# Forward Engineering Object Recognition: A Scalable Approach

by

Nicolas Pinto

Ingénieur (M.S.) de l'Université de Technologie, Belfort-Montbéliard, 2006
and
M.S. Ecole Nationale Supérieure d'Ingénieurs Sud Alsace, 2007

Submitted to the Department of Brain and Cognitive Sciences
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN COMPUTATIONAL NEUROSCIENCE
AT THE
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2011



Signature of Author: _____
Nicolas Pinto
Department of Brain and Cognitive Sciences
December 3rd, 2010

Certified by: _____
James J. DiCarlo
Associate Professor of Neuroscience
Thesis Supervisor

Accepted by: _____
Earl K. Miller
Picower Professor of Neuroscience
Director, BCS Graduate Program

*To my family*

**Forward Engineering Object Recognition: A Scalable Approach**

Simple Baselines, Efficient Benchmarks, High-Throughput Solution Discovery and Large-Scale Applications

by
Nicolas Pinto

Submitted to the Department of Brain and Cognitive Sciences
on December 3rd, 2010, in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computational Neuroscience

## Abstract

The ease with which we recognize visual objects belies the computational difficulty of this feat. Despite the concerted efforts of both biological and computer vision research communities over the last forty years, human-level visual recognition remains an unsolved problem. The impact of a robust yet inexpensive solution would dramatically change computer science and neuroscience, unleashing a host of innovative applications in our modern society.

In this thesis, we identify two operational barriers that have obstructed progress towards finding a solution – namely the lack of clear indicators and operational definitions of success, and the currently limited exploration of the staggeringly large hypothesis space of biologically-inspired solutions. To break down these barriers, we first establish new neuroscience-motivated baselines and new suites of fully-controlled benchmarks for object and face recognition. We also compare and contrast a variety of high-level visual systems, both artificial (state-of-the-art computer vision) and biological (humans). Then, we propose a simple high-throughput approach to undertake a systematic exploration of the biologically-inspired model class. By leveraging recent advances in massively parallel computing, we show that it is possible to generate a multitude of candidate models, screen them for desirable properties and discover robust solutions. Finally, we validate the scalability of our approach by showing its potential on large-scale "real-world" applications.

Taken together, this thesis represents a humble attempt towards an integrated approach to the problem of brain-inspired object recognition – spanning the engineering, specification, evaluation, and application of an interesting set of biologically-inspired ideas, driven and enabled by massively parallel technology. Even relatively early instantiations of this approach yield algorithms that achieve state-of-the-art performance in object recognition tasks and can generalize to other image domains. In addition, it offers insight into which computational ideas may be important for achieving this performance. Such insights can then be "fed back" into the design of new candidate models, constraining the search space and suggesting improvements, further guiding "evolutionary" progress.

We hope that our results will point a new way forward, both in the creation of powerful yet simple computer vision systems and in providing insights into the computational underpinnings of biological vision.

Thesis Supervisor: James J. DiCarlo
Title: Associate Professor of Neuroscience

# Acknowledgments

Working on this PhD thesis was an exciting and uniquely rewarding experience, and many people have helped tremendously, both directly and indirectly.

First and foremost, I would like to thank my wonderful parents, sister and grandparents. This adventure would not have been possible – nor even conceivable – without their unwavering support and unconditional love. *To them, I dedicate this thesis.*

I am deeply grateful to my lovely girlfriend, Ya-Ting Amy "Mega Mouh" Chuang, for being such a great companion, and for her generous understanding when I worked many long hours. I am thankful to my friend Nicolas Poilvert – who has encouraged and inspired me to apply to MIT – for the innumerable enlightening discussions about science, technology and everything else. For their great friendship, advices, and for keeping me sane, I would like to thank my friends at 253 Washington: Simon "Lord of the Night" Laflamme and Dustin Smith; at 89 Plymouth: Benjamin Gathier, Herve Martins-Rivas, and Matthieu Varagnat; at MIT/Harvard: Hila Hashemi, Tilke Judd, Justin Riley and Zak Stone; at JP: John Acosta and Jonah Feldman; at the "Keep It Real Asso": Pierre Lardoux, Sebastien Ranouille and Julien Vitte; at "la Ouf Malade": Fopa, Forik, Jimmy Jym, Les Ours, Lizzie, Pti Biket, Spat, Yas; at the "MPCrew": Thomas Hillion and Bertrand Pallier.

I want to express my most sincere gratitude to James DiCarlo, my PhD advisor, for

welcoming me in the incredible intellectual environment he created, and for providing the right balance of independence, challenge, supportive guidance, and constructive – yet gentle – criticism. Special consideration also goes out to David Cox for his dedicated mentoring throughout this thesis, for emphasizing the critical importance of careful engineering in life sciences, and for his inspiring "can-do" / "thinking-out-of-the-box" attitude. I would also like to acknowledge the members of my PhD committee: Pawan Sinha and Antonio Torralba for their insightful discussions, guidance and patience. The creativity, unprecedented rigor and brilliance of my entire committee continue to astound me. Thank you for reminding us of the importance of optimizing scientific progress – not credit nor profit.

Thanks to my companions at the DiCarlo Lab: Arash Afraz, Jennie Deutsch, Ha Hong, Chou Hung, Elias Issa, Ben Kennedy, Nuo Li, Najib Majaj, Marie Maloof, Daniel Oreper, Marino Pagan, Alex Papanastassiou, Nicole Rust, Kailyn Schmidt, Ethan Soloman and Davide Zoccolan – for their abundant help and heated discussions. Thanks to my MS interns: Youssef Barhomi, Abhijit Bendale, Benoit Corda, David Doukhan, Mehdi Mirza-Mohammadi, Pantea Moghimi and Xiyuan Phil Zhang – for their patience, contributions and for shaping my mentorship skills. Thanks to the BCS department and especially Denise Heintze for her invaluable assistance.

Finally, I would like to acknowledge the generous technical, financial and moral support provided by NVIDIA (David Luebke, Bill Dally, David Kirk, Joe Stam, John Roberts), NCSA/UIUC (Wen-mei Hwu), Harvard (Hanspeter Pfister, Robert Parrot), and Microsoft Research (Zhengyou Zhang).

*Thank you, I owe it all to you.*

# Table of Contents

# Part I

# Introduction

# Chapter 1

# Problem Statement

> "Of all (our) senses, vision is the first, the most extensive; accordingly, if they (our eyes) were given to us for discovering truth, it (vision) would have a greater role by itself than all the others combined."
>
> *[Malebranche, 1997]*

Vision is incontestably our richest sense and it goes far beyond the mechanisms of seeing [1]. Human vision is striking in its effortlessness as we perceive our visual environment almost instantaneously [Potter and Levy, 1969; Thorpe *et al.*, 1996], and it is surely not a coincidence that meticulous evolutionary refinement has dedicated a large fraction of the primate brain to the processing of visual information (e.g. approximately 50% of the macaque neocortex). Understanding the cortical principles underlying this remarkable feat has been a primary focus of neuroscience, but despite decades of collaborative multi-disciplinary research, the "vision problem" remains unsolved [Masland and Martin, 2007].

[1] Don't we say "an *image* is worth a thousand words", "I *see* what you mean" when we understand a new concept, or "Albert Einstein was a *visionary* scientist"?

## 1.1 Goals and Motivations

> "But why do you want to study vision? Vision is trivial!"
>
> *A friend*

In this thesis, we will focus on the *computational* aspects of one of the major stumbling blocks in vision research: *object recognition* (i.e. detecting and identifying single objects in visual scenes as well as categorizing or grouping sets of similar objects together). To completely solve this problem from a computational perspective means instantiating and understanding models that replicate or outperform natural systems while retaining the essential computational principles that biology exploits.

Until the early seventies, artificial intelligence researchers seriously underestimated what a difficult task tackling this problem would be [Papert, 1966], essentially because humans are so good at it that it prevents them from performing critical introspection. Today, even the most sophisticated computer vision system cannot rival a child's object recognition capabilities, but we now have a more accurate understanding of what makes the problem so challenging. First, it is important to note that even though we perceive our visual world as three-dimensional (3-D) our retinas are only two-dimensional (2-D). In addition, any 3-D structure (object) can produce a virtually infinite number of 2-D retinal projections (images) depending on its pose or distance, the lighting and illumination of the environment, the surrounding objects, etc. It is very unlikely that the viewer's retina, which contains millions of sensors, will see the exact same image twice in its lifetime, and even if similar objects can cast very different images, his visual system must group them together in a meaningful way and in a fraction of a second. Ultimately, a successful system must solve this high-dimensional ill-posed and ill-conditioned inverse problem [Bertero *et al.*, 1988; Ullman, 1996; Edelman, 1999] by maintaining a high tolerance ("invariance") to a wide range of identity-preserving transformations [DiCarlo and Cox, 2007], being highly selective to complex objects (e.g. discriminating two faces, even though they share very similar shapes), recognizing with high accuracy tens of thousands of images following a single exposure [Shepard, 1967; Standing *et al.*, 1970; Standing, 1973; Brady *et al.*, 2008], and storing/retrieving tens of

thousands of object categories [Biederman, 1987] – including objects that it has never seen before.

Considering that we evolved to the point where vision is absolutely critical to the survival of our species, it is legitimate to speculate that brain-inspired object recognition, if solved, would have a substantial impact on our society. The solution would certainly elicit new insights into how the brain works, affect the way we think about cortical information processing and learning, and profoundly influence cognitive, systems and molecular neuroscience, computer science and artificial intelligence. Enabling computers to become visually aware of their environment would also enhance or unlock innovative applications and technologies ranging from data mining to medical image analysis through fully autonomous vehicles.

# Chapter 2

# Background

> "Science is built up with facts, as a house is with stones. But a collection of facts is no more a science than a heap of stones is a house"
>
> *Henri Poincaré*

Biological brains (e.g. in primates) are the only systems to convey robust solutions to the vision problems such as object recognition. Since our goal is to mimic their abilities, it would seem productive to draw inspiration from their design, architecture and function. Unfortunately, we do not have access to their blueprints, making "forward-engineering" impossible. Instead, neuroscientists have attempted to "reverse-engineer" natural systems [Hapgood, 2006; Cox, 2007; Adee, 2008] and better characterize what constitutes a solution for the brain, how it solves it, why it behaves the way it does, and how it evolved up to this point.

Even though the subject has been extensively studied and documented in a large body of literature for decades, it is still poorly understood. In the following sections, we provide a brief overview of what is known and highlight important "broad-stroke" properties of biological visual object recognition.

## 2.1 From the Retina to the Neocortex

When photons from a visual scene hit the retina, the light is transduced into electrochemical signals by its photoreceptors and ultimately encoded by retinal ganglion cells [Koch *et al.*, 1982; Meister *et al.*, 1995; Meister, 1996] into action potentials[1]. The information then goes through the lateral geniculate nucleus (LGN) in the thalamus [Reinagel *et al.*, 1999; Sherman, 2001; Alonso *et al.*, 1996; Lesica and Stanley, 2004], and culminates in the first cortical area: the primary visual cortex (V1).

From V1, two somewhat parallel processing pathways emerge: the *dorsal* stream and the *ventral* stream [Felleman and Van Essen, 1991; Haxby *et al.*, 1991; Ungerleider *et al.*, 1982; Mishkin *et al.*, 1983; DeYoe and Van Essen, 1988; Ungerleider and Haxby, 1994; Goodale and Milner, 1992] (see Figure 2.1). The dorsal stream, usually described as the "where" pathway, is thought to build an action-oriented object representation and to process motion information and object locations. The ventral stream, also known as the "what" pathway, is assumed to form an invariant object representation that supports highly selective and robust recognition [Logothetis and Sheinberg, 1996; Tanaka, 1996a; Rolls, 2000; Gross, 2002].

This dissociation between "what" and "where" seems natural, especially considering that David Marr, one of the most prominent computational vision neuroscientists, defines the "purpose of vision" as "knowing *what* is *where* by looking" [Marr, 1982]. However, it is probably an over-simplification to think that there is a complete dichotomy between the functions of these two processing streams. The cortical areas in the dorsal and ventral pathways are in fact heavily interconnected [Farivar, 2009], making the functional dissociation hypothesis "difficult if not impossible to test" [Cardoso-Leite and Gorea, 2010].

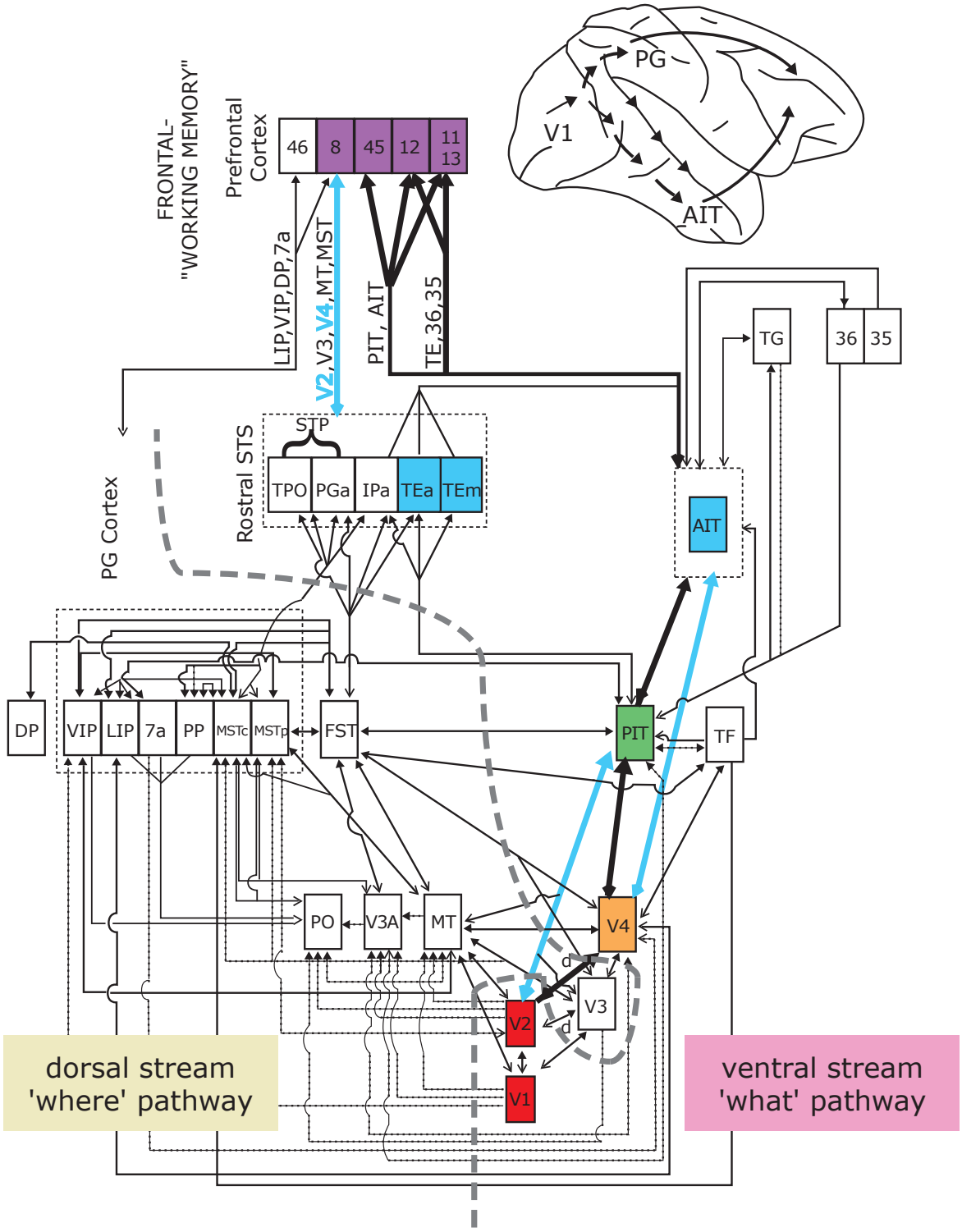



**Figure 2.1: The ventral and dorsal streams in the visual cortex (rhesus monkey).** Figure and schematic modified from [Felleman and Van Essen, 1991; Serre *et al.*, 2007a].

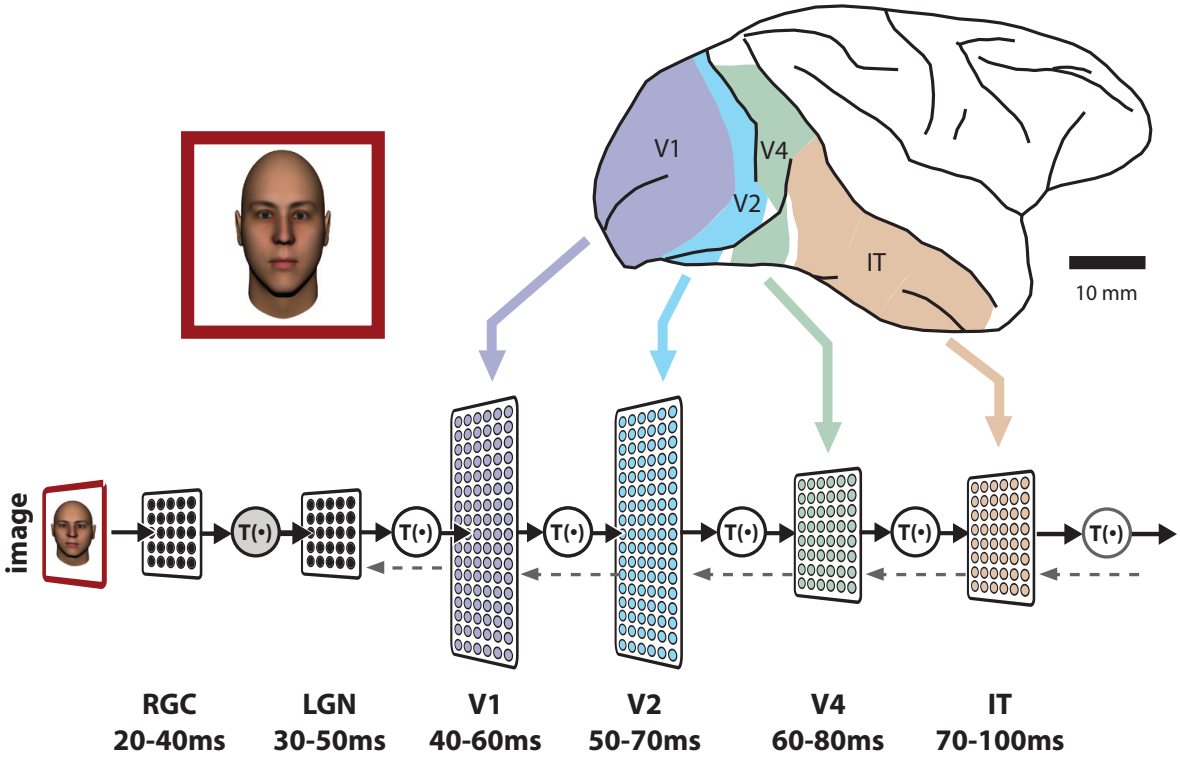



**Figure 2.2: Neuronal populations along the ventral visual processing stream.** Although we ultimately seek to understand how object recognition is accomplished by the human brain, the rhesus monkey is our current best model system. Like humans, this species has high visual acuity, relies heavily on vision (approx. 50% of macaque neocortex is devoted to vision) and easily performs visual recognition tasks. Moreover, the monkey visual areas have been mapped and are hierarchically organized [Felleman and Van Essen, 1991], and the ventral visual stream is known to be critical for complex object discrimination (colored areas, see text). We show a lateral schematic of a rhesus monkey brain (adapted from [Felleman and Van Essen, 1991]). We conceptualize each stage of the ventral stream as a new population representation. The lower panels schematically illustrate these populations in early visual areas and at successively higher stages along the ventral visual stream – their relative size loosely reflects their relative output dimensionality (approximate number of feed-forward projection neurons). A given pattern of photons from the world (here, a face) is transduced into neuronal activity at the retina and is progressively and rapidly transformed and re-represented in each population, perhaps by a common canonical transformation (T). Solid arrows indicate the direction of visual information flow based on neuronal latency (approx. 100 ms latency in IT), but this does not preclude fast feedback both within and between areas. Figure and caption modified from [DiCarlo and Cox, 2007], feed-forward timing data from [Thorpe and Imbert, 1989].

## 2.2 Object Recognition in the Ventral Visual Stream

### 2.2.1 Architecture

A large body of experimental work done in primates has established the ventral visual stream as the fundamental support for object recognition in the neocortex [Logothetis and Sheinberg, 1996; Tanaka, 1996a; Rolls, 2000; Gross, 2002]. This pathway is organized as a hierarchical structure (see Figure 2.2, modified from [Felleman and Van Essen, 1991; Serre *et al.*, 2007a]) composed of a series of complex neuronal transformations from V1 to extrastriate visual areas V2, V4, and to the inferior temporal cortex (IT, which can be further divided into two parts: posterior and anterior or PIT and AIT) [Gross *et al.*, 1972; Tanaka, 1996a; Quiroga *et al.*, 2005; DiCarlo and Cox, 2007].

IT is considered the last purely visual area in this hierarchy [Gross, 1994]. Lesions in IT cause severe deficits in visual recognition behavior [Dean, 1976, 1982; Holmes and Gross, 1984; Weiskrantz and Saunders, 1984] and electrical microstimulation of IT neurons can bias object discrimination performance [Afraz *et al.*, 2006; Kawasaki and Sheinberg, 2008]. IT is also thought to play a role in short-term memory, visual memory and their consolidation by projecting to areas in the medial temporal lobe (MTL) and the pre-frontal cortex (PFC) [Miyashita, 1988; Sakai and Miyashita, 1991; Zola-Morgan and Squire, 1993; Miyashita, 1993; Miller and Desimone, 1994]. PFC receives a major input from IT and is thought to be responsible for linking/controlling perception and action [Miller, 2000; Miller and Cohen, 2003]. In the context of visual recognition, PFC is where the final object identification/categorization is thought to happen [Freedman *et al.*, 2001, 2002].

### 2.2.2 Processing Speed, Feed-forward vs. Feed-back

When confronted with rapid categorization tasks, the primate visual system is able to recognize objects with high accuracy in less than 200 ms [Potter and Levy, 1969; Biederman, 1972; Biederman *et al.*, 1974; Potter, 1975, 1976; Potter *et al.*, 2002; Oram

[1] Action potentials are short-lasting events in which the electrical membrane potential of a cell rapidly rises and falls, following a stereotyped trajectory [Wikipedia].

and Perrett, 1992; Thorpe *et al.*, 1996; Hung *et al.*, 2005], which imposes a stringent constraint on how many steps can be performed and how long they can take. At this speed, the information processing required for immediate recognition along the ventral stream hierarchy has to be mostly feed-forward [Serre *et al.*, 2007a]. Although *local* feed-back connections within cortical areas are highly likely to happen within this period (e.g. to implement signal normalization), there may simply be no time for significant interactions between cortical areas (i.e. long-range feed-back, e.g. PFC to V1).

It is obvious that visual object recognition does not only happen in a glimpse, and complex categorization tasks (e.g. when heavy clutter is present) may require feed-back processing (e.g. to disambiguate uncertainties in the scene). In fact, feed-back processing is omnipresent in the brain. For example, feed-back from V1 to LGN is ten times as numerous as feed-forward connections from LGN to V1 [Sherman and Koch, 1990]. Interestingly, V1 receives more top-down projections from a larger number of extrastriate cortical areas than it sends bottom-up projections (e.g. see [Boussaoud *et al.*, 1990; Barone *et al.*, 2000]), and, although V1 does not project directly to IT, it receives signals from IT (e.g. see [Rockland and Van Hoesen, 1994]).

Feed-forward and feed-back projections have very different properties. Bottom-up connections are more restricted and focused whereas top-down connections are more pervasive and diffuse [Callaway, 1998; Zeki and Shipp, 1989; Shipp and Zeki, 1989; Salin and Bullier, 1995]. This contrast suggests an important division of their function with respect to visual object recognition: feed-forward processing probably builds up a structured object representation from simple to complex (see below) with activity from lower areas driving activity in higher regions, while feed-back processing enhances the representation from lower areas by controlling and biasing their activity depending on global and contextual information from higher regions.

In this thesis, we will focus on feed-forward processing.

### 2.2.3 Selectivity and Tolerance

As we discussed in Chapter 1, object recognition requires both *selectivity*, so that objects or categories of objects that appear similar can still be distinguished, and *tolerance*

("invariance"), so an object that undergoes rigid or non-rigid transformations, or if it is subject to complex interactions with its environment (e.g. lighting or reflection), can still be perceived as the same entity.

Although we do not know exactly how these two conflicting properties are gradually built up and kept balanced in the visual cortex, many essential elements have been discovered over the last decades. There is ample evidence for a general increase of the complexity of the neurons' preferred stimuli (leading to more selectivity) as well as an expansion of the size of their receptive fields[2] (associated with more tolerance to variation in position, scale, viewpoint, etc.) along the ventral stream hierarchy [Hubel and Wiesel, 1968; Desimone, 1991; Perrett and Oram, 1993; Kobatake and Tanaka, 1994; Logothetis *et al.*, 1995; Logothetis and Sheinberg, 1996; Tanaka, 1996b; Anzai *et al.*, 2007; Rust and DiCarlo, 2010].

The seminal experiments of Hubel and Wiesel over fifty years ago revealed that some neurons in V1 – called "simple cells" – were highly selective to bars/edges of specific orientations, spatial frequencies and phases, while other neurons – called "complex cells" – were invariant to the specific phase or position of the optimal oriented bar (note that other cells, sometimes dubbed "hypercomplex cells", were also shown to have end-inhibition and side-inhibition properties, as well as tuning to bars of particular lengths). V1 neurons were also shown to form retinotopic maps of the visual environment [Hubel and Wiesel, 1959, 1962, 1968, 1977, 1998].

Neurons in V2, in addition to their orientation tuning [Hubel and Wiesel, 1965], are also thought to be sensitive to specific angles or contours (including illusory borders) by encoding combinations of local oriented bars [Von der Heydt *et al.*, 1982; Boynton and Hegdé, 2004; Ito and Komatsu, 2004; Anzai *et al.*, 2007]. The spatial organization of V2 is also believed to be retinotopic.

Neurons in V4 have been implicated in the processing of shapes of intermediate complexity [Kobatake and Tanaka, 1994] like texture patterns or boundary fragments [Pasupathy and Connor, 1999, 2001, 2002]. V4 neurons are generally thought to be selective to Cartesian gratings (e.g. bars or sinusoidal waves) [Desimone and Schein,

[2]The receptive field of a sensory neuron is a region of space in which the presence of a stimulus will alter the firing of that neuron [Wikipedia].

1987] and non-Cartesian gratings (e.g polar or hyperbolic) [Gallant *et al.*, 1993, 1996] (with a bias for non-Cartesian ones). They also exhibit three-dimensional orientation tuning [Hinkle and Connor, 2002]. The retinotopy in V4 (and beyond) has not been fully validated yet (the subject is still controversial – see [Wandell *et al.*, 2005] for example).

Although both V4 and IT neurons seem to encode natural images equally well, the IT population exhibits increased (negative) sensitivity to statistical scrambling of those images (i.e. "less natural") [Rust and DiCarlo, 2010], hence devoting more resources to the construction of an invariant representation for "behaviorally relevant" stimuli. Interestingly, this scrambling sensitivity is proportional to the receptive field size in both V4 and IT.

Building upon the V4 neuronal representation, neurons in IT become selective for more complex feature conjunctions *and* more robust to many identity-preserving stimulus transformations. Since more than forty years ago [Gross *et al.*, 1967, 1969, 1972], single-unit electrophysiological recordings have established that neurons in IT are indeed tuned to complex shapes including faces and other body parts [Rolls *et al.*, 1982; Perrett *et al.*, 1984, 1987; Logothetis and Sheinberg, 1996; Tanaka, 1996a; de Beeck *et al.*, 2001; Brincat and Connor, 2004] . This high selectivity is preserved within the neurons' receptive fields [Rust and DiCarlo, 2010] and is robust to eye movements [DiCarlo and Maunsell, 2000] or to the task being performed [Chelazzi *et al.*, 1998]. It is also tolerant to changes in position and scale [Desimone *et al.*, 1984; Rolls, 1984, 1991; Ito *et al.*, 1995; Hung *et al.*, 2005], rotation and pose/view [Perrett *et al.*, 1985; Perrett and Oram, 1993; Tovee *et al.*, 1994; Logothetis *et al.*, 1995; Booth and Rolls, 1998; Op De Beeck and Vogels, 2000; Quiroga *et al.*, 2005], illumination, texture and low-level shape cues [Sary *et al.*, 1993; Vogels and Orban, 1996], clutter [Zoccolan *et al.*, 2005], etc. Interestingly, adult IT does not necessarily require extensive previous experience with novel objects or views to maintain this tolerance [Logothetis *et al.*, 1995; Hung *et al.*, 2005]. Moreover, not all IT neurons are highly selective or highly tolerant and a trade-off was recently found between these two properties: individual neurons with high selectivity have low tolerance and vice versa [Zoccolan *et al.*, 2007]. Finally, even though many studies have tried to gain insight into the critical features and shape di-

mensions preferred by IT neurons (e.g. by studying the neurons' responses to complex objects and gradual decomposition of objects' "parts" – effectively trying to come up with a theory similar to the one found by Hubel and Wiesel in V1 with orientation and spatial frequency tunings) [Gross *et al.*, 1972; Kobatake and Tanaka, 1994; Tsunoda *et al.*, 2001; Pollen *et al.*, 2002; Kayaert *et al.*, 2003; Yamane *et al.*, 2008], we still lack a clear fundamental understanding.

### 2.2.4 Development, Learning and Plasticity

Genetically controlled processes and experience-dependent processes are deeply intertwined throughout the development of the visual cortex (which involves laminar/columnar organization, neuron wiring and synaptic weight "specification"). When newborns open their eyes for the first time, their brains begin to understand the surrounding visual environment at an incredible speed. Within only a few days after birth, human infants begin to exhibit sophisticated visual skills – primary amongst these is the ability to preferentially orient towards complex stimuli like faces [Johnson *et al.*, 1991]. To what extent primates' visual skills are innate or learned is still under debate in the community, and precisely identifying the endogenous and exogenous factors remains an important open research area [Morton and Johnson, 1991; Johnson and Aslin, 1996; Fiser and Aslin, 2002; Johnson, 2005; Sugita, 2008]. Interestingly, studies with human patients who acquire sight late in life have suggested that the development of these visual abilities is unlikely to be merely the result of maturation in the brain [von Senden, 1960; Gregory and Wallace, 1963; Valvo, 1968; Valvo *et al.*, 1971; Kellman *et al.*, 1986; Fine *et al.*, 2003; Maurer *et al.*, 2005; Ostrovsky *et al.*, 2006, 2009]. Instead, visual development seems to be relying more on learning about the environment (i.e. exogenous factors – not only visual but also multi-modal) than previously appreciated. Experiments in the developing brain involving "rewiring" of visual inputs to the auditory cortex reached the same conclusion by showing significant activity-dependent remodeling of the auditory cortex to process visual information (e.g. neurons developed receptive fields similar to the ones normally found in the visual cortex) [Horng and Sur, 2006].

Learning and plasticity have been reported in all levels of the ventral stream in humans and monkeys [Ghose, 2004; Kourtzi and DiCarlo, 2006; Hoffman and Logothetis, 2009; Kourtzi, 2010]:

- in V1 and V2 [Wiesel and Hubel, 1963; Singer *et al.*, 1982; Karni and Sagi, 1991; Reber *et al.*, 1998; Aizenstein *et al.*, 2000; Schuett *et al.*, 2001; Crist *et al.*, 2001; Yao and Dan, 2001; Lee *et al.*, 2002; Reber *et al.*, 2003], even though the contribution of V1/V2 after training remains controversial [DeAngelis *et al.*, 1995; Schoups *et al.*, 2001; Ghose *et al.*, 2002];
- in V4 [Rainer *et al.*, 2004; Yang and Maunsell, 2004];
- in IT [Logothetis *et al.*, 1995; Rolls, 1995; Dolan *et al.*, 1997; Kobatake *et al.*, 1998; Booth and Rolls, 1998; Erickson *et al.*, 2000; Jagadeesh *et al.*, 2001; Baker *et al.*, 2002; Sigala and Logothetis, 2002; Freedman *et al.*, 2003, 2006; Op de Beeck *et al.*, 2006; Gauthier *et al.*, 1999; DeGutis and D'Esposito, 2007; Jiang *et al.*, 2007; Li and DiCarlo, 2008, 2010];
- and in higher areas like PFC [Rainer and Miller, 2000; Freedman *et al.*, 2001, 2002, 2003; Pasupathy and Miller, 2005].

The specifics of how learning is taking place along the hierarchy and how it is shaping the visual representations are still largely unclear, but significant progress has been made. Learning during object discrimination tasks seems to begin at higher levels of the hierarchy (e.g. IT) for easy tasks and proceeds to lower levels (e.g. V1, V2 or V4) for finer and more complex tasks [Sigman *et al.*, 2005; Hochstein and Ahissar, 2002; Ahissar and Hochstein, 2004; Ghose, 2004] (and these changes are presumably happening at faster time scales in higher areas). For instance, studies have reported that V4 neurons change their tuning for (fine) orientation discrimination tasks [Yang and Maunsell, 2004] while IT neurons do not [Vogels and Orban, 1994]. IT neurons are tuned for specific views or parts of complex objects, and this tuning and organization depend partly on *supervised* visual experience (i.e. IT cells have higher selectivity for object views that have been presented during training, and they tend to form clusters with similar preferences) [Logothetis *et al.*, 1995; Tanaka, 1996b; Kobatake *et al.*, 1998; DiCarlo

and Maunsell, 2000; Erickson *et al.*, 2000; Sheinberg and Logothetis, 2001; Jagadeesh *et al.*, 2001; Baker *et al.*, 2002; Sigala and Logothetis, 2002; Freedman *et al.*, 2003]. Interestingly, view-tuning of IT neurons has also been observed during *unsupervised* (i.e. passive) exposure to novel complex objects [Booth and Rolls, 1998; Freedman *et al.*, 2006]. In this context, time (and motion) could provide an implicit supervised signal and thus clues to learning which transformations are object identity-preserving or not (i.e. two retinal projections following each other are very likely to describe the same visual scene and hence contain the same objects). Recently, the hypothesis that the ventral stream might learn to be "invariant" using the coherent evolution of spatio-temporal statistics [Földiak, 1991; Wallis *et al.*, 1993; Wallis, 1996; Wallis and Rolls, 1997; Wiskott and Sejnowski, 2002] (see Section 2.3.3 for more details) has been confirmed psychophysically [Wallis and Bülthoff, 2001; Brady and Kersten, 2003; Cox *et al.*, 2005] and physiologically [Li and DiCarlo, 2008, 2010]. More specifically, Cox, Li and DiCarlo showed that careful alteration of the temporal contiguity under natural vision conditions caused specific changes in human behavior as well as IT position and size tolerance. This learning occurred very rapidly (1-2 hours) and is compatible with Hebbian learning [Hebb, 1949].

## 2.3 Biologically-Inspired Models of Visual Object Recognition

> "Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful."
>
> *George E. P. Box*

Efforts towards modeling visual object recognition with various degrees of inspiration from biology and the brain started around the seventies. Since then, there has been considerable debate in the neuroscience and computer vision communities regarding the most appropriate way to tackle the problem. Two main computational theories have emerged to try to describe how humans recognize objects and categories of objects.

On the one hand, there is the *part-based* approach (also called *correspondence-based* or *component-based*) which uses three-dimensional (CAD-like) canonical "mental" models representing different objects' structure and relationships between their visual parts. Recognition is then performed by means of correspondences between the various parts.

On the other hand, there is the *feature-based* approach (also called *view-based* or *holistic*) where different objects' "views" are learned and encoded into an invariant visual representation. Recognition is then performed by matching the views stored in memory.

Today, the distinction between the two directions may appear blurry in the computer vision literature as objects can share "parts", "components" or "fragments" that can sometimes be encoded by image "features" or "descriptors" that essentially act as (local) "template-matching" operators.

### 2.3.1 Part-Based Models

The part-based approach seeks structural correspondences between image regions and parts of object canonical models. As a consequence, this approach relies on (1) detecting and identifying volumetric primitives (i.e. the parts), and (2) building intrinsic geometrical relations between them (i.e. above or below, left or right, larger or smaller, etc.) [Fischler and Elschlager, 1973; Marr and Nishihara, 1978; Marr *et al.*, 1980; Marr, 1982]. Many volumetric primitives have been proposed including generalized cylinders [Binford, 1971], deformable superquadrics [Pentland, 1986, 1987; Dickinson *et al.*, 1992] with their extensions [Zhou and Kambhamettu, 1999], and "geons" (geometric primitives) [Biederman, 1987; Biederman and Cooper, 1991; Hummel and Biederman, 1992; Zhou and Kambhamettu, 2002]. Part-based models are intuitively organized as hierarchies where smaller and simpler parts are at the lower layers and more complex parts are gradually build up.

In the brain, converting objects from a retinal representation to canonical models by finding correspondences between components is not straightforward to model. It could be implemented by dynamically linking and synchronizing patterns of neuronal

activities representing the components [von der Malsburg, 1981; Feldman, 1982] and selectively routing the information that will activate the next layer. This mechanism explicitly builds up an object representation that is tolerant to image variations with minimal loss of information [Pitts and McCulloch, 1947]. Routing architectures have been proposed with a wide range of neuro-plausibility [Postma *et al.*, 1997; Arathorn, 2002]. Examples include the "shifter-circuit" [Anderson and Van Essen, 1987; Olshausen *et al.*, 1993, 1995] or the "gain-field" model [Salinas and Abbott, 1997] where dedicated control neurons act as routing circuits to regulate/modulate the flow of information in the hierarchy.

Many computer vision systems that do not necessarily seek neuro-plausible implementations have exploited the part-based approach [Lanitis *et al.*, 1995; Burl and Perona, 1996; Felzenszwalb and Huttenlocher, 2000; Fergus *et al.*, 2003; Fei-Fei *et al.*, 2003, 2004b; Holub and Perona, 2005; Crandall *et al.*, 2005; Crandall and Huttenlocher, 2006; Felzenszwalb and Huttenlocher, 2005; Felzenszwalb *et al.*, 2010b,a], and some have even provided invaluable insights into computational principles that biological object recognition might use [Ullman, 1989; Yuille, 1991; Amit and Geman, 1999; Heisele *et al.*, 2001; Mohan *et al.*, 2001; Ullman *et al.*, 2002; Harel *et al.*, 2007; Ullman, 2007; Lerner *et al.*, 2008].

### Advantages

Part-based models are attractive because they can encode the size and position of objects and their parts explicitly, so there is no need for an additional system to decode this information.

In addition, they require minimal information loss to achieve their tolerance to image variations, so they can generalize very well to new stimuli and new visual conditions.

### Disadvantages

Nevertheless, they suffer from the fact that detecting volumetric primitives in natural images is (still) a very hard (object segmentation or recognition) problem on its own [Bulthoff *et al.*, 1995; Edelman and Duvdevani-Bar, 1997].

They also need time consuming layer-to-layer communications (with complex top-down mechanisms) to build/use the canonical "mental" models, and, as a result they do not fit well into the physiological constraints required to perform highly accurate rapid categorization tasks.

### 2.3.2 Feature-Based Models

The feature-based approach aims to gradually build up an "invariant" visual representation that encodes views of different objects. Most feature-based models are organized as deep hierarchies (that are usually more rigid and specific than the ventral stream's). They are composed of multiple layers that (1) selectively detect (AND-like operation) features in the image, and (2) combine (OR-like operation) the convergent outputs from simpler feature detectors into more and more complex ones. This approach thus explicitly separates *selectivity* to complex object shapes and *tolerance* to object identity-preserving image transformations (see Section 2.2.3) into two operations that are alternated throughout the hierarchy.

The computational principles behind most feature-based models can be traced back to the pioneering (and Nobel Prize winning) work of Hubel and Wiesel [Hubel and Wiesel, 1959, 1962, 1968, 1977, 1998]. They proposed a model based on the responses of the neurons they found in V1:

1. The *simple cells*' orientation and position tuning for bars within their (small) receptive fields could be achieved by combining aligned "center-surround" cells from LGN (that are known to be highly responsive to circular spots of light); see Figure 2.3.

2. The *complex cells*' tolerance to the exact position of the bars within their (larger) receptive fields could be achieved by pooling over adjacent simple cells that share the same preferred orientation but have shifted receptive fields; see Figure 2.4.

Although claims have been made that (1) "beyond this early insight, systems neuroscience has not provided a breakthrough" for the modeling of object recognition [DiCarlo and Cox, 2007], (2) we do not yet completely understand the behavior of *all*

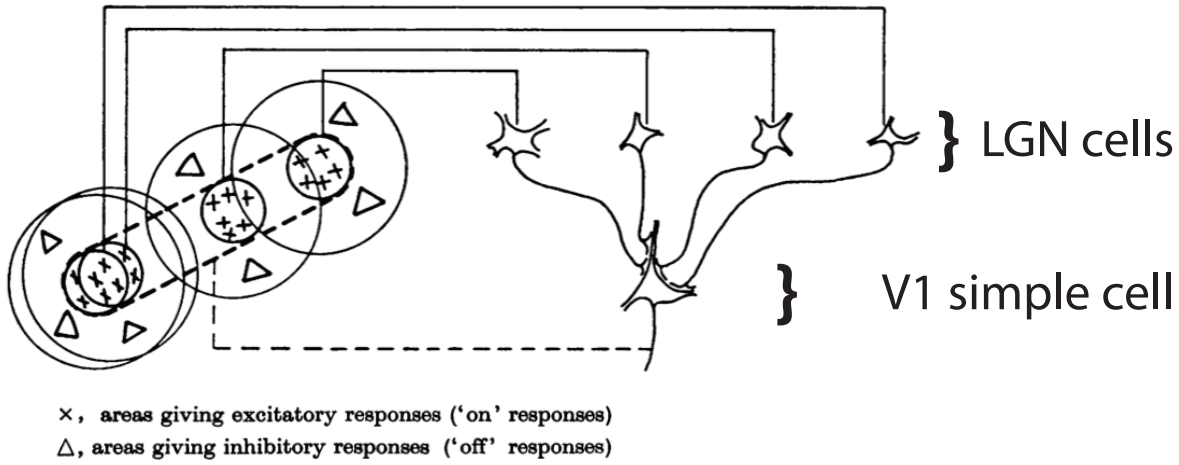



**Figure 2.3: Possible scheme proposed by Hubel and Wiesel for explaining the organization of simple cells in V1.** A large number of cells from the lateral geniculate nucleus (LGN), of which four are illustrated in the upper right in the figure, have receptive fields with "on" centers arranged along a straight line on the retina. All of these project upon a single cortical cell, and the synapses are supposed to be excitatory. The receptive field of the simple cell will then have an elongated "on" center (selective for oriented bars) indicated by the interrupted lines in the receptive-field diagram to the left of the figure. Figure and caption modified from [Hubel and Wiesel, 1962].

V1 neurons [Olshausen and Field, 2005; Carandini *et al.*, 2005], and (3) this description of V1 is certainly an over simplification of what is currently known [Jones and Palmer, 1987; Carandini *et al.*, 1997; Simoncelli and Olshausen, 2001], this intuitive idea of describing V1's simple cells' responses as specific combinations of afferent LGN neurons, and complex cells as specific combinations of simple cells could presumably be extended and applied again to model the responses of neurons in V2 (from V1), V4 (from V2), and IT (from V4). Some recent work has been done in this direction for V2 [Plebe, 2007], and V4 [Cadieu *et al.*, 2007].

In parallel with Hubel and Wiesel's studies in the fifties and sixties, Selfridge proposed a multi-layer feature detector pattern recognition system called "Pandemonium" [Selfridge, 1966], and Rosenblatt described his featured-based multi-layer *Perceptron* [Rosenblatt, 1958, 1961; Minsky and Papert, 1987]. Inspired by these early insights, many feature-based multi-layer hierarchical models have been presented in the seventies [Fukushima, 1969; Marko and Giebel, 1970; Fukushima, 1970; Giebel, 1971; Marko,

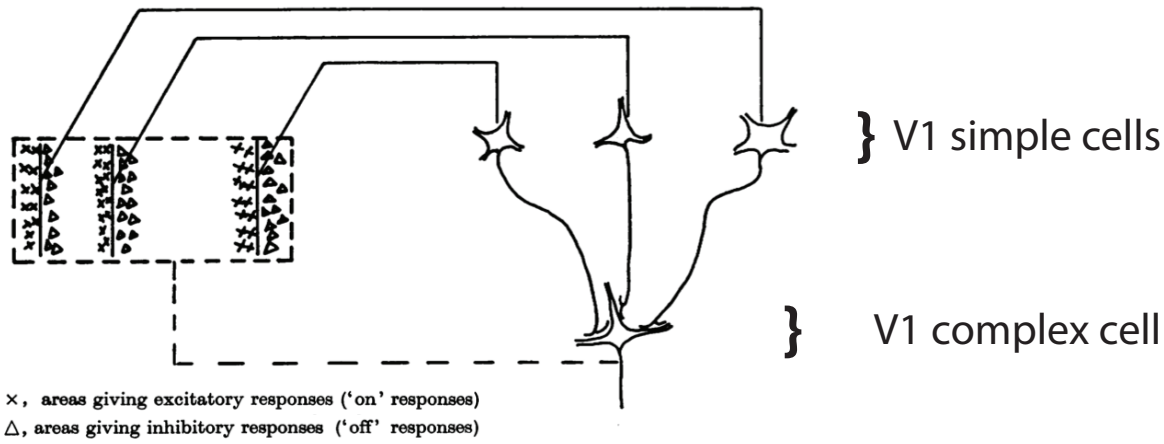



**Figure 2.4: Possible scheme proposed by Hubel and Wiesel for explaining the organization of complex cells in V1.** A number of simple cells, of which three are shown schematically, are imagined to project to a single complex cell. Each projecting neuron has a receptive field arranged as shown to the left: an excitatory region to the left and an inhibitory region to the right of a vertical straight-line boundary. The boundaries of the fields are staggered within an area outlined by the interrupted lines. Any vertical-edge stimulus falling across this rectangle, *regardless of its position*, will excite some simple cells, leading to excitation of the complex cell. Figure and caption modified from [Hubel and Wiesel, 1962].

1974], and in the eighties, when Fukushima proposed the *Neocognitron* [Fukushima, 1980; Fukushima and Miyake, 1982; Fukushima, 1988, 1989] as an extension of his earlier work on the *Cognitron* [Fukushima, 1975]. The Neocognitron (see Figure 2.5) is a feed-forward model that has a retinotopically organized input layer representing pixels and multiple layers of processing with alternating "S-cells" (AND-like operations) and "C-cells" (OR-like operations) with progressively larger receptive fields. These cells correspond to Hubel and Wiesel's simple and complex cells. Layers of S-cells extract spatially localized features from previous layers (i.e. they have topographically organized receptive fields) and these features become more complex and specific (i.e. the selectivity of S-cells increases along the hierarchy). Layers of C-cells pool over afferent S-cells with similar selectivity but adjacent positions hence building tolerance to position transformations.

In the last two decades, many feature-based models inspired by the Neocognitron have emerged, including a model by Perret and Oram showing that the principle be-

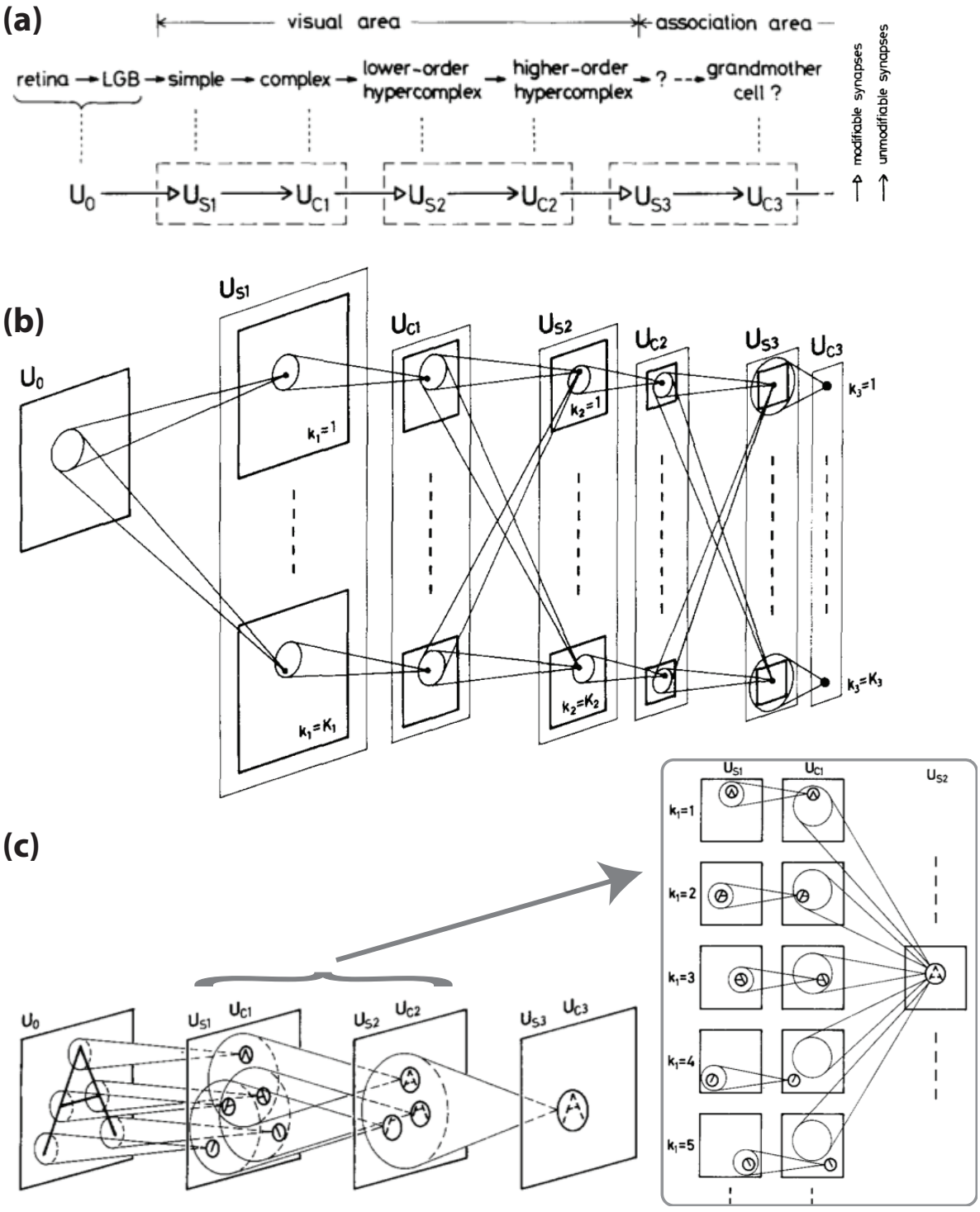



**Figure 2.5: Schematic diagrams of the Neocognitron.** (a) Correspondence between the simple/-complex model by Hubel and Wiesel [Hubel and Wiesel, 1962] and the neural network of the Neocognitron. (b) Schematic diagram illustrating the interconnections between layers in the Neocognitron. (c) An example of the interconnections between cells and the response of the cells after completion of self-organization. Figure and caption modified from [Fukushima, 1980].

hind C-cells' pooling could also be extended to scale invariance [Perrett and Oram, 1993; Oram and Perrett, 1994], the *VisNet* model (see Figure 2.6) which was based on some of the same ideas [Rolls *et al.*, 1992; Wallis *et al.*, 1993; Wallis and Rolls, 1997; Rolls and Milward, 2000; Stringer and Rolls, 2002; Elliffe *et al.*, 2002; Deco and Rolls, 2004], or *SEEMORE* [Mel, 1997]. In the meantime, the advancement of a more formal mathematical approach to learning in multi-layer architectures, called the *back-propagation* algorithm [LeCun, 1985; Parker, 1986; Rumelhart *et al.*, 1986; LeCun, 1988], led to the *Convolutional Neural Networks* (CNN) models developed by LeCun and colleagues [LeCun *et al.*, 1989, 1998; LeCun and Bengio, 1998; LeCun *et al.*, 2004; Osadchy *et al.*, 2004; Chopra *et al.*, 2005; Kavukcuoglu *et al.*, 2009; Jarrett *et al.*, 2009; Hadsell *et al.*, 2009; Boureau *et al.*, 2010a; LeCun *et al.*, 2010]. As indicated by their name, CNN models use convolutional layers with a bank of filters to achieve their selectivity and spatial subsampling layers to achieve their invariance to position (see Figure 2.7). Even though these layers are similar to the Neocognitron's S-cells and C-cells, these layers are usually denoted "C" (for convolution) and "S" (for subsampling), which can be confusing.

Another comprehensive and advanced line of research, also based on Hubel and Wiesel's (S)imple and (C)omplex cells and on the S/C cascade of Fukushima's Neocognitron, has been carried out by Poggio and colleagues with their *HMAX* models. Originally proposed in 1999 [Riesenhuber and Poggio, 1999b, 2000] and refined until today [Riesenhuber and Poggio, 2002b; Serre *et al.*, 2005b, 2007c,b; Mutch and Lowe, 2008], this class of models seeks close agreement with what is known about the anatomy and physiology of the ventral stream [Logothetis *et al.*, 1995; Cadieu *et al.*, 2007] as well as human behavioral performance during rapid visual categorization [Serre *et al.*, 2007a]. The goal of their approach is to start with simple ideas and intuitive computational principles and gradually build more elaborate models to finally establish a "Standard Model" [Riesenhuber and Poggio, 2002a], like in Physics. Figure 2.8 shows the most recent HMAX instantiation. Layers of S units perform a *Gaussian-like* tuning operation (sometimes implemented with a convolution) with a set of filters while layers of C units perform a *max-like* pooling operation across scale and space. In the HMAX models, the primary visual cortex (V1) is hard-coded at the bottom of the hierarchy

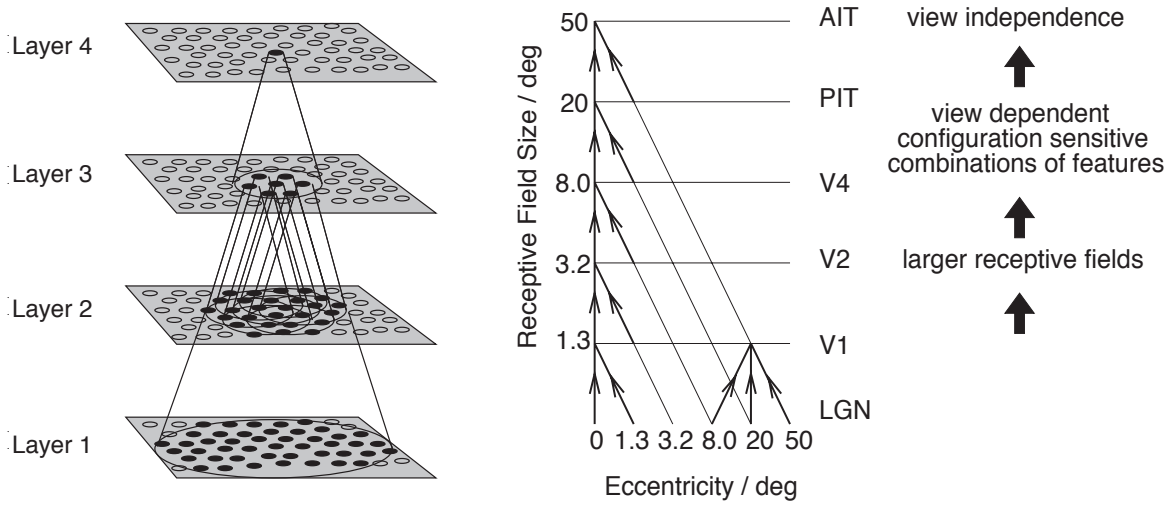



**Figure 2.6: Schematic diagram of VisNet.** (left) Stylized image of the VisNet four-layer network. Convergence through the network is designed to provide fourth-layer neurons with information from across the entire input retina. (right) Convergence in the visual system (adapted from [Rolls *et al.*, 1992]). V1: primary visual cortex area V1; PIT: posterior inferior temporal cortex; AIT: anterior inferior temporal cortex. Figure and caption modified from [Stringer and Rolls, 2002].

with a bank of two-dimensional Gabor bandpass filters [Gabor, 1946] with different orientations and spatial frequencies. As described previously, the receptive fields of both types of units become progressively larger throughout the hierarchy, and S units become selective for more and more complex shapes while C units exhibit higher degrees of tolerance to position and scale variations. Interestingly, it has been suggested that both Gaussian-like and max-like operations could be implemented with similar biophysically-plausible canonical neural circuits involving divisive normalization and polynomial nonlinearities with different parameters [Kouh and Poggio, 2008]. Moreover, a recent extension to HMAX has attempted to include "where" information by modeling the attentional processes happening in the brain [Chikkerur *et al.*, 2010]. A rigorous mathematical framework has also been developed to understand the tolerance and discrimination properties of HMAX models [Smale *et al.*, 2009; Bouvrie *et al.*, 2009; Wibisono *et al.*, 2010].

This review of feature-based models does not pretend to be complete, but it is worth mentioning that various researchers have developed similar hierarchical models,

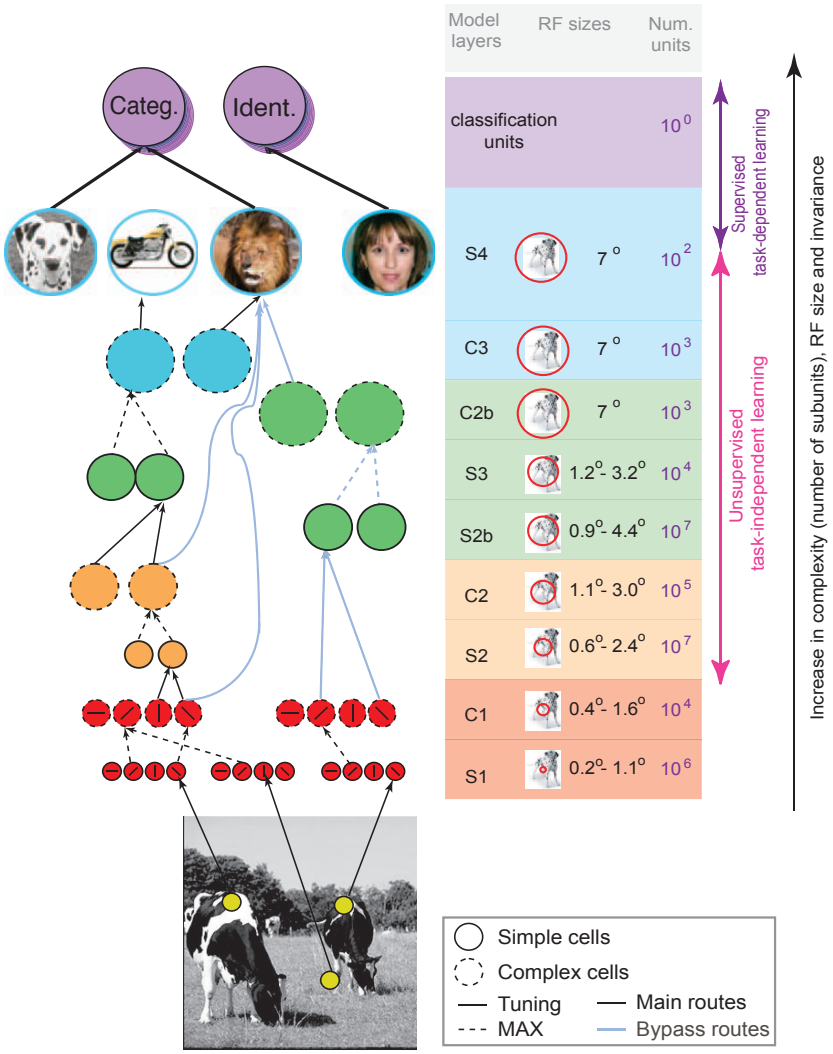



**Figure 2.8: Schematic diagram one of the most recent variant of HMAX.** This model is mostly feedforward, apart from local recurrent circuits. It attempts to describe the initial stage of visual processing and immediate recognition, corresponding to the output of the top of the hierarchy and to the first ∼150 ms in visual recognition. Colors encode the tentative correspondences between model layers and brain areas (see Figure 2.1). The model assumes that one of the main functions of the ventral stream is to achieve an optimal trade-off between selectivity and invariance. It is important to point out that the hierarchy is probably not as strict as depicted here. In addition there may be cells with relatively complex receptive fields already in V1. Stages of simple "S" units with Gaussian-like tuning (plain circles and arrows), which provide generalization [Poggio and Edelman, 1990; Poggio and Smale, 2003; Poggio and Bizzi, 2004], are interleaved with layers of complex "C" units (dashed circles and arrows), which perform MAX-like operation on their inputs and provide invariance to position and scale (pooling over scales is not shown here). Both operations may be performed by the same local recurrent circuits of lateral inhibition [Kouh and Poggio, 2008]. The major extension in this model relative to [Riesenhuber and Poggio, 1999b] is that unsupervised learning, on a set of natural images unrelated to the task, determines the tuning (e.g., the synaptic weights) of the simple units in the S2 and S3 layers (corresponding to V4 and PIT, respectively). Learning of the synaptic weights from S4 to the top classification units is the only task-dependent supervised learning stage in this architecture. The total number of units in the model is in the order of $10^7$. The table on the right provides a summary of the main properties of the units at the different levels of the model. Figure and caption modified from [Serre *et al.*, 2007a,b].

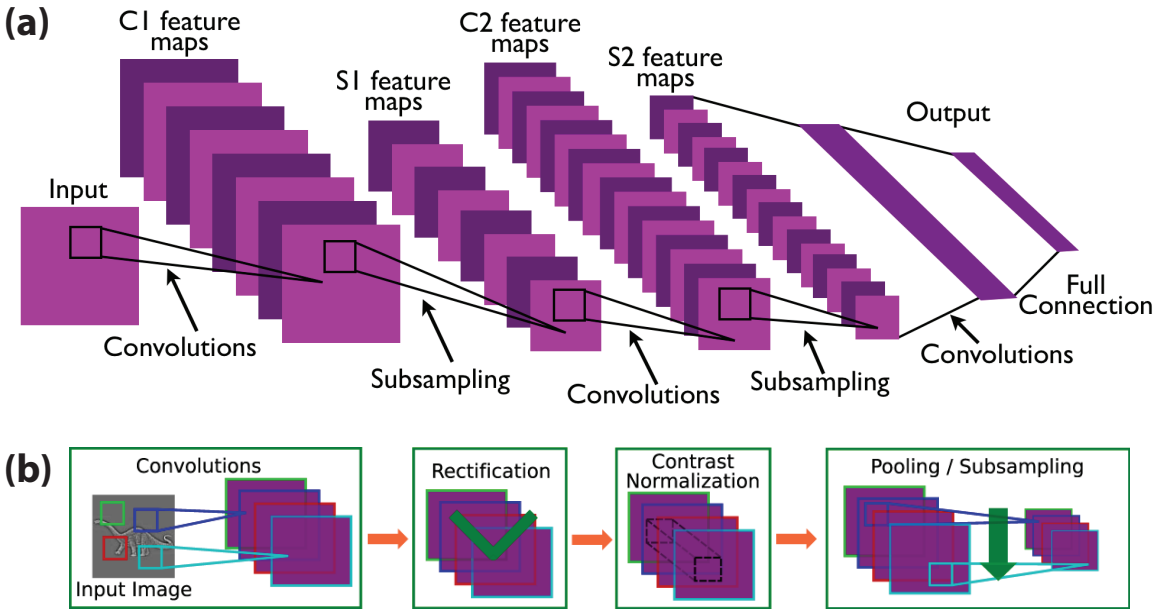



**Figure 2.7: Schematic diagram of Convolutional Neural Nets (CNNs).** (a) A typical CNN architecture with two feature stages. (b) Example of a recent extension of the feature extraction stage. An input image (or a feature map) is passed through a filter-bank, followed by a non-linear activation function, local subtractive and divisive contrast normalization (inspired by the Methods of Chapter 4), and spatial pooling/sub-sampling. Figure and caption modified from [LeCun *et al.*, 2010].

emphasizing different aspects and without necessarily seeking neuro-plausibility. For example: Thorpe et al.'s *SpikeNet* (focusing on neurons' spikes with temporal order coding instead of firing rate coding) [Thorpe and Gautrais, 1997; Van Rullen *et al.*, 1998; Gautrais and Thorpe, 1998; Thorpe *et al.*, 2001; Thorpe and Fabre-Thorpe, 2001; Thorpe, 2002; Masquelier and Thorpe, 2007]; Wersing, Körner et al.'s models [Wersing and Körner, 2003; Schneider *et al.*, 2005; Kirstein *et al.*, 2008, 2009]; Wyss, König and Verschure's models [Wyss *et al.*, 2003, 2006]; Bengio, Hinton, LeCun and Ng's *Deep Neural Networks* (such as *Belief Nets*, *Auto-Encoders*, *Restricted Boltzmann Machines* (RBMs), etc.) [Hinton, 2005; Hinton and Salakhutdinov, 2006; Hinton, 2007a; Bengio and LeCun, 2007; Bengio, 2009; Lee *et al.*, 2009; Raina *et al.*, 2009; Nair and Hinton, 2009]; or Dileep and Hawkins' *Hierarchical Temporal Memory* (HTM) [George, 2008; George and Hawkins, 2009].

#### Advantages

The S/C cascade implemented in many hierarchical feature-based models provides a gradual and parallel increase of selectivity and tolerance, which may be critical to avoid a combinatorial explosion in the number of neurons to represent many object views without being affected by the "binding problem" (i.e. the failure to discriminate among objects sharing the same features) [Malsburg, 1995; Treisman, 1996; Ullman and Soloviev, 1999; Riesenhuber and Poggio, 1999a; Mel and Fiser, 2000].

In addition, due to their "invariant" properties, these models can recognize objects without necessarily knowing their positions and sizes. As a consequence, they can be much faster than part-based models and more compatible with primates' behavioral performance on rapid categorization tasks.

#### Disadvantages

The strict separation and alternation between the selectivity-building (AND-like) operation (S) and the tolerance-building (OR-like) operation (C) is somewhat of an over-simplification of the trade-off. Increasing selectivity and *then* tolerance is not a joint optimization. Instead, the fact that these two operations could be implemented with the similar biophysically-plausible circuit with different parameters might suggest that there exists a *unique* canonical operation that can jointly balance selectivity and tolerance (and this could be achieved by learning the statistical regularities of the visual environment, see Section 2.2.4).

Another problem with most feature-based models is that information is lost as tolerance is gained. For example, max-like pooling throws away the exact position and size of the detected features. This shortcoming could however be overcome by modeling the dorsal "where" stream or by the inclusion of visual attentional processes (as in [Walther and Koch, 2007; Chikkerur *et al.*, 2010]).

### 2.3.3 Learning

"Arguably, the problem of learning represents a gateway to understanding intelligence in brains and machines, to discovering how the human brain works, and to making

intelligent machines that learn from experience and improve their competences as children do" [Poggio and Smale, 2003]. Simply put, learning in biologically-inspired models is "the process by which [a certain class of] free parameters [of the model] are adapted through a process of stimulation by the environment in which the [model] is embedded" [Haykin, 1994].

The most common class of adjustable parameters that are learned in these models are the synaptic weights of each simulated neuron (or population of neurons), and they are consequently in very large number. Depending on the values of these variables, the visual representation encoded at each layer will be different. There are many ways to learn, but we will focus on the learning of the statistical regularities present in the visual environment itself, an idea originally proposed by Barlow who formulated it as the "detection of associations, or covariation, or *suspicious coincidences*, among the inputs" (i.e. learn to anticipate these coincidences so that they are no longer non-accidental) [Barlow, 1961, 1985a, 1994]. This type of learning is generally performed *unsupervised* where the learning signal is implicit in the input data (also called "learning without a teacher", as opposed to "learning with a teacher" or *supervised* where the learning signal is explicitly given by the teacher, e.g. the labels associated with a given image).

Since Barlow's early insights, many ideas have been proposed to learn the high-order statistical regularities in the visual input and to constrain the format of the representation produced by each layer. We provide a brief and non-exhaustive overview below.

### Redundancy Reduction and Sparse Coding

Densely distributed or highly redundant representations make it difficult to determine when a coincidence is "suspicious" because combination of features will be represented by complex activity patterns over many detectors and detecting coincidences among these detectors requires keeping track of complex higher-order statistical dependencies. To increase the efficiency of pattern recognition for upstream layers and to facilitate the detection of non-accidental coincidences, it seems necessary to reduce redundancies in the representation.

A solution has been originally proposed by Barlow [Barlow, 1989, 1994] and it con-

sists in imposing a *sparse* constraint on the representation encoded by each layer (which means that only a few of the many elements in the representation will be active at any given time to encode a given stimulus). In addition to producing outputs with higher degree of statistical independence, a sparse representation has many other computational benefits compared to its dense counterpart including good signal-to-noise ratio, efficient separation between combinations of features, and higher storage capacity [Field, 1987, 1994].

There is physiological evidence suggesting that the visual cortex is using sparse codes [Barlow, 1972, 1985b] (even though the goal of the ventral stream does not seem to be one of obtaining sparser and sparser representations, as suggested by a recent study showing that V4 and IT populations have constant sparseness [Rust and DiCarlo, 2008]). Overcomplete sparse coding on natural images has been shown to produce the spatially localized, oriented bandpass receptive fields of simple cells [Olshausen *et al.*, 1996; Olshausen and Field, 1997; Simoncelli and Olshausen, 2001]. In addition, sparse coding also saves energy and "given the actual energy constraints of the mammalian cortex, sparse coding would seem to be a necessity" [Olshausen and Field, 2004].

Many methods (with a wide range of neuro-plausibility) have been proposed to learn sparse representations from natural images: local *anti-Hebbian* learning [Földiák, 1990]; *Infomax*, *Independent Component Analysis* (ICA) and extensions [Comon, 1994; Bell and Sejnowski, 1997; Te-Won, 1998; van Hateren and van der Schaaf, 1998; Hyvärinen and Oja, 2000; Hyvärinen and Hoyer, 2001; Hyvärinen *et al.*, 2001; Hoyer and Hyvärinen, 2002; Stone, 2004; Hyvärinen, 2010]; *Non-negative Matrix Factorization* [Lee and Seung, 1999; Eggert and Korner, 2004; Mairal *et al.*, 2010]; *Intrinsic Plasticity* / homeostasis [Triesch, 2007; Weber and Triesch, 2008; Savin *et al.*, 2010]; among others. As a consequence, many recent object recognition models use sparse coding and approximations at their core [Wersing and Körner, 2002; Hasler *et al.*, 2005; Marc-Aurelio Ranzato *et al.*, 2006; Marc Aurelio Ranzato *et al.*, 2007; Mairal *et al.*, 2008; Yang *et al.*, 2009; Boureau *et al.*, 2010b; Gregor and Lecun, 2010; Boureau *et al.*, 2010a]. Note that sparse coding is also related to *Compressed Sensing* [Donoho, 2006], a rapidly developing signal processing research area.

### Smoothness Assumption and Temporal Coherence

Another approach comes from considering object recognition as an inverse problem and constraining the hierarchical representations to gradually extract the hidden causes that produce the visual environment. Knowing critical properties of our physical world and how we observe it can help us to formulate intuitive assumptions. For instance, one can presume that important hidden causes for visual recognition are *smoothly varying* in time (i.e. two retinal projections following each other have a very high probability of describing similar visual scenes with the same objects but slightly different position, pose, etc.), even though the raw measurements acquired by the photoreceptors and encoded by the retinal ganglion cells vary rapidly in time (e.g. a small change in the position or pose of an object will produce a very different pattern of activation in the retina). It seems intuitive to exploit this temporal smoothness assumption as a surrogate for recovering the hidden causes by gradually extracting smoothly varying representations from the rapidly varying visual inputs.

Learning criterions relying on temporal smoothness have been originally introduced in the eighties, including by Sutton and Barto [1981]; Sutton [1988] (for associative learning / classical conditioning), by Perrett *et al.* [1984, 1985] (for invariance to pose), or by Hinton [1989]. Early biologically-inspired vision models applying this principle and showing its usefulness were subsequently implemented by Mitchison [1991] and Földiak [1991]. To learn tolerance to position variations, Földiak proposed an algorithm based on local competition, short-term memory, and most importantly, a *trace* learning rule. This rule is essentially Hebbian learning on *low-pass filtered* pre- or post-synaptic signals, effectively minimizing the variance of the signals' time derivatives. In addition to translation and pose [Földiak, 1991; Einhauser *et al.*, 2002; Einhäuser *et al.*, 2005; Spratling, 2005], similar algorithms can be used to learn other "invariances", such as deformation [De Sa and Ballard, 1998], or viewpoint and depth [Becker and Hinton, 1992; Becker, 1993; Stone, 1996]. Many variants of the trace learning rule exist and some were also implemented in full hierarchical models like *VisNet* [Wallis *et al.*, 1993; Wallis and Rolls, 1997; Rolls and Milward, 2000; Stringer and Rolls, 2002; Elliffe *et al.*, 2002; Deco and Rolls, 2004].

Another important instantiation of this idea is the *Slow Feature Analysis* (SFA), independently developed by Wiskott and Sejnowski [Wiskott, 1998; Wiskott and Sejnowski, 2002; Wiskott, 2003]. What differentiates SFA from earlier approaches is its closed-form derivation that relies on batch processing instead of online gradient-descent-based approximations. Consequently, the original SFA algorithm can not be applied incrementally, but it will find globally optimal solutions instead of being trapped in local optima. Interestingly, SFA has also been formally connected to other algorithms, including Földiak's trace learning (i.e. under some mathematical assumptions, SFA can actually be approximated by an online trace learning rule, see [Sprekeler *et al.*, 2007]), sparse coding with ICA [Blaschke *et al.*, 2006], or predictive coding [Shaw, 2005; Creutzig and Sprekeler, 2008]. Probabilistic interpretations within a Bayesian framework have also been established and extended to deal with noise and missing data. For example, Turner and Sahani [2007] showed that SFA can be seen as "maximum-likelihood learning in a linear Gaussian state-space model, with an independent Markovian prior".

Temporal coherence and predictive coding have also been used in HTM models within a Bayesian belief propagation framework [George, 2008], and they have been connected to mathematical models for cortical circuits (e.g. an "HTM node is abstracted using a *coincidence* detector and a mixture of Markov chains", see [George and Hawkins, 2009] for details).

There is ample support in the neuroscience literature for learning from temporal coherence in the visual cortex (see e.g. [Cox *et al.*, 2005; Li and DiCarlo, 2008, 2010]), and many computational studies have revealed that learning from natural image sequences under temporal smoothness assumptions leads to the emergence of simple, complex *and* hypercomplex cell receptive fields (e.g. [Kayser *et al.*, 2001; Einhauser *et al.*, 2002; Hurri and Hyvärinen, 2003a,b; Kording *et al.*, 2004; Einhäuser *et al.*, 2005; Berkes and Wiskott, 2005; Masquelier *et al.*, 2007]), effectively giving this principle more computational neuroscience support than sparse coding.

Finally, it is also possible to imagine that sparse representations in lower layers of the hierarchy could arise as a by-product of temporal coherence optimization This is strongly suggested by the fact that in going from V4 to IT, generalization to natural stimuli increases [Rust and DiCarlo, 2010] but sparseness does not [Rust and DiCarlo,

2008].

### Notes on Information Preservation

To avoid trivial solutions to sparseness or smoothness optimizations (e.g. producing random binary patterns or constant signals that have nothing to do with the inputs), it is important to impose some sort of restriction (explicitly or not) on the information that will get lost in the process. This type of constraint, in combination with others, will also implicitly discover and represent complex statistical regularities in the visual inputs.

Virtually all of the methods described above use some sort of information preservation. Other related approaches use auto-encoders / auto-associators, or generative models (see e.g. [Yuille and Kersten, 2006; Petrovic *et al.*, 2006; Hinton and Salakhutdinov, 2006; Hinton *et al.*, 2006; Hinton, 2007b; Bengio *et al.*, 2007; Bengio and LeCun, 2007; Vincent *et al.*, 2008; Bengio, 2009; Hinton, 2010]).

### Notes on Learning with Weight Sharing

Recent hierarchical models that use convolution-like operation to build up their selectivity (e.g. Poggio et al.'s HMAX or LeCun et al.'s CNN) effectively use filters (i.e. local receptive fields) and a concept known as "weight-sharing" where each neuron and its synaptic connectivity gets duplicated at different locations or scales[3] within a given layer, and thus perform the same operation but on different parts of the input. As a result, weight-sharing allows tolerance to position, scale and orientation *by construction* and not by learning statistical regularities.

This structuring of domain knowledge about "invariant" object recognition has the benefit of greatly reducing the number of adjustable parameters (synaptic weights, i.e. filter kernels) and thus the number of examples required to learn them (this concept is related to the VC-dimension of the learning algorithm, see [Vapnik and Chervonenkis, 1971]).

[3]Note that weight-sharing can also be applied to in-plane rotation, symmetry, etc.

#### Notes on Learning with Gaussian-like Tuning

In addition to relying on neuro-plausible mechanisms [Logothetis *et al.*, 1995; Kouh and Poggio, 2008], models that use Gaussian-like tuning operations have been shown to learn efficiently and generalize well to new stimuli (e.g. *Radial Basis Function Networks* [Poggio and Edelman, 1990; Poggio and Girosi, 1990; Bishop, 1995; Poggio and Smale, 2003; Poggio and Bizzi, 2004]).

During learning, these models effectively combine training examples to form prototypes (equivalent of neurons' receptive fields or convolution filters), and to generalize to new examples, they simply interpolate among the learned prototypes. Note that tuning does not have to be exactly Gaussian and conceptually simpler tuning operations have been shown to work as effectively (e.g. normalized-dot products with sigmoidal outputs, see Appendix A.4 in [Serre *et al.*, 2005a] for details).

### 2.3.4 Performance on Standard Computer Vision Tasks

To date, only relatively modest progress has been made towards building artificial systems that approach the abilities of biological visual systems under real-world conditions. Efforts to build algorithms capable of object and face recognition from both the computer vision and the neuroscience communities have produced performance improvements on *some* restricted types of visual tasks, but performance is still arguably far below that of the human brain and other biological systems.

While very few visual recognition models from neuroscience have generated "game-changing" predictions, they have nevertheless shown that they can "fit" some piece of known anatomy and/or physiology. One particularly common example is to demonstrate that a model can produce responses similar to simple cells and/or complex cells. Many experiments also continue to rely on small-scale, constrained recognition tasks.

Only a small group of models inspired by the brain have shown competitive results on the type of benchmarks used by the computer vision community (which is arguably more interested in overall recognition performance than any particular relationship with biology). In particular, some biologically-inspired models have been shown to perform well on digits classification [LeCun *et al.*, 1989, 1998], face detection [Wiskott *et al.*,

1997; Serre *et al.*, 2002; Osadchy *et al.*, 2004], face identification [Chopra *et al.*, 2005], object classification [LeCun *et al.*, 2004; Serre *et al.*, 2007c; Mutch and Lowe, 2008; Kavukcuoglu *et al.*, 2009; Jarrett *et al.*, 2009; Boureau *et al.*, 2010a], behavior classification [Jhuang *et al.*, 2007], or autonomous driving [Hadsell *et al.*, 2009] among others. However, specialized methods from computer vision have historically outperformed more biologically-inspired models – for example in face verification [Wolf *et al.*, 2008, 2009; Taigman *et al.*, 2009; Kumar *et al.*, 2009; Cao *et al.*, 2010], object detection [Felzenszwalb *et al.*, 2010b,a], object classification [Zhang *et al.*, 2006; Varma and Ray, 2007; Lazebnik *et al.*, 2009; Gehler and Nowozin, 2009; Van De Sande *et al.*, 2010], etc.

### 2.3.5 Comparisons with Standard Computer Vision

#### Standard Computer Vision: Differences from Biology

Undoubtedly, standard computer vision models exploit very different mechanisms than those observed in the brain to achieve the reported high levels of performance. One particularly insightful observation is that they tend to rely heavily on high resolution data to perform well, in contrast to humans that can correctly recognize faces [Sinha *et al.*, 2007] or objects [Torralba *et al.*, 2008] at very low resolutions, even when the objects in isolation can not be recognized.

#### Standard Computer Vision: Similarities with Biologically-Inspired Models

Most of the standard object recognition models from computer vision use SIFT-like features [Lowe, 2004] at their lowest level. Even though SIFT was originally proposed as a well-engineered "computational model for object recognition in IT cortex" [Lowe, 1999, 2000], it included many hardcoded components, based on mathematical and algorithmic tricks not easily accessible to biological circuits, such as scale-invariant interest point detection or complete rotation invariance. These specializations were subsequently discarded to provide better *scalability* while improving *generalization* and suitability for later stages of processing (e.g. bag-of-words approaches or their spatial pyramid extensions, see below). Specifically, SIFT-like descriptors are now applied densely and their rotation is not normalized anymore, thus avoiding the loss of potentially useful

information and the trap of gaining too much tolerance too quickly at the expense of selectivity to increasingly larger number of objects and categories. Furthermore, many variants or simplifications to SIFT features, such as *Histograms of Gradients* (HOG) [Dalal and Triggs, 2005], have been proposed and are now used extensively in recent successful efforts (e.g. [Gehler and Nowozin, 2009; Felzenszwalb *et al.*, 2010b]). They are generally both faster to compute and more compact.

Operationally, SIFT-like features (1) capture local orientations by extracting image gradient directions, and (2) pool over spatial regions to build a compact histogram. Here we see an intuitive relationship with the Hubel and Wiesel's "S/C cascade" of feature-based biologically-inspired models described in Section : (1) build up *selectivity* and tuning to orientated edges (like simple cells or the first "S" layer of Neocognitron-like models), and (2) build up *tolerance* to position variation by pooling over spatially arranged afferents (like complex cells or the first "C" layer of Neocognitron-like models). Interestingly, this local orientation tuning and spatial pooling can be obtained through unsupervised learning of spatio-temporal statistics, possibly with the same canonical circuit, as we have discussed in Section 2.3.3.

The next layers of processing in most of the standard computer vision models mentioned above, such as bag-of-word approaches and extensions [Vogel and Schiele, 2004; Sivic *et al.*, 2005; Fei-Fei and Perona, 2005; Bosch *et al.*, 2007; Grauman and Darrell, 2006; Lazebnik *et al.*, 2009; Van De Sande *et al.*, 2010], include some sort of density estimation / vector quantization step like K-means clustering and another histogramming step. This is striking because it looks exactly like another layer of selectivity/tolerance or "S/C cascade" with some stimulus-driven unsupervised learning stage (K-means-like) that could be be implemented with simple Hebbian-like online learning [McQueen, 1967; Bottou and Bengio, 1995] (e.g. Winner-Take-All) to build up selectivity to specific redundant inputs, and spatial pyramid pooling to build up invariance to position and scale. For example, Serre *et al.* [2007c]'s HMAX model include an input "imprinting" stage for learning S2 features, and this can be seen as the first iteration of a specific online K-means with a high-learning rate (i.e. equal to 1). CNNs, RBMs and Auto-encoders also have related mechanisms to learn selective/generative feature spaces layer-wise [Hinton and Salakhutdinov, 2006; Bengio and LeCun, 2007; Vincent

*et al.*, 2008; Kavukcuoglu *et al.*, 2009; Lee *et al.*, 2009].

Other interesting relationships, similarities and convergence of ideas can be made and mapped into a conceptually simpler computational framework with a biologically-inspired canonical circuit and learning rule at its core. However, the details lie outside the scope of this thesis and will be subject to a more comprehensive review paper in the future.

Chapter 3

# Moving Forward

"Tryin' to paint a perfect picture and excel. In case you didn't know. Never movin' backwards. Complicated. Know what I mean?"

*Rakim / DJ Premier (1999)*

In this chapter, we identify operational barriers that have obstructed progress towards finding a solution to the object recognition problem, and we provide an overview of the thesis research program designed to break down these barriers.

## 3.1 Major Challenges

### 3.1.1 The Lack of Clear and Measurable Indicators of Progress

To support efforts in developing biologically-inspired visual models and guide future progress in the right direction, one fundamental problem that must be addressed is the development of clear and measurable indicators of performance on object recognition benchmarks. In practical terms, this boils down to defining the task that must be solved, for example: the set of (labeled) visual data (e.g. images, videos) that must be correctly categorized.

Historically, a wide range of benchmarks have been used to evaluate the progress of visual recognition algorithms, which has made it difficult to compare results across approaches from different research groups. A comparative approach in which success is objectively judged on a level playing field is required to accurately evaluate how much improvement has been made and to disentangle the best conceptual ideas and insights from various groups.

A recent popular, yet controversial approach [Felsen and Dan, 2005; Rust and Movshon, 2005] is the use of large databases of "natural" images both in the study of biological vision (both theoretical e.g. [Masquelier and Thorpe, 2007] and experimental e.g. [Gallant *et al.*, 1998; Reinagel, 2001; Einhäuser *et al.*, 2007]) and artificial vision (e.g. [Fei-Fei *et al.*, 2004a; Griffin *et al.*, 2007; Huang *et al.*, 2007; Mutch and Lowe, 2008; Deng *et al.*, 2009; Jarrett *et al.*, 2009; Felzenszwalb *et al.*, 2010a; Everingham *et al.*, 2010] among many others), in part because they ostensibly capture the essence of problems encountered in the "real" world. The logic behind these "natural" sets is that the sheer number of categories and the apparent diversity of those images place a high bar for object recognition systems and require them to solve the computational crux of object recognition.

However, many of these standardized tests may have not been given much thought since the images are usually loosely collected from the web without any guarantee of capturing the core computational problem of object recognition: tolerance to identity-preserving image transformations. Performance on these benchmarks are indeed not predictive of the performance across a large variety of tasks. There are clear signs of over-fitting in the literature: no wide generalization is shown, and models are usually "tuned" to benchmarks that are usually "cherry picked" to show high performance with different metrics of success. In addition, these tests are not computationally efficient to run due to the large number of images they contain.

In sum, we lack *good* and *efficient* operational definitions of the problem we aim to solve.

### 3.1.2 The Hypothesis Space is Largely Unexplored

As we have described in Section 2.2, neuroscience research has gradually provided a better qualitative understanding of the neural architecture and computational principles that biological vision systems rely upon to process visual information and accomplish recognition. When these basic insights from neuroscience were recently instantiated in computational models, surprising levels of performance were obtained that rivaled performance of state-of-the-art artificial systems (see Section 2.3.4) and even matched human performance under extremely limited conditions [Serre *et al.*, 2007a]. These models are still far below general human recognition capabilities, but they are only individual instances drawn from a very large class of biologically-inspired models and thus represent a lower bound on how far the existing neuroscience principles can take us.

In particular, the hypothesis space of possible computational vision models is staggeringly large and even the restricted class of biologically-inspired models has dozens to hundreds of explicit and implicit free parameters, including filter kernel sizes, normalization neighborhoods and exponents, learning rule parameters, etc. Unfortunately, systems neuroscience data currently provide very few constraints on the values those parameters can take.

The size of the hypothesis space is further compounded by the extreme computational expense of instantiating models that approach realistic scales. Primate visual systems are composed of billions of neurons and trillions of synapses, thus attempting to simulate a system even a fraction of that size is computationally daunting. To effectively evaluate the performance of a machine vision system, one must evaluate thousands of images, and potentially larger amounts of training images (e.g. video), which further increases the computational cost. As a result, the region of hypothesis space that has been explored is biased towards smaller, more tractable models, and biologically-inspired computational ideas tend to only get tested at a very small scale.

As a consequence, the space of possible models has gone largely unexplored. When current instantiations fail to approach the performance of biological visual systems

in unfettered conditions[1], we are left uncertain whether this failure is because the underlying computational principles are wrong, or some fundamental element is missing from the model class[2], or because the correct parts have not been tuned correctly, assembled at sufficient scale or provided with sufficient experience.

## 3.2 Scope and Thesis Outline

In the remaining of this thesis we will attempt to address these challenges as follows:

- In Part II, we provide new neuroscience-motivated baselines and new fully-controlled benchmarks for object recognition (Chapter 4) and face recognition (Chapters 5 and 6). We also compare and contrast a variety of state-of-the-art recognition systems (Chapter 7) as well as human observers (Chapter 8) on the same benchmarks.

- In Part III, we propose a high-throughput screening approach inspired by molecular biology and genetics to (1) explore the large hypothesis space of possible feature-based visual recognition models inspired by the brain, (2) discover promising instantiations, and (3) systematically study them on various tasks (Chapter 9). We also provide details regarding some of the engineering and programming techniques we use to leverage massively parallel computing resources required to apply this approach at low time and cost while allowing the experimenter/developer to keep high degrees of flexibility and adaptability necessary in research environments (Chapter 10).

- In Part IV, we validate the scalability and applicability of our simple approach to large-scale "real-world" applications (e.g. face recognition in social networks or "in the wild") without loss of generality (Chapters 11 and 12), and by effectively solving the problem of interest defined in Part II (Chapter 13).

[1] For example, [Serre *et al.*, 2007a] could only approach human performance under unnaturally short image presentation times with uncontrolled natural image sets.

[2] For example, sophisticated feedback [Walther and Koch, 2007; Chikkerur *et al.*, 2010], spiking neurons [Thorpe and Gautrais, 1997; Van Rullen *et al.*, 1998; Gautrais and Thorpe, 1998; Thorpe *et al.*, 2001; Thorpe and Fabre-Thorpe, 2001; Thorpe, 2002; Masquelier and Thorpe, 2007], etc.

# Part II

# Simple Baselines and Efficient Benchmarks

Chapter 4

# Why is Real-World Visual Object Recognition Hard?*

"We will know what is the problem we are trying to solve once we solve it."

*Jitendra Malik, MIT BCS Colloquium (2010)*

Progress in understanding the brain mechanisms underlying vision requires the construction of computational models that not only emulate the brain's anatomy and physiology, but ultimately match its performance on visual tasks. In recent years, "natural" images have become popular in the study of vision and have been used to show apparently impressive progress in building such models. Here, we challenge the use of uncontrolled "natural" images in guiding that progress. In particular, we show that a simple V1-like model – a neuroscientist's "null" model, which should perform poorly at real-world visual object recognition tasks – outperforms state-of-the-art object recognition systems (biologically inspired and otherwise) on a standard, ostensibly natural image recognition test. As a counterpoint, we designed a "simpler" recognition test to better span the real-world variation in object pose, position, and

*This chapter is modified from a study published in the open-access journal PLoS Computational Biology in collaboration with David D. Cox and James J. DiCarlo [Pinto et al., 2008b],

scale, and we show that this test correctly exposes the inadequacy of the V1-like model. Taken together, these results demonstrate that tests based on uncontrolled natural images can be seriously misleading, potentially guiding progress in the wrong direction. Instead, we reexamine what it means for images to be natural and argue for a renewed focus on the core problem of object recognition – real-world image variation.

## 4.1 Introduction

Visual object recognition is an extremely difficult computational problem. The core problem is that each object in the world can cast an infinite number of different 2-D images onto the retina as the object's position, pose, lighting, and background vary relative to the viewer (e.g., [DiCarlo and Cox, 2007]). Yet the brain solves this problem effortlessly. Progress in understanding the brain's solution to object recognition requires the construction of artificial recognition systems that ultimately aim to emulate our own visual abilities, often with biological inspiration (e.g., [Weber *et al.*, 2000; Arathorn, 2002; Lowe, 2004; Serre *et al.*, 2007c; Zhang *et al.*, 2006]). Such computational approaches are critically important because they can provide experimentally testable hypotheses, and because instantiation of a working recognition system represents a particularly effective measure of success in understanding object recognition. However, a major challenge is assessing the recognition performance of such models. Ideally, artificial systems should be able to do what our own visual systems can, but it is unclear how to evaluate progress toward this goal. In practice, this amounts to choosing an image set against which to test performance.

Although controversial ([Felsen and Dan, 2005; Rust and Movshon, 2005]), a popular recent approach in the study of vision is the use of "natural" images [Bell and Sejnowski, 1997; Gallant *et al.*, 1998; Reinagel, 2001; Simoncelli and Olshausen, 2001; Felsen and Dan, 2005], in part because they ostensibly capture the essence of problems encountered in the real world. For example, in computational vision, the Caltech101 image set has emerged as a gold standard for testing "natural" object recognition performance [Fei-Fei *et al.*, 2004a]. The set consists of a large number of images divided into 101 object categories (e.g., images containing planes, cars, faces, flamingos, etc.; see Figure

4.1A) plus an additional "background" category (for 102 categories total). While a number of specific concerns have been raised with this set (see [Ponce *et al.*, 2006] for more details), its images are still currently widely used by neuroscientists, both in theoretical (e.g., [Serre *et al.*, 2007c; Masquelier and Thorpe, 2007]) and experimental (e.g., [Einhäuser *et al.*, 2007]) contexts. The logic of Caltech101 (and sets like it; e.g., Caltech256 [Griffin *et al.*, 2007]) is that the sheer number of categories and the diversity of those images place a high bar for object recognition systems and require them to solve the computational crux of object recognition. Because there are 102 object categories, chance performance is less than 1% correct. In recent years, several object recognition models (including biologically inspired approaches) have shown what appears to be impressively high performance on this test – better than 60% correct [Zhang *et al.*, 2006; Wang *et al.*, 2006; Mutch and Lowe, 2006; Lazebnik *et al.*, 2006; Grauman and Darrell, 2006], suggesting that these approaches, while still well below human performance, are at least heading in the right direction.

However, we argue here for caution, as it is not clear to what extent such "natural" image tests actually engage the core problem of object recognition. Specifically, while the Caltech101 set certainly contains a large number of images (9,144 images), variations in object view, position, size, etc., between and within object category are poorly defined and are not varied systematically. Furthermore, image backgrounds strongly covary with object category (see Figure 4.1B). The majority of images are also "composed" photographs, in that a human decided how the shot should be framed, and thus the placement of objects within the image is not random and the set may not properly reflect the variation found in the real world. Furthermore, if the Caltech101 object recognition task is hard, it is not easy to know what makes it hard – different kinds of variation (view, lighting, exemplar, etc.) are all inextricably mixed together. Such problems are not unique to the Caltech101 set, but also apply to other uncontrolled "natural" image sets (e.g., Pascal VOC [Everingham *et al.*, 2010]).

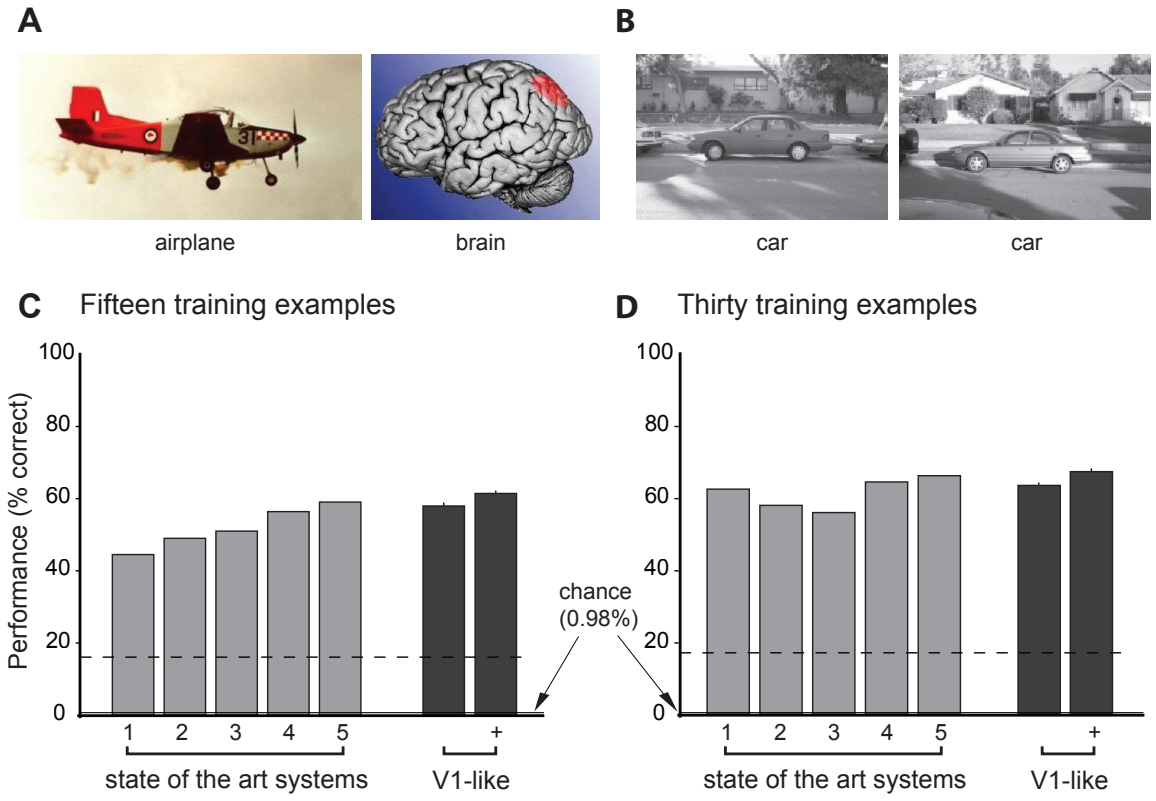



**Figure 4.1: Performance of a Simple V1-like Model Relative to Current Performance of State-of-the-Art Artificial Object Recognition Systems (Some Biologically Inspired) on an Ostensibly "Natural" Standard Image Database (Caltech101).** (A) Example images from the database and their category labels. (B) Two example images from the "car" category. (C) Reported performance of five state-of-the-art computational object recognition systems on this "natural" database are shown in gray (1=[Wang *et al.*, 2006]; 2=[Grauman and Darrell, 2006]; 3=[Mutch and Lowe, 2006]; 4=[Lazebnik *et al.*, 2006]; 5=[Zhang *et al.*, 2006]). In this panel, 15 training examples were used to train each system. Since chance performance on this 102-category task is less than 1%, performance values greater than 40% have been taken as substantial progress. The performance of the simple V1-like model is shown in black (+ is with "ad hoc" features; see Methods Section 4.4). Although the V1-like model is extremely simple and lacks any explicit invariance-building mechanisms, it performs as well as, or better than, state-of-the-art object recognition systems on the "natural" databases. (D) Same as (C) with 30 training examples. The dashed lines indicates performance achieved using an untransformed grayscale pixel space representation and a linear SVM classifier (15 training examples: 16.1%, SD 0.4; 30 training examples: 17.3%, SD 0.8). Error bars (barely visible) represent the standard deviation of the mean performance of the V1-like model over ten random training and testing splits of the images. The authors of the state-of-the-art approaches do not consistently report this variation, but when they do they are in the same range (less than 1%). The V1-model also performed favorably with fewer training examples (see Figure 4.5).

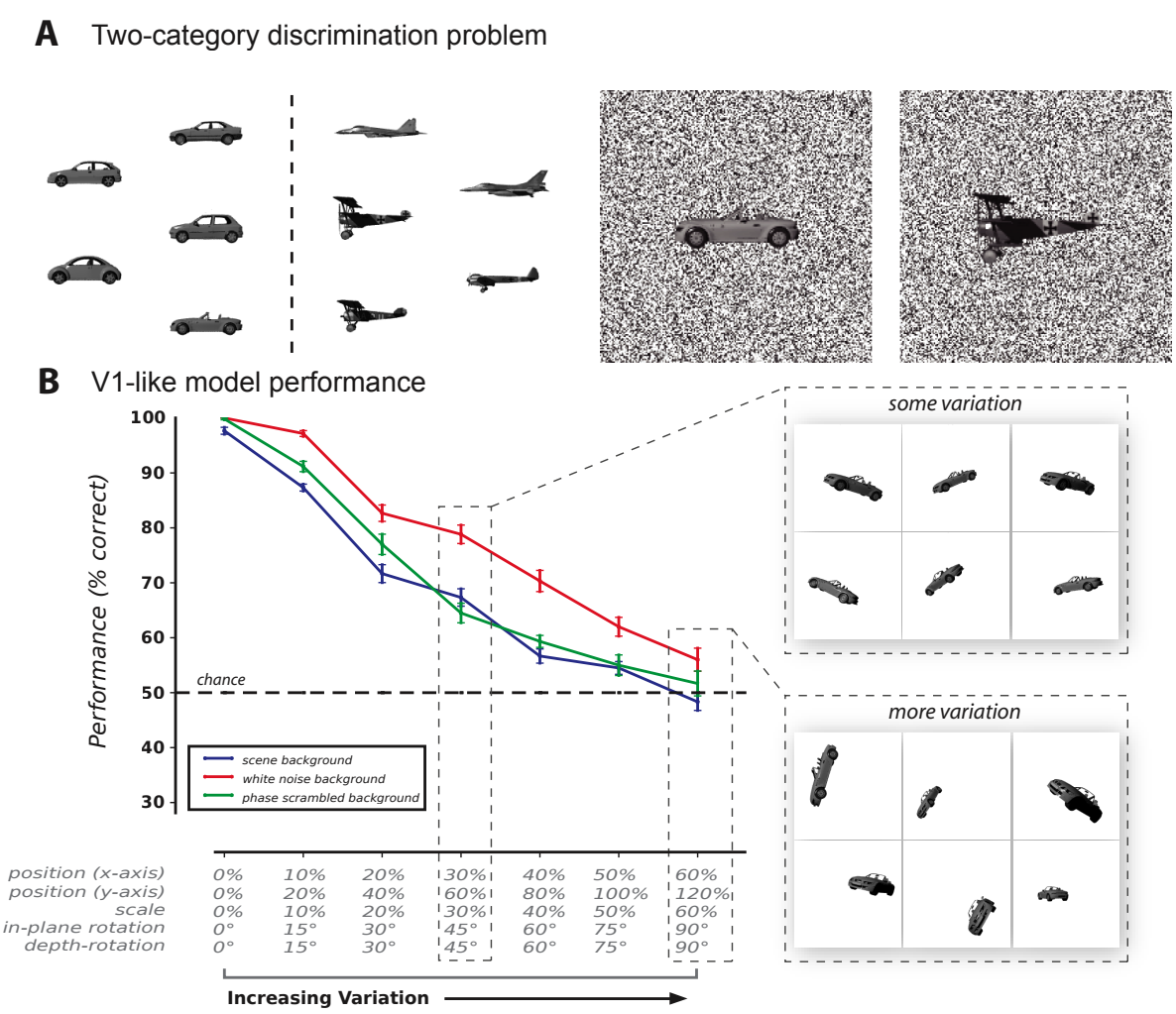



**Figure 4.2: The Same Simple V1-like Model That Performed Well in Figure 4.1 Is Not a Good Model of Object Recognition – It Fails Badly on a "Simple" Problem That Explicitly Requires Tolerance to Image Variation.** (A) We used 3-D models of cars and planes to generate image sets for performing a cars-versus-planes two-category test. By using 3-D models, we were able to parametrically control the amount of identity-preserving variation that the system was required to tolerate to perform the task (i.e., variation in each object's position, scale, pose). The 3-D models were rendered using ray-tracing software (see Methods Section 4.4), and were placed on either a white noise background (shown here), a scene background, or a phase scrambled background (these backgrounds are themselves another form of variation that a recognition system must tolerate; see Figure 4.3). (B) As the amount of variation was increased (x-axis), performance drops off, eventually reaching chance level (50%). Here, we used 100 training and 30 testing images for each object category. However, using substantially more exemplar images (1,530 training, 1,530 testing) yielded only mild performance gains (e.g., 2.7% for the fourth variation level using white noise background), indicating that the failure of this model is not due to under-training. Error bars represent the standard error of the mean computed over ten random splits of training and testing images (see Methods Section 4.4). This result highlights a fundamental problem in the current use of "natural" images to test object recognition models. By the logic of the "natural" Caltech101 test set, this task should be easy, because it has just two object categories, while the Caltech101 test should be hard (102 object categories). However, this V1-like model fails badly with this "easy" set, in spite of high performance on the Caltech101 test set (Figure 4.1).

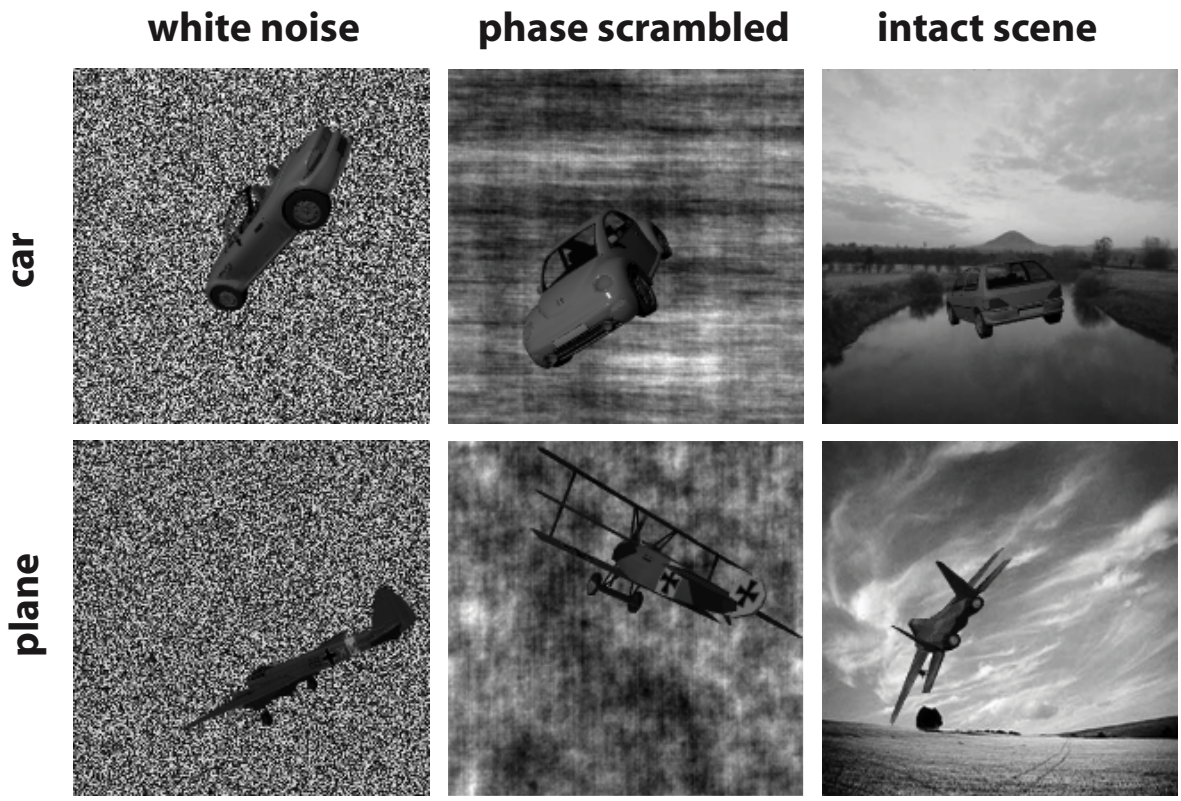



**Figure 4.3: Backgrounds Used.** Model performance for our "simple" two class image set was assessed with the 3-D models rendered onto three types of backgrounds – white noise, phase-scrambled scene images ($\sim \frac{1}{f}$ noise), and intact scene images. Performance with each of these types of background is shown in Figure 4.2.

## 4.2 Results

To explore this issue, we used the simplest, most obvious starting point for a biologically inspired object recognition system – a "V1-like" model based roughly on the known properties of simple cells of the first primate cortical visual processing stage (area V1). In particular, the model was a population of locally normalized, thresholded Gabor functions spanning a range of orientations and spatial frequencies (see Methods Section 4.4 for details). This is a neuroscience "null" model because it is only a first-order description of the early visual system, and one would not expect it to be good for real-world object recognition tasks. Specifically, it contains no explicit mechanisms to enable recognition to tolerate variation in object position, size, or pose, nor does it

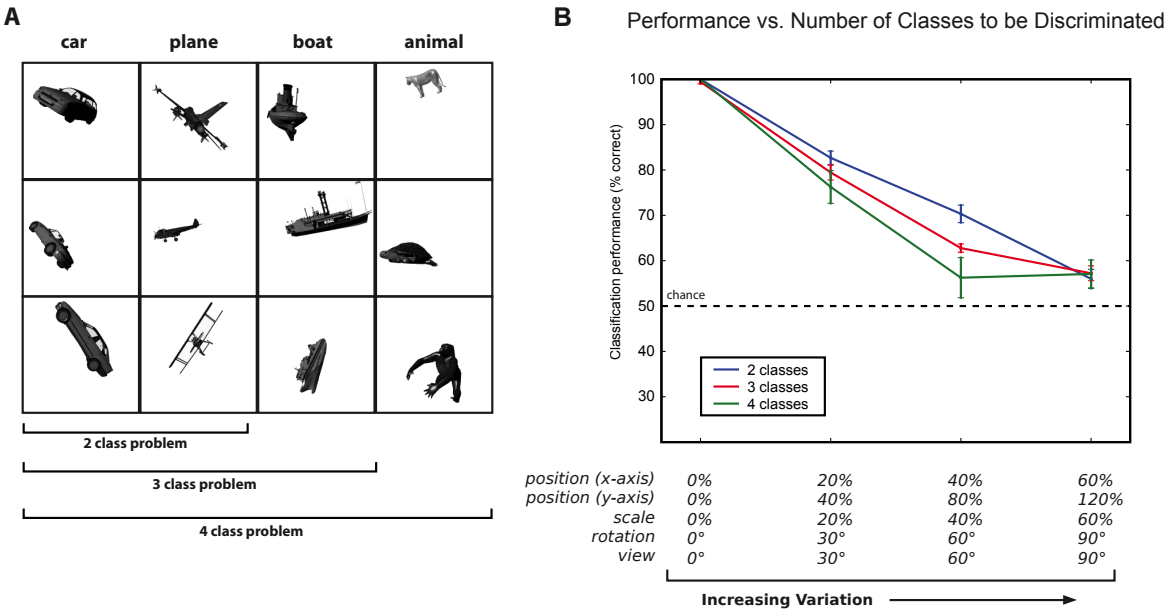



**Figure 4.4: Performance Fall-Off for Increasing Numbers of Object Categories.** Figure 4.2 shows that relatively modest amounts of image transformation push the performance of our simple V1-like model down to chance. Here we show that this fall-off becomes slightly steeper as more categories-to-be-discriminated are added. (A) Four categories of objects (cars, planes, boats, and animals) were used to measure performance when 2, 3, or 4 categories are considered. (B) Average identification performance ("is object category X present or not") is plotted as a function of view variation and number of object categories to be discriminated. Chance performance is 50% for all three lines, because average one-versus-all performance is shown here, not n-way recognition performance (i.e., "which object is present").

contain a particularly sophisticated representation of shape. Nevertheless, null models are useful for establishing baselines, and we proceeded to test this null model on a gold-standard "natural' object recognition task (i.e., Caltech101 [Fei-Fei *et al.*, 2004a]), using standard, published procedures [Grauman and Darrell, 2006]).

We found that this simple V1-like model performed remarkably well on the Caltech101 object recognition task – indeed, it outperformed reported state-of-the-art computational efforts (biologically inspired or not). Figure 4.1 shows the cross-validated performance of two versions of this simple model: one where only the model's outputs are fed into a standard linear classifier, and one where some additional ad-hoc features are also used (e.g., local feature intensity histograms; see Methods Section 4.4 for details). In both cases, performance is surprisingly good (61% and 67% correct with 15

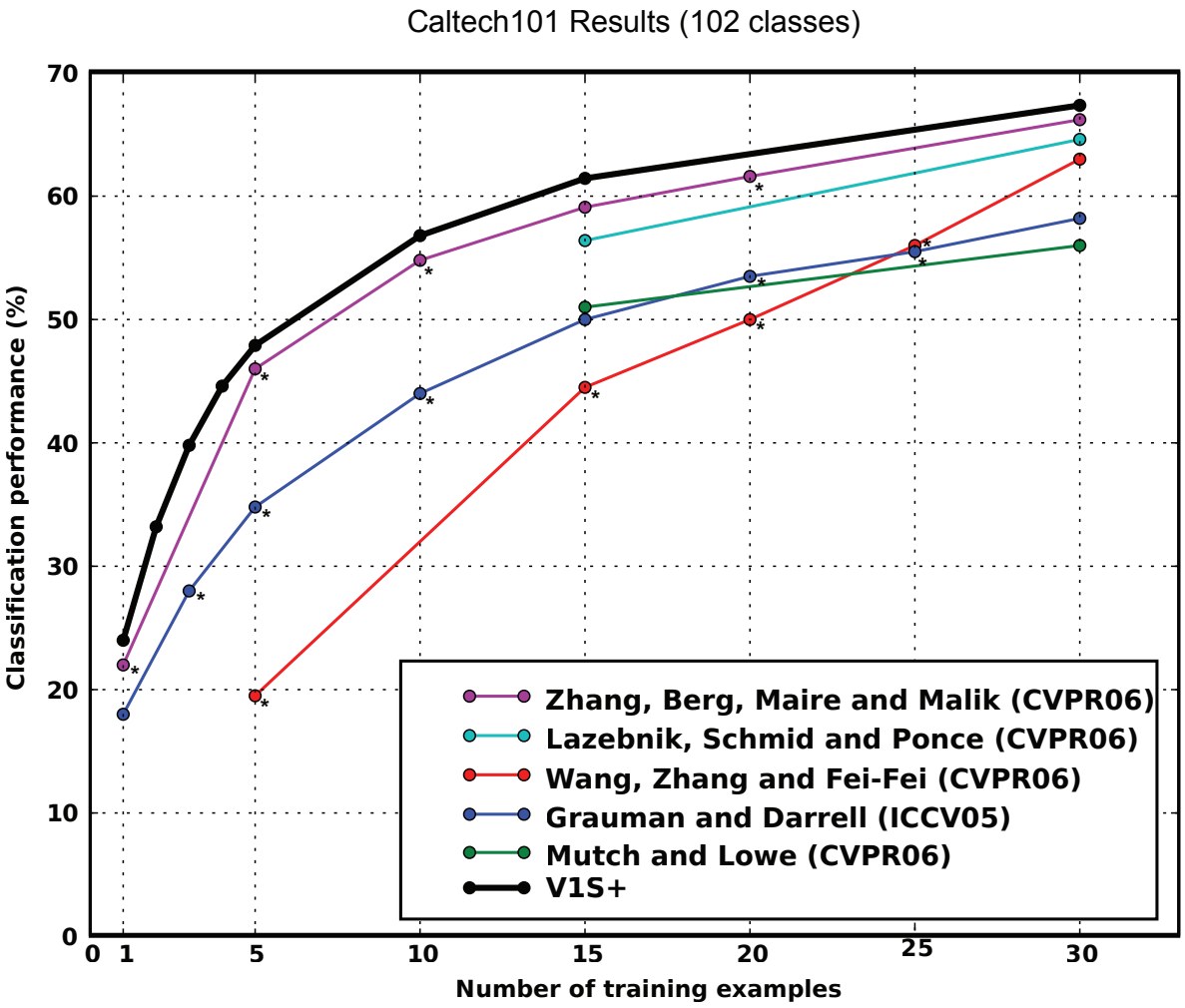



**Figure 4.5: Performance on the Caltech101 as a Function of the Number of Training Examples, Including Small Numbers of Training Examples.** Points marked with asterisks are not exact, but were estimated from published plots.

and 30 training examples), and comparable to, or better than, the current reported performance in the literature ([Zhang *et al.*, 2006; Wang *et al.*, 2006; Mutch and Lowe, 2006; Lazebnik *et al.*, 2006; Grauman and Darrell, 2006]).

Given the V1-like model's surprisingly good performance on this "natural" image set (Figure 4.1), there are two possibilities. Either this model is a previously overlooked good model of visual object recognition, or current "natural" tests do not appropriately engage the object recognition problem. Given that our V1-like model contains no special

machinery for tolerating image variation (and it would generally be considered a "straw man" model by neuroscientists), we were suspicious that this result had more to do with the test set, than the model itself. Nevertheless, to distinguish between these two possibilities, we designed a second more carefully controlled object recognition test that directly engages the core problem of object recognition.

Specifically, we constructed a series of two-category image sets, consisting of rendered images of plane and car objects. By the logic of the Caltech101 "natural" image test, this task should be substantially easier – there are only two object categories (rather than 102), and only a handful of specific objects per category (Figure 4.2A). In these sets, however, we explicitly and parametrically introduced real-world variation in the image that each object produced (see Methods Section 4.4). In spite of the much smaller number of categories that the system was required to identify, the problem proved substantially harder for the V1-like model, exactly as one would expect for an incomplete model of object recognition. Figure 4.2 shows how performance rapidly degrades toward chance-level as even modest amounts of real-world object image variation are systematically introduced in this simple two-category problem (see Figure 4.4 for a comparable demonstration with more than two object categories). Given this result, we conclude that the "V1-like" model performed well on the "natural" object recognition test (Figure 4.1), not because it is a good model of object recognition, but because the "natural" image test is inadequate.

These results (re-)emphasize that object recognition is hard, not because images are natural or complex, but because each object can produce a very wide range of retinal images. Although the Caltech101 and other such "natural" sets were useful in that they encouraged the use of common performance tests on which all recognition models should compete, the results presented here show that a different direction is needed to create the content of those tests. This question is not simply an academic concern – great effort is now being expended to test object recognition models against a new, larger image set: the Caltech256 [Griffin *et al.*, 2007]. However, as with its predecessor, it fails to reflect real-world variation, and our "null" V1 model also performs well above chance (24% accuracy with 15 training examples to discriminate 257 categories), and competitively with early published performance estimates on this new set (Figure 4.6).

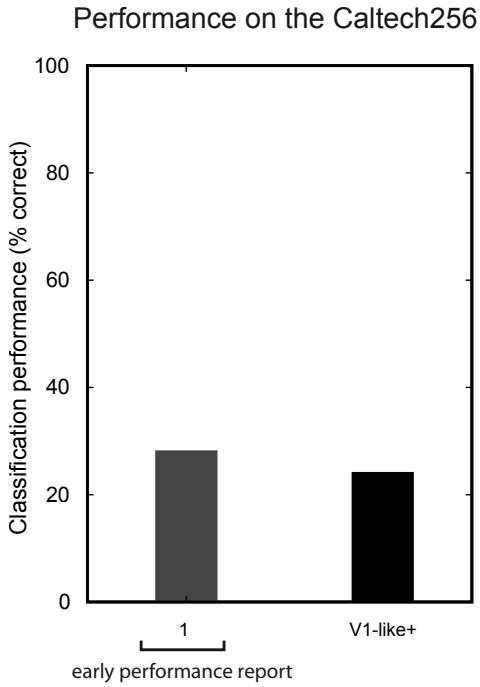



**Figure 4.6: Performance on the Caltech256.** 1=[Griffin *et al.*, 2007]

## 4.3 Discussion

**How should we gauge progress in solving object recognition ?** First, the results presented here underscore that simple chance performance level is far from a good baseline and that our intuitions about "hard" and "easy" recognition problems are often far from correct. Indeed, it is disconcerting how little variation we needed to introduce to break a model that performs quite well according to current "natural" object recognition tests. Thus, simple "null" models (that are able to exploit regularities in the image database) are needed to objectively judge the difficulty of recognition tasks and to establish a baseline for each such task. The V1-like model presented here provides one possible "null" model, and portable code for building and evaluating it is freely available upon request.

Second, the development of appropriate recognition tests is critical to guiding the development of object recognition models and testing performance of neuronal populations that might support recognition [Hung *et al.*, 2005]. The construction of such tests

is not trivial because the issues cut deeper than simple performance evaluation – this is a question of how we think about the problem of object recognition and why it is hard [DiCarlo and Cox, 2007]. Because the number of images in any practical recognition database will be small relative to the dimensionality of the problem domain, test images must be chosen in a manner that properly samples this domain so as to capture the essence of the recognition problem and thus avoid "solutions" that rely on trivial regularities or heuristics.

One approach would be to generate a very large database of "natural" images, like the Caltech sets, but captured in an unbiased way (i.e., with great care taken to avoid the implicit biases that occur in framing a snapshot). Done correctly, this approach has the advantage of directly sampling the true problem domain. However, annotating such an image set is extremely labor-intensive (but see the LabelMe project [Russell *et al.*, 2008], Peekaboom [Von Ahn *et al.*, 2006], and the StreetScenes dataset [Bileschi, 2006; Serre *et al.*, 2007c]). More importantly, a set that truly reflects all real-world variation may be too stringent of an assay to guide improvement in recognition models. That is, if the problem is too hard, it is not easy to construct a reduced version that still engages the core problem of object recognition.

Another approach, an extension of the one taken here, would be to use synthetic images, where ground truth is known by design. Paradoxically, such synthetic image sets may in many ways be more natural than an arbitrary collection of ostensibly "natural" photographs, because, for a fixed number of images, they better span the range of possible image transformations observed in the real world (see also the NORB dataset [LeCun *et al.*, 2004]). The synthetic image approach obviates labor-intensive and error-prone labeling procedures, and can be easily used to isolate performance on different components of the task. Such an approach also has the advantage that it can be parametrically made more difficult as needed (e.g., when a given model has achieved the ability to tolerate a certain amount of variation, a new instantiation of the test set with greater variation can be generated). Given the difficulty of real-world object recognition, this ability to gradually "ratchet" task difficulty, while still engaging the core computational problem, may provide invaluable guidance of computational efforts.

While standardized benchmarks are important for assessing progress, designing

benchmarks that properly define what constitutes *progress* is extremely difficult. On one hand, a benchmark that captures too little of the complexity of the real world (no matter how complex it may seem at first glance) invites over-optimization to trivial regularities in the test set (e.g., Caltech101). On the other hand, a benchmark that embraces too much of the "real" problem can be too difficult for any model to gain traction, giving little insight on which approaches are most promising. This problem is compounded by the fact that there are many more kinds of image variation in the real world beyond those used in our simple synthetic test set (e.g., lighting, occlusion, deformation, etc.). At the center of this challenge is the need to clearly define what the problem is, why it is difficult, and what results would constitute success. The path forward will not be easy, but it is time for the field to give this problem much more central attention.

## 4.4 Methods

### 4.4.1 A *V1-like* Recognition System

Area V1 is the first stage of cortical processing of visual information in the primate and is the gateway of subsequent processing stages. We built a very basic representation inspired by known properties of V1 "simple" cells (a subpopulation of V1 cells). The responses of these cells to visual stimuli are well-described by a spatial linear filter, resembling a Gabor wavelet [Hubel and Wiesel, 1959, 1962, 1965, 1968], with a nonlinear output function (threshold and saturation) and some local normalization (roughly analogous to "contrast gain control"). Operationally, our V1-like model consisted of the following processing steps.

#### Image Preparation

First we converted the input image to grayscale and resized by bicubic interpolation the largest edge to a fixed size (150 pixels for Caltech datasets) while preserving its aspect ratio. The mean was subtracted from the resulting two-dimensional image and we divided it by its standard deviation. The resulting image had zero mean, unit variance,

and a size of HxW. Because images have different aspect ratios, H and W vary from image to image.

### Local Input Divisive Normalization

For each pixel in the input image, we subtracted the mean of the pixel values in a fixed window (3x3 pixels, centered on the pixel), and we divided this value by the euclidean norm of the resulting 9-dimensional vector (3x3 window) if the norm was greater than 1 (i.e., roughly speaking, the normalization was constrained such that it could reduce responses, but not enhance them).

### Linear Filtering With A Set Of Gabor Filters

We convolved the normalized images with a set of two-dimensional Gabor filters of fixed size (43x43 pixels), spanning 16 orientations (equally spaced around the clock) and six spatial frequencies ($\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$, $\frac{1}{6}$, $\frac{1}{11}$, $\frac{1}{18}$ cycles/pixel) with a fixed Gaussian envelope (standard deviation of 9 cycles/pixel in both directions) and fixed phase (0) for a total of $N = 96$ filters. Each filter had zero-mean and euclidean norm of one. This dimensionality expansion approximates the roughly 100-fold increase in the number of primate V1 neurons relative to the number of retinal ganglion cell axons. To speed this step, the Gabor filters were decomposed via singular value decomposition into a form suitable for use in a separable convolution (this is possible because the Gabor filters are of low rank), and the decomposed filters retained at least 90% of their original variation.

### Thresholding And Saturation

The output of each Gabor filter was passed through a standard output non-linearity – a threshold and response saturation. Specifically, all negative output values were set to 0 and all values greater than 1 were set to 1.

### Local Output Divisive Normalization

The result of the Gabor filtering was a three-dimensional matrix of size HxWxN where each two-dimensional slice (HxW) is the output of each Gabor filter type. For each

filter output, we subtracted the mean of filter outputs in a fixed spatial window (3x3 pixels, centered) across all orientations and spatial scales (total of 864 elements). We then divided by the euclidean norm of the values in this window (864 elements), except when the norm was less than 1.

### 4.4.2 Comparison To Other Biologically Inspired Recognition Models

Some of the other models whose performance is shown in Figure 4.1 were biologically inspired, and thus also have V1-like stages contained within them, as well as additional machinery intended to allow invariant object recognition (e.g., [LeCun *et al.*, 2004; Mutch and Lowe, 2006; Serre *et al.*, 2007c]). Thus, it might be surprising that the simple V1-like model presented here outperforms those models. Although detailed comparisons are beyond the scope of this study and tangential to our main point, we note that the V1-like model presented here contains a number of differences from the V1-like portions of these other models (higher dimensionality, larger receptive fields, inclusion of threshold nonlinearities, local normalization, etc.) that probably produce better performance than these models.

### 4.4.3 Classification

To test the utility of our V1-like representation for performing object recognition tasks, we performed a standard cross-validated classification procedure on the high-dimensional output of the model.

#### Dimensionality Reduction

To speed computation and improve classification performance, we reduced the dimensionality of the model output prior to classification. The output of V1-like model (above) was a stack of 96 output images, one per Gabor filter type. Because the dimensionality of this stack can be very high (up to 2,160,000 output values per input image depending on its size), standard dimensionality reduction techniques were used

to prepare the data for classification. Specifically, each of the 96 output images was low-pass filtered (17x17 boxcar) and down-sampled to a smaller size (30x30). Thus, regardless of the original input image size, the total dimensionality for classification was always 86,400 (30x30x96). The data were then sphered (i.e., each filter output was standardized by subtraction of its mean and division by its standard deviation across the training image set; see below), and the dimensionality of the representation was further reduced by principal components analysis (PCA), keeping as many dimensions as there were data points in the training set. For the Caltech101 experiments (Figure 4.1), the dimensionality of the final feature vector was 1530 or 3060 (depending on the number of training examples: 15 or 30, respectively).

### Additional "Ad Hoc" Features

To further explore the utility of this V1-like model, we generated some additional easy-to-obtain features and concatenated these to the final feature vector, prior to PCA dimensionality reduction. These features included: raw grayscale input images (downsampled to 100x100 by bicubic interpolation; 10,000 features), and model output histograms for some intermediate stages of the model: pre-normalization (one local histogram per quadrant of the image), post-normalization (full image), and post down-sampling (full image) – roughly 30,000 features total. No color information was used in these additional features. Throughout the text, results from the system containing these extra "ad hoc" features are reported separately from those obtained with the system that did not have these extra features. These extra features were added to demonstrate what was possible using additional obvious, "cheap" (but still fair) tricks that improve performance without incurring additional conceptual complexity.

### Training

Training and test images were carefully separated to ensure proper cross-validation. 15 training example images, and 30 testing example images were drawn from the full image set. Sphering parameters and PCA eigenvectors were computed from the training images (see *Dimensionality Reduction*, above), and the dimensionality-reduced training

data were used to train a linear support vector machine (SVM) using libsvm-2.82 [Chang and Lin, 2001]. A standard one-versus-all approach was used to generate the multi-class SVM classifier from the training images.

### Testing Protocol

Following training, absolutely no changes to the representation or classifier were made. Each test image was sphered using parameters determined from the training images, projected through the V1-like model onto the eigenvectors computed from the training images, and the trained SVM was used to report the predicted category of the test image

### Number Of Training And Testing Images

Classifiers were trained using a fixed number of examples (15 and 30 example images; see Figure 4.1C and 4.1D). The performance scores reported here are the average of performances obtained from ten random splits of training and testing sets. For testing, 30 images were classified per category, except in categories where there were not enough images available, in which case the maximum number of available images was used (e.g., "inline_skate", the smallest category, has only 31 examples; when 30 examples were used for training, then only one example was available for testing). Since the Caltech101 sets contains a different number of images for each category, care must be taken to ensure that per-category performance is normalized by the number of test examples considered in each category – otherwise, average performance can be biased toward the performance obtained from categories with larger numbers of images available. This is a particular problem for the Caltech101 set, because some of the largest categories are also empirically the easiest (e.g., cars, airplanes, faces, motorbikes). For the performance values reported in this paper, average performance was computed per category, and then these performances were averaged together to obtain an overall performance value (reported in the text and figures).

### Further Controls

To ensure the validity of our results, we undertook a number of checks to verify that the classification procedure used here was correct. Two different SVM front-ends were used (PyML and libsvm command line tools) and produced identical results. To confirm proper cross-validation, we manually inspected training and test set splits to certify that there were no images in common between the two sets (this control was partially motivated by the fact that an earlier version of the Caltech101 dataset contained duplicates). The classification procedure was also repeated with noise images, and for image sets with category labels scrambled; both tests yielded chance performance, as expected.

## 4.4.4 Synthetic Dataset Generation

Synthetic images of cars and planes were generated using POV-Ray, a free, high-quality ray tracing software package [1]. 3-D models of cars and planes (purchased from Dosch Design[2] and TurboSquid[3]) were converted to the POV-Ray format. This general approach could be used to generate image sets with arbitrary numbers of different objects, undergoing controlled ranges of variation. For example, in Figure 4.2 each "pooled variation" level on the x-axis shows the maximum deviation of each of five object viewing parameters (zero variation is shown in Figure 4.2A assuming centering in the image). Given a "pooled variation" level, a set of images was generated by randomly sampling each viewing parameter uniformly within its specified maximum deviation (e.g., $\pm 30°$ in plane rotation). Each image in the set was the result of using one such parameter sample to render the view of the object on a given background (see Figure 4.3). 100% position variation is a full non-overlapping shift of the object's bounding box; 100% scale variation is one octave of change.

While this image set is useful for demonstrating the inadequacy of our V1-like model (in spite of its apparent success at the Caltech101 test), we do not believe it represents any sort of new "standard" against which models of object recognition should be tested.

[1] `http://www.povray.org`
[2] `http://www.doschdesign.com`
[3] `http://www.turbosquid.com`

Instead, we believe that the approach is more important – identifying the problem, generating sets that span limited regions of the problem space, building models, and then "ratcheting" the problem to a higher difficulty level once the limited version of the problem has been solved.

## Acknowledgments

We would like to thank J. Maunsell, J. Mutch, T. Poggio, E. Simoncelli, and A. Torralba for helpful comments and discussion. This work was supported by The National Eye Institute (NIH-R01-EY014970), The Pew Charitable Trusts (PEW UCSF 2893sc), and The McKnight Foundation.

Chapter 5

# Establishing Good Benchmarks and Baselines for Face Recognition*

"What I cannot create, I do not understand"

*Richard Feynman*

Progress in face recognition relies critically on the creation of test sets against which the performance of various approaches can be evaluated. A good set must capture the essential elements of what makes the problem hard, while conforming to practical scale limitations. However, these goals are often deceptively difficult to achieve. In the related area of object recognition, we demonstrated in Chapter 4 the potential dangers of using a large, uncontrolled natural image set, showing that an extremely rudimentary vision system (inspired by the early stages of visual processing in the brain) was able to perform on par with many state-of-the-art vision systems on the popular Caltech101 object set [Fei-Fei *et al.*, 2004a]. At the same time, this same rudimentary system was easily defeated by an ostensibly "simpler" synthetic recognition test designed to better span the range of real world variation in object pose, position, scale, etc. These results suggested that image sets that look "natural" to human ob-

**This chapter is modified from a study published in the proceedings of the "Faces in Real-Life Images" Workshop at the European Conference on Computer Vision (ECCV) in collaboration with James J. DiCarlo and David D. Cox [Pinto et al., 2008a].*

servers may nonetheless fail to properly embody the problem of interest, and that care must be taken to establish baselines against which performance can be judged.

Here, we repeat this approach for the "Labeled Faces in the Wild" (LFW) dataset [Huang *et al.*, 2007], and for a collection of standard face recognition tests. The goal of the present work is not to compete in the LFW challenge, per se, but to provide a baseline against which the performance of other systems can be judged. In particular, we found that our rudimentary "baseline" vision system was able to achieve ∼68% correct performance on the LFW challenge, substantially higher than a pure "chance" baseline. We argue that this value might serve as a more useful baseline against which to evaluate absolute performance and argue that the LFW set, while perhaps not perfect, represents an improvement over other standard face sets.

## 5.1 Introduction

Highly accurate, "in-the-wild" face recognition is one of the holy grail applications in the field of artificial vision. While substantial progress has been made in the last several decades, the problem of face recognition in real-world images remains a largely unsolved challenge. At the heart of this challenge is the considerable amount of image variation (e.g. position, size, orientation, lighting, clutter, occlusion, etc.) that a successful recognition system must tolerate, while maintaining its specificity for individual faces.

As with any engineering effort, it is essential to lay out a specification of what the problem is and what would constitute its solution. In the context of face recognition in real-world environments, this operationally amounts to constructing image test sets and "challenges" that capture the problem of interest. In practice, this is a daunting task, both because of the substantial effort associated with building the set (e.g. collecting and labeling images), and because it is difficult to construct a test set that is fully representative of the staggering image variation that is present in the real world.

At a deeper level, a fundamental problem is that no test set can practically be large enough to span the full range of variation observed in the "wild" (cf. the work of Torralba and colleagues [Ponce *et al.*, 2006]). Compounding this problem, it is difficult to escape bias in the selection of images — most photographs are implicitly or explicitly

centered and framed; frontal views of faces are typically over-represented, either by accident or by design (see the work . Sometimes, individual identity is correlated with background features (indeed, Shamir recently showed that relatively high performance was possible on a variety of standard face recognition sets using image patches taken from the background of images [Shamir, 2008]). Taken together, these factors make it difficult to know whether or not "cheats" (i.e. trivial regularities, which exist in the test set, but not in the real world) exist for a given test set. Likewise, it is difficult to know what fraction of the performance achieved by a particular approach arises from exploitation of these low-level regularities, as opposed to from real progress towards a robust, general solution. This problem is compounded by the increasing complexity of artificial vision systems and the power of machine learning approaches [Hand, 2006], both of which make it difficult to determine which aspects of an image set a given system is actually utilizing to achieve its performance.

In recent years, it has become increasingly popular to evaluate artificial vision systems using large collections of "natural" images, such as can be harvested from the internet. Such image collections are appealing because they are relatively easy to assemble, and they typically include a wide range of sources, settings, etc. However, there is no guarantee that sets that "look" like they span the range of situations that would be encountered in the real world actually do so in reality. Due the nature of these tests, it is practically impossible to control for low-level statistical regularities that may significantly bias the results and potentially lead to wrong interpretations and conclusions.

In the related domain of object recognition, we previously demonstrated in Chapter 4 some of the potential dangers associated with large uncontrolled image sets, showing that an extremely rudimentary vision system (inspired by the early stages of visual processing in the brain) was able to perform on par with many state-of-the-art vision systems on the popular Caltech101 object recognition set [Fei-Fei *et al.*, 2004a]. At the same time, this same rudimentary system was easily defeated by an ostensibly "simpler" synthetic recognition test designed to better span the range of real world variation in object pose, position, scale, etc. These results suggest that a substantial fraction of the Caltech101 problem can be solved using low-level features, without solving the core

problem of image variation. It is important to note that this does not necessarily mean that the Caltech101 set is not useful or that systems that perform well on the Caltech101 do not contain good ideas; rather, it suggests that performance reports might better be judged relative to a baseline that takes these low-level regularities into account.

In the present work, we undertake a similar approach to investigate what might constitute a reasonable baseline benchmark for various face recognition image sets. In particular, we focus on a collection of old "standard" publicly-available face image sets in wide use: ORL [Olivetti Research Laboratory, 1994], Yale [Yale Center for Computational Vision and Control, 1997], AR [Martinez and Benavente, 1998], CVL [Computer Vision Lab at the University of Ljubljana, 1999] and on the relatively new "Labeled Faces in the Wild" challenge set [Huang *et al.*, 2007].

## 5.2 Simple baseline models: Pixel and V1-like representations

In the following experiments, we considered three basic representations to serve as potential baselines: 1) a raw grayscale pixels representation, 2) a "V1-like" representation, inspired by the known properties of cortical area V1, and 3) a "V1-like+" representation, which includes all of the V1 features, plus a grab-bag of easily-computed additional features (e.g. histograms).

The "V1-like" and "V1-like+" representations were constructed as described in Chapter 4, without any optimization or modification for the task at hand. Briefly, the model was composed of a population of locally-normalized, thresholded Gabor functions spanning a range of orientations and spatial frequencies. From a neuroscientist's perspective, these models are "null" models, because they include only a first-order description of the earliest stage of visual processing in the brain. Importantly, these models do not contain any particularly sophisticated representation of shape, nor do they possess any explicit mechanism designed to tolerate image variation (e.g. changes in view, lighting, position, etc.).

For the purposes of the analyzes that follow, the processing of images was divided

into two phases: a *representation* phase, and a *classification* stage. For the pixel representation, the *representation* phase consisted of resizing each image to 150x150 pixels (by bicubic resampling) and unwrapping the pixels into a 22,500 dimensional vector. For the V1-like models, each element in the representation corresponded to the "activity" of a simulated V1-simple-cell-like unit. Each response was computed by first locally normalizing the image (dividing each pixel's intensity value by the norm of the pixels in the 3x3 neighboring region), then applying a set of 96 spatially local (43x43 pixel) Gabor wavelet filters to the image (with a 1 pixel stride), and normalizing the output values (dividing by the norm of the output values of all 3x3 spatial region across all Gabor filter types). Output values were finally thresholded (values below 0 were clipped to zero) and clipped (values above 1 were clipped). The 96 Gabor filters were chosen such that they spanned an exhaustive cross of 16 orientations (evenly spaced "around the clock") and 6 spatial frequencies ($\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$, $\frac{1}{6}$, $\frac{1}{11}$, $\frac{1}{18}$ cycles/pixel). See Chapter 4 (Section 4.4) for a detailed description of these methods.

After each image was converted to an n-dimensional vector, for each of the "standard" face datasets, these vectors were then used as inputs to a linear SVM after dimensionality reduction by PCA. Where required, multi-class classification was implemented using a one-against-rest approach. For the LFW challenge set, a slightly different procedure was used (see below).

## 5.3 Commonly-used face datasets

In this section we briefly present the performance of these baseline models with previous face image sets (ORL, YALE, AR and CVL). To facilitate comparison with previous results, we followed established testing protocols in the literature for each image set (described below).

### 5.3.1 Olivetti Research Lab (ORL) dataset

The ORL face dataset [Olivetti Research Laboratory, 1994] consists of images of 40 subjects, with 10 grayscale images (92x112) per subject, with random variations in

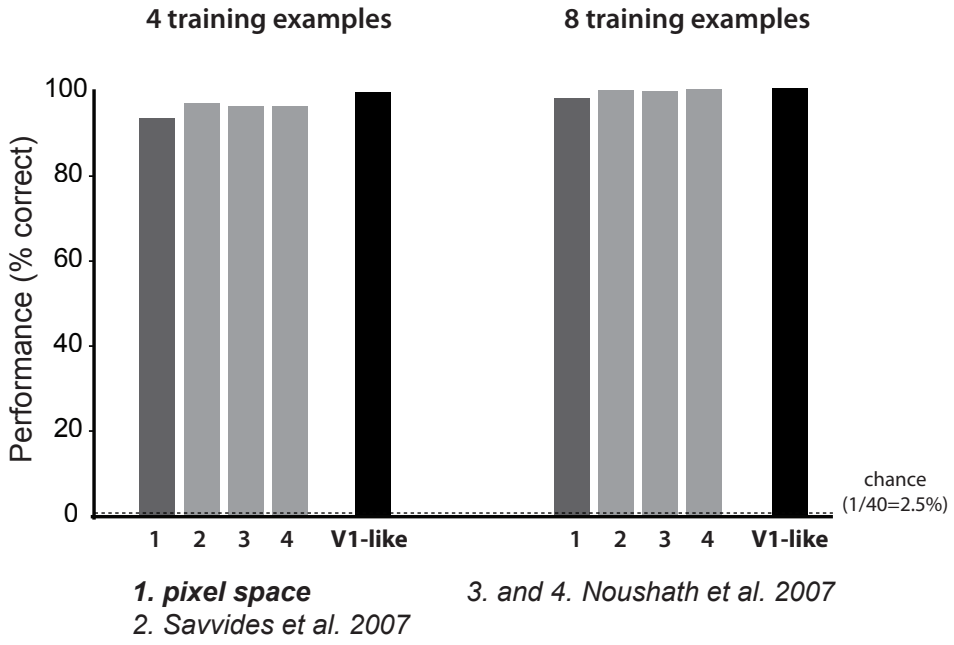



**Figure 5.1:** Performance of Baseline Models on the ORL Set

facial expression, pose, and lighting. The standard task for this set is to identify which individual is present in a given image, based on some number of training examples. Because there are 40 individuals, theoretical chance (i.e. from guessing) is $\frac{1}{40}$, or 2.5%.

Following previously published protocols, classifiers were trained using 4 or 8 training examples per individual (reserving the remaining 6 or 2, respectively, for testing), with a 10-trial random subsampling cross-validation scheme. Figure 5.1 shows the performance using our three baseline representations, along with several performance reports from the literature. In general, in spite of their simplicity the "baseline" models perform very well on the ORL set, with the pixel representation yielding better than 98% correct (with 8 training examples), and the V1-like model achieving *perfect* performance. Given the triviality of these baseline models, these results call into question whether the ORL database provides any real leverage in evaluating face recognition models.

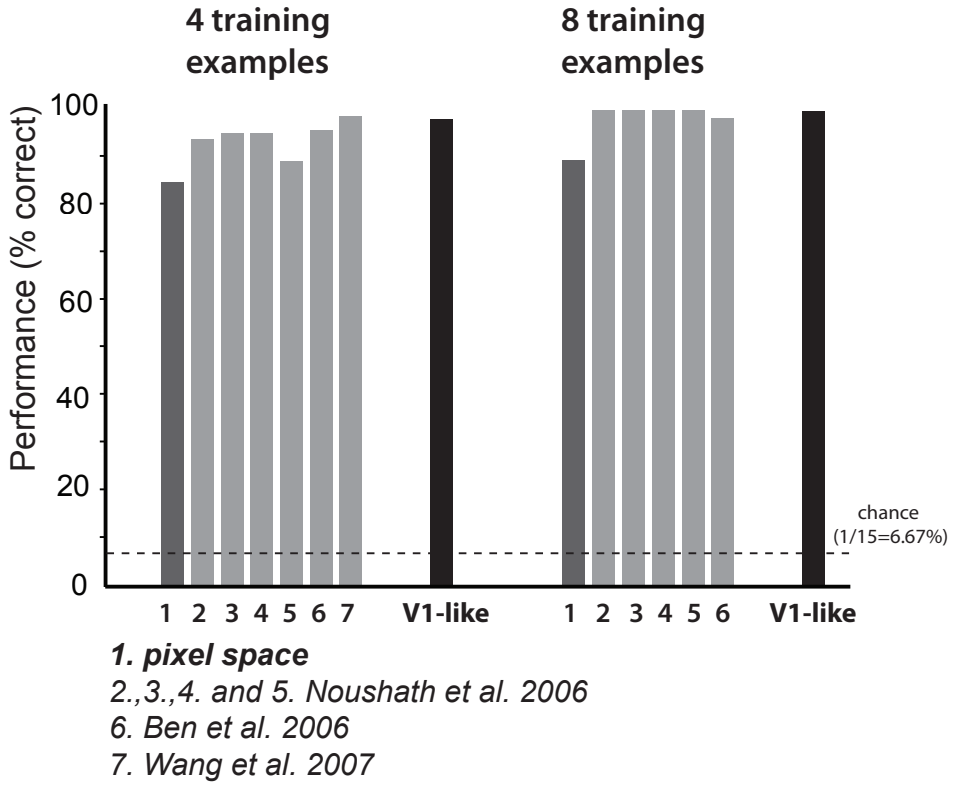



**Figure 5.2:** Performance of Baseline Models on the Yale Set

### 5.3.2 Yale dataset

The Yale face set [Yale Center for Computational Vision and Control, 1997] consists of images of 15 subjects, with 11 gray scale images (320x243) per subject with fixed variations in lighting (e.g. center, right, or left lighting) and expression (e.g. neutral, sad, sleepy, happy). As with the ORL set, the standard task is to identify the individual on the basis of some number of training examples. Theoretical chance is $\frac{1}{15}$, or 6.67%.

Performance was assessed in a manner comparable to that described above. Classifiers were trained with 4 or 8 training examples per individual (reserving the remaining 7 or 3 images, respectively, for testing), with 10-trial random subsampling cross-validation. Results are shown in Figure 5.2. Again, the "baseline" models perform extremely well — the V1-like model achieves near perfect performance ($> 99\%$) with 8 training examples.

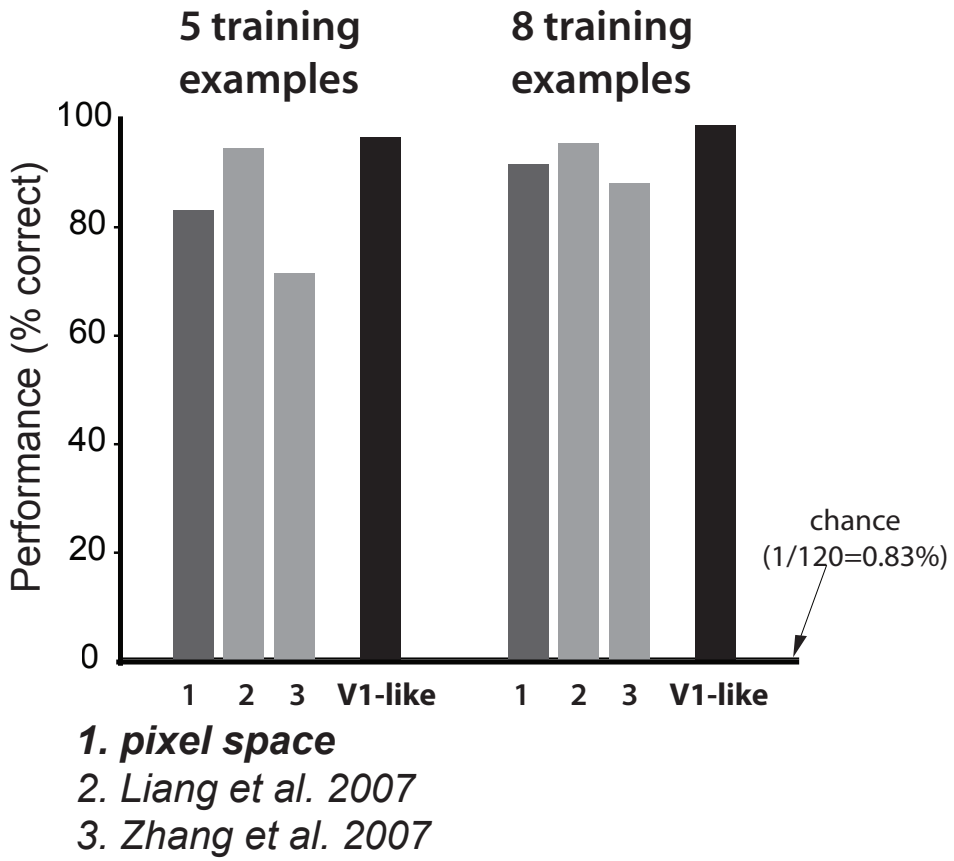



**Figure 5.3:** Performance of Baseline Models on the AR Set

### 5.3.3 Aleix and Robert (AR) dataset

The AR face set [Martinez and Benavente, 1998] consists of over 4,000 color face images (768x576) of 126 subjects. The majority of subjects (65 men and 55 women) participated in two session, separated by two weeks, where 26 images per subject (13 per session) were taken with fixed variations in facial expressions (e.g. neutral, smile or anger), illumination conditions (e.g. left or right light), and occlusions (e.g sun glasses and scarf). Theoretical chance performance on this set is $\frac{1}{120}$, or 0.83%.

Again, a 10-trial random subsampling cross-validation procedure was used. To facilitate comparison with existing literature, we trained with 5 and 8 training examples per individual (reserving the remaining images in each cross-validation split for testing). Results are shown in Figure 5.3. Once again, performance of the baseline models is

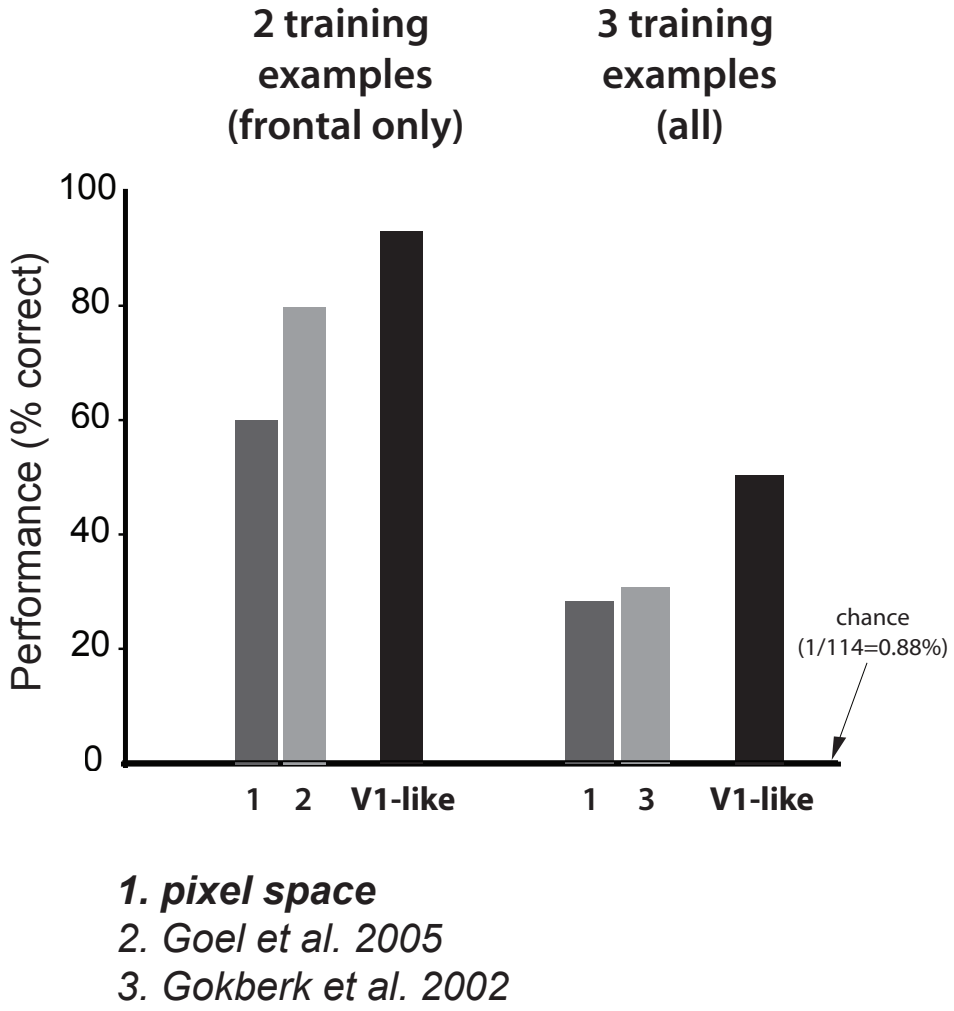



**Figure 5.4:** Performance of Baseline Models on the CVL Set

comparable to or better than previous performance report from the literature, with the V1-like model achieving greater than 98% performance with 8 training examples.

### 5.3.4 Computer Vision Laboratory (CVL) dataset

The CVL image set [Computer Vision Lab at the University of Ljubljana, 1999] contains 114 subjects, 7 color images (640x480) per subject with fixed variations in facial expression (i.e. smile or laugh) and pose (i.e. right, midright, frontal, midleft and left). Theoretical chance performance on this set is $\frac{1}{114}$, or 0.88%.

Following the existing literature using the CVL data set, we report performance using two distinct protocols. First, we trained and tested using only frontal views (2 training examples, 1 test example). In the second protocol, we used three examples drawn randomly from the 7 available images, and tested performance using the remaining 4. In both train/test protocols, 10-trial random subsampling was used for cross-validation. Results are shown in Figure 5.4. Performance of the baseline models was again quite high, with the V1-like model outperforming reported performances from the literature in both training/testing protocols. In the case of protocol based on frontal views only, the V1-like model achieved better than 90% correct, indicating that the task under this protocol can be be largely solved using trivial regularities in the test set images. In the case of the protocol using all views, performance of the V1-like model was substantially lower (∼50% correct), though still substantially higher than the "chance" baseline (0.88%)

## 5.4 Labeled Faces in the Wild (LFW) dataset

The recent Labeled Faces in the Wild (LFW) face set [Huang *et al.*, 2007] contains 13,233 images of 5,749 individuals. This database is described by the creators as "unconstrained", meaning that face images are subject to a large range of "natural" variation (pose, lighting, focus, facial expression, background, age, etc).

The operational goal of this new image set is different from those presented above; it is aimed at studying the problem of face pair matching (i.e. given two face images, decide if they are from the same person or not). To accommodate this alternate goal, we took the vectors produced as the output of each representation (150x150 grayscale pixels, V1-like, V1-like+), and for each pair, we computed the element-wise squared difference. For each training pair, these squared-difference vectors were labeled as "same" and "different," and the task of labeling new (test) examples was thus treated as a two-class classification problem (theoretical chance is 50%).

Prior to training a two-class linear SVM, training data were sphered (zero-mean and unit-variance feature wise), and dimensionality was reduced using principal components analysis (PCA), keeping as many dimensions as there were data points in the training

**Table 5.1:** Performance of Baseline Models on the LFW Challenge Set

| | Pixels | V1-like | V1-like+ |
|---|---|---|---|
| mean (%) | 59.95 | 64.21 | 68.08 |
| std. error | 0.64 | 0.69 | 0.45 |

set. Test data were transformed in an identical manner, using parameters (mean, standard deviation and principal components) computed exclusively from the training set.

Table 5.1 summarizes the mean classification accuracy on the "View 2" portion of the LFW set, using three baseline models: Pixels, V1-like and V1-like+. It is important to note that the V1-like models we used were taken verbatim from what has been described in Chapter 4 — no attempt was made to optimize model parameters for the LFW challenge (i.e. using "View 1" images). Portable code (written in Python) and a minimal virtual machine environment (available in VMware or Amazon EC2 AMI format) are available upon request to facilitate reproducing these results.

While not quite as good, these baseline results are nonetheless reasonably close to early reported results on the LFW set (e.g. ∼72-73% [Nowak and Jurie, 2007]).

## 5.5 Counterpoint: A "simple" synthetic face dataset

To rule out the logical possibility that our baseline models (particularly the V1-like models) are actually effective face recognition systems, we constructed a synthetic image set that spans a range of view variation by design. In particular, the set consisted of two textured 3D face meshes (one male, one female; created with FaceGen, Singular Inversions [1]) that were ray-traced using POV-Ray [2] and randomly overlaid on a variety of backgrounds (as described in Chapter 4). Critically, because the faces were rendered, known amount of variation in view, lighting, etc. could be introduced into the set, and this variation can be parametrically controlled. By the logic of most face recognition

[1] `http://www.facegen.com`
[2] `http://povray.org`

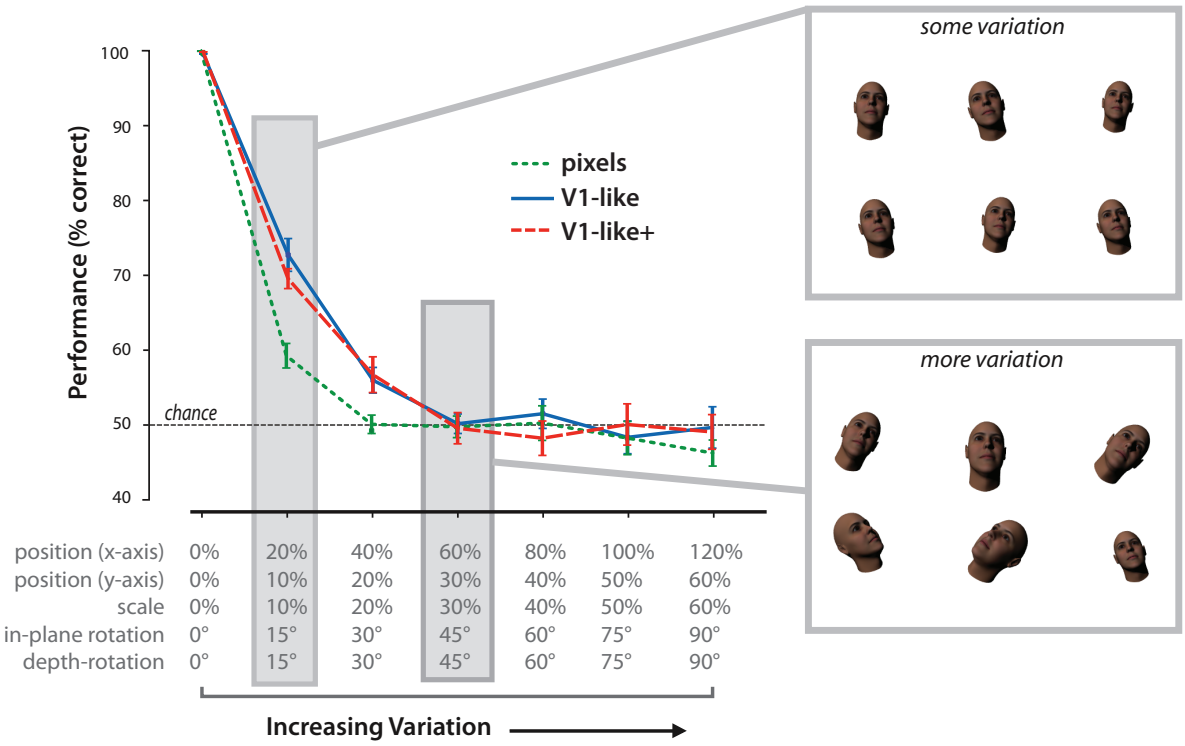



**Figure 5.5:** Performance of Baseline Models on a Synthetic Face Set

challenges, the set should be easy as only two faces must be discriminated, and ample training examples are available for each face.

Figure 5.5 shows the performance of the various baseline models with this synthetic set, as a function of the amount of view variation parametrically applied. As even modest amounts of view variation are included, performance rapidly declines. This rapid decline verifies that the baseline representations are not able to tolerate the sorts of image variation observed in the real world.

## 5.6 Discussion

Our results show that a simple V1-like vision system can perform extremely well on a variety of standard face recognition tests, and that it can perform moderately well on the LFW challenge test set. At the same time, we have shown that this same simple

model performs at or near chance in a "simpler" face recognition task comprised of just two synthetic faces undergoing a wider range of view transformations. Taken together, these results suggest that while the V1-like model is demonstrably not a good general face recognition system, sufficient low-level regularities exist in each test set such that it can nonetheless perform surprisingly well. In the case of the "standard" face recognition sets that we tested (ORL, Yale, AR and CVL), the V1-like model can perform at or near 100%.

Interestingly, the V1-like model performs at ∼68% correct on the new Labeled Faces in the Wild challenge, indicating that some, but not all, of the problem can be solved using a simple system relying on low-level cues. Clearly, there remains a substantial gap between this performance level and 100%, indicating that the LFW set has potential for guiding face recognition progress. We would argue, however, that performance reports on this set should be considered with this number in mind. Given that a trivial algorithm can perform at close to 70% correct, models should ideally target substantially higher performance.

## Acknowledgments

This work was funded in part by The National Institutes of Health (NIH-R01-EY014970), The McKnight Endowment Fund for Neuroscience, and The Rowland Institute at Harvard. We would like to thank David Doukhan and Youssef Barhomi for help in assembling the face databases for this work.

# Chapter 6

# V1-like Features Gone Wild![*]

"S.e.k.y.a.f.o.s.r?"

*Simon Laflamme (2010)*

In recent years, large databases of natural images have become increasingly popular in the evaluation of face and object recognition algorithms. However, we previously illustrated an inherent danger in using such sets, showing that an extremely basic recognition system, built on a trivial feature set, was able to take advantage of low-level regularities in popular object (Chapter 4) and face (Chapter 5) recognition sets, performing on par with many state-of-the-art systems. Recently, several groups have raised the performance "bar" for these sets, using more advanced classification tools. However, it is difficult to know whether these improvements are due to progress towards solving the core computational problem, or are due to further improvements in the exploitation of low-level regularities.

Here, we show that even modest optimization of the simple model introduced in Chapters 4 and 5 using modern multiple kernel learning (MKL) techniques once again yields "state-of-the-art" performance levels on a standard face recognition set ("Labeled Faces in the Wild" [Huang *et al.*, 2007]). However, at the same time, even with the

[*] *This chapter is modified from a study published in the proceedings of the IEEE Computer Vision and Pattern Recognition Conference (CVPR) in collaboration with James J. DiCarlo and David D. Cox [Pinto* et al.*, 2009b].*

inclusion of MKL techniques, systems based on these simple features still fail on a synthetic face recognition test that includes more "realistic" view variation by design. These results underscore the importance of building test sets focused on capturing the central computational challenges of real-world face recognition.

## 6.1 Introduction

The development of a robust face recognition algorithm capable of functioning in unconstrained, real-world environments will have far-reaching applications in our modern digital world. While considerable progress has been made towards building an artificial system that can match human performance, no clear solution has emerged. At the core of this challenge is the extreme diversity in viewpoint, lighting, clutter, occlusion, etc. present in real-world images of faces, which allows any given face to produce a virtually infinite number of different images. A successful recognition system will have to accurately recognize many individuals while tolerating these variations.

To guide any serious effort towards solving face recognition, one needs to define detailed specifications of what the problem is and what would constitute a solution, so that incremental progress can be precisely quantified and different approaches can be compared through a standard procedure. For the purposes of a recognition system, defining a specification amounts to choosing a test set against which an algorithm's performance is evaluated. Recently, it has become increasingly popular to evaluate models on large test sets of "natural" images [Fei-Fei *et al.*, 2004a; Griffin *et al.*, 2007; Huang *et al.*, 2007]. Such an approach is appealing, as it is relatively easy to collect many images from the Internet, and it is relatively efficient to label them (e.g. [Russell *et al.*, 2008; Von Ahn *et al.*, 2006; Collins *et al.*, 2008]). However, there are significant downsides to this approach as well. Importantly, there is no guarantee that such a set accurately captures the range of variation (e.g. view, lighting, etc.) found in the real-world. A variety of factors conspire to limit the range of variation found in such image sets — e.g. posing and "framing" of photographs from the web, implicit or explicit selection criteria in choosing images for the set, etc. Images collected in this manner may also have subtle low-level confounds that "give away" the task, such as

image artifacts or backgrounds that covary with face identity.

As a consequence, it is difficult to know if a given model achieves its recognition performance by robustly solving the problem (i.e., genuinely tolerating image variation), or by exploiting accidental low-level regularities present in the test set. This danger was demonstrated in Chapters 4 and 5 and by the study of Shamir [Shamir, 2008] on popular face and object recognition test sets.

Specifically, Shamir showed that relatively high performance was possible on various face recognition sets using image patches taken from the background, indicating that there was significant, diagnostic covariation of background content with face identity. At the same time, we demonstrated that an extremely rudimentary algorithm was able to match or exceed the performance of many state-of-the-art vision systems (on the Caltech101 [Fei-Fei *et al.*, 2004a], Caltech256 [Griffin *et al.*, 2007], AR [Martinez and Benavente, 1998], ORL [Olivetti Research Laboratory, 1994], CVL [Computer Vision Lab at the University of Ljubljana, 1999], YALE [Yale Center for Computational Vision and Control, 1997], and LFW [Huang *et al.*, 2007] sets). Interestingly, the same "null" model was easily defeated by ostensibly "simpler" synthetic recognition tests specifically designed to better span the range of real world variation. These results indicate that performance reports might better be judged relative to simple baseline models (e.g. based on pixels or wavelets) that are able take these low-level regularities into account.

Recently, with the advent of large scale machine learning techniques [Sonnenburg *et al.*, 2006], it has become possible to significantly outperform the "trivial" baselines set forth in Chapters 4 and 5 on several object and face recognition test sets. These approaches work by optimally combining many image features (e.g. [Varma and Ray, 2007; Huang *et al.*, 2008; Wolf *et al.*, 2008; Bosch *et al.*, 2007]). However, it unclear whether these approaches tap into some deeper solution to the underlying problem, or derive their increased performance from enhanced exploitation of low-level regularities.

To offer insight into this problem, we here apply a similar large-scale approach ("out-of-the-box" multiple kernel learning, [Sonnenburg *et al.*, 2006]) to the trivial representations described in Chapters 4 and 5. Thus while the underlying representation ("front-end") remains unsophisticated in its processing of shape, lacking any mechanism to help tolerate image variation, we have added highly sophisticated "back-end"

processing. We combine variants of the trivial features we proposed earlier to investigate whether more low-level regularities can be captured using a large-scale (but not necessarily smarter) classifier backend. We evaluate this method on "Labeled Faces in the Wild", a large natural face recognition set publicly-available [Huang *et al.*, 2007] and contrast the results with a small synthetic face recognition set, specifically designed to include controlled image variations.

## 6.2 Combining Trivial Features

In the following experiments, the processing of images was divided into two phases: a *representation* phase, in which images were transformed into feature vectors, and a *classification* phase. Since multiple kernel learning techniques (see below) rely on blending of multiple representations, we generated a series of variants based on two basic classes of representation:

1. *Pixels*: a representation based on raw pixel values (with optional spatial resampling, and Gaussian blurring)

2. *V1-like*: a simple representation inspired by the known properties of cortical area V1 (see Chapter 4).

### 6.2.1 Trivial Representations

#### Pixel-based Representations

Here, the *Pixels* representation is simply based on unrolling a preprocessed image into a n-dimensional feature vector. Simple preprocessing steps were added as follows:

1. use color information if present or convert the image to grayscale (2 variants: grayscale or color),

2. normalize the original image to have zero-mean and unit-variance,

3. blur the image with a Gaussian filter (3 variants: no blur, $\sigma = 1$, $\sigma = 2$).

By exhaustively crossing all possible variants of these three steps, one can produce up to six pixel-based feature representations (2 color spaces by 3 blurs).

### V1-like Features

*V1-like* models are composed of a population of locally-normalized, thresholded Gabor wavelets spanning a range of orientations and spatial frequencies. For our purposes, these models are intended as "null" models, as they only represent first-order descriptions of the primary visual cortex, and do not contain any particularly sophisticated representation of shape, nor do they possess any explicit mechanism designed to tolerate image variation (e.g. from variation in view, lighting, etc.).

We previously described two *V1-like* representations: *V1-like* and *V1-like+*; code for both representations is available upon request. In the "default" *V1-like* representation, each input image is first resized by bicubic interpolation (the largest edge is resized to 150 pixels while preserving the aspect ratio), before conversion to grayscale and normalization to zero-mean and unit-variance. Each element in the output representation correspond to the "activity" of a simulated V1-simple-cell-like unit. Each response is computed by:

1. first locally normalizing the image (dividing each pixel's intensity value by the norm of the pixels in the 3x3 neighboring region),

2. applying a set of 96 spatially local (43x43 pixels) Gabor wavelets to the image (with a one pixel stride),

3. and normalizing the output values (dividing by the norm of the output values of all 3x3 spatial region across all Gabor filter types);

4. output values are finally thresholded (values below zero were clipped to zero) and clipped (values above one were clipped).

The *V1-like+* representation includes all of the *V1-like* features, plus a grab-bag of easily-computed additional features (e.g. color and output histograms, see Chapter 4 and 5 for details).

In this study, we refer to the original versions of these representations as "V1-like(A)" and "V1-like(A)+" and describe six new instances, as follows.

- Both "V1-like(B)" and "V1-like(B)+" resize the largest edge of their input images by 75 pixels instead of 150. "V1-like(B)+" concatenates 37x37 raw grayscale pixels to the feature vector instead of 75x75. Other parameters are unchanged from (A);

- "V1-like(C)" and "V1-like(C)+" use slightly bigger Gabor filters (63x63 instead of 43x43) and cover an enlarged panel of 8 spatial frequencies ($\frac{1}{2}$, $\frac{1}{3}$, $\frac{1}{4}$, $\frac{1}{6}$, $\frac{1}{11}$, $\frac{1}{18}$, $\frac{1}{23}$, $\frac{1}{35}$ cycles/pixel), for a total of 128 Gabor filters). Their output stack is downsampled to 10x10x128 with a 21x21 box-car filter instead of the original 30x30x96 with a 17x17 filter. The other parameters are unchanged from (A);

- "V1-like(D)" and "V1-like(D)+" use much larger Gabor filters (125x125 instead of 43x43), and cover an enlarged panel of 24 spatial frequencies ($\frac{1}{2}$, $\frac{1}{5}$, $\frac{1}{8}$, $\frac{1}{11}$, $\frac{1}{14}$, $\frac{1}{18}$, $\frac{1}{22}$, $\frac{1}{27}$, $\frac{1}{31}$, $\frac{1}{36}$, $\frac{1}{41}$, $\frac{1}{46}$, $\frac{1}{52}$, $\frac{1}{58}$, $\frac{1}{64}$, $\frac{1}{70}$, $\frac{1}{76}$, $\frac{1}{82}$, $\frac{1}{89}$, $\frac{1}{96}$, $\frac{1}{103}$, $\frac{1}{110}$, $\frac{1}{117}$, $\frac{1}{125}$) and 36 orientations (equally spaced "around the clock"), for a total of 864 Gabor filters. Their output stack is downsampled to 10x10x864 with a 21x21 box-car filter instead of the original 30x30x96 with a 17x17 filter. The other parameters are unchanged from (A).

These variants represent modest departures from the original *V1-like* representations described in Chapters 4 and 5. Since MKL-based blends benefit from the inclusion of as much diversity as possible, the use of these variants represents just a first step in optimization of the use of the V1-like representation class.

### 6.2.2 Classification by Optimally Combining Kernels

The classification of face images was performed using *Multiple Kernel Learning* (MKL) associated with a *Support Vector Machine* (SVM). MKL allows the practitioner to optimize jointly over a convex linear combination of $p$ kernels $K^* = \sum_{k=1}^{p} \beta_k K_k$ and the SVM parameters $\alpha \in \mathbb{R}^n$ and $b \in \mathbb{R}$, where $n$ is the number of training examples. The

value of the coefficients $\beta$, $\alpha$ and $b$ are obtained by solving the following optimization problem:

$$\begin{cases} \min\limits_{\beta,\alpha,b} & \frac{1}{2}\left(\sum\limits_{k=1}^{p}\beta_k\alpha^T K_k\alpha\right) + C\sum\limits_{i=1}^{n}\xi_i \\ \text{s.t.} & \sum\limits_{k=1}^{p}\beta_k = 1 \quad \text{and} \quad \beta_k \geq 0 \quad \forall k \\ \text{with} & \xi_i = \max(0, 1 - y_i(\sum_{k=1}^{p}\beta_k K_k(x_i)^T\alpha + b)) \end{cases}$$

Where $y_i$ is the binary label $\in \{-1, +1\}$ associated with the $i$-th training example $x_i$.

We solve this problem using the *Semi-Infinite Linear Problem* (SILP) formulation described in [Sonnenburg *et al.*, 2006]. The implementation was taken "out-of-the-box" from the shogun-toolbox [1]. The combined kernels were all linear and were obtained after sphering the data – e.g. features were made to be zero-mean and unit-variance, with sphering parameters being estimated from the training examples. To avoid the MKL optimization unduly favoring any one kernel during training, their traces were normalized to one (i.e. by dividing each element of the training and testing matrix by the sum of the training matrix diagonal). The SVM's regularization parameter $C$ was fixed to $10^4$ for all experiments. All the other parameters were set to their default values.

A full discussion of MKL methods is outside of the scope of the present paper, and is well covered elsewhere [Bach *et al.*, 2004; Sonnenburg *et al.*, 2006; Rakotomamonjy *et al.*, 2007]. For the purpose of this work, MKL methods simply represent an expedient and powerful means to more fully exploit a large collection of features.

### 6.2.3 Hardware and Implementation

Due to the large number of images included in recent "natural" sets (e.g. LFW has over ten thousand images) and the high-dimensionality of the baseline models, the computations involved add up significantly. As a consequence, care must be taken to avoid being limited by throughput. To solve this "speed issue", we harnessed the power of commodity graphics hardware (i.e. NVIDIA GPUs and PlayStation 3's) and leveraged the

[1] `http://www.shogun-toolbox.org`

large scale resources offered by cloud computing services (i.e. Amazon EC2). We coupled these heterogeneous, but powerful, architectures with the flexibility of the Python programming language to collect the data presented below in approximately one week at a reasonable cost.

More specifically, the output of the V1-like models were computed using graphics hardware while the kernel generation and MKL procedures used a combination of cloud computing and "home" computing. Portable code (written in Python) and a minimal virtual machine environment (VMware or Amazon EC2 AMI format) to reproduce our results can be made available upon request.

## 6.3 Experiments

### 6.3.1 Labeled Faces in the Wild Set

We first conducted experiments on the recent "Labeled Faces in the Wild" (LFW) face set (using the "View 2" subset from the LFW "funneled" version, see [Huang *et al.*, 2007] for details). This set contains 13,233 images (250x250 pixels) of 5,749 individuals (see Figure 6.1 for examples) and was created to study the problem of face pair matching in unconstrained environments (i.e., given two face images, decide if they are from the same person or not). At a surface level, face images from the LFW set appear to be quite varied in appearance, and this is hailed as one of set's primary advantages.

#### Pair Matching

In this pair-matching setting, each representation variant described in Section 6.2 (i.e. each of the six variants for the *Pixels* representation and eight variants for the *V1-like* representation) was used to produce six linear kernels as follows.

- The first kernel was the same as in Chapter 5 where the feature mapping is the element-wise squared difference of the representation outputs computed on a given pair of 250x250 images.
- The second and third kernels were also computed from 250x250 images but using


(a) Examples of one individual from LFW. (b) Examples of "same" and "different" pair of faces in LFW.

**Figure 6.1: Examples taken from the "Labeled Faces in the Wild" (LFW) test set.**

an absolute-value difference or a square-root absolute-value difference respectively.

- The last kernels were computed using these three different element-wise differences (i.e. squared, absolute-value and square-root absolute-value) on 150x150 pair of images (cropped from the center).

Finally, for each training pair, the resulting feature vector was labeled as "same" or "different," and the task of labeling new (test) examples was treated as a two-category classification problem (theoretical chance being 50%). We followed the standard procedure described in [Huang *et al.*, 2007] and we report the mean classification accuracy ± s.e.m. computed from the ten random folds of 5,400 training and 600 testing examples from the "View 2" portion of the full LFW set.

### Results

Table 6.1 summarizes the performance using MKL to combine variants of the *Pixels* baseline model. The best performance achieved is 68.33%±0.50 correct, using non-blurred color images. This is substantially more than theoretical chance (50%). More importantly, already this simple pixel-based approach outperforms some previously

| | Grayscale | Color |
|---|---|---|
| no blur | 66.02%±0.53 | **68.33%±0.50** |
| Gaussian blur($\sigma = 1$) | 66.12%±0.54 | 67.47%±0.53 |
| Gaussian blur($\sigma = 2$) | 66.12%±0.55 | 66.45%±0.64 |
| All variants | **68.22%±0.41** | |

**Table 6.1: Performance on the "Labeled Faces in the Wild" (LFW) set using multiple-kernel learning (MKL) with kernels computed from the *Pixels* representations.** The score of each cell is the result of the optimal combination of six kernels (see methods). All the variants add up to 36 kernels. Note that using all kernels doesn't improve performance significantly over the optimal blend of non-blurred color images.

| | V1-like | V1-like+ |
|---|---|---|
| Variant (A) | 76.55%±0.49 | 78.52%±0.49 |
| Variant (B) | 73.23%±0.57 | 76.16%±0.56 |
| Variant (C) | 74.65%±0.38 | 77.30%±0.62 |
| Variant (D) | 73.43%±0.36 | 75.78%±0.49 |
| All variants | **79.35%±0.55** | |

**Table 6.2: Performance on LFW set using MKL with kernels computed from the *V1-like* representations.** The score of each cell is the result of the optimal combination of 6 kernels (see methods). All the variants add up to 48 kernels. Note that using all kernels, our approach can get close to 80% accuracy.

reported methods (e.g. see [Wolf *et al.*, 2008] for details).

The recognition accuracy of the *V1-like* model variants is presented in Table 6.2, and a corresponding ROC curve is shown in Figure 6.2. Interestingly, an MKL blend of only six *V1-like(A)+* kernels (i.e., the representation taken, without modification, from Chapters 4 and 5) scored 78.52%±0.49, which is not significantly different from the current state-of-the-art [Wolf *et al.*, 2008].

When all 48 *V1-like* kernels were blended, performance reached 79.35%±0.55, establishing a new record (as of the time of writing of this manuscript) on this test set. Combining all 36 *Pixels* and 48 *V1-like* kernels did not improve performance further.

| Reference | Methods | Performance |
|---|---|---|
| Huang08 [Huang *et al.*, 2008] | Nowak [Nowak and Jurie, 2007] | 73.93%±0.49 |
| | MERL | 70.52%±0.60 |
| | Nowak+MERL | 76.18%±0.58 |
| Wolf08 [Wolf *et al.*, 2008] | descriptor-based | 70.62%±0.57 |
| | one-shot-learning* | 76.53%±0.54 |
| | hybrid* | 78.47%±0.51 |
| This paper | Pixels/MKL | 68.22%±0.41 |
| | V1-like/MKL | **79.35%±0.55** |

**Table 6.3: Average performance comparison with the current state-of-the-art on LFW.** *note that the "one-shot-learning" and "hybrid" methods from [Wolf *et al.*, 2008] cannot directly be compared to ours as they exploit the fact that individuals in the training and testing sets are mutually exclusive (i.e. using this property, you can build a powerful one-shot-learning classifier knowing that each test example is *different* from all the training examples, see [Wolf *et al.*, 2008] for more details. Our decision not to use such techniques effectively handicaps our results relative to reports that use them).

### 6.3.2 Synthetic Face Set

At this point, we have shown that a combination of MKL techniques with previously described "trivial" feature representations is able to yield record levels of performance on a standard face recognition test set. However, this high level of performance could be due one of to two possible causes: 1) the powerful MKL back-end could be extracting a sophisticated, robust solution to face recognition from the relatively unsophisticated "parts" provided by the V1-like representation, or 2) the LFW set itself could contain more low-level regularities than previously appreciated, which the MKL-based back-end is more adept at exploiting.

To investigate whether the large-scale combination of kernels based on *Pixels* or *V1-like* representations represents a robust solution to the face recognition problem, we conducted experiments using an ostensibly simpler parametric face set described in Chapter 5, using the a similar protocol as described in that work. Briefly, the image set consisted of two individual 3D faces meshes (one male, one female generated using the FaceGen software package, rendered using the POV-Ray ray-tracing package (see Figure

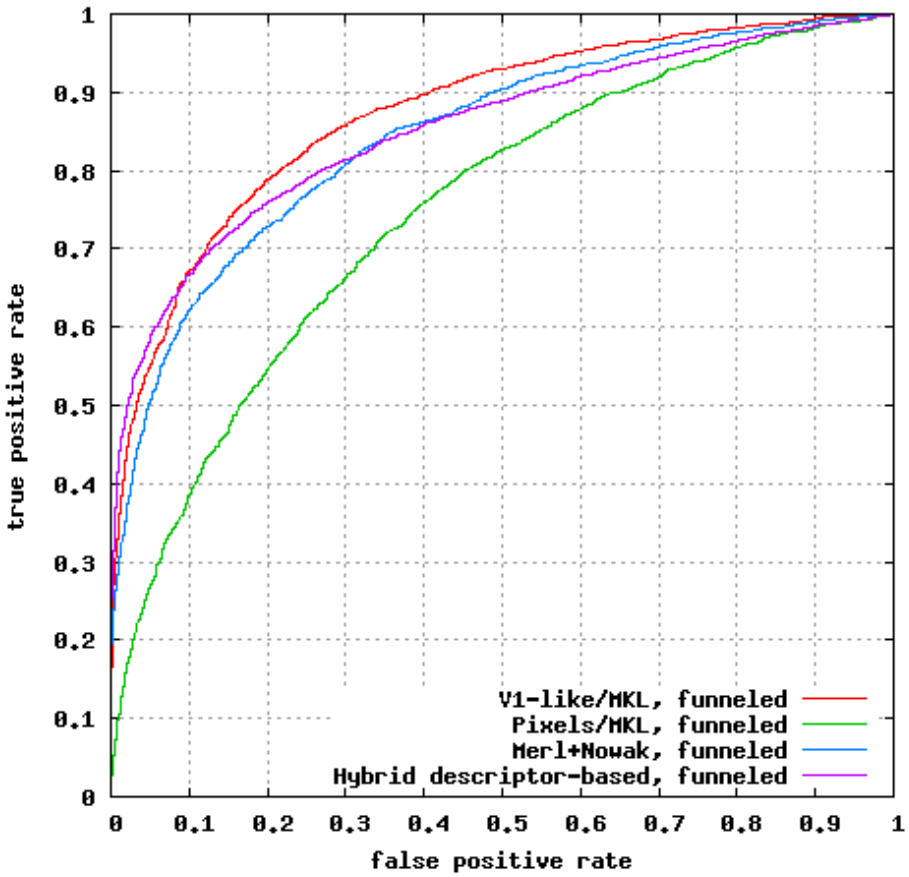



**Figure 6.2:** ROC curve comparison with the current state-of-the art on LFW. These curves were generated using the standard procedure described in [Huang *et al.*, 2007].

6.3 for examples). Because this image set only contains two individuals, it is arguably simpler than most other face recognition sets, which typically contain many individuals (e.g. almost 6,000, in the case of the LFW set). Critically, however, these synthetic faces were rendered with parametrically increasing amounts of variation in rotation, 2D position, and size, so that the performance of a system can be assessed as a function of the amount of variation present in the set. Here, as above, we used MKL-based classifiers, with a combination of kernels from the six *Pixels* representation variants and the eight *V1-like* variants (see Materials and Methods). Test sets corresponding to seven levels of increasing variation (see Figure 6.3, x-axis labels) were created. For

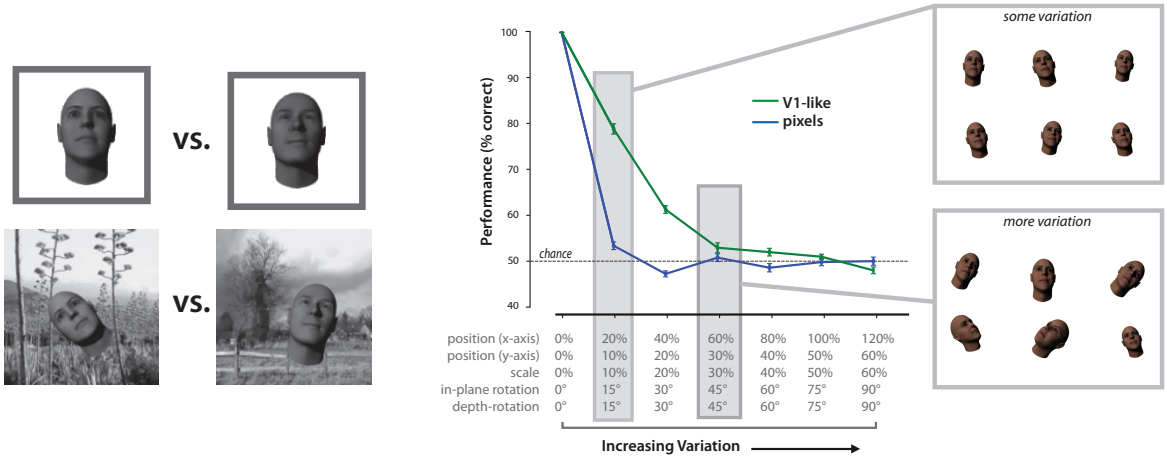



**Figure 6.3: Performance of the *Pixels* and *V1-like* Representations with a MKL back-end on a synthetic face recognition task.** Left top: examples of the faces to be discriminated in their default views, without any background (shown for illustration purposes); Left Bottom: examples of face images used here. The faces could appear in variety of sizes, positions, and orientations, and were randomly composited onto natural image backgrounds. For a human observer, this task is trivial, however even modest amounts of controlled view variation severely degrade performance of the MKL-backed *Pixels* and *V1-like* representations, confirming that these representations are not well suited for real-world face recognition, even with the addition of a more sophisticated back-end.

each level of variation, classifiers were trained with 150 randomly generated faces per individual and were tested using 150 examples.

Figure 6.3 shows the performance of the MKL combinations of the "Pixels" and "V1-like" baseline models with this synthetic set, as a function of the amount of parametric image variation. (i.e. position, viewpoint, scale, etc.). Echoing the results of Chapter 5, performance degrades rapidly as a function of image variation, with even modest amounts of variation resulting in chance performance. Interestingly, performance falls to a level statistically indistinguishable from chance at the same variation level as in Chapter 5 (the fourth data point in 6.3) and the use of a powerful large-scale classifier back-end does not rescue performance at this level. While the addition of an MKL back-end did produce some gains at smaller levels of image variation relative to that reported in Chapter 5 (e.g. the second and third points in Figure 6.3), it is clear that an MKL-based classifier built atop these simple features does not represent a particularly robust solution to the problem of unconstrained face recognition.

## 6.4 Discussion

In this study, we combined variants of the *Pixels* and *V1-like* baseline models using a large-scale statistical learning tool ("out-of-the-box" MKL, [Sonnenburg *et al.*, 2006]) to investigate how far you can get using only simple features. We presented evidence that this simple approach is capable of performing at a state-of-the-art level on the large "Labeled Faces in the Wild" (LFW) face recognition set, while failing on (an ostensibly simpler) synthetic set that includes more realistic view variation by design. Taken together, these results again urge for caution, as more sophisticated large scale kernel learning-based classifiers have the power to leverage good performance even from collections of relatively unsophisticated features. While it is still possible that this powerful machinery is building something "deeper" out of the simple parts provided to it, the extent of this sophistication is limited, at the very least. The MKL-backed system's inability to tolerate even modest amounts of variation (trivial for a human observer), raises the possibility that the MKL-backed system's gains on the LFW set may have more to do with extraction of low-level regularities than with progress towards the "core" problem.

### 6.4.1 The Importance of Good Benchmark Test Sets

These results underscore the importance of building test sets focused on capturing the central computational challenges of real-world face and object recognition. The use of very large sets of "natural" images, while important, may not necessarily be optimal if used alone, as there is no clear way to ensure a realistic range of variation is present and there is no obvious way to control for undesired low-level regularities. A central concern with databases of "found" images from the internet is that photographers typically pose and frame their photos such that a limited range of views are highly over-represented. This effect may be further amplified by the manner in which the sets are assembled. For example, every face image included in the LFW set was the product of a successful detection by the Viola Jones algorithm applied to a set of pictures gathered from news articles on the Internet [Huang *et al.*, 2007]. Even if the image diversity in LFW seems large, applying this face detector "filter" leads to an under-representation of lighting

conditions and face views where the Viola-Jones detector does not excel (e.g. views from above, below or side; which can arguably be more challenging than frontal views). Obviously, such concerns are subject to practical trade-offs — though this automated procedure has biases, it enabled the authors to collect more than ten thousand images at a reasonable cost in terms of labor.

Large-scale methods are undoubtedly very powerful. However, this power represents a double-edged sword. On one hand, the use of large scale methods are now routinely responsible for the highest levels of performance in a variety of object and face recognition tests (e.g. [Varma and Ray, 2007; Bosch *et al.*, 2007]). On the other hand, while such methods are adept at "wringing" substantial performance out of a test set and representation, there is no guarantee that such an exercise brings us closer to a real solution. Indeed, while large scale methods allowed us to achieve a high level of performance gains on the LFW set, we are unconvinced that these gains represent real progress. The cost of potentially false progress is magnified by the computational expense of large scale methods, which favor massive computational and memory footprints.

It is important to note that we are *not* claiming that any previously reported result necessarily represents "false" progress. Previously reported methods may very well represent significant progress towards a solution. However, we argue that this progress will be difficult to see until, as a field, we are able to develop test sets that include realistic ranges of image variation. This will not be an easy task.

One approach that we advocate here is the complementary use of parametric, rendered image sets along with natural photographic sets. While synthetic sets have in some circles fallen out of favor, considered to be "toy" sets, our results here (along with the ones presented in Chapters 4 and 5) suggest that synthetic sets may in some ways be paradoxically more "natural" than a database of "found" photographic images, because they can span a realistic range of view, lighting, etc. variation, *by design*.

In addition, because ground-truth is known, one can assess performance as a function of that variation. Finally, as computer graphics continue to become ever more realistic and accessible, the lines between natural and synthetic images are increasingly blurred, allowing a more natural interplay between both kinds of sets.

Of course, using synthetic images is not the only way to achieve controlled image variation. An alternative approach would be to use (or create) controlled photographic sets such as the PIE Face Set [Sim *et al.*, 2003] (or the NORB Object Set [LeCun *et al.*, 2004]), which systematically vary parameters such as camera and lighting angle. However, while such sets have the appeal of being "real," it is extremely difficult and time-consuming to create a set that spans a sufficient number of axes of variation (i.e. six degrees of freedom in view, multiple light sources, different backgrounds, etc.), and failure to span enough axes results in an incomplete surrogate for the full range of variation in the real world. As a point of reference, for the PIE set, a simple unblended V1-like(A)+ already achieves 87.9%±0.3 performance [2], indicating that low-level regularities are likely nonetheless present. While a controlled photographic set with adequate variation is certainly theoretically possible, we are not aware of a set that meets this goal. Meanwhile, synthetic sets offer extreme practicality and flexibility.

### 6.4.2 New Baselines for Face Recognition

As we previously argued in Chapters 4 and 5, one function for low-level "baseline" models, such as the *V1-like* model, is to set a baseline mark against which performance of other systems can be compared. Test sets where a "trivial" model performs well can still be highly useful, provided the level of performance of that "trivial" model is taken into account when evaluating performance, and provided that there is still "headroom" left with respect to the test set (i.e. the trivial model doesn't perform at 100%). That is, to be reassured that a purpose-built system is going beyond low-level regularities, the performance of the purpose-built vision system should ideally be substantially higher than the performance of a "trivial" model.

The nature of multiple kernel methods also opens up an additional avenue for integrating trivial baselines directly into the discovery process. In particular, if the simple *V1-like* representation presented here were added to the collection of representations under evaluation (i.e. including the purpose-built representation under study), then the *V1-like* representation can "soak up" some of the performance gains due to low-

[2] 68-way one-against-all, chance is at 1.5%

level regularities, making clearer what contributions are made by the purpose-built representation. In such a scenario, one would want the inclusion of the purpose-built representation to result in substantial improvement over the V1-like representation alone. To some extent, interpretation of the weights produced by the MKL approach [Sonnenburg *et al.*, 2006, 2005] could offer valuable insights into what contributions the purpose-built representation is making.

We are clearly not the first to identity the importance of evaluation in driving progress in face and object recognition [Ponce *et al.*, 2006]; our results add to a long-standing process of evaluation and re-evaluation of how algorithms and systems are evaluated. Going forward, large-scale techniques such as MKL will have an important role to play in face and object recognition, however, their use will also require redoubled efforts in collecting and creating test sets that properly channel and direct that power.

### 6.4.3 Future Work

Future work will determine whether it's possible to get even further using many different trivial features on various face and object recognition sets [Fei-Fei *et al.*, 2004a; Griffin *et al.*, 2007; Everingham *et al.*, 2010]. Many possibilities could be explored at close-to-zero human cost thanks to the use of commodity graphics hardware and cloud computing strategies (see Methods Section 6.2.3). A potential approach may be an evolutionary guided generation of baseline model instantiations that uses the interpretable MKL weights to "understand" performance and increase the exploitation of low-level regularities.

We are clearly not the first to identity the importance of evaluation in driving progress in face and object recognition [Ponce *et al.*, 2006]; our results add to a long-standing process of evaluation and re-evaluation of how algorithms and systems are evaluated. Going forward, large-scale techniques such as MKL will have an important role to play in face and object recognition, however, their use will also require redoubled efforts in collecting and creating test sets that properly channel and direct that power.

# Acknowledgments

We would like to thank Antonio Torralba and Ce Liu for encouraging this work, and NVIDIA Corporation for hardware support. This study was funding in part by The National Institutes of Health (NEI R01EY014970), The McKnight Endowment for Neuroscience, Dr. Gerald Burnett and Marjorie Burnett, and The Rowland Institute of Harvard.

Chapter 7

# Comparing State-of-the-Art Visual Features on Invariant Object Recognition Tasks*

"A great deal more is known than has been proved."

*Richard Feynman*

Tolerance ("invariance") to identity-preserving image variation (e.g. variation in position, scale, pose, illumination) is a fundamental problem that any visual object recognition system, biological or engineered, must solve to be successful. While standard natural image database benchmarks can be useful for guiding progress in computer vision, they can fail to probe the ability of a recognition system to solve the invariance problem as we have seen in Chapters 4, 5 and 6. Thus, we do not fully understand which computational approaches are succeeding at solving the invariance problem.

* *This chapter is modified from a study that will be published in the proceedings of the IEEE Workshop on Applications of Computer Vision (WACV) in collaboration with Youssef Barhomi, David D. Cox and James J. DiCarlo [Pinto and Cox, 2010].*

To get traction on this issue, we compared and contrasted a variety of state-of-the-art visual representations using synthetic recognition tasks designed to systematically probe invariance. We successfully re-implemented many state-of-the-art visual representations and confirmed their published performance on a natural image benchmark. We here report that most of these representations perform very poorly on invariant recognition, but that only one representation [Mutch and Lowe, 2008] shows significant performance gains over two baseline representations.

With a relatively small image set and minimal effort, we also show how this approach can more deeply illuminate the strengths and weaknesses of different visual representations and thus guide progress on invariant object recognition.

## 7.1 Introduction

Visual object recognition is an extremely difficult problem and a great deal of effort continues to be expended to reach the goal of discovering visual representations that solve that problem (identification and categorization). Indeed, some of those representations are yielding performance that appears to be quite impressive [Kavukcuoglu *et al.*, 2009; Gehler and Nowozin, 2009; Jarrett *et al.*, 2009; Van De Sande *et al.*, 2010], perhaps even approaching human object recognition performance under very limited conditions [Serre *et al.*, 2007a]. However, understanding what ideas are key to that progress, requires a clear focus on the computational crux problem and a critical, systematic evaluation of how much progress is being made on that problem by each state-of-the-art approach. The goal of the present study is to tackle this issue. For example, are current state-of-the-art representations all performing equally well, or are some consistently better than others? Do they each have weaknesses that might be overcome from learning from the strengths of each? Should we be satisfied with the single performance figure provided by a given natural images database, or can we more precisely determine what components of the object recognition problem are easily handled by each representation and what components are limiting performance?

The computational crux of object recognition is known as the "invariance" problem: any given object in the world can cast an essentially infinite number of different

two-dimensional images onto the retina as the object's position, pose, lighting and background vary relative to the viewer. Thus, to critically evaluate a visual representation for object recognition, we must have ways of measuring its ability to solve the invariance problem. Even though performance evaluation is central in computer vision [Christensen and Phillips, 2002; Ponce *et al.*, 2006], we do not believe that any previously study has directly and systematically tested state-of-the-art algorithms on solving the invariance problem.

In particular, some groups [Murase and Nayar, 1995; Kim and Kweon, 2006] have employed tests that try to directly engage the invariance problem, but these test sometimes miss important components (e.g. failure to use appropriate backgrounds), and they have not been applied to compare and contrast state-of-the-art representations. Other groups [Mikolajczyk and Schmid, 2005; Moreels and Perona, 2007] have compared various visual descriptors (including SIFT, steerable filters, spin images or shape context), but the focus of these studies was not directly on the invariant object recognition problem, but on correspondence matching using local features similar in nature (i.e. "distribution-based" representations). A number of other recent evaluation studies (e.g. [Van De Sande *et al.*, 2010]), including a comprehensive study by [Zhang *et al.*, 2007] have evaluated the performance of state-of-the-art visual representations (and combinations of those representations) using so-called "natural" image databases (esp. Caltech101 [Fei-Fei *et al.*, 2004a] or PASCAL VOC [Everingham *et al.*, 2010]). However, because image variation is not explicitly controlled, these tests may lack real-world image variation (e.g. due to posing of photographs), making difficult or impossible to know how well the visual representations have solved the invariance problem. Moreover, performance on such tests may reflect successful exploitation of low-level regularities (e.g. due to covariation of object identity with background texture or color) and artifacts hidden in the test sets (e.g. cropping cues, etc.). While these problems have been pointed out in recent studies on natural image sets in object and face recognition [Ponce *et al.*, 2006; Shamir, 2008] (in addition to Chapters 4, 5 and 6), systematic tests that expose or circumvent them have not yet emerged.

To illuminate the progress of state-of-the-art visual representations in solving invariant object recognition, we re-implemented five state-of-the-art visual representa-

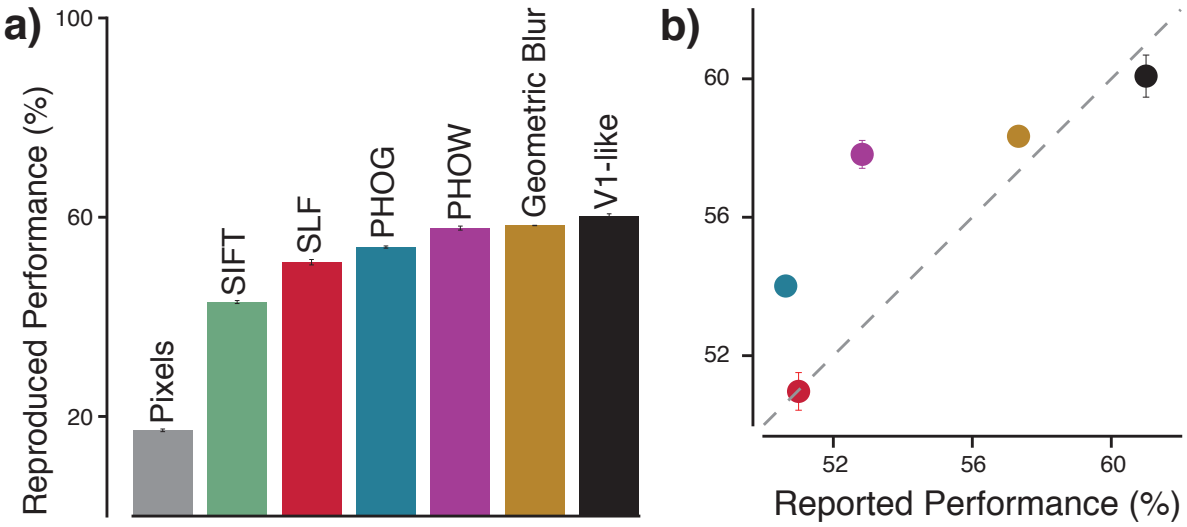



**Figure 7.1: Reproducing the state-of-the-art on Caltech101.** a) Average accuracy with 15 training and 15 testing examples for five state-of-the-art algorithms and two baselines (see Methods Section 7.2). b) Reported vs reproduced performance showing the successful re-implementation of published methods. The reported numbers come from [Mutch and Lowe, 2008] (*SLF*), [Varma and Ray, 2007] (*PHOG*, *PHOW* and *Geometric Blur*) and Chapter 4 (*V1-like*).

tions (some bio-inspired, some not), and we probed the ability of each representation to solve invariant object recognition tasks in which ground truth is known. Specifically, we used a synthetic image approach outlined in Figure 7.2 because it allows: parametric control of the invariance problem, control of shape similarity, control of the number of object exemplars in each category, control of background and color covariance. We compared the obtained performance with both a *Pixels* baseline representation and a well-established baseline representation that approximates the first level of primate visual processing (*V1-like* representation). We also used this approach to ask how well each visual representation handle each of these underlying types of invariance, which is difficult or impossible using prevailing "natural" object recognition tests.

## 7.2 Methods

### 7.2.1 Visual Representations

In the following, we give an overview of the visual features used in our experiments along with their key parameters. We refer to the corresponding publications for more details. Note that the terms "descriptors", "features" and "representations" are used interchangeably throughout the chapter.

#### "Baseline" Features

We used two simple image representations – *Pixels* and *V1-like* – designed to serve as baselines against which the performance of state-of-the-art features can be measured. For both baseline representations, training and testing data were normalized to have zero-mean and unit-variance feature-wise (using the training data only), and a simple linear kernel was used for classification (see Section 7.2.1).

**Pixels:** In the *Pixels* representation, each image was simply rescaled to 150 by 150 pixels, converted to grayscale and then unrolled as a feature vector. The resulting feature vector represents an almost entirely unprocessed representation of the original image.

**V1-like:** In the *V1-like* representation, features were taken without any additional optimization from Chapter 4's V1S+. This visual representation consists of a collection of locally-normalized, thresholded Gabor wavelet functions spanning a range of orientations and spatial frequencies and is based on a first-order description of primary visual cortex V1. *V1-like* features have been proposed by neuroscientists as a "null" model for object recognition since they do not contain a particularly sophisticated representation of shape or appearance, nor do they possess any explicit mechanism designed to tolerate image variation (e.g. changes in view, lighting, position, etc. [DiCarlo and Cox, 2007]). In spite of their simplicity, these features have been shown to be among the best-performing non-blended features set on standard natural face and object recognition benchmarks (i.e. Caltech101, Caltech256, ORL, Yale, CVL, AR, PIE, LFW –

see Chapters 4, 5 and 6), and are a key component of the best blended solutions for some of these same benchmarks [Gehler and Nowozin, 2009]. We used publicly available code for these features with two minor modifications to the published procedure. Specifically, no PCA dimensionality reduction was performed prior to classification (the full vector was used) and a different regularization parameter was used ($C = 10,000$ instead of $C = 10$).

### State-of-the-art Features

We considered a diverse set of five state-of-the-art features. Most were chosen on the basis of their high-performance on Caltech101 (arguably still the most widely used multi-class object recognition benchmark today [Kavukcuoglu *et al.*, 2009; Jarrett *et al.*, 2009; Gehler and Nowozin, 2009]). Effort was made to span a wide range of different approaches to object recognition: models that were bio-inspired, and those that are not; distribution-based and non-distribution-based models, and models with a custom kernel (e.g. Spatial Pyramid) and models with a simple linear one. To promote experimental reproducibility and ease distribution, we re-implemented all but one of these models (*SLF*, see below) from the ground up using only free, open-source software (e.g. Python, NumPy, SciPy, Shogun, OpenCV, etc.).

**SIFT:** *SIFT* descriptors [Lowe, 2004] were computed on a uniform dense grid from a 150 by 150 pixels grayscale image with a spacing of 10 pixels and a single patch size of 32 by 32 pixels. The result was then unwrapped as a feature vector. Training and testing data were normalized to have zero-mean and unit-variance feature-wise (using the training data only), and SVM classification was done using a linear kernel.

**PHOW:** *PHOW* (Pyramid Histogram Of visual Words) is a spatial pyramid representation of appearance [Bosch *et al.*, 2007; Varma and Ray, 2007; Lazebnik *et al.*, 2009]. To compute these features, a dictionary of visual words was first generated by quantizing the *SIFT* descriptors with k-means clustering. We fixed the dictionary size to 300 elements, and the SVM kernel to a three-level spatial pyramid kernel with $\chi^2$ distance [Lazebnik *et al.*, 2009].

**PHOG:** *PHOG* (Pyramid Histogram Of Gradients) is a spatial pyramid representation of shape [Bosch *et al.*, 2007; Varma and Ray, 2007] based on orientations gradients (HOG [Dalal and Triggs, 2005]) of edges extracted with a Canny detector. We fixed the angular range to 360 degrees, the number of quantization bins to 40, and the SVM kernel to a four-level spatial pyramid kernel with $\chi^2$ distance.

**Geometric Blur:** The *Geometric Blur* shape descriptors [Berg and Malik, 2001; Zhang *et al.*, 2006] are generated by applying spatially varying blur on the surrounding patch of edge points in the image (extracted by the boundary detector of [Martin *et al.*, 2004]). We fixed the blur parameters to $\alpha = 0.5$ and $\beta = 1$, the number of descriptors to 300 and the maximum radius to 50 pixels. For the SVM classification, we used the kernelized distance $D^A$ from [Zhang *et al.*, 2006] (Eq. 1) with no texture term as described in [Varma and Ray, 2007].

**SLF:** The bio-inspired *Sparse Localized Features* (SLF) [Mutch and Lowe, 2008] are an extensions of the *C2* features from the Serre et al. HMAX model [Riesenhuber and Poggio, 1999b; Serre *et al.*, 2007c]. For this representation, we took advantage of the MATLAB code provided by the authors (FHLib[1]). Here, the SVM classification was based on a linear kernel with normalized training and testing data (zero-mean and unit-variance feature-wise). Interestingly, we found that it was unnecessary to use the feature selection procedure described in [Mutch and Lowe, 2008] to match the level of Caltech101 performance achieved in that work. We suspect that our slightly higher observed performance level was due to differences in SVM formulation and regularization parameters.

### Classification

For classification we used L2-regularized Support Vector Machines (libsvm solver from the Shogun Toolbox[2]) with a regularization constant $C = 10,000$. Each representation was used to produce either a simple linear, or custom (for *PHOW*, *PHOG* and *Geometric*

[1] `http://www.mit.edu/~jmutch/fhlib`
[2] `http://www.shogun-toolbox.org`

*Blur*) kernel. Multi-class problems were addressed with a one-versus-rest formulation.

Classifiers were trained using a fixed number of examples. Except when stated otherwise, we use 150 training and 150 testing examples for each class. The performance scores reported are the average of performances obtained from five random splits of training and testing sets, the error bars represent the standard error of the mean. The same image splits were used for all the representations.

### 7.2.2 Synthetic Image Set Generation

A key feature of the evaluation procedure described in this study is the use of object test sets where the ground-truth range of variation in object view is known. In particular, we chose to use rendered three-dimensional objects, which allow for large numbers of test images to be generated with minimal effort, while preserving tight controls on the distribution of view variation within the set.

For each category of objects (cars, planes, boats, animals), five 3D meshes (purchased from Dosch Design and Turbosquid.com) were rendered using the POV-Ray ray-tracing package onto a transparent background, and this image was overlaid onto a randomly selected background image from a set of more than 2,000 images of natural scenes (Figure 7.2a). Background images were selected randomly, and no background was ever reused within a given training / test set. While backgrounds often contain information that is helpful for recognizing objects, we made no effort to associate objects with context-appropriate background, in order to better focus the test set on object recognition *per se*. All images were made to be grayscale to avoid any color confound.

In Figures 7.3a and 7.4a, object views were varied simultaneously ("composite variation") along four axes: position (horizontal and vertical), scale, in-plane rotation and in-depth rotation. In order to roughly equate the effects of each of these kinds of view variation, we defined a view change "quantum" for each axis of variation, such that each kind of variation, on average, produced an equivalent pixel change in the image, as defined by a pixel-wise Euclidean distance. The average pixel change associated with a full, non-overlapping translations of the objects' bounding boxes was taken as the "standard" unit of pixel variation, and all other view change units were equalized

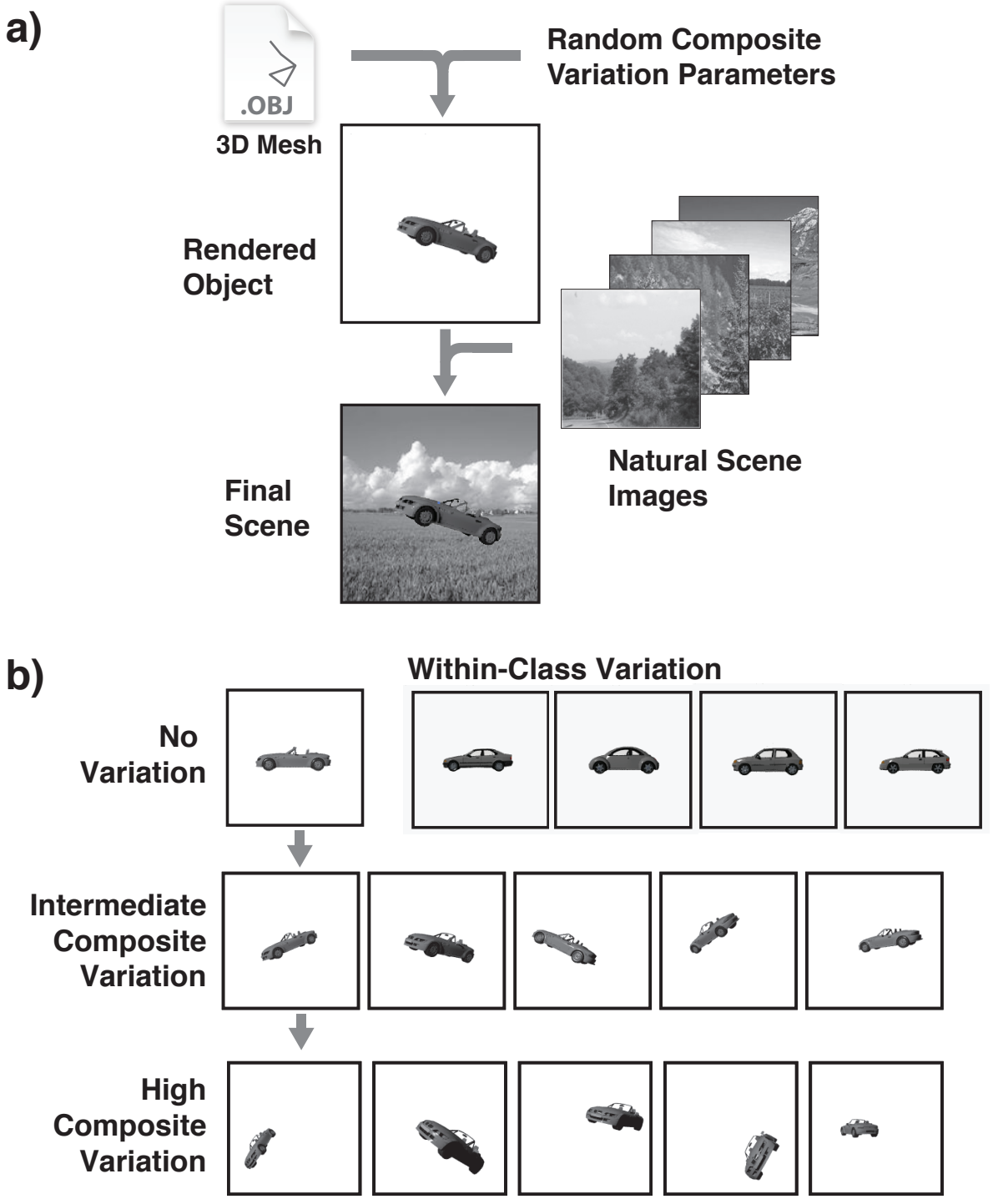



**Figure 7.2: Object rendering procedure.** a) 3D object meshes were rendered using view parameters drawn from a uniform random distribution, and then composited onto randomly selected natural background images. b) Examples of view variation ranges used in this study, spanning from no view variation (top) to relatively large amounts of "composite variation" (i.e. all four types of variation included: position, scale, in-plane rotation and in-depth rotation).

to this unit.

Separate test and training sets were generated for each of a series of view-variation ranges, spanning from no view variation (Figure 7.2b, top) to a relatively large amount of variation (Figure 7.2b, bottom). For each range, view parameters were drawn independently along each of the four axes, with a uniform random distribution, and all object exemplars were included as part of the random image draw. Importantly, for a given range, a successful recognition system must not only correctly recognize objects with view parameters at the extremes of the range, but must also correctly recognize objects across the entire range.

## 7.3 Results

The main goal of this study was to test state-of-the-art artificial visual representations on truly difficult, systematic tests of invariant object recognition where ground truth is known. To do this, we first verified that we had successfully re-implemented each visual representations (see Methods Section 7.2). We used the Caltech101 image categorization task [Fei-Fei *et al.*, 2004a] as a point of reference. Despite many serious concerns raised about the Caltech101 set [Ponce *et al.*, 2006] (Chapter 4), that test is still widely used in the object recognition community and thus most state-of-art algorithms have reported accuracy on Caltech101 in the literature [Berg and Malik, 2001; Zhang *et al.*, 2006; Varma and Ray, 2007; Mutch and Lowe, 2008; Lazebnik *et al.*, 2009]. Specifically, for each representation, we compared the Caltech101 performance of our re-implementation with the performance reported in the literature. As Figure 7.1b shows, in all cases, we succeeded in matching (or slightly exceeding) the reported performance of all representations on the Caltech101 set. These results provide an independent replication of the original authors' results, and that we have succeeded in re-implementing these state-of-the-art algorithms.

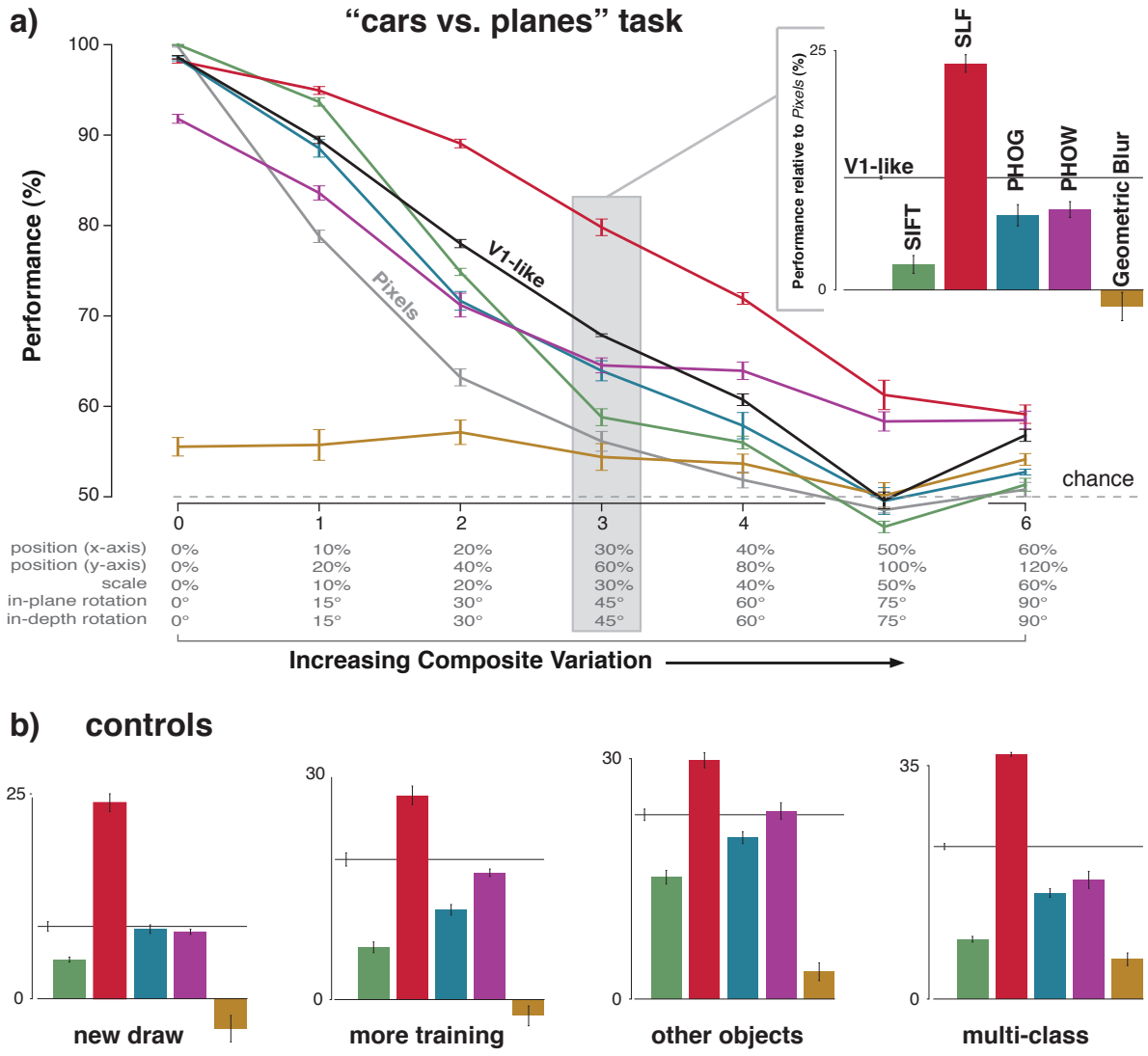



**Figure 7.3: Performance on object recognition tasks with controlled composite variation.** a) Average accuracy of each representation on a series of "cars vs planes" tasks in which the composite variation is gradually increased. The inset shows the performance of the five state-of-the-art features from the literature relative to the *Pixels* representation. b) Performance relative to *Pixels* on the composite variation 3 (cf. inset in a)) with a new draw of "cars" and "planes" images, more training examples, other objects ("animals vs boats"), and more object classes ("animals vs boats vs cars vs planes").

### 7.3.1 Basic-level Object Recognition

With successful re-implementations of a collection of state-of-the-art algorithms in hand, we proceeded to test each representation on basic-level invariant object recognition tasks and to benchmark these results against a simple *Pixels* baseline representation and the *V1-like* baseline representation (see Methods Section 7.2).

In Figure 7.3, we show the performance of all seven visual representations (including the *Pixels* and *V1-like* baseline representations) as we gradually increase the difficulty of the "cars vs. planes" task by increasing four types of object variation (position, scale, in-plane rotation and in-depth rotation) at the same time in a fixed mixture ("composite invariance", see Methods Section 7.2.2). Even though all of the state-of-the-art representations consistently outperformed *Pixels* and performed approximately equally well on the standard Caltech101 "natural" task (Figure 7.1), the results in Figure 7.3 reveal clear differences among the models. Several state-of-the-art models are clearly below the *V1-like* baseline and, in some case, below the *Pixels* baseline. Most interestingly, the results show that one representation (*SLF*) has made clear gains on the composite invariance problem (see Discussion Section 7.4).

Given that there is no single test of basic-level recognition, we next considered the possibility that the results in Figure 7.3a are simply due to particular parameter choices one necessarily has to make when testing a visual representation (e.g. number of training examples, number of objects, particular choice of objects, etc). Specifically, we picked an intermediate level of composite variation that best revealed the differences among the representations (see highlighted section in Figure 7.3a and, using this level of composite variation, we created four new object recognition tests using: a new set of "cars" and "planes" images, three times as many example images for training (450 instead of 150), two completely new objects ("animals" and "boats"), and a test with four basic object categories ("animals", "boats", "cars", "planes") instead of two (Figure 7.3b). In all cases, we found that the relative performance of each representation (i.e. performance relative to the other representations) was largely unaffected by these testing parameter choices. We quantified this robustness by computing Spearman's rank correlation of performance in all pairs of these basic level recognition tasks, and found very high

values (mean = 0.95, min = 0.86). In sum, these results show that, at least for the currently considered set of state-of-the-art models, our tests of basic-level recognition are largely robust to: the exact set of images (at a given level of composite variation) the number of training examples, the exact categories of basic object used and the number of categories.

### 7.3.2 Subordinate-level Object Recognition (Faces)

The absolute level of recognition performance must depend on the degree of 3D structural similarity of the objects in the test set. Specifically, while objects involved in tests of basic-level recognition (e.g. cars vs. planes vs. boats, etc.) are moderately to highly dissimilar in terms of 3D structure, objects in so-called, subordinate-level [Mervis and Rosch, 1981] tasks of recognition (e.g. one face vs. another face) share common 3D structure that makes tasks intrinsically more challenging. Thus, we used the same approach as in Figure 7.3 (but with lower absolute levels of view variance) to test the performance of the state-of-the-art representations on a face recognition task. The results are shown in Figure 7.4. As with the basic-level recognition task, we found that most, but not-all, state-of-the-art representations performed below the *V1-like* baseline representation and that the relative performance of the representations on the face task was largely robust to the number of training examples, the particular choice of faces, and the number of faces (mean Spearman's rank correlation = 0.92, min = 0.85).

To ask if a representation's performance on basic-level recognition is predictive of its performance on subordinate-level recognition, we directly compared the results in the "cars vs. planes" task (Figure 7.3) and the "face vs. face" task (Figure 7.4). Figure 7.5 shows the performance of all representations on both tasks using a range of different testing conditions (as outlined in Figures 7.3 and 7.4). We found that the absolute performance level on each task is highly correlated (Figure 7.5a). However, when performance is plotted relative to the *Pixels* (Figure 7.5b) and *V1-like* baselines (Figure 7.5c), the data reveal that one of the state-of-the-art representations (*PHOW*, see purple points) is reasonably good at basic-level invariant object recognition but quite poor at subordinate-level (face) recognition, and that one representation (*SLF*,

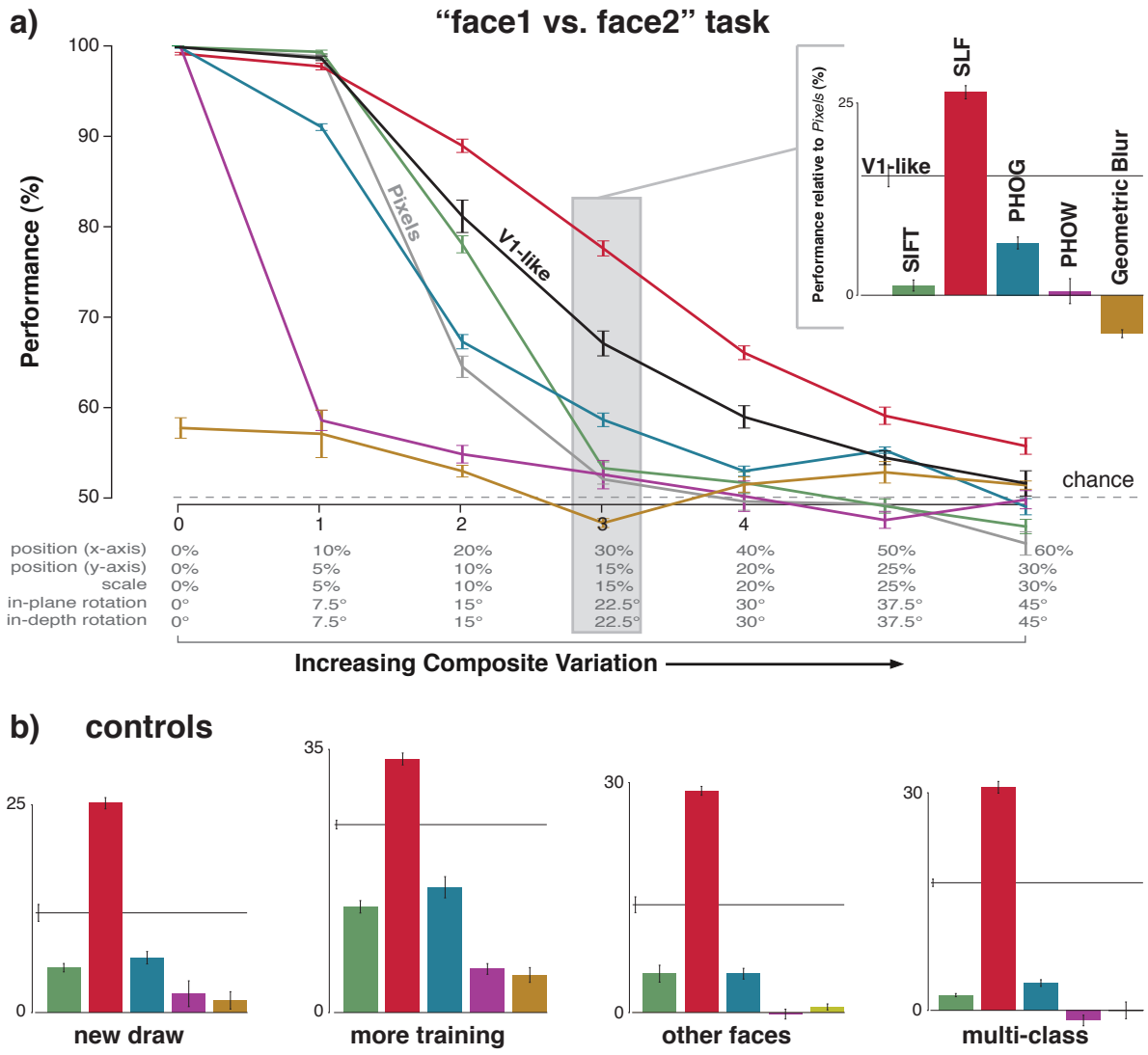



**Figure 7.4: Performance on subordinate-level object recognition tasks (face discrimination) with gradually increasing amounts of composite variation.** Plotting conventions as in Figure 7.3, but note that, because this is a more difficult that, view variation parameters are lower than those used in Figure 7.3.

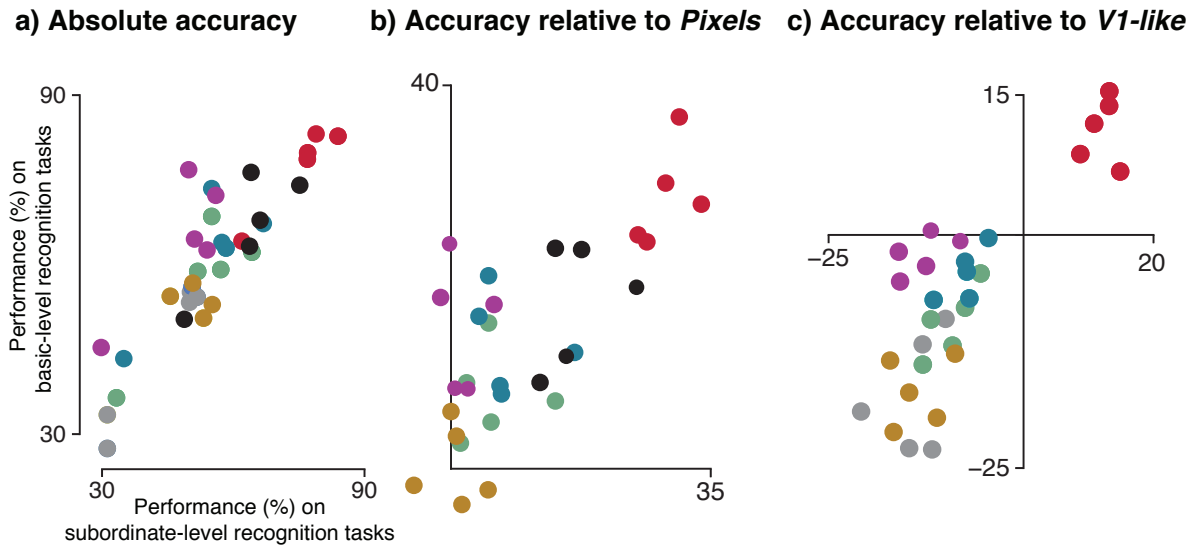



**Figure 7.5: Comparison of performance on basic-level and subordinate-level object recognition tasks.** a) We plot the accuracy of the representations on five tasks at composite variation 3 (original, new draw, more training, other objects and more objects; see Figures 7.3 and 7.4 for details). b) Same data re-plotted relative to *Pixels* representation. c) Same data re-plotted relative to *V1-like* representation.

see red points) is a clear stand-out with respect to the *V1-like* baseline on both basic- and subordinate-level recognition tasks (see Discussion Section 7.4).

### 7.3.3 Individual Types of View Variation

We next considered the possibility that our tests (e.g. Figure 7.3) were over-weighting some types of variation relative to others (see Methods Section 7.2.2, and Chapter 4 for a description of how the relative mix of types of object variation in Figure 7.3 was chosen). Without an operational goal (e.g. matching human performance), it is impossible to exactly determine if one type of variation is under- or over-weighted in recognition tasks, even when ground truth is known. However, for the present study, we sought to determine if our conclusions about the relative performance of the state-of-the-art representations would be strongly altered by our current weighting of each of these four type of view variation. Specifically, we created four new basic-level recognition tests in which we fully removed one type of variation from each test (and made sure that

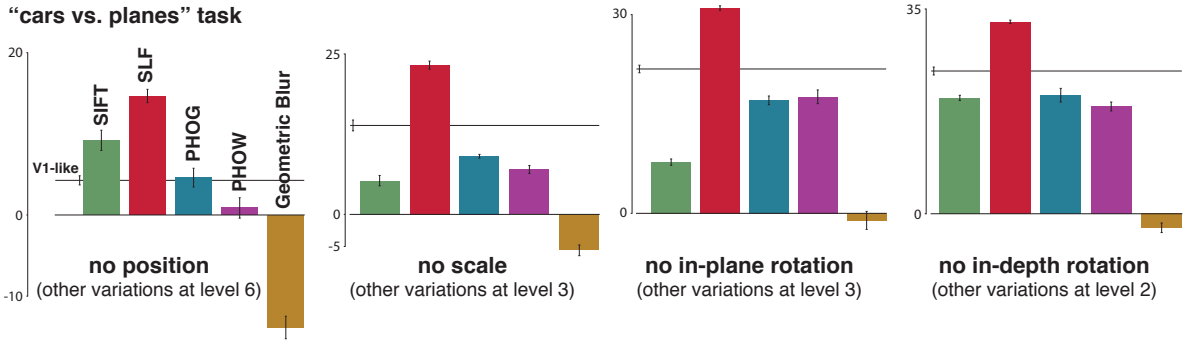



**Figure 7.6: Performance rank order of the representations after removal of each type of view variation (position, scale, in-plane rotation and in-depth rotation).**

the remaining composite variation was at a level that put all the representations in a performance regime that was not on the ceiling or the floor, analogous to the highlighted region in Figure 7.3). We found that the relative performance of all the state-of-the-art representations on these four basic-level object recognition tasks (Figure 7.6) was very similar to that found with the full composite invariance tests (Figure 7.3; mean Spearman's rank correlation between results in these two figures was 0.90, min = 0.64). This suggests that, at least for the currently considered set of state-of-the-art models, our tests of basic-level recognition are not strongly dependent on composite variation "mixture" in the test.

To ask what type of tolerance is least difficult and most difficult for each representation without regard to absolute performance, we created four new tests of basic level recognition ("cars vs. planes") that each contained only one type of object variation (position-only, scale-only, in-plane-rotation-only, and in-depth-rotation-only tests). To fairly compare each representation's degree-of-difficulty in handling each type of variation, we equated these four tests in that the amount of variation produced an equal degree of difficulty for the pixel representation (10% absolute performance drop, see Figure 7.7). The results show that most of the state-of-the-art representations have the least difficulty with position variation, and tend to have more difficulty with (e.g.) in-depth variation. For example, these tests reveal that the representation of [Mutch and Lowe, 2008] which was designed with position and scale variation in mind

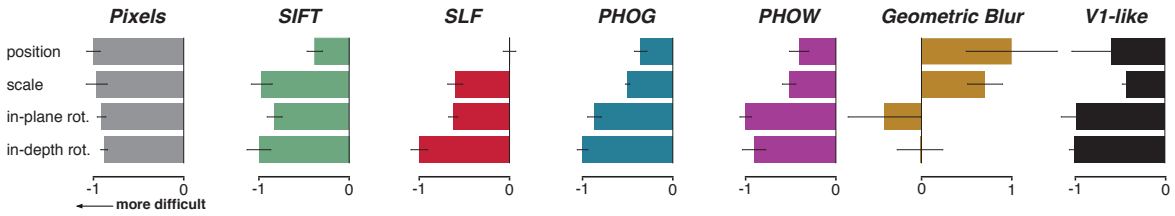



**Figure 7.7: Degree of difficulty of each type of view variation for each representation.** The amount of each type of variation was chosen to produce a 10% approximate decrease in performance for the *Pixels* baseline. We here show the change in average performance due to each type of variation (i.e. change relative to each representation's performance in the "no variation" task (composite variation 0) in Figure 7.3). The axes are normalized by the maximum decrease (or increase, for *Geometric Blur*) of performance for each representation. Thus, each plot shows the relative degree of difficulty for each type of variation (from the representation's point of view; -1 is most difficult). Even though this figure suggests that *Geometric Blur* benefits from more position and scale variations, that is only a by-product of the overall poor performance of this representation and floor effects (see Figure 7.3).

[Riesenhuber and Poggio, 1999b; Serre *et al.*, 2007c] is much less sensitive to position variation than it is to in-depth rotation (i.e. pose) variation.

### 7.3.4 The influence of background

Because background structure and its covariance with object identity are fully known, the testing methods used here can also expose visual representations that rely strongly on these cues. For example, Figures 7.3 and 7.4 show that one of the state-of-the-art visual representations, *Geometric Blur*, performs very poorly on most of our tests, but Figure 7.8 shows that, when we perform the tests on no background, the same representation now performs at a very high level. Taken together, this suggests that this visual representation is seriously impaired by clutter or leans heavily on background features to perform categorization. When natural images are used and background covariance is brought to zero (as in all our testing), this limitation of the representation is revealed. We emphasize that these effects are difficult or impossible to uncover in standard "natural" tests (e.g. Caltech101 or PASCAL VOC), but are very easy to uncover using a synthetic test set approach (see Discussion Section 7.4).

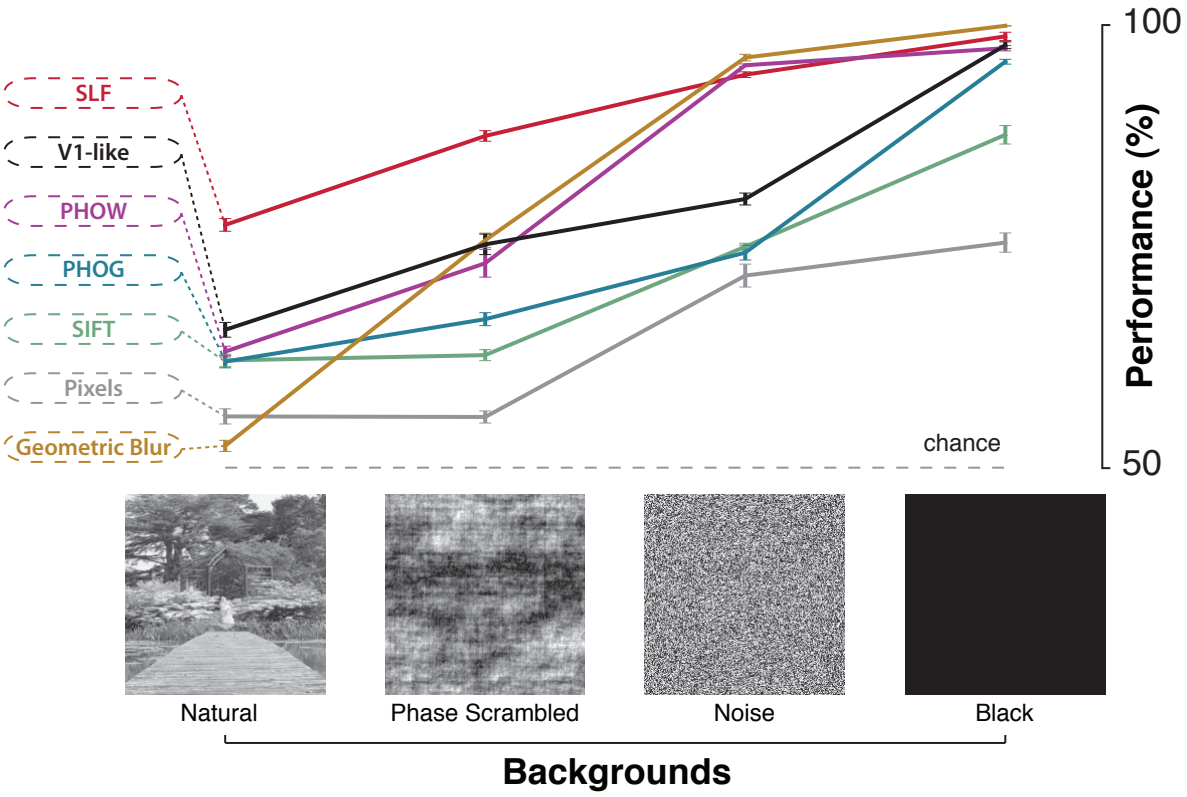



**Figure 7.8: Effect of the background type on each representation's performance on a basic level recognition task ("cars vs. planes") at composite object variation 3 (see Figure 7.3a).** More natural backgrounds (more "clutter") produce a higher degree of difficulty for all representations. Note that, although *Geometric Blur* and *PHOW* look very promising with simple backgrounds, they are particularly disrupted by natural backgrounds that are uncorrelated with object identity.

## 7.4 Discussion

Our testing of invariant object recognition revealed that most of the state-of-the-art representations consistently performed at or below the performance of the *V1-like* baseline representation (which also achieves the highest performance on the Caltech101 set). To the extent that this model represents a "null" baseline that lacks mechanisms to perform invariant recognition, this suggests that other state-of-the-art representations perhaps also rely heavily on view-specific information, or covariation with backgrounds, to achieve their performance. Interestingly, the bio-inspired model *SLF* (an extension

of Serre et al.'s C2 features [Serre *et al.*, 2007c]) stood out in all of our tests, performing consistently better than both baselines, suggesting that it contains computational ideas that are useful for solving invariant object recognition. It remains to be seen how this visual representation and other emerging representations compare to unfettered human performance and the performance of high-level neuronal representations on these tasks.

Our results also revealed that the performance of some representations was highly dependent on the details of the task under test. For example, the performance of *Geometric Blur* descriptors degraded rapidly with the inclusion of background content uncorrelated to object identity, and *PHOW*, while reasonably good at basic-level object categorization tasks, was no better than the pixel representation at the subordinate-level task (face identification).

The synthetic testing approach used here is partly motivated by previous work on photographic approaches like NORB [LeCun *et al.*, 2004]. However, while NORB-like databases are challenging and costly to obtain, the synthetic approach offers the potential to draw on large numbers of objects and generate an essentially infinite number of images with precise control of all key variables at low cost. The approach easily allows exploration of the individual underlying difficulty variables (e.g. position, scale, background, etc.) to better learn from the best ideas of each representation. Because the synthetic approach offers the ability to gradually "ratchet up" the task difficulty (e.g. increasing levels of composite variation in the test) and because only hundreds of images are needed instead of thousands, it can be used to efficiently search for better visual representations (see Chapter 9).

Although it is widely understood that performance evaluation is critical to driving progress (e.g. [Dollár *et al.*, 2009]), such performance evaluation is much easier said than done. Over the last decade, tests based on known ground truth have fallen out of fashion in computer vision [Dickinson, 2009] while the field has rallied around a number of "natural" image test sets (e.g. Caltech101, PASCAL VOC) continue to be used almost exclusively as evidence of progress in solving object recognition [Kavukcuoglu *et al.*, 2009; Jarrett *et al.*, 2009; Gehler and Nowozin, 2009; Van De Sande *et al.*, 2010]. While these test sets are laudable because they encourage systematic comparison of various algorithms, they can also be dangerous when hidden confounds exist in

the sets, or when it is not clear *why* the sets are difficult. Indeed, despite the fact that these representations are highly competitive on large, complex "natural" image sets and the expectation from leaders in the field [LeCun *et al.*, 2008] that many of these representations should be capable of dealing "fairly well" with simpler synthetic invariance tests, our results show that many of these representations are surprisingly weak on these tests, even though these synthetic sets remain trivially easy for human observers (see Chapter 8).

While there is no perfect evaluation tool, we believe that a synthetic testing approach is an important complement to ever-improving photographic-based image sets (e.g. LabelMe [Russell *et al.*, 2008]). More deeply, we share the desire and ultimate goal of evaluating algorithms on "real-world" tasks, and we share the concern that ill-considered "sloppy" synthetic testing approaches may have their own artifacts and, if not carefully considered, may not be predictive of real world performance. But in the world of modern computer graphics, a synthetic testing approach offers a powerful path forward as it can ultimately produce images that are indistinguishable from real-world photographs, yet still have all ground truth variables known and under parametric control.

Future work will be aimed at fully closing any gap between synthetic testing approaches and the idealized notion of "real world" tasks.

## Acknowledgements

We would like to thank the Python community for supporting critical scientific packages with no commercial dependencies and Justin Riley for developing starcluster, a tool we used to run our experiments efficiently on Amazon EC2's infrastructure. We would also like to thank Roman Stanchak for technical assistance, and Jim Mutch for providing his source code.

This work was funded by the NVIDIA Graduate Fellowship, the Singleton Graduate Fellowship, the Amazon AWS Research Grant, the National Institutes of Health (NEI R01-EY014970), the McKnight Endowment Fund for Neuroscience, and The Rowland Institute of Harvard.

Chapter 8

# Human vs. Machine: Comparing Visual Object Recognition Systems on a Level Playing Field*

> "There are two possible outcomes: if the result confirms the hypothesis, then you've made a measurement. If the result is contrary to the hypothesis, then you've made a discovery."
>
> *Enrico Fermi*

It is received wisdom that biological visual systems easily outmatch current artificial systems at complex visual tasks like object recognition. But have the appropriate comparisons been made? Since artificial systems are improving every day, they may surpass human performance some day, thus it is crucial to understand the progress toward reaching that day because success is only one of several necessary requirements for "understanding" visual object recognition.

** This chapter presents preliminary work presented at the Learning Workshop 2010 and the Computational Systems Neuroscience Conference (COSYNE) [Pinto et al., 2010] in collaboration with Najib J. Majaj, Ethan A. Solomon, David D. Cox and James J. DiCarlo.*

**How large (or small) is the difference in performance between current state-of-the-art object recognition systems and the primate visual system?**

As we discussed before, the performance comparison of any two object recognition systems requires a focus on the computational crux of the problem and sets of images that engage it. Although it is widely believed that tolerance ("invariance") to identity-preserving image variation (e.g. variation in object position, scale, pose, illumination) is critical, systematic comparisons of state-of-the-art artificial visual representations almost always rely on "natural" image databases that might fail to probe the ability of a recognition system to solve the invariance problem (see Chapters 4, 5 and 6). Thus, to understand how well current state-of-the-art artificial visual representations perform relative to each other and relative to low-level baseline representations (e.g. retinal-like and V1-like), we tested all of them on a common set of controlled visual recognition tasks that directly engage the "invariance problem" (Chapter 7).

In this chapter, we present an early attempt to test human visual recognition abilities on the same set of invariance controlled tasks. The preliminary data briefly summarized below demonstrate that previous attempts using arbitrary"natural" images may not properly capture the performance of high-level representations (e.g. human observers), and that image variation is also critical in quantifying how far we are from a human-level solution to visual recognition – thus providing tools and guidance for cognitive and systems neuroscience.

## 8.1 Motivation

Serre *et al.* [2007a] started the important effort of comparing human to machine using an animal vs. non-animal categorization task. The authors arranged "natural" grayscale images of animals into four distinct categories (head, close-body, medium-body and far-body) and measured rapid categorization performance of humans as well as supervised classification performance of a specific biologically-inspired feed-forward hierarchical model implementation (HMAX extension including many layers of processing, bypass routes, etc.).

In particular, stimuli were presented to humans for 20 ms followed by a mask,

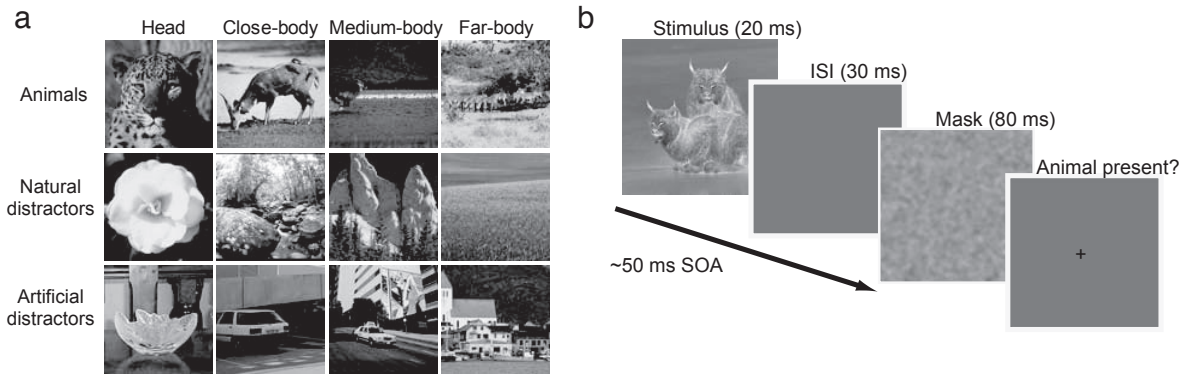



**Figure 8.1: Animal- vs. non-animal-categorization task used in Serre *et al.* [2007a].** (a) The four classes of stimuli. Animal images were *manually* arranged into four groups (150 images each) based on the distance of the animal from the camera: head (close-up), close-body (animal body occupying the whole image), medium-body (animal in scene context), and far-body (small animal or groups of animals). Each of the four classes corresponds to different animal sizes and the task difficulty is *probably* modulated by the different amount of clutter relative to the object size. A set of matching distractors (300 each from natural and artificial scenes) was selected to try to prevent human observers and computational models from relying on low-level cues (artifactual regularities). (b) Schematic of the task. A gray-level image stimulus is flashed for 20 ms, followed by a blank screen for 30 ms for a stimulus onset asynchrony (SOA) of 50 ms, and followed by a mask for 80 ms. Subjects ended the trial with an answer of "yes" or "no" by pressing one of two keys. Figure and caption modified from [Serre *et al.*, 2007a].

subjects then responded "yes" or "no" if they saw an animal (see Figure 8.1). The authors observed that the performance of their machine model was on par with human performance and concluded that the model could predict the level *and* the pattern of performance achieved by humans on this rapid categorization task. More specifically, they showed that difficult stimulus groups, such as "Far-body", were equally difficult for humans and machine, and easy stimulus groups were similarly easy for both (see Figure 8.2).

Of course, the authors do not claim that the object recognition problem is solved in sight of these results. Instead, they merely argue that their HMAX extension, which fits some known physiology and anatomy of the visual cortex, correlates well with humans and exhibits comparable accuracy on this presumably *difficult* rapid categorization task. Nevertheless, given the results presented earlier in Chapters 4, 5 and 6, one might ask (1) if a simpler "baseline" model would also predict the level and pattern of human performance, and (2) if this ostensibly difficult "natural" benchmark is really capturing

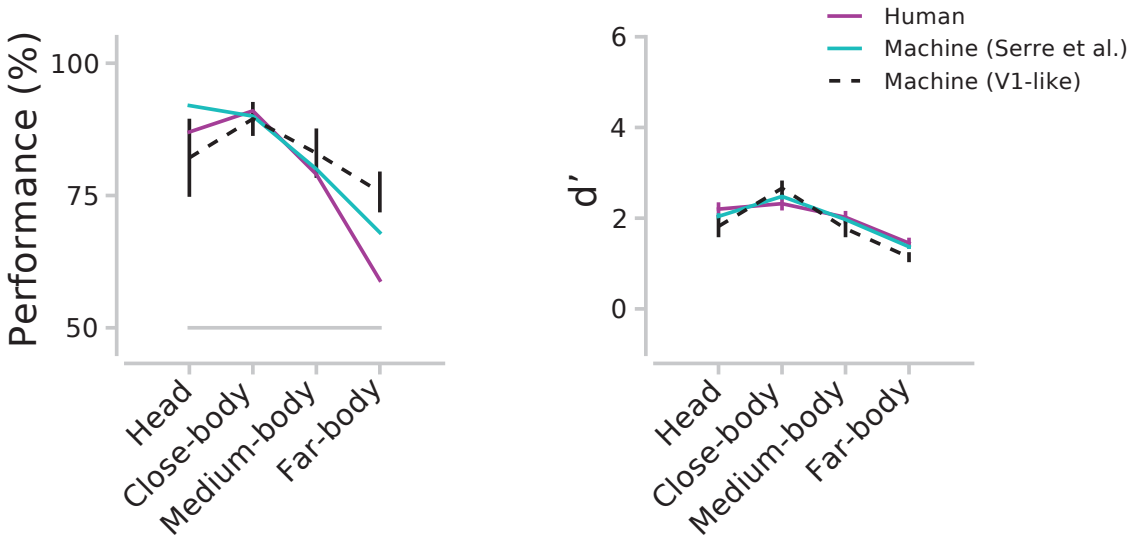



**Figure 8.2: Comparison between machines and human observers.** . Models vs. human-level performance in percent correct (left), and d-prime sensitivity measure (right). Both the complex (Serre et al.) and baseline (V1-like) models exhibit pattern of performance very similar to human observers.

the problem of interest.

To answer these questions, we first test the "V1-like" neuroscientist baseline model on the same animal vs. non-animal categorization task. Then, we measure human performance on carefully controlled visual recognition tasks that include a lot of image variation by design and that have been shown to be particularly difficult for current machine vision systems (see Chapter 7).

## 8.2 Natural Animal vs. Non-Animal Task

Figure 8.2 shows that similar to the multi-layer model of Serre *et al.* [2007a], the single-layer "V1-like" model also matches human accuracy and pattern of performance on the animal vs non-animal task (see Serre *et al.* [2007a] for details regarding the methods).

One possible reason behind this observation is the presence of low-level artifacts in the animal vs. non-animal benchmark as recently illustrated by Landecker *et al.*

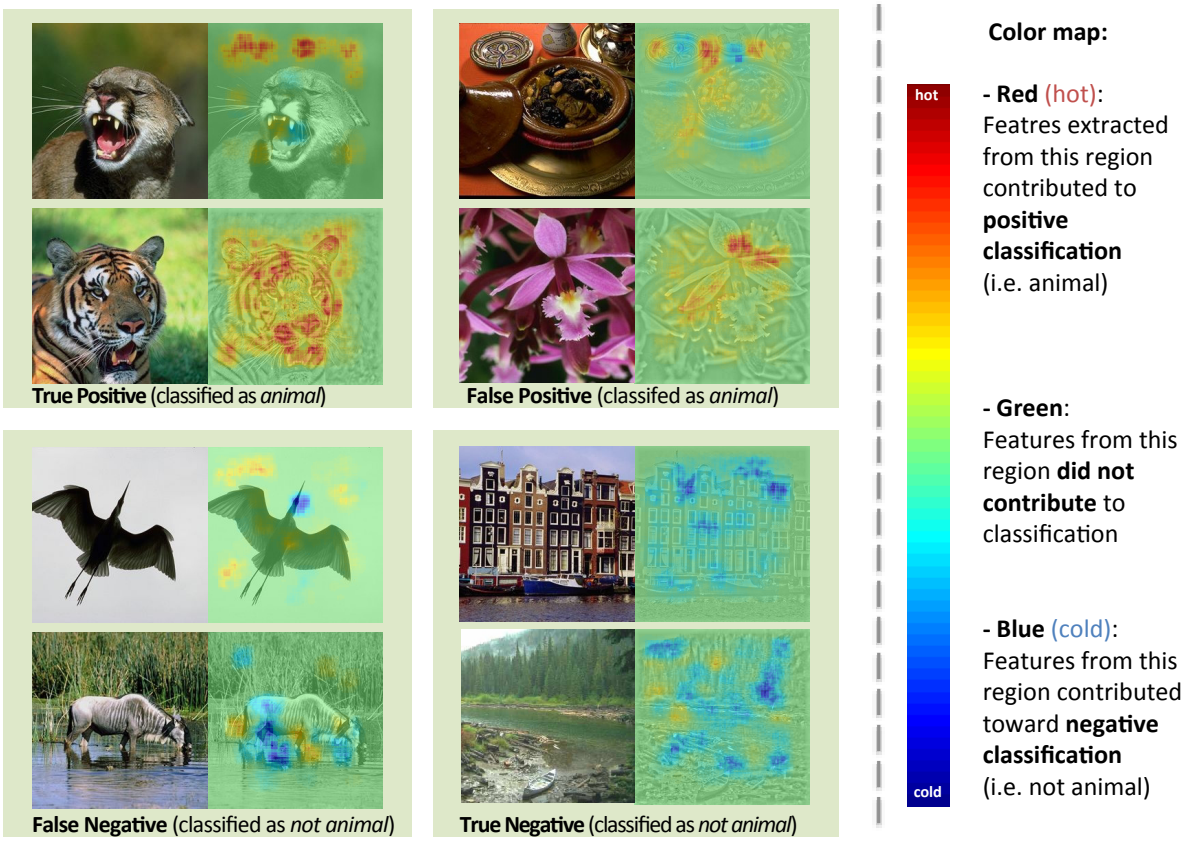



**Figure 8.3: Visualizing the classification weights of the Serre *et al.* [2007a] model on the animal vs. non-animal task.** The first rows of the top-left and bottom-left quadrants show examples where positive feature weights, contributing to an "animal is present" classification, were placed on the background instead of the object of interest (see Figure 8.3), suggesting possible artifactual confounds in the animal vs non-animal image set. Figure modified from [Landecker *et al.*, 2010].

[2010] when they looked at the classification weights of the model used by Serre *et al.* [2007a] on this task. In particular, they showed examples where positive feature weights, contributing to an "animal is present" classification, were placed on the background instead of the object of interest (see Figure 8.3). It is thus possible that the presence of low-level regularities may have biased the pattern of results plotted in Figure 8.2 for both models. Interestingly, this confound, which involves backgrounds and possible artifactual covariation with category label, is similar to the one we discussed in Chapter 7 (see Figure 7.8).

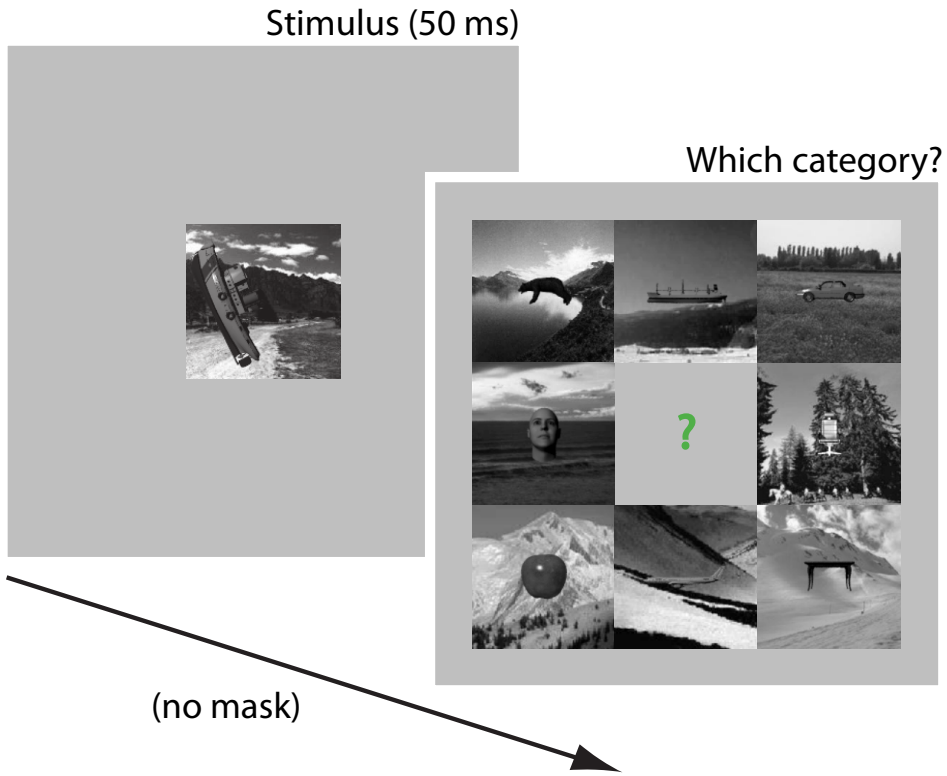



**Figure 8.4: Schematic of the synthetic controlled variation task.** A grayscale image stimulus is flashed for 50 ms, immediately followed by a response screen. Subjects ended the trial by selecting the stimulus category they think they saw.

## 8.3 Controlled Synthetic Recognition Tasks

As in previous chapters, we use a synthetic testing approach that allows direct engagement of the invariance problem, as well as knowledge and control of some of the key parameters that make object recognition challenging. For human observers, we present 600 stimuli, divided into 8 categories and 6 variation levels, for a total of approximately 12 stimuli per condition. Each stimulus is flashed for 50ms with *no mask*. A response screen depicting an exemplar of each category in a canonical view is then shown and the subjects are asked to decide which category they thought they saw (see Figure 8.4). For V1-like, we use the same training/testing protocol as in Chapter 7. Performance is reported as percent correct (chance is at 12.5%) and d-prime sensitivity measure.

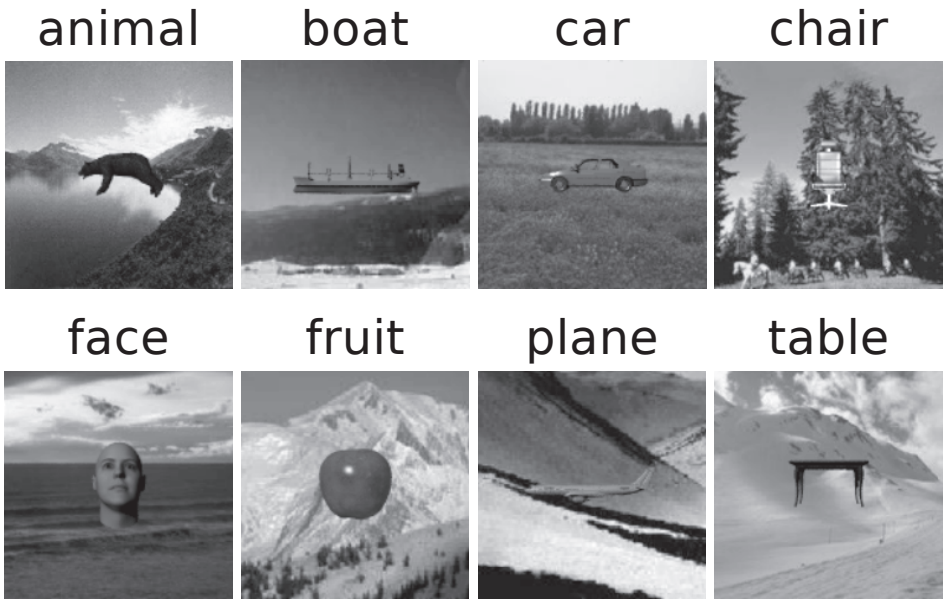



**Figure 8.5: Exemplars from the 8 object categories used in the basic-level recognition tasks.** Stimuli shown in their canonical view on random natural backgrounds.

### 8.3.1 Basic-Level Recognition

In the basic-level recognition experiments, we use 8 categories with 8 exemplars per category (see Figure 8.5). Figure 8.6 shows the results of our basic-level recognition experiments. We observed that the V1-like machine model is only slightly worse than humans at very low variation. However, as the amount of variation increases, machine performance drops *precipitously* while human performance drops *gradually*.

### 8.3.2 Subordinate-Level Recognition

We designed two different identification tasks with two arbitrary level of difficulty: one with car meshes[1], presumably easier since their shapes appear more "different", and the other one with face meshes[2], likely more difficult as they are more "similar" in shape.

Figure 8.7 shows stimuli examples of the 8 car models (alfa, astra, beetle, bmw, bora, celica, clio, z3) in their canonical view on random natural backgrounds. Figure

[1]From Dosch Design (`http://www.doschdesign.com`).
[2]Generated with FaceGen (`http://www.facegen.com`).

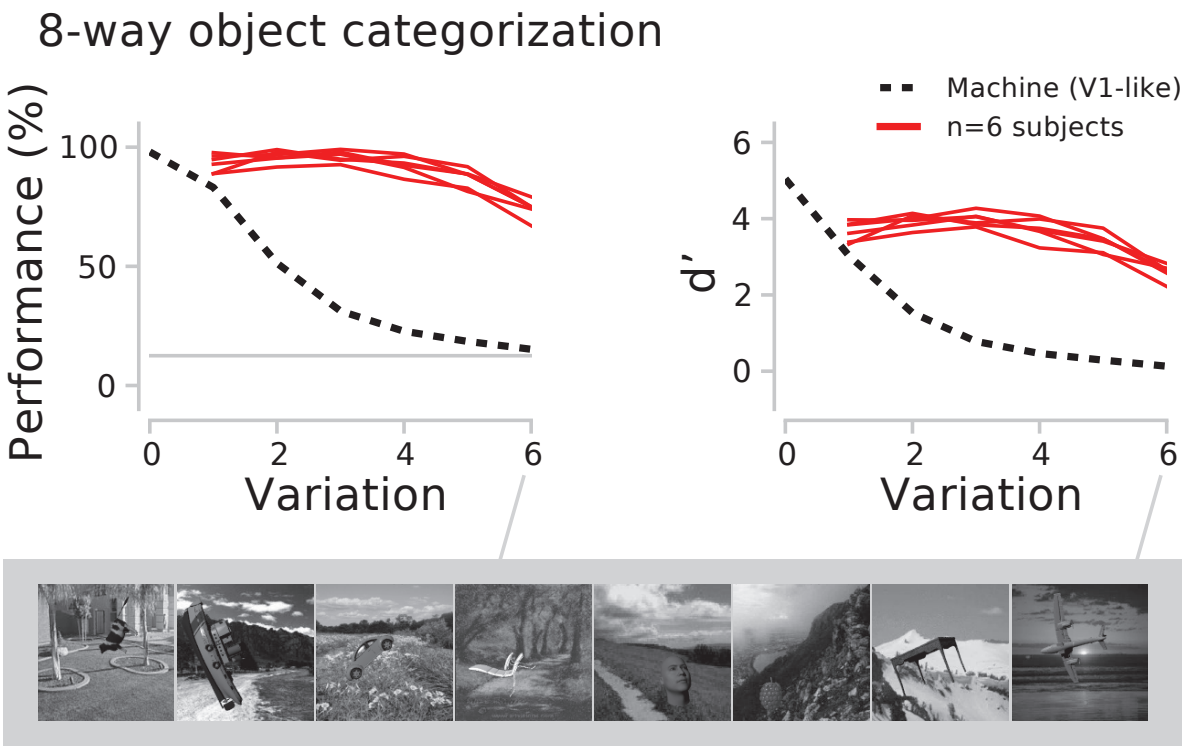



**Figure 8.6: Human vs machine: 8-way object categorization task.** (top-left) accuracy in percent correct, chance is at 12.5%. (top-right) d-prime sensitivity measure. (bottom) Examples of stimuli at variation 6.

8.8 shows the results of the 8-way car identification controlled variation experiments. At low variation, the V1-like machine baseline performs *better* than the inexperienced human subjects, but, as but as the amount of variation increases, machine performance degrades *abruptly* while human performance does not, and is consequently much better at high variation. Interestingly, the most experienced human observer, who has an amount of training similar to the machine on this task, performs significantly better than V1-like at low variations but reaches the same level as inexperienced subjects at the highest variation.

Figure 8.9 shows stimuli examples of the 8 face meshes in their canonical view on random natural backgrounds. Figure 8.10 shows the results of the 8-way face identification controlled variation experiments. Again, at low variation, the machine model performs *better* than the inexperienced human subjects, but, as the amount of varia-

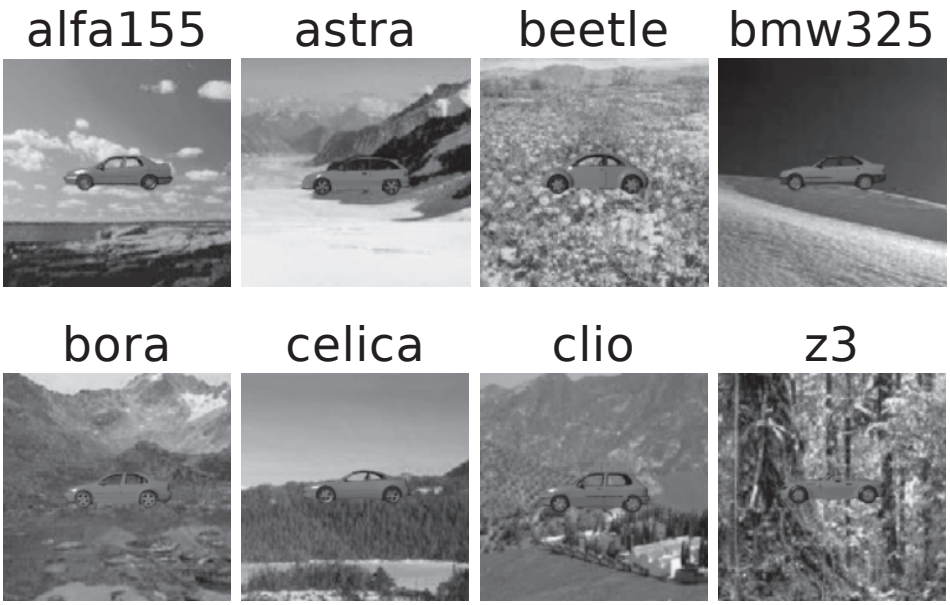



**Figure 8.7: Examples of the 8 specific cars used in one of the subordinate-level recognition tasks.** Stimuli shown in their canonical view on random natural backgrounds.

tion increases, machine performance deteriorates *drastically* while human performance does not. We can see that this task is generally much harder than the previous ones (e.g. some subjects are even at chance level), especially at high variation. This observation seems consistent with our prediction and may reflect the inherent tradeoff between high selectivity to very similar stimuli with different identity (e.g. non-familiar faces) and high tolerance to image variation. The human observer with most training clearly outperforms every contestants on this task, even at high variation.

## 8.4 Summary

To summarize our results, we (loosely) define a "Turing ratio" as the ratio of machine d-prime to human d-prime (to evaluate machine performance on different recognition tasks presented in this study). In Figure 8.11, we draw this "Turing ratio" as a function of variation and we plot its distribution. Regardless of the task, the turing ratio drops consistently with variation. Interestingly, the baseline machine tend to perform better at very low variations, which is presumably the regime at which uncontrolled "natural"

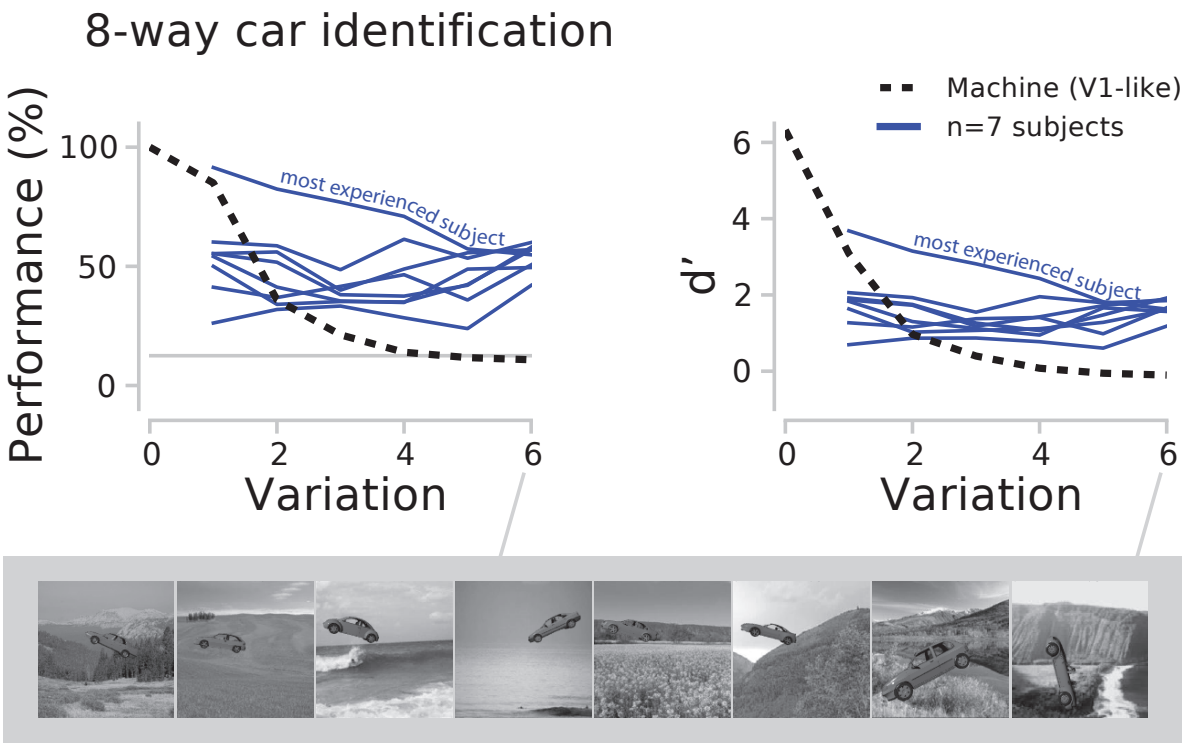



**Figure 8.8: Human vs machine: 8-way car identification task.** (top-left) accuracy in percent correct, chance is at 12.5%. (top-right) d-prime sensitivity measure. (bottom) Examples of stimuli at variation 6. Note that the most experienced human observer is clearly an outlier.

benchmarks "operate". Taken together, our results suggest that variation is also a critical component in exposing the inadequacy of machines relative to humans, and in quantifying progress toward human-level solutions.

While in aggregate, we found that the performance of machines pales in comparison to human performance, humans and computers seem to fail in different and potentially enlightening ways when faced with the problem of invariance. In combination with Chapter 7, we show how our synthetic testing approach can further illuminate the strengths and weaknesses of different visual representations and thus guide progress on invariant object recognition.


**Figure 8.9: Examples of the 8 specific faces used in one of the subordinate-level recognition tasks.** Stimuli shown in their canonical view on random natural backgrounds.

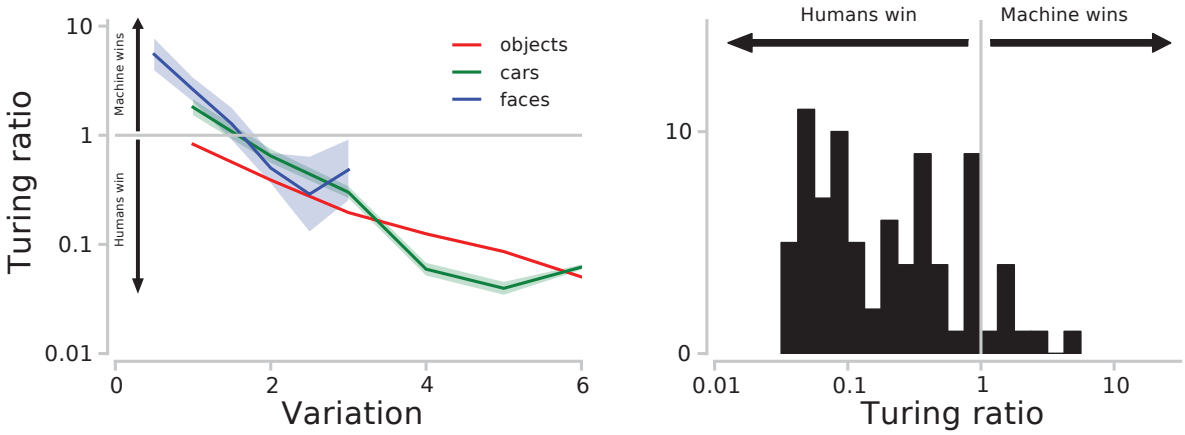



**Figure 8.11: Invariant object recognition "Turing ratio".** Ratio of machine d-prime to human d-prime on all the tasks of this study. (left) "Turing ratio" as a function of variation. (right) Distribution of "Turing ratio" values across all the tasks.

# Acknowledgments

This work was funded by the NVIDIA Graduate Fellowship, the Singleton Graduate Fellowship, the Amazon AWS Research Grant, the National Institutes of Health (NEI

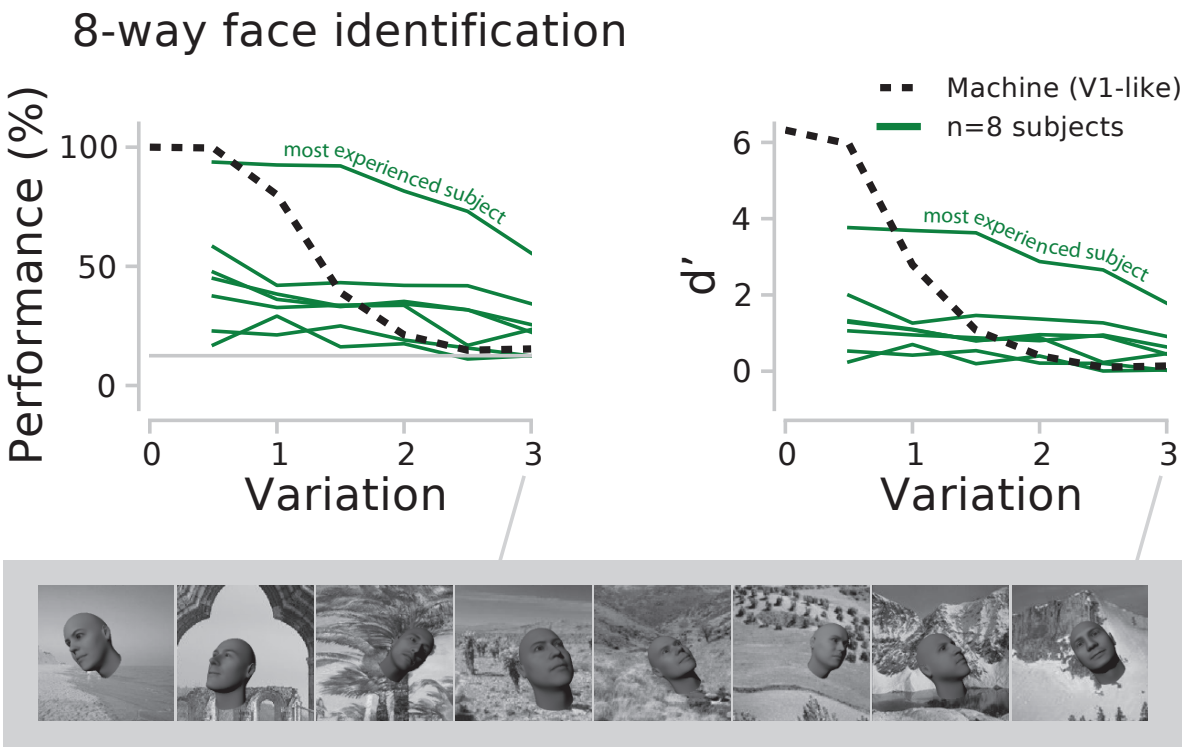



**Figure 8.10: Human vs machine: 8-way face identification task.** (top-left) accuracy in percent correct, chance is at 12.5%. (top-right) d-prime sensitivity measure. (bottom) Examples of stimuli at variation 3. Note the human observer with most training is clearly an outlier.

R01-EY014970), the McKnight Endowment Fund for Neuroscience, and The Rowland Institute of Harvard.

# Part III

# High-Throughput Solution Discovery

Chapter 9

# A High-Throughput Screening Approach to Discovering Good Forms of Biologically-Inspired Visual Representation*

"If you want to have good ideas you must have many ideas. Most of them will be wrong, and what you have to learn is which ones to throw away."

*Linus Pauling*

"The more comfortable we become with being stupid, the deeper we will wade into the unknown and the more likely we are to make big discoveries."

*[Schwartz, 2008]*

*This chapter is modified from a study published in the open-access journal PLoS Computational Biology in collaboration with David Doukhan, James J. DiCarlo and David D. Cox [Pinto et al., 2009a]. This work also appeared in* Science Editor's Choice *[Chin, 2010].*

While many models of biological object recognition share a common set of "broad-stroke" properties, the performance of any one model depends strongly on the choice of parameters in a particular instantiation of that model – e.g. the number of units per layer, the size of pooling kernels, exponents in normalization operations, etc. Since the number of such parameters (explicit or implicit) is typically large, and the computational cost of evaluating one particular parameter set is high, the space of possible model instantiations goes largely unexplored. Thus, when a model fails to approach the abilities of biological visual systems, we are left uncertain whether this failure is because we are missing a fundamental idea, or because the correct "parts" have not been tuned correctly, assembled at sufficient scale, or provided with enough training. Here, we present a high-throughput approach to the exploration of such parameter sets, leveraging recent advances in stream processing hardware (high-end NVIDIA graphic cards and the PlayStation 3's IBM Cell Processor). In analogy to high-throughput screening approaches in molecular biology and genetics, we explored thousands of potential network architectures and parameter instantiations, screening those that show promising object recognition performance for further analysis. We show that this approach can yield significant, reproducible gains in performance across an array of basic object recognition tasks, consistently outperforming a variety of state-of-the-art purpose-built vision systems from the literature. As the scale of available computational power continues to expand, we argue that this approach has the potential to greatly accelerate progress in both artificial vision and our understanding of the computational underpinning of biological vision.

## 9.1 Introduction

The study of biological vision and the creation of artificial vision systems are naturally intertwined – exploration of the neuronal substrates of visual processing provides clues and inspiration for artificial systems, and artificial systems, in turn, serve as important generators of new ideas and working hypotheses. The results of this synergy have been powerful: in addition to providing important theoretical frameworks for empirical investigations (e.g. [Fukushima, 1980; Hinton, 1989; Haykin, 1994; Riesenhuber and

Poggio, 1999b; Rolls and Milward, 2000; Rolls and Deco, 2002]), biologically-inspired models are routinely among the highest-performing artificial vision systems in practical tests of object and face recognition [LeCun *et al.*, 2004; Serre *et al.*, 2007c; Mutch and Lowe, 2008] (see also Chapters 4, 5 and 6).

However, while neuroscience has provided inspiration for some of the "broad-stroke" properties of the visual system, much is still unknown. Even for those qualitative properties that most biologically-inspired models share, experimental data currently provide little constraint on their key parameters. As a result, even the most faithfully biomimetic vision models necessarily represent just one of many possible realizations of a collection of computational ideas.

Truly evaluating the set of biologically-inspired computational ideas is difficult, since the performance of a model depends strongly on its particular instantiation – the size of the pooling kernels, the number of units per layer, exponents in normalization operations, etc. Because the number of such parameters (explicit or implicit) is typically large, and the computational cost of evaluating one particular model is high, it is difficult to adequately explore the space of possible model instantiations. At the same time, there is no guarantee that even the "correct" set of principles will work when instantiated on a small scale (in terms of dimensionality, amount of training, etc.). Thus, when a model fails to approach the abilities of biological visual systems, we cannot tell if this is because the ideas are wrong, or they are simply not put together correctly or on a large enough scale.

As a result of these factors, the availability of computational resources plays a critical role in shaping what kinds of computational investigations are possible. Traditionally, this bound has grown according to Moore's Law [Moore Gordon, 1965], however, recently, advances in highly-parallel graphics processing hardware (such as high-end NVIDIA graphics cards, and the PlayStation 3's IBM Cell processor) have disrupted this status quo for some classes of computational problems. In particular, this new class of modern graphics processing hardware has enabled over hundred-fold speed-ups in some of the key computations that most biologically-inspired visual models share in common. As is already occurring in other scientific fields [Yang, 2004; Kurzak *et al.*, 2008], the large quantitative performance improvements offered by this new class of

hardware hold the potential to effect qualitative changes in how science is done.

In the present work, we take advantage of these recent advances in graphics processing hardware [Owens *et al.*, 2007, 2008] to more expansively explore the range of biologically-inspired models – including models of larger, more realistic scale. In analogy to high-throughput screening approaches in molecular biology and genetics, we generated and trained thousands of potential network architectures and parameter instantiations, and we "screened" the visual representations produced by these models using tasks that engage the core problem of object recognition – tolerance to image variation [DiCarlo and Cox, 2007] (see Chapters 4, 5, 6, 7 and 8). From these candidate models, the most promising were selected for further analysis.

We show that this large-scale screening approach can yield significant, reproducible gains in performance in a variety of basic object recognitions tasks and that it holds the promise of offering insight into which computational ideas are most important for achieving this performance. Critically, such insights can then be fed back into the design of candidate models (constraining the search space and suggesting additional model features), further guiding evolutionary progress. As the scale of available computational power continues to expand, high-throughput exploration of ideas in computational vision holds great potential both for accelerating progress in artificial vision, and for generating new, experimentally-testable hypotheses for the study of biological vision.

## 9.2 Methods

### 9.2.1 A Family of Candidate Models

In order to generate a large number of candidate model instantiations, it is necessary to parametrize the family of all possible models that will be considered. A schematic of the overall architecture of this model family, and some of its parameters, is shown in Figure 9.2. The parametrization of this family of models was designed to be as inclusive as possible – that is, the set of model operations and parameters was chosen so that the family of possible models would encompass (as special cases) many of the biologically-inspired models already described in the extant literature (e.g. [Fukushima,

1980; Hinton, 1989; Haykin, 1994; Riesenhuber and Poggio, 1999b; Rolls and Milward, 2000; Rolls and Deco, 2002; LeCun *et al.*, 2004; Serre *et al.*, 2007c]). For instance, the full model includes an optional "trace" term, which allows learning behavior akin to that described in previous work (e.g. [Földiak, 1991; Wallis *et al.*, 1993; Wallis and Rolls, 1997; Rolls and Milward, 2000; Stringer and Rolls, 2002; Elliffe *et al.*, 2002; Einhauser *et al.*, 2002; Einhäuser *et al.*, 2005; Spratling, 2005; Sprekeler *et al.*, 2007; Franzius *et al.*, 2008]). While some of the variation within this family of possible models might best be described as variation in parameter tuning within a fixed model architecture, many parameters produce significant architectural changes in the model (e.g. number of filters in each layer). The primary purpose of this report is to present an overarching approach to high-throughput screening. While precise choices of parameters and parameter ranges are clearly important, one could change which parameters were explored, and over what ranges, without disrupting the integrity of the overarching approach. An exhaustive description of specific model parameters used here is included in the Supplemental Text S1, and is briefly described next.

Model parameters were organized into four basic groups. The first group of parameters controlled structural properties of the system, such as the number of filters in each layer and their sizes. The second group of parameters controlled the properties of nonlinearities within each layer, such as divisive normalization coefficients and activation functions. The third group of parameters controlled how the models learned filter weights in response to video inputs during an *Unsupervised Learning Phase* (this class includes parameters such as learning rate, trace factors, etc.; see *Phase 2: Unsupervised Learning* below). A final set of parameters controlled details of how the resulting representation vectors are classified during screening and validation (e.g. parameters of dimensionality reduction, classification parameters, etc.). For the purposes of the work presented here, this class of classification-related parameters was held constant for all analyzes below. Briefly, the output values of the final model layer corresponding to each test example image were "unrolled" into a vector, their dimensionality was reduced using Principal Component Analysis (PCA) keeping as many dimensions as there were data points in the training set, and labeled examples were used to train a linear Support Vector Machine (SVM).

Each model consisted of three layers, with each layer consisting of a "stack" of between 16 and 256 linear filters that were applied at each position to a region of the layer below. At each stage, the output of each unit was normalized by the activity of its neighbors within a parametrically-defined radius. Unit outputs were also subject to parametrized threshold and saturation functions, and the output of a given layer could be spatially resampled before being given to the next layer as input. Filter kernels within each stack within each layer were initialized to random starting values, and learned their weights during the *Unsupervised Learning Phase* (see below, see Supplemental Text S1). Briefly, during this phase, under parametric control, a "winning" filter or filters were selected for each input patch, and the kernel of these filters was adapted to more closely resemble that patch, achieving a form of online non-parametric density estimation. Building upon recent findings from visual neuroscience [Yao and Dan, 2001; Cox *et al.*, 2005; Li and DiCarlo, 2008, 2010], unsupervised learning could also be biased by temporal factors, such that filters that "won" in previous frames were biased to win again (see Supplemental Text S1 for details).

It should be noted that while the parameter set describing the model family is large, it is not without constraints. While our model family includes a wide variety of feed-forward architectures with local intrinsic processing (normalization), we have not yet included long-range feedback mechanisms (e.g. layer to layer). While such mechanisms may very well turn out to be critically important for achieving the performance of natural visual systems, the intent of the current work is to present a framework to approach the problem. Other parameters and mechanisms could be added to this framework, without loss of generality. Indeed, the addition of new mechanisms and refinement of existing ones is a major area for future research (see Discussion Section 9.4).

### 9.2.2 Parallel Computing Using Commodity Graphics Hardware

> "Science is driven more by new tools than new ideas."
>
> *Freeman Dyson*

While details of the implementation of our model class are not essential to the theoretical implications of our approach, attention must nonetheless be paid to speed in order to ensure the practical tractability, since the models used here are large (i.e. they have many units), and because the space of possible models is enormous. Fortunately, the computations underlying our particular family of candidate models are intrinsically parallel at a number of levels. In addition to coarse-grain parallelism at the level of individual model instantiations (e.g. multiple models can be evaluated at the same time) and video frames (e.g. feed-forward processing can be done in parallel on multiple frames at once), there is a high degree of fine-grained parallelism in the processing of each individual frame. For instance, when a filter kernel is applied to an image, the same filter is applied to many regions of the image, and many filters are applied to each region of the image, and these operations are largely independent. The large number of arithmetic operations per region of image also results in high arithmetic intensity (numbers of arithmetic operations per memory fetch), which is desirable for high-performance computing, since memory accesses are typically several orders of magnitude less efficient than arithmetic operations (when arithmetic intensity is high, caching of fetched results leads to better utilization of a processor's compute resources). These considerations are especially important for making use of modern graphics hardware (such as the Cell processor and GPUs) where many processors are available. Highly-optimized implementations of core operations (e.g. linear filtering, local normalization) were created for both the IBM Cell Processor (PlayStation 3), and for NVIDIA graphics processing units (GPUs) using the Tesla Architecture and the CUDA programming model [Lindholm *et al.*, 2008]. These implementations achieve highly significant speed-ups relative to conventional CPU-based implementations (see Table 9.1 and Supplemental Figure S1). High-level "outer loop" coordination of these highly optimized operations was ac-

complished using the Python programming language (e.g. using PyCUDA [1] [Klöckner *et al.*, 2009]), allowing for a favorable balance between ease of programming and raw speed (see Supplemental Text S2). In principle, all of the analyzes presented here could have been performed using traditional computational hardware; however, the cost (in terms of time and/or money) of doing so with current CPU hardware is prohibitive.

| | CPUs | | | GPUs | | | |
|---|---|---|---|---|---|---|---|
| **Hardware** | | | | | | | |
| Manufacturer | Intel | Intel | Intel | NVIDIA | Sony, IBM, Toshiba | NVIDIA | NVIDIA |
| Model | Q9450 | Q9450 | Q9450 | 7900 GTX | PlayStation 3 | 8800 GTX | GTX 280 |
| # cores used | 1 | 4 | 4 | 4x96 | 2+6 | 4x128 | 4x240 |
| Implementation | MATLAB | MATLAB | SSE2 | Cg | Cell SDK | CUDA | CUDA |
| Year | 2008 | 2008 | 2008 | 2006 | 2007 | 2007 | 2008 |
| **Performance / Cost** | | | | | | | |
| Full System Cost (approx.) | $1,500** | $2,700** | $1,000 | $3,000* | $400 | $3,000* | $3,000* |
| Relative Speedup | **1x** | **4x** | **80x** | **544x** | **222x** | **1544x** | **2712x** |
| Relative Perf. / $ | **1x** | **2x** | **120x** | **272x** | **833x** | **772x** | **1356x** |

**Table 9.1: Performance and Cost of various CPU and GPU implementations of a Critical Component of Our Model Family.** Our implemented performance speed-ups for a key filtering operation in our biologically-inspired model implementation. Performance and price are shown across a collection of different GPUs, relative to a commonly used MATLAB CPU-based implementation (using a single CPU core with the *filter2* function, which is coded in C++). We contrast this standard implementation with a multi-core MATLAB version, a highly-optimized C/SSE2 multi-core implementation on the same CPU, and highly-optimized GPU implementations. We have implemented speedups of over thousands of times with GPUs, resulting in qualitative changes in what kinds of model investigations are possible. More technical details and a throughout discussion of the computational framework enabling these speedups can be found in Supplemental Figure S1 and Supplemental Text S2.
* These costs are based on multi-GPU systems containing four GPUs in addition to the quad-core CPU (Q9450).
** These costs include both the hardware and MATLAB yearly licenses (based on an academic discount pricing, for one year).

Table 9.1 shows the relative speedup and performance / cost of each implementation (IBM Cell on Sony's PlayStation 3 and several NVIDIA GPUs) relative to traditional MATLAB and multi-threaded C code for the linear filtering operation (more details such as the raw floating point performance can be found in the Supplemental Figure

[1] `http://mathema.tician.de/software/pycuda`

S1). This operation is not only a key component of the candidate model family (see below) but it's also the most computationally demanding, reaching up to 94% of the total processing time (for the PlayStation 3 implementation), depending on model parameters (average fraction is 28%). The use of commodity graphics hardware affords orders-of-magnitude increases in performance. In particular, it should be noted that the data presented in this work took approximately one week to generate using our PlayStation 3-based implementation (222x speedup with one system) on a cluster of 23 machines. We estimate that producing the same results at the same cost using a conventional MATLAB implementation would have taken more than two years (see Supplemental Figure 9.9).

### 9.2.3 Screening for Good Forms of Representation

Our approach is to sample a large number of model instantiations, using a well-chosen "screening" task to find promising architectures and parameter ranges within the model family. Our approach to this search was divided into four phases (see Figure 9.1): Candidate Model Generation, Unsupervised Learning, Screening, and Validation/Analysis of high-performing models.

#### Phase 1: Candidate Model Generation

Candidate model parameter sets were randomly sampled with a uniform distribution from the full space of possible models in the family considered here (see Figure 9.2 and Figure S2 for a schematic diagram of the models, and Supplemental Materials for an exhaustive description of model parameters and value ranges that were explored; Supplemental Text S1).

#### Phase 2: Unsupervised Learning

All models were subjected to a period of unsupervised learning, during which filter kernels were adapted to spatiotemporal statistics of a stream of input images. Since the family of models considered here includes features designed to take advantage of the temporal statistics of natural inputs (see Supplementary Methods), models were learned

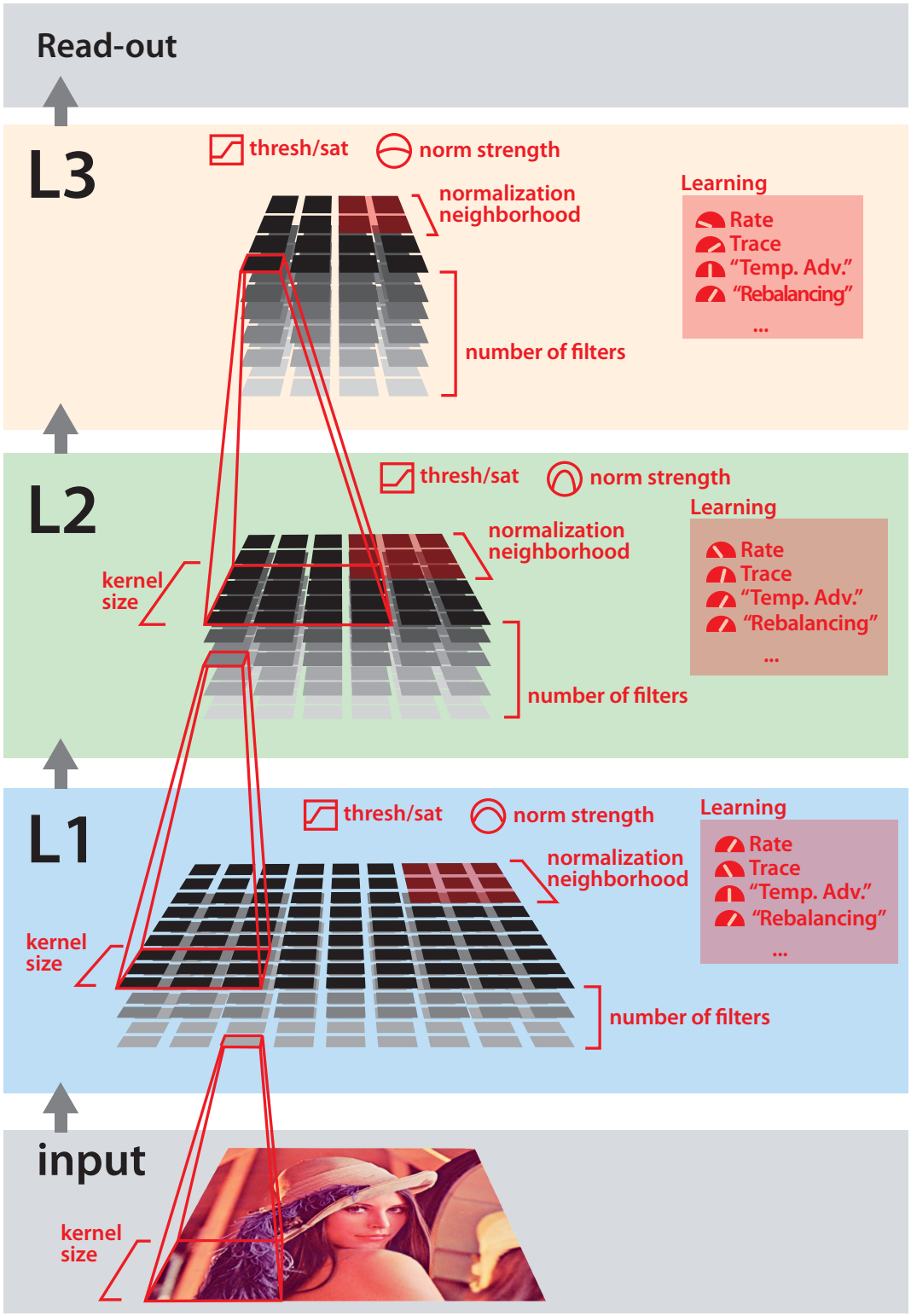



**Figure 9.2: A schematic diagram of the system architecture of the family of models considered.** The system consists of three feed-forward filtering layers, with the filters in each layer being applied across the previous layer. Red colored labels indicate a selection of configurable parameters (only a subset of parameters are shown).

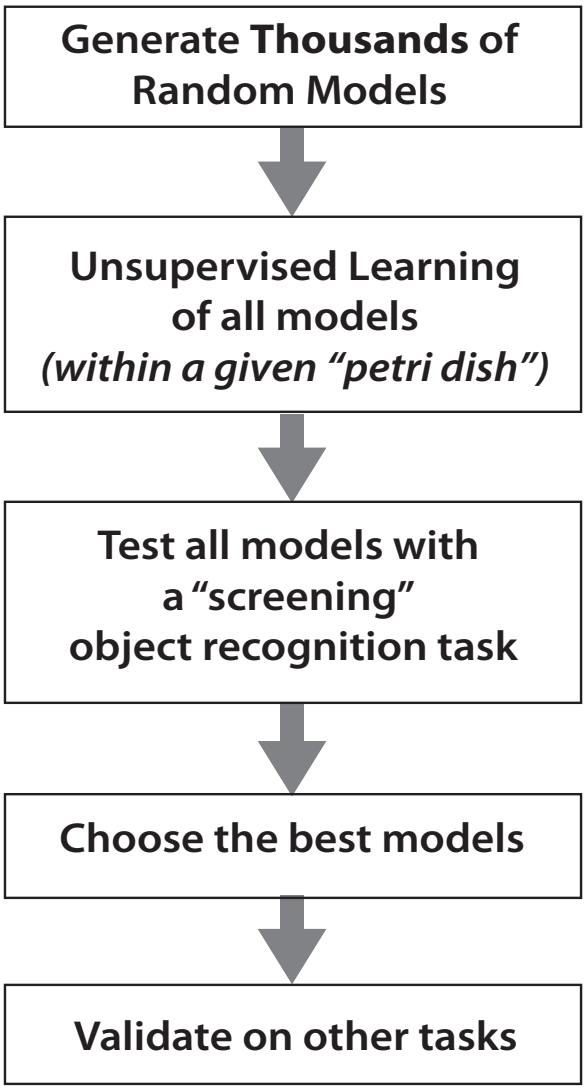



**Figure 9.1: Experimental flow** The experiments described here consist of five phases. (a) First, a large collection of model instantiations are generated with randomly selected parameter values. (b) Each of these models then undergoes an unsupervised learning period, during which its filter kernels are adapted to spatio-temporal statistics of the video inputs, using a learning algorithm that is influenced by the particular parameter instantiation of that model. After the *Unsupervised Learning Phase* is complete, filter kernels are fixed, and (c) each model is subjected to a screening object recognition test, where labeled images are represented using each model instantiation, and these re-represented images are used to train an SVM to perform a simple two-class discrimination task. Performance of each candidate model is assessed using a standard cross-validation procedure. (d) From all of the model instantiations, the best are selected for further analysis. (e) Finally, these models are tested on other object recognition tasks.

using video data. In the current version of our family of models, learning influenced the form of the linear kernels of units at each layer of the hierarchy, but did not influence any other parameters of the model.

We used three video sets for unsupervised learning: "Cars and Planes", "Boats", and "Law and Order". The "Law and Order" video set consisted of clips from the television program of the same name (© NBC Universal), taken from DVDs, with clips selected to avoid the inclusion of text subtitles. These clips included a variety of objects moving through the frame, including characters' bodies and faces. Examples from these clips are shown in Figure 9.3(a).

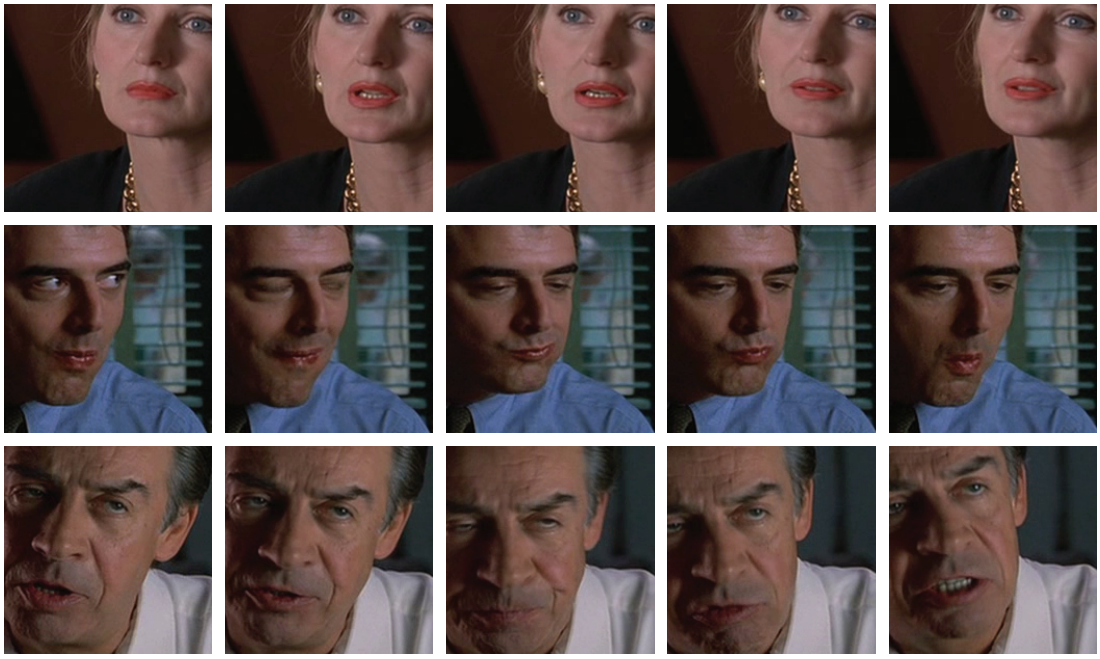
The "Cars and Planes" and "Boats" video sets consisted of 3D ray-traced cars, planes and boats undergoing 6-degree-of-freedom view transformations (roughly speaking, "tumbling" through space). These same 3D models were also used in Chapters 4, 7 and 8. Video clips were generated where an object would appear for approximately 300 frames, performing a random walk in position (3 degrees of freedom) and rotation (3 degrees of freedom) for a total of 15,000 frames. Examples are shown in Figures 9.3(b) and 9.3(c).

For the sake of convenience, we refer to each unsupervised learning video set as a "petri dish," carrying forward the analogy to high-throughput screening from biology. In the results presented here, 2,500 model instantiations were independently generated in each "petri dish" by randomly drawing parameter values from a uniform distribution (a total of 7,500 models were trained). Examples of filter kernels resulting from this unsupervised learning procedure are shown in Supplemental Figures S3, S4, S5 and S6.

After the end of the *Unsupervised Learning Phase*, the linear filter kernels were not modified further, and the resulting model was treated as a fixed transformation (e.g. a static image is entered as input, and a vector of responses from the units of the final layer is outputted).

### Phase 3: Screening

Following the *Unsupervised Learning Phase*, each "petri dish" was subjected to a *Screening Phase* to determine which model instantiations produced image representations that are well-suited for performing invariant object recognition tasks.



(a)

(b)

(c)

**Figure 9.3: Example video frames used as input during the *Unsupervised Learning Phase*.** (a) Short video clips taken from the television series "Law and Order". (b) Sequences of a rendered car undergoing a random walk through the possible range of rigid body movements. (c) A similar random walk with a rendered boat.

During the *Screening Phase*, individual static images were supplied as input to each model, and the vector of responses from the units of its final layer were taken as that model's "representation" of the image. The labeled, "re-represented" images were then reduced in dimensionality by PCA and taken as inputs (training examples) for a classifier (in our case, a linear SVM).

We used a simple "Cars vs. Planes" synthetic object recognition test as a screening task (see Chapters 4 and 7 for details). In this task, 3D models from two categories (cars and planes), were rendered across a wide range of variation in position, scale, view, and background. The rendered grayscale images (200 by 200 pixels) were provided as input to each model, and a classifier was trained to distinguish car images from plane images (150 training images per category). Performance of each model was then tested on a new set of unlabeled re-represented car and plane images (150 testing images per category). This recognition test has the benefit of being relatively quick to evaluate (because it only contains two classes), while at the same time having previous empirical grounding as a challenging object recognition test due to the large amount of position, scale, view, and background variation (see Figure 9.4a).

### Phase 4: Validation

The best models selected during the *Screening Phase* were submitted to validation tests using other image sets, to determine if the representations generated by the models were useful beyond the immediate screening task. For the present work, four validation sets were used:

1. a new set of rendered cars and planes (generated by the same random process that generated the screening set, but with different specific exemplars),

2. a set of rendered boats and animals,

3. a set of rendered images of two synthetic faces (one male, one female, see Chapters 5 and 6),

4. a modified subset of the standard MultiPIE face recognition test set ([Gross *et al.*, 2007]; here dubbed the "MultiPIE Hybrid" set).

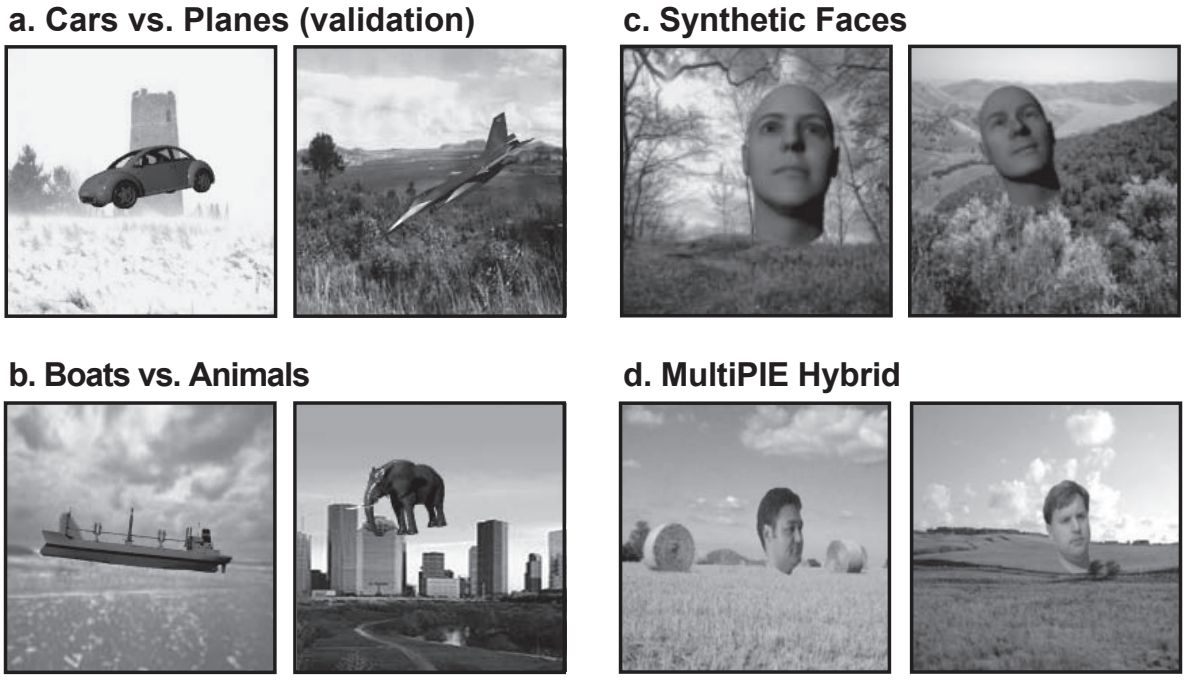



**Figure 9.4: Examples of images from the validation test sets.** (a) A new set of rendered cars and planes composited onto random natural backgrounds. (b) Rendered boats and animals. (c) Rendered female and male faces. (d) A subset of the MultiPIE face test set [Gross *et al.*, 2007] with the faces manually removed from the background, and composited onto random image backgrounds, with additional variation in position, scale, and planar rotation added.

In the case of the rendered sets (sets 1-3), as with the screening set, the objects were rendered across a wide range of views, positions, and scales.

For the "MultiPIE hybrid" set, 50 images each of two individuals from the standard MultiPIE set were randomly selected from the full range of camera angles, lighting, expressions, and sessions included in the MultiPIE set. These faces were manually removed from their backgrounds and were further transformed in scale, position, planar rotation and were composited onto random natural backgrounds. Examples of the resulting images are shown in Figure 9.4.

For all sets (as with the screening set) classifiers were trained with labeled examples to perform a two-choice task (i.e. Cars vs. Planes, Boats vs. Animals, Face 1 vs. Face 2), and were subsequently tested with images not included in the training set.

While a number of standardized "natural" object and face recognition test sets exist [Olivetti Research Laboratory, 1994; Yale Center for Computational Vision and Control, 1997; Computer Vision Lab at the University of Ljubljana, 1999; Martinez and Benavente, 1998; Fei-Fei *et al.*, 2004a; Griffin *et al.*, 2007; Huang *et al.*, 2007], we made a deliberate choice not to use these sets. Chapters 4, 5, 6 and 7, as well as previous investigations [Shamir, 2008; Ponce *et al.*, 2006], have raised concerns with many of these sets, calling into question whether they appropriately capture the problem of interest. As a result, we chose to focus here on image sets that include substantial image variation by design, be they synthetic (as in our rendered set) or natural (as in the MultiPIE Hybrid set) in origin.

### 9.2.4 Performance Comparison with Other Algorithms

#### "V1-like" Baseline

Since object recognition performance measures are impossible to interpret in a vacuum, we used a simple *V1-like* model to serve as one baseline against which model performance can be compared. This *V1-like* model was taken, without modification, from Chapter 4, and was shown previously to match or exceed the performance of a variety of purpose-built vision systems on the popular (but, we argue, flawed as a test of invariant object recognition) Caltech101 object recognition set and a wide variety of

standard face recognition sets (ORL, Yale, CVL, AR, and Labeled Faces in the Wild (Chapters 5 and 6). Importantly, this model is based on only a first-order description of the first stage of visual processing in the brain, and it contains no mechanisms that should allow it to tolerate the substantial image variation that makes object recognition hard in the first place [DiCarlo and Cox, 2007]. Here, this model serves as a lower bound on the amount of trivial regularity that exists in the test set. To be considered promising object recognition systems, models should at least exceed the performance of the *V1-like* model.

### Comparison with State-of-the-art Algorithms

To facilitate comparison with other models in the literature, we obtained code for, or re-implemented five "state-of-the-art" object recognition algorithms from the extant literature: "Pyramid Histogram of Oriented Gradients" (PHOG) [Dalal and Triggs, 2005; Bosch *et al.*, 2007; Varma and Ray, 2007], "Pyramid Histogram of Words" (PHOW) (also known as the Spatial Pyramid [Bosch *et al.*, 2007; Varma and Ray, 2007; Lazebnik *et al.*, 2009]), the "Geometric Blur" shape descriptors [Berg and Malik, 2001; Zhang *et al.*, 2006], the descriptors from the "Scale Invariant Feature Transformation" (SIFT) [Lowe, 2004], and the "Sparse Localized Features" (SLF) features of Mutch and Lowe [2008] (a sparse extension of the C2 features from the Serre et al. HMAX model [Serre *et al.*, 2007c]). In all cases, we were able to reproduce or exceed the authors' reported performance for each system on the Caltech101 test set, which served as a sanity check that we had correctly implemented and used each algorithm as intended by its creators (see Figure 7.1 in Chapter 7).

Each algorithm was applied using an identical testing protocol to our validation sets. In cases where an algorithm from the literature dictated that filters be optimized relative to each training set (e.g. PHOW and SLF), we remained faithful to the authors' published descriptions and allowed this optimization, resulting in a different individually tailored model for each validation set. This was done even though our own high-throughput-derived models were not allowed such per-set optimizations (i.e. the same representation was used for all validation sets), and could therefore theoretically be "handicapped" relative to the state-of-the-art models.

## 9.3 Results

### 9.3.1 Object Recognition Performance

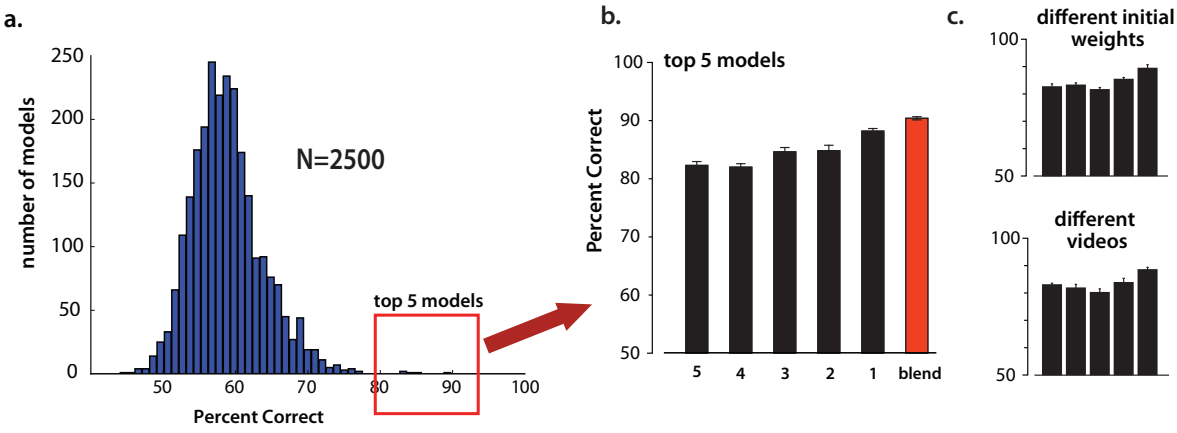



**Figure 9.5: High-throughput screening in the "Law and Order" Petri Dish.** (a) Histogram of the performance of 2,500 models on the "Cars vs. Planes" screening task (averaged over 10 random splits; error bars represent standard error of the mean). The top five performing models were selected for further analysis. (b) Performance of the top five models (1-5), and the performance achieved by averaging the five SVM kernels (red bar labeled "blend"). (c) Performance of the top five models (1-5) when trained with a different random initialization of filter weights (top) or with a different set of video clips taken from the "Law and Order" television program (bottom).

As a first exploration of our high-throughput approach, we generated 7,500 model instantiations, in three groups of 2,500, with each group corresponding to a different class of unsupervised learning videos ("petri dishes"; see Methods Section 9.2). During the *Screening Phase*, we used the "Cars vs. Planes" object discrimination task (Chapter 4 to assess the performance of each model, and the most promising five models from each set of 2,500 models was submitted to further analysis. The raw computation required to generate, train and screen these 7,500 models was completed in approximately one week, using 23 PlayStation 3 systems. Results for models trained with the "Law and Order" petri dish during the *Unsupervised Learning Phase* are shown in Figure 9.5a. As expected, the population of randomly-generated models exhibited a broad distribution of performance on the screening task, ranging from chance performance (50%) to better than 80% correct. Figure 9.5b shows the performance of the best five models drawn from

the pool of 2,500 models in the "Law and Order" petri dish. These models consistently outperformed the *V1-like* model baseline (Figure 9.6), and this performance was roughly maintained even when the model was retrained with a different video set (e.g. a different clip from Law and Order), or with a different random initialization of the filter kernel weights (Figure 9.5c).

Since these top models were selected for their high performance on the screening task, it is perhaps not surprising that they all show a high level of performance on that task. To determine whether the performance of these models generalized to other test sets, a series of *Validation* tests were performed. Specifically, we tested the best five models from each Unsupervised Learning petri dish on four test sets: two rendered object sets, one rendered face set, and a modified subset of the MultiPIE face recognition image set (see *Validation Phase* in Methods Section 9.2). Performance across each of these validation sets is shown in Figure 9.6 (black bars). While the exact ordering of model performance varied somewhat from validation set to validation set, the models selected during the *Screening Phase* performed well across the range of validation tasks.

The top five models found by our high-throughput screening procedure generally outperformed state-of-the-art models from the literature (see Methods Section 9.2) across all sets, with the best model found by the high-throughput search uniformly yielding the highest performance across all validation sets. Even greater performance was achieved by a simple summing of the SVM kernels from the top five models (red bar, Figure 9.6). Of note, the nearest contender from the set of state-of-the-art models is another biologically-inspired model [Serre *et al.*, 2007c; Mutch and Lowe, 2008].

Interestingly, a large performance advantage between our high-throughput-derived models and state-of-the-art models was observed for the MultiPIE hybrid set, even though this is arguably the most different from the task used for screening, since it is composed from natural images (photographs), rather than synthetic (rendered) ones. It should be noted that several of the state-of-the-art models, including the sparse C2 features ("SLF" in Figure 9.6), which was consistently the nearest competitor to our models, used filters that were individually tailored to each validation test – i.e. the representation used for "Boats vs. Planes" was optimized for that set, and was different from the representation used for the MultiPIE Hybrid set. This is in contrast

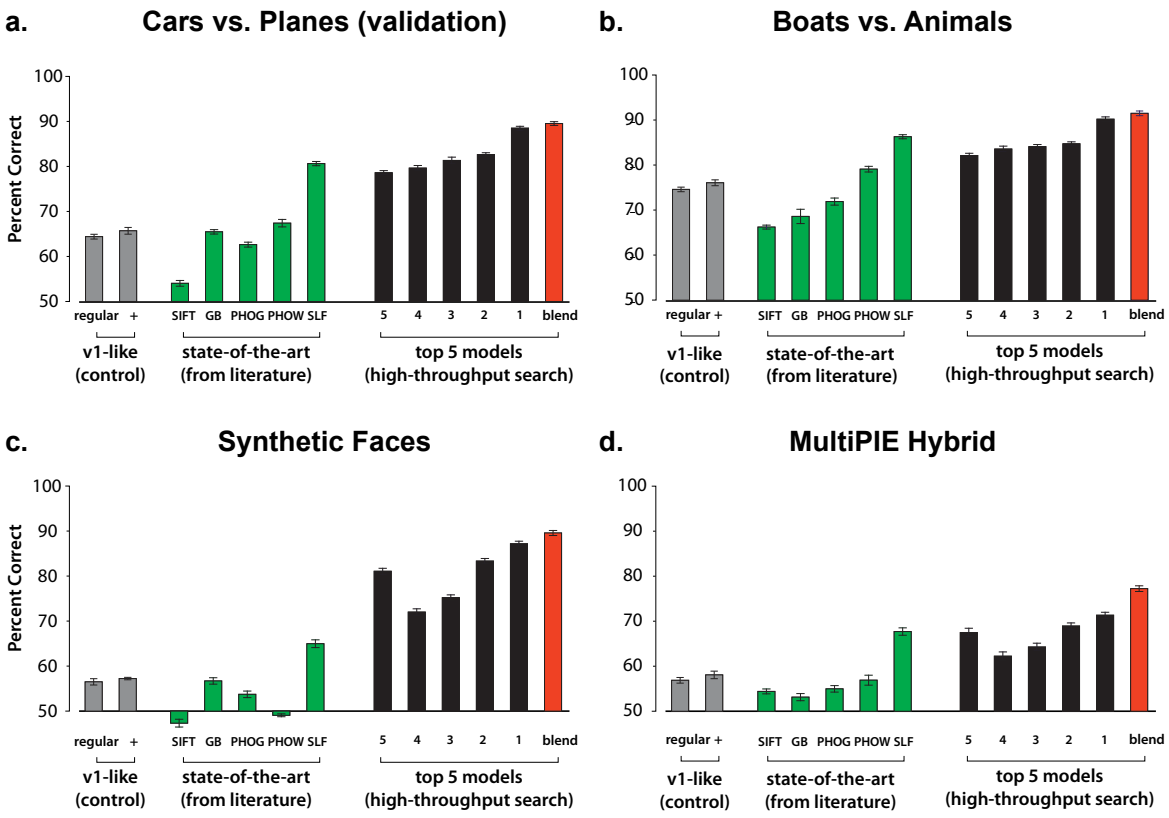



**Figure 9.6: Validation.** Performance of the top five models from the *Screening Phase* on a variety of other object recognition challenges. Example images from each object recognition test are shown in Figure 9.4. For each validation set, the performance (averaged over 10 random splits; error bars represent standard error of the mean) is first plotted for *V1-like* and *V1-like+* baseline models (see Chapters 4, 5 and 6 for a detailed description of these two variants) (gray bars), and for five state-of-the-art vision systems (green bars): Scale Invariant Feature Transform (SIFT, [Lowe, 2004]), Geometric Blur Descriptor (GB, [Berg and Malik, 2001; Zhang *et al.*, 2006]), Pyramidal Histogram of Gradients (PHOG, [Dalal and Triggs, 2005; Bosch *et al.*, 2007; Varma and Ray, 2007]), Pyramidal Histogram of Words (PHOW, [Bosch *et al.*, 2007; Varma and Ray, 2007; Lazebnik *et al.*, 2009]), and a biologically-inspired hierarchical model ("Sparse Localized Features" SLF, [Mutch and Lowe, 2008]). Finally, performance of the five best models derived from the high-throughput screening approach presented in this paper (black bars), and the performance achieved by averaging the five SVM kernels (red bar labeled "blend"). In general, high-throughput-derived models outperformed the *V1-like* baseline models, and tended to outperform a variety of state-of-the-art systems from the literature. Model instantiation 3281 and the blend of all five top models uniformly produced the best results across all test sets considered here.

to our models, which learned their filters from a completely unrelated video data set (Law and Order) and were screened using an unrelated task ("Cars vs. Planes", see Methods Section 9.2). While even better performance could no doubt be obtained by

screening with a subset taken from each individual validation test, the generalizability of performance across a range of different tasks argues that our approach may be uncovering features and representations that are broadly useful. Such generality is in keeping with the models' biological inspiration, since biological visual representations must be flexible enough to represent a massive diversity of objects in order to be useful.

Results for the 2,500 models in each of the other two "petri dishes" (i.e. models trained with alternate video sets during unsupervised learning) were appreciably similar, and are shown in Supplemental Figures S7 and S8, using the same display conventions set forth in Figures 9.5 and 9.6.

## 9.4 Discussion

We have demonstrated a high-throughput framework, within which a massive number of candidate vision models can be generated, screened, and analyzed. Models found in this way were found to consistently outperform an experimentally-motivated baseline model (a *V1-like* model; see Chapter 4), and the representations of visual space instantiated by these models were found to be useful generally across a variety of object recognition tasks. The best of these models and the blend of the five best models were both found to consistently outperform a variety of state-of-the-art machine vision systems for all of the test sets explored here, even without any additional optimization.

This work builds on a long tradition of machine vision systems inspired by biology (e.g. [Fukushima, 1980; Hinton, 1989; Haykin, 1994; Riesenhuber and Poggio, 1999b; Rolls and Milward, 2000; Rolls and Deco, 2002; LeCun *et al.*, 2004; Serre *et al.*, 2007c]). However, while this past work has generated impressive progress towards building artificial visual systems, it has explored only a few examples drawn from the larger space of biologically-inspired models. While the task of exploring the full space of possible model instantiations remains daunting (even within the relatively restricted "first-order" class of models explored here), our results suggest that even a relatively simple, brute-force high-throughput search strategy is effective in identifying promising models for further study. In the parameter space used here, we found that a handful of model instantiations performed substantially better than the rest, with these "good" models occurring

at a rate of approximately one in five-hundred. The relative rarity of these models underscores the importance of performing large-scale experiments with many model instantiations, since these models would be easy to miss in a "one-off " mode of exploration. Importantly, these rare, high-performing models performed well across a range of object recognition tasks, indicating that our approach does not simply optimize for a given task, but can uncover visual representations of general utility.

Though not conceptually critical to our approach, modern graphics hardware played an essential role in making our experiments possible. In approximately one week, we were able to test 7,500 model instantiations, which would have taken approximately two years using a conventional (e.g. MATLAB-based) approach. While it is certainly possible to use better-optimized CPU-based implementations, GPU hardware provides large increases in attainable computational power (see Table 9.1 and Supplemental Figure S1).

An important theme in this work is the use of parametrically controlled objects as a way of guiding progress. While we are ultimately interested in building systems that tolerate image variation in real-world settings, such sets are difficult to create, and many popular currently-available "natural" object sets have been shown to lack realistic amounts of variation. Our results show that it is possible to design a small synthetic set to screen and select models that generalize well across various visual classification tasks, suggesting that parametric sets can capture the essence of the invariant object recognition problem. Another critical advantage of the parametric screening approach presented here is that task difficulty can be increased on demand – that is, as models are found that succeed for a given level of image variation, the level of variation (and therefore the level of task difficulty), can be "ratcheted up" as well, maintaining evolutionary "pressure" towards better and better models.

While we have used a variety of synthetic (rendered) object image sets, images need not be synthetic to meet the requirements of our approach. The modified subset of the MultiPIE set used here ("MultiPIE Hybrid", Figure 9.4) is an example of how parametric variation can also be achieved using carefully controlled photography.

### 9.4.1 Future Directions

While our approach has yielded a first crop of promising biologically-inspired visual representations, it is another, larger task to understand how these models work, and why they are better than other alternatives. While such insights are beyond the scope of the present paper, our framework provides a number of promising avenues for further understanding.

One obvious direction is to directly analyze the parameter values of the best models in order to understand which parameters are critical for performance. Figure 9.7 shows distributions of parameter values for four arbitrarily chosen parameters. While in no way conclusive, there are hints that some particular parameter values may be more important for performance than others (for quantitative analysis of the relationship between model parameters and performance, see Supplemental Text S3, Figures S9 and S10). The speed with which large collections of models can be evaluated opens up the possibility of running large-scale experiments where given parameters are held fixed, or varied systematically. Insights derived from such experiments can then be fed back into the next round of high-throughput search, either by adjusting the parameter search space or by fundamentally adjusting the algorithm itself. Such iterative refinement is an active area of research in our group.

The search procedure presented here has already uncovered promising visual representations, however, it represents just the simplest first step one might take in conducting a large-scale search. For the sake of minimizing conceptual complexity, and maximizing the diversity of models analyzed, we chose to use random, brute-force search strategy. However, a rich set of search algorithms exist for potentially increasingly the efficiency with which this search is done (e.g. evolutionary algorithms [Deb, 2001; Hansen and Ostermeier, 2001; Igel *et al.*, 2007], simulated annealing [Rutenbar, 1989], or particle swarm techniques [Kennedy and Eberhart, 1995] among others). Interestingly, our brute-force search found strong models with relatively high probability, suggesting that, while these models would be hard to find by "manual" trial-and-error, they are not especially rare in the context of our high-throughput search.

While better search algorithms will no doubt find better instances from the model

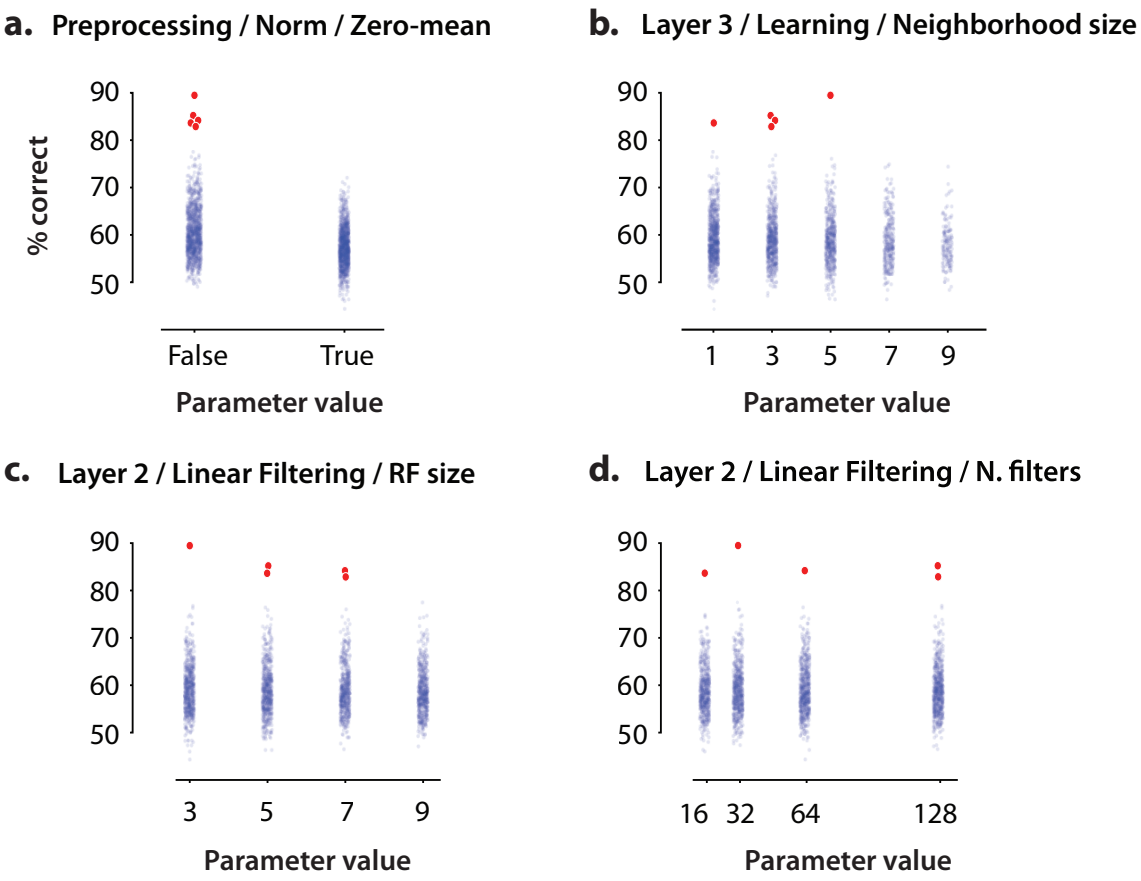



**Figure 9.7: Distributions of Screening Task performance, as a function of parameter values for four arbitrarily-chosen parameters.** See Supplemental Text S1 for an exhaustive description of the meaning of each parameter. The top five best performing models are plotted in red, with the other models overplotted in semi-transparent blue. The parameters considered in (a) and (b) show hints of a relationship between parameter value and inclusion in the top five. In (a) all of the five best models had the same value of the parameter, and in (b) best models were clustered in lower ranges of parameter value. (c) and (d) show parameters where the best models were distributed across a range of parameter values. Such examinations of parameter values are in no way conclusive, but can provide hints as to which parameters might be important for performance.

class used here, an important future direction is to refine the parameter-ranges searched and to refine the algorithms themselves. While the model class described here is large, the class of all models that would count as "biologically-inspired" is even larger. A critical component of future work will be to adjust existing mechanisms to achieve better performance, and to add new mechanisms (including more complex features such as

long-range feedback projections). Importantly, the high-throughput search framework presented here provides a coherent means to find and compare models and algorithms, without being unduly led astray by weak sampling of the potential parameter space.

Another area of future work is the application of high-throughput screening to new problem domains. While we have here searched for visual representations that are good for object recognition, our approach could also be applied to a variety of other related problems, such as object tracking, texture recognition, gesture recognition, feature-based stereo-matching, etc. Indeed, to the extent that natural visual representations are flexibly able to solve all of these tasks, we might likewise hope to mine artificial representations that are useful in a wide range of tasks.

Finally, as the scale of available computational resources steadily increases, our approach naturally scales as well, allowing more numerous, larger, and more complex models to be examined. This will give us both the ability to generate more powerful machine vision systems, and to generate models that better match the scale of natural systems, providing more direct footing for comparison and hypothesis generation. Such scaling holds great potential to accelerate both artificial vision research, as well as our understanding of the computational underpinnings of biological vision.

## 9.5 Supplemental Text S1: Search Space of Candidate Models

Candidate models were composed of a hierarchy of three layers, with each layer including a cascade of linear and nonlinear operations that produce successively elaborated nonlinear feature-map representations of the original image. A diagram detailing the flow of operations is shown in Figure 9.8, and, for the purposes of notation, the cascade of operations is represented as follows:

$$Layer^0:$$
$$\mathbf{Input} \overset{\textbf{Grayscale}}{\longrightarrow} \overset{\textbf{Normalize}}{\longrightarrow} \mathbf{N^0}$$
$$Layer^1:$$
$$\mathbf{N^0} \overset{\textbf{Filter}}{\longrightarrow} \mathbf{F^1} \overset{\textbf{Activate}}{\longrightarrow} \mathbf{A^1} \overset{\textbf{Pool}}{\longrightarrow} \mathbf{P^1} \overset{\textbf{Normalize}}{\longrightarrow} \mathbf{N^1}$$

and generally, for all $\ell \geq 1$:

$$Layer^\ell:$$
$$\mathbf{N^{\ell-1}} \overset{\textbf{Filter}}{\longrightarrow} \mathbf{F}^\ell \overset{\textbf{Activate}}{\longrightarrow} \mathbf{A}^\ell \overset{\textbf{Pool}}{\longrightarrow} \mathbf{P}^\ell \overset{\textbf{Normalize}}{\longrightarrow} \mathbf{N}^\ell$$

Details of these steps along with the range of parameter values included in the random search space are described below. We varied 52 parameters (described below), with a total of $2.807930 \times 10^{25}$ possible unique combinations of parameter values.

### 9.5.1 Input and Pre-processing

The input of the model was a $200 \times 200$ pixel image. In the pre-processing stage, referred to as $Layer^0$, this input was converted to grayscale and locally normalized:

$$\mathbf{N^0} = \mathbf{Normalize}(\mathbf{Grayscale}(\mathbf{Input})) \tag{9.1}$$

where the **Normalize** operation is described in detail below. Because this normalization is the final operation of each layer, in the following sections, we refer to $N^{\ell-1}$ as the input of each $Layer^{\ell>0}$ and $N^\ell$ as the output.

### 9.5.2 Linear Filtering

**Description:** The input $N^{\ell-1}$ of each subsequent layer (i.e. $Layer^\ell, \ell \in \{1, 2, 3\}$) was first linearly filtered using a bank of $k^\ell$ filters to produce a stack of $k^\ell$ feature maps, denoted $F^\ell$. In a biologically-inspired context, this operation is analogous to the

weighted integration of synaptic inputs, where each filter in the filterbank represents a different cell.

**Definitions:** The filtering operation for $Layer^{\ell}$ is denoted:

$$\mathbf{F}^{\ell} = \mathbf{Filter}(\mathbf{N}^{\ell-\mathbf{1}}, \mathbf{\Phi}^{\ell}) \tag{9.2}$$

and produces a stack, $F^{\ell}$, of $k^{\ell}$ feature maps, with each map, $F_i^{\ell}$, given by:

$$F_i^{\ell} = N^{\ell-1} \otimes \Phi_i^{\ell} \qquad \forall i \in \{1, 2, \ldots, k^{\ell}\} \tag{9.3}$$

where $\otimes$ denotes a correlation of the output of the previous layer, $N^{\ell-1}$ with the filter $\Phi_i^{\ell}$ (e.g. sliding along the first and second dimensions of $N^{\ell-1}$). Because each successive layer after $Layer^0$, is based on a stack of feature maps, $N^{\ell-1}$ is itself a stack of 2-dimensional feature maps. Thus the filters contained within $\Phi^{\ell}$ are, in turn, 3-dimensional, with the their third dimension matching the number of filters (and therefore, the number of feature maps) from the previous layer (i.e. $k^{\ell-1}$).

**Parameters:**

- The filter shapes ${f_s}^{\ell} \times {f_s}^{\ell} \times {f_d}^{\ell}$ were chosen randomly with ${f_s}^{\ell} \in \{3, 5, 7, 9\}$ and ${f_d}^{\ell} = k^{\ell-1}$.
- Depending on the layer $\ell$ considered, the number of filters $k^{\ell}$ was chosen randomly from the following lists:
  - In $Layer^1$, $k^1 \in \{16, 32, 64\}$
  - In $Layer^2$, $k^2 \in \{16, 32, 64, 128\}$
  - In $Layer^3$, $k^3 \in \{16, 32, 64, 128, 256\}$

All filters were initialized to random starting values, and their weights were then learned during the *Unsupervised Learning Phase* (described below; an example of a set of learned filterbanks from one model instance is shown in Figure 9.13).

### 9.5.3 Activation Function

**Description:** Filter outputs were subjected to threshold and saturation activation function, wherein output values were clipped to be within a parametrically defined range. This operation is analogous to the spontaneous activity thresholds and firing saturation levels observed in biological neurons.

**Definitions:** We define the activation function:

$$\mathbf{A}^{\ell} = \mathbf{Activate}(\mathbf{F}^{\ell}) \tag{9.4}$$

that clips the outputs of the filtering step, such that:

$$\mathbf{Activate}(\mathbf{x}) = \begin{cases} {\gamma_{max}}^{\ell} & \text{if } x > {\gamma_{max}}^{\ell} \\ {\gamma_{min}}^{\ell} & \text{if } x < {\gamma_{min}}^{\ell} \\ x & \text{otherwise} \end{cases} \tag{9.5}$$

Where the two parameters ${\gamma_{min}}^{\ell}$ and ${\gamma_{max}}^{\ell}$ control the threshold and saturation, respectively. Note that if both minimum and maximum threshold values are $-\infty$ and $+\infty$, the activation is linear (no output is clipped).

**Parameters:**

- ${\gamma_{min}}^{\ell}$ was randomly chosen to be $-\infty$ or 0
- ${\gamma_{max}}^{\ell}$ was randomly chosen to be 1 or $+\infty$

### 9.5.4 Pooling

**Description:** The activations of each filter within some neighboring region were then pooled together and the resulting outputs were spatially downsampled.

**Definitions:** We define the pooling function:

$$\mathbf{P}^{\ell} = \mathbf{Pool}(\mathbf{A}^{\ell}) \tag{9.6}$$

such that:

$$\mathbf{P}_\mathbf{i}^\ell = \mathbf{Downsample}_\alpha(\sqrt[p^\ell]{(A_i^\ell)^{p^\ell} \odot \mathbf{1}_{a^\ell \times a^\ell}}) \tag{9.7}$$

Where $\odot$ is the 2-dimensional correlation function with $\mathbf{1}_{a^\ell \times a^\ell}$ being an $a^\ell \times a^\ell$ matrix of ones ($a^\ell$ can be seen as the size of the pooling "neighborhood"). The variable $p^\ell$ controls the exponents in the pooling function.

**Parameters:**

- The stride parameter $\alpha$ was fixed to 2, resulting in a downsampling factor of 4.
- The size of the neighborhood $a^\ell$ was randomly chosen from $\{3, 5, 7, 9\}$.
- The exponent $p^\ell$ was randomly chosen from $\{1, 2, 10\}$.

Note that for $p^\ell = 1$, this is equivalent to blurring with a $a^\ell \times a^\ell$ boxcar filter. When $p^\ell = 2$ or $p^\ell = 10$ the output is the $L^{p^\ell}$-norm [2].

### 9.5.5 Normalization

**Description:** As a final stage of processing within each layer, the output of the Pooling step were normalized by the activity of their neighbors within some radius (across space and across feature maps). Specifically, each response was divided by the magnitude of the vector of neighboring values if above a given threshold. This operation draws biological inspiration from the competitive interactions observed in natural neuronal systems (e.g. contrast gain control mechanisms in cortical area V1, and elsewhere [Geisler and Albrecht, 1992; Rolls and Deco, 2002]).

**Definitions:** We define the normalization function:

$$\mathbf{N}^\ell = \mathbf{Normalize}(\mathbf{P}^\ell) \tag{9.8}$$

such that:

[2] The $L^{10}$-norm produces outputs similar to a *max* operation (i.e. *softmax*).

$$N^\ell = \begin{cases} \rho^\ell \cdot C^\ell & \text{if } \rho^\ell \cdot \left|\left|C^\ell \otimes \mathbf{1}_{b^\ell \times b^\ell \times k^\ell}\right|\right|_2 < \tau^\ell \\ \frac{C^\ell}{\left|\left|C^\ell \otimes \mathbf{1}_{b^\ell \times b^\ell \times k^\ell}\right|\right|_2} & \text{otherwise} \end{cases} \tag{9.9}$$

with

$$C^\ell = P^\ell - \delta^\ell \cdot \frac{P^\ell \otimes \mathbf{1}_{b^\ell \times b^\ell \times k^\ell}}{b^\ell \cdot b^\ell \cdot k^\ell} \tag{9.10}$$

Where $\delta^\ell \in \{0, 1\}$, $\otimes$ is a 3-dimensional correlation over the "valid" domain (i.e. sliding over the first two dimensions only), and $\mathbf{1}_{b^\ell \times b^\ell \times k^\ell}$ is a $b^\ell \times b^\ell \times k^\ell$ array full of ones. $b^\ell$ can be seen as the normalization "neighborhood" and $\delta^\ell$ controls if this neighborhood is centered (i.e. subtracting the mean of the vector of neighboring values) before divisive normalization. $\rho^\ell$ is a "magnitude gain" parameter and $\tau^\ell$ is a threshold parameter below which no divisive normalization occurs.

**Parameters:**

- The size $b^\ell$ of the neighborhood region was randomly chosen from $\{3, 5, 7, 9\}$.
- The $\delta^\ell$ parameter was chosen from $\{0, 1\}$.
- The vector of neighboring values could also be stretched by gain values $\rho^\ell \in \{10^{-1}, 10^0, 10^1\}$. Note that when $\rho^\ell = 10^0 = 1$, no gain is applied.
- The threshold value $\tau^\ell$ was randomly chosen from $\{10^{-1}, 10^0, 10^1\}$.

### 9.5.6 Final model output dimensionality

The output dimensionality of each candidate model was determined by the number of filters in the final layer, and the x-y "footprint" of the layer (which, in turn, depends on the subsampling at each previous layer). In the model space explored here, the possible output dimensionalities ranged from 256 to 73,984.

### 9.5.7 Unsupervised Learning

**Description:** During the *Unsupervised Learning Phase*, filter weights are learned from input video sequences. This procedure bears similarity to nonparametric density

estimation, e.g. online K-means clustering. The algorithm for this phase additionally contains simple mechanisms for taking advantage of temporal information in a video sequence, and thus *Unsupervised Learning* was conducted on sequences of video frames. In this work, 15,000 video frames were used.

**Definitions:** For each incoming video frame, an output for each filter at each location was computed, and a "winning" filter $\Phi^{\ell}_{winner}$ was selected:

$$winner = \arg\max_{i}(F^{\ell}_{i}) \tag{9.11}$$

This winning filter was adapted to the input, by adding the corresponding input patch, times a fixed learning rate $\lambda$, to the filter weights:

$$\Phi^{\ell}_{winner}{}' = (1 - \lambda^{\ell}) \cdot \Phi^{\ell}_{winner} + \lambda^{\ell} \cdot patch \tag{9.12}$$

The resulting updated filter was then re-normalized to zero-mean and unit-length:

$$\Phi^{\ell}_{winner}{}'' = \frac{\Phi^{\ell}_{winner}{}' - \langle \Phi^{\ell}_{winner}{}' \rangle}{\left|\left| \Phi^{\ell}_{winner}{}' - \langle \Phi^{\ell}_{winner}{}' \rangle \right|\right|_2} \tag{9.13}$$

Where $\langle \Phi^{\ell}_{winner}{}' \rangle$ represents the mean of the winner's weights and $\Phi^{\ell}_{winner}{}''$ is the filter carried forward into the next learning iteration.

The incoming patch could be normalized (i.e. $||patch||_2 = 1$), or not, under parametric control, and multiple patches could enter into one "round" of competition at the same time (e.g. filter stack outputs corresponding to multiple patches could be evaluated, and the largest output across all patches could decide the winner). The selection of the number of patches simultaneously competing was governed by the *Competition Neighborhood Size* and *Competition Neighborhood Stride* parameters, which served tile a set of competing filter stacks across the input.

**Parameters:**

- *Learning rate* parameter $\lambda^{\ell} \in \{10^{-4}, 10^{-3}, 10^{-2}\}$

- *Patch Normalization*: normalize *patch* to unit-length, or do not normalize (2 choices)
- *Competition Neighborhood Size* $\in \{1, 3, 5, 7, 9\}$
- *Competition Neighborhood Stride* $\in \{1, 3, 5, 7, 9\}$
- *"Rebalancing"*: if the relative winning ratio [3] of a given filter $\Phi_i^\ell$ is less than {1%, 10% or 50%} (3 choices), its weights are reinitialized to the values of the most-winning filter plus a random jitter. This prevents filters from never winning.
- *"Temporal Advantage"* (or *"trace"*, see also [Földiak, 1991; Rolls and Milward, 2000; Einhäuser *et al.*, 2005; Franzius *et al.*, 2008] for variants): the output score of the last-winning filter is multiplied by {1, 2 or 4} (3 choices) prior to determining which filter "wins." A value of 1 is the equivalent of no advantage; a value of 2 doubles the effective output of the filter for the purposes of competition, biasing it to win again.

### 9.5.8 Classification during Screening and Validation Phases

During the *Screening* and *Validation Phases*, the representations generated during the *Unsupervised Learning Phase* were evaluated in a variety of object recognition tasks (see main text). This *Classification Phase* consisted of the following steps, with *fixed parameters* across all model instantiations:

- A random sampling of up to 5,000 outputs from the full representation were taken (to accelerate processing).
- Dimensionality was further reduced by PCA (using training data only, keeping the full eigensubspace projection, i.e. as many dimensions as training examples).
- A linear SVM (using the LIBSVM[4] solver, with regularization parameter $C = 10$) was used with a 10-trial random subsampling cross-validation scheme (150

[3] the number of times $\Phi_i^\ell$ won multiplied by the number of filters, divided by the running count of completed updates

[4] `http://www.csie.ntu.edu.tw/~cjlin/libsvm`

training and 150 testing examples).

### 9.5.9 Random Exploration

Note that the parameters and parameter ranges described here are clearly not the most comprehensive search space; rather they represent a starting point intended to demonstrate the utility of the overarching approach. While a brute force search procedure was used here, other more elaborate optimization schemes (e.g. evolutionary algorithms [Deb, 2001; Hansen and Ostermeier, 2001; Igel *et al.*, 2007]) could also be used.

## 9.6 Supplemental Text S2: Technical Details of the Computational Framework

### 9.6.1 Coarse-to-fine Parallelism

The high-throughput search described in this paper takes advantage of multiple levels of parallelism, from coarse to fine-grained. Roughly speaking, fine-grained parallelism is exploited by allocating one core (of which modern graphics hardware have many; and their number is exponentially growing over the years) to one or more virtual neurons, while coarse-scale parallelism is achieved by allocating one model instantiation to each of many multi-core pools (i.e. CPUs, Cell Processors, GPUs). Because we evaluated thousands of model instantiations, it was straightforward to spread these evaluations across a cluster of GPU-enabled nodes, with the throughput of each node maximized by taking full advantage of fine-grained parallelism.

In practice, as a general rule in modern high-performance computing, the level of speed-up that is achievable depends more fundamentally on the ability to bring relevant memory fetches to the parallel arithmetic processing units, than on the number of these units per se. For this reason, stream processing architectures contain special mechanisms for explicit manipulation of fast local caches (e.g. the Local Stores on the Cell processor and Shared Memory on NVIDIA GPUs). Importantly for maximizing the usage of the parallel resources in these processors, some of the largest computational

bottlenecks present in our model class (e.g. filtering operations), lend themselves to the usage of such caches by loading small tiles of data into local caches. More broadly, because each layer of each model maintains a notion of x-y space, and because most of the operations operate over spatially local regions (e.g. normalization occurs within a spatially-restricted neighborhood), our coarse-to-fine-grain parallelism can be exploited throughout a large portion of our model implementation.

### 9.6.2 Distributed Job System

More specifically, we implemented a simple distributed job system where all candidate model instantiations in a given "petri dish" were submitted. Each job consisted in the unsupervised learning of the model followed by performance evaluation on the screening task. Computing agents were responsible for fetching these jobs one by one, running them until completion and submitting their results back to a relational database for further analysis (we now use NoSQL-like distributed databases). Each multi-core was responsible for keeping one computing agent alive. In our experiments, we used 23 computing agents running on 23 PlayStation 3 systems. The screening throughput of this cluster was roughly 40 models per hour, for a total cost of $12,000.

### 9.6.3 Software Engineering and Programming

The use of commodity graphics hardware has drastically reduced the cost of a scientific exploration of this scale (see Table 9.1 and Supplemental Figure S1), but writing reliable code for these multi-core platforms can be a demanding task. On one hand, scientific software can be difficult to handle because of constantly changing requirements. On the other hand, these architectures are advancing at a very rapid pace and we have experienced three different paradigms in three years (i.e. programming GPUs with graphics primitives in 2006, programming the PlayStation 3 using low-level Cell intrinsics in 2007 and programming GPUs with compute primitives in 2008; see below). To overcome these difficulties, we combined careful engineering, high-level languages (such

as Python[5] and its numerous scientific bindings[6],[7]) and template meta-programming techniques. Inspired by automatically-tuned scientific libraries such as the Automatically Tuned Linear Algebra Software (ATLAS[8]) or the Fastest Fourier Transform in the West library (FFTW[9]), we found that empirical optimization through automatic run-time code generation was a useful way to abstract the low-level details away from the end-user. Integrating these heterogeneous technologies in our large-scale computational software shows us that it is possible to achieve a favorable balance between ease-of-use, ease-of-programming and peak computing speed.

### Details

An extensive description on how we used these techniques is well beyond the scope of this paper, however, we highlight a few important high-level points here:

- *Meta-programming and combining high-level with low-level languages.* In our implementation, each core operation (see Text S1) has two levels of abstraction. The high-level abstraction is designed for the user and is written in a high-level language (we used Python). The low-level abstraction is designed to achieve maximum throughput on heterogeneous hardware and as a consequence it must be able to handle low-level languages and "close to the metal" code optimization techniques (e.g. involving assembly) if needed. The interface between the two abstractions is a templating engine (we used Cheetah[10]) that is responsible for dynamically generating optimized low-level code at runtime, many specialized versions of which are compiled and auto-tuned prior to running a given model simulation. Such an approach is equivalent to "just-in-time" (JIT[11]) compilation techniques used elsewhere for portability and dynamic specialization[12].

  The primary goal of the developer is to come up with various optimization strate-

[5] http://www.python.org
[6] http://numpy.scipy.org
[7] http://www.scipy.org
[8] http://www.netlib.org/atlas
[9] http://www.fftw.org
[10] http://www.cheetahtemplate.org
[11] http://en.wikipedia.org/wiki/Just-in-time_compilation
[12] http://psyco.sourceforge.net

gies that instrument low-level code and manipulate it from a high-level language (using templates). These strategies may involve loop unrolling[13], software pipelining[14], register pressure[15], communication and computation load distribution (aka "latency hiding"), to name just a few.

Producing a large number of hand-tuned implementations, corresponding to optimized lower-level code across a range of implementations would be impractically time-consuming. A meta-programming approach circumvents this difficulty by producing code that can itself generate a variety of specialized compiled versions under parametric control. This large number of candidate implementations of the meta-program can be empirically tested to find which is the fastest (see *Auto-tuning* below)

It is important to note that the high-level language must be mature and general enough to allow a seamless interaction between all the components of the system, from the distributed job system and its database to its template meta-programming capabilities and its interaction with other (low-level) languages. The Python programming language was a natural choice as it is often referred as a versatile "glue" language (i.e. used to connect software components of different levels together), and allows quick prototyping and experimentation.

While MATLAB, by itself, does not support easy meta-programming on GPUs, commercial companions to MATLAB like AccelerEyes's Jacket[16] could potentially enable some of the gains necessary for our approach. However, such solutions typically do not necessarily achieve the full performance of stream processing hardware [17].

- *Auto-tuning* To auto-tune our instrumented (i.e. templated) code, we used the simplest approach: random search on a coarse grid. Using this simple approach, we achieved comfortable speed-ups, and thus we did not explore more complex

[13] http://en.wikipedia.org/wiki/Loop_unwinding
[14] http://en.wikipedia.org/wiki/Software_pipelining
[15] http://en.wikipedia.org/wiki/Register_allocation
[16] http://www.accelereyes.com
[17] http://www.nvidia.com/object/matlab_acceleration.html

schemes before launching the experiments presented in this study. In the future, we plan to investigate the use of machine learning techniques to auto-tune the code, an approach recently undertaken by IBM's Milepost GCC[18].

- *Use of specialized extensions and libraries:*

  Our first implementation in 2006 on the NVIDIA 7900GTX was probably the most challenging to complete. At this time, there was no compute language to exploit the horsepower of GPUs and we were forced map our problem into a graphics domain where matrices were textures and computations were quad drawings. We had to translate our algorithms using OpenGL[19] primitives and Cg[20] shaders, while trying to maintain our code under unstable graphics drivers releases and scarce GNU/Linux support. Interfacing these programs with Python was straightforward with PyOpenGL[21] bindings and home-made Cg swig[22] bindings.

  Our PlayStation 3 implementation was created using tools provided in IBM's Cell SDK (Software Development Kit[23]) were mature and comprehensive (e.g. availability of a simulator, profiler, debugger, etc.), interfaced using ctypes[24] from the Python standard library. These tools allow one to program primarily in C (or in a language that can bind to an underlying C implementation), but require specialized knowledge of the architecture of the Cell processor in order to achieve high levels of performance.

  In 2007, NVIDIA released the CUDA (Compute Unified Device Architecture) technology. They extended the function of their GPUs and provided a new level of control to address a wide range of computationally intensive problems. At this point, the use of graphics primitives became obsolete and the GPU parallel computing power was made more accessible through NVIDIA's extensions

[18] `http://www.milepost.eu`
[19] `http://www.khronos.org/opengl`
[20] `http://developer.nvidia.com/page/cg_main.html`
[21] `http://pyopengl.sourceforge.net`
[22] `http://www.swig.org`
[23] `http://www.ibm.com/developerworks/power/cell`
[24] `http://docs.python.org/library/ctypes.html`

to the standard C programming language. Today, CUDA is very mature and many optimized libraries are available: Fourier transform (CUFFT), linear algebra (CUBLAS), standard parallel primitives (CUDPP), templating (Thrust), etc. We first interfaced CUDA programs using home-made ctypes bindings.

Our GPU implementations are now managed via PyCUDA [Klöckner *et al.*, 2009] and python-cuda, Python libraries that bind to the underlying NVIDIA CUDA libraries. Meta programs were created using the Cheetah template library (see above) that would emit specialized CUDA code which was compiled on the fly and run on the GPU.

While the efforts described here relied on vendor-specific software development kits (which arguably imposes a significant barrier-to-entry for developer scientists), efforts are underway in industry to provide a unified programming model and tool set for developing applications of the sort presented here. In particular, the lack of general-purpose programming standard for heterogeneous systems was recently addressed through the introduction of OpenCL[25] (Open Computing Language) by the Khronos Group. OpenCL is being driven by industry-leading companies including AMD/ATI, Apple, ARM, Codeplay, Ericsson, Freescale, Imagination Technologies, IBM, Intel, Nokia, NVIDIA, Motorola, RapidMind and Texas Instruments. Interfacing OpenCL with Python is already supported by PyOpenCL[26]. We anticipate future work to utilize these tools will enable us to target more platforms, and will ease the cost of incorporating ideas of the sort presented here into the work of other groups.

- *Understanding the hardware:* Even though achieving peak performance may require deep understanding of the underlying architecture, it is usually possible to get one order of magnitude speed-up by just "porting" your code and exposing its parallelism. With more knowledge of the threading and memory / communication hierarchies, it is possible to achieve two orders of magnitude by maximizing the arithmetic intensity (i.e. the ratio of mathematical operations per memory fetch)

[25] `http://www.khronos.org/opencl`
[26] `http://pypi.python.org/pypi/pyopencl`

to amortize the latency of transferring data. Up to three orders of magnitude can be obtain with a deeper understanding including, for example, latencies of the ISA (Instruction Set Architecture) assembly operands, instruction level parallelism, or number of registers per multi-processors, and how to fine-tune their usage.

- *Learning these techniques:* With the recent introduction of programming "standards" and documentation online, it has become increasingly easy to learn how to exploit graphics hardware for general computing. For example, early in 2009, we had the opportunity to teach a one-month intensive course[27] on the subject to undergraduate and graduate students at MIT, and in only a few weeks, they were able to learn and apply these techniques, and finally achieve up to two orders of magnitude speed-ups in scientific applications such as Lipid Bilayers Simulation, H.264 Compression, Particle Interaction Simulation, High-Definition Pedestrian Detection, Bio-Inspired Computer Vision or Regression Analysis (see the course's website for more information).

## 9.7 Supplemental Text S3: First-Order Analyzes of Model Parameters and Behavior

The results presented in the main text show that the five best model instantiations found by a screening procedure are well-suited to a variety of object recognition tasks, but they do not speak to *how* these models achieve their performance. While fully answering this question is beyond the scope of the present paper, as a first step in understanding model performance, we asked a series of first-order questions about the relationship between model parameters and performance. These analyzes are in no way intended to be definitive; rather, they primarily suggest directions and challenges for future experiments.

First, we asked which (if any) of the parameters were predictive of model performance, using simple linear regression. While complex interdependencies between

[27] `http://sites.google.com/site/cudaiap2009`

parameters can (and almost certainly do) exist, linear regression provides a first-order tool to identify parameters that are especially important to performance. Significance values for individual parameters are shown in a histogram in Figure 9.16. A handful of parameters were found to be significantly predictive of model performance. To determine if a particular category of model parameters were more important than any other, we divided the parameters into three groups: linear filter parameters, normalization/activation/pooling nonlinear parameters, and learning parameters. We found that normalization/activation/pooling parameters shared a trend toward being over-represented in the set of significantly predictive parameters, but that the distribution of significant parameters from each of these three categories were not significantly different than would be predicted by chance ($p = 0.338$; Fisher's exact test).

Another reasonable first-order question to ask is whether the top models are somehow similar to one another. In this context, similarity might be assessed along a number of axes. One possibility is to simply compare the parameter values for the best models, to see if they share more parameter settings in common with each other than one would expect by chance. To do this, an expanded binary parameter vector was first created in which each parameter value combination was included as a binary element (e.g. if a parameter $\omega$ could take of values 3, 5, and 7, three binary values $[\omega = 3]$, $[\omega = 5]$, and $[\omega = 7]$ were generated for each model). The Hamming distance was then computed between these vectors to assess the similarity between models. To determine whether the top five models were more similar to each other than to the population of models, we computed the median pairwise Hamming distance among the top five, and among randomly chosen sets of five models ($N = 100,000$) taken from the remaining (non-top-five) models (Figure 9.17a). By this measure, the median distance between the top five trended toward higher than expected similarity but was not found to be significantly different from the median distance over the full population of models ($p = 0.136$; permutation test). Thus, at least by this simple measure, we could find no evidence that the best models were any more similar to each other than would be expected by chance. Attempts to compare parameter vector using $\ell_1$ and $\ell_2$ distances also failed to find any increased similarity amongst the best models, though these analyzes are intrinsically difficult to interpret, as it is unclear how to scale one parameter

relative to another.

Another approach to comparing models is to compare the structure of the space of their outputs. That is, for a given set of images, do the best models somehow transform these images in a similar way? To explore this issue, we transformed 600 images from the screening set for each model, and then formed the similarity (Euclidean distance) matrix for the set of transformed image vectors. We then computed the Euclidean distance between the upper triangular part of these symmetric matrices (similar to the Frobenius distance) to assess their similarity. As before, we computed distributions of pairwise distances within the top five models ($N = \binom{5}{2} = 10$), and in the random sampling from the full population ($N = 10,000$) in order to test whether the top five models were more similar to each other than would be expected from random draws of five models (Figure 9.17b,c). We found that the similarity matrices of the top five models tended to be more similar to each other, but that this effect was not significant ($p = 0.082$; permutation test).

Taken together, these analyzes of model parameters and performance show that model comparison is not a straightforward endeavor, but that there are clues to which parameters may be important to focus on in achieving greater performance.

## 9.8 Supplemental Figures

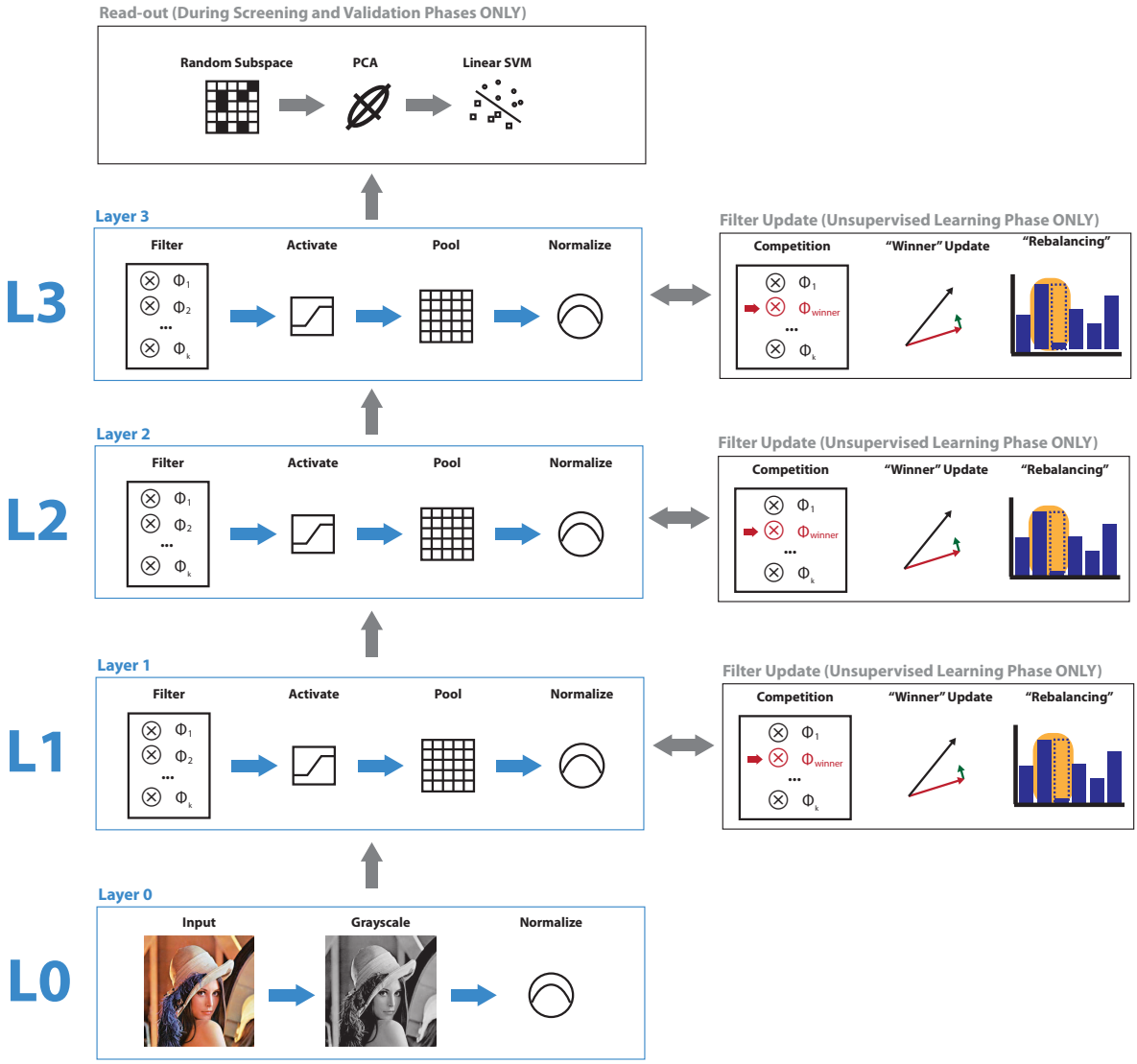
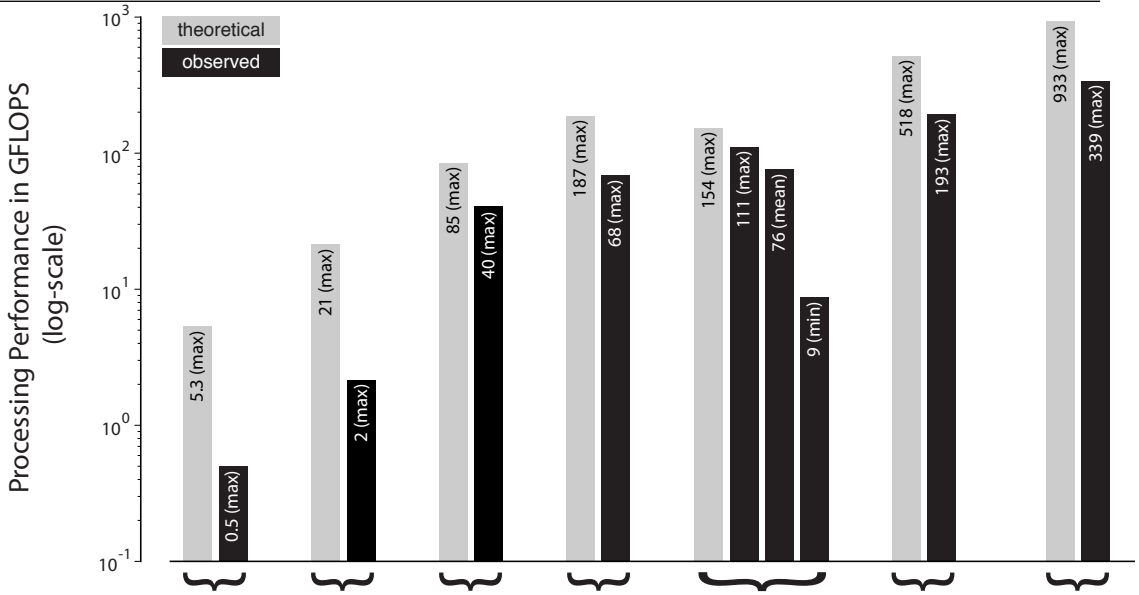



**Figure 9.8: A schematic of the flow of transformations performed in our family of biologically-inspired models.** Blue-labeled boxes indicate the cascade of operations performed in each of the three layers in the canonical model. Gray-labeled boxes to the right indicate filter weight update steps that take place during the *Unsupervised Learning Phase* after the processing of each input video frame. The top gray-labeled box shows processing steps undertaken during the *Screening* and *Validation Phases* to evaluate the performance achievable with each model instantiation.




| Hardware | | | | | | | |
|---|---|---|---|---|---|---|---|
| Manufacturer | Intel | Intel | Intel | NVIDIA | Sony, IBM, Toshiba | NVIDIA | NVIDIA |
| Model | Q9450 | Q9450 | Q9450 | 7900 GTX | PlayStation 3 (Cell) | 8800 GTX | GTX 280 |
| # cores used | 1 | 4 | 4 | 96 | 8 (2+6) | 128 | 240 |
| **Software** | | | | | | | |
| Languages | MATLAB / C | MATLAB / C | C | Python / C++ | Python / C | Python / C | Python / C |
| Extensions / Libraries | MEX | MEX | SSE2 / pthread | OpenGL / Cg | Cell SDK | PyCUDA | PyCUDA |
| Year of implementation | 2008 | 2008 | 2008 | 2006 | 2007 | 2007 | 2008 |
| **Performance / Cost** | | | | | | | |
| GFLOPS (max) | 0.5 | 2 | 40 | 68-272* | 111 | 193-772* | 339-1356* |
| Full System Cost (approx.) | $1,500** | $2,700** | $1,000 | $1,500-$3,000* | $400 | $1,500-$3,000* | $1,500-$3,000* |
| $ / GFLOPS | 3000 | 1350 | 25 | 22-11* | 4 | 8-4* | 4-2* |
| Relative Speedup | 1 | 4 | 80 | 136-544* | 222 | 386-1544* | 678-2712* |
| Relative GFLOPS / $ | 1 | 2 | 120 | 136-272* | 833 | 386-772* | 678-1356* |

**Figure 9.9: Processing Performance of the Linear Filtering Operation.** The theoretical and observed processing performance in GFLOPS (billions of floating point operations per second) is plotted for a key filtering operation in our biologically-inspired model implementation. Theoretical performance numbers were taken from manufacturer marketing materials and are generally not achievable in real-world conditions, as they consider multiple floating operations per clock cycle, without regard to memory communication latencies (which typically are the key determinant of real-world performance). Observed processing performance for the filtering operation varied across candidate models in the search space, as input and filter sizes varied. Note that the choice of search space can be adjusted to take maximum advantage of the underlying hardware at hand. We plot the "max" observed performance for a range of CPU and GPU implementations, as well as the "mean" and "min" performance of our PlayStation 3 implementation observed while running the 7,500 models presented in this study. The relative speedup denotes the peak performance ratio of our optimized implementations over a reference MATLAB code on one of the Intel QX9450's core (e.g. using *filter2*, which is itself coded in C++), whereas the relative GFLOPS per dollar indicates the peak performance *per dollar* ratio. Costs of typical hardware for each approach and cost per FLOPS are shown at the bottom. * These ranges indicate the performance and cost of a single system containing from one (left) to four (right) GPUs. ** These costs include both the hardware and MATLAB yearly licenses (based on an academic discount pricing, for one year).

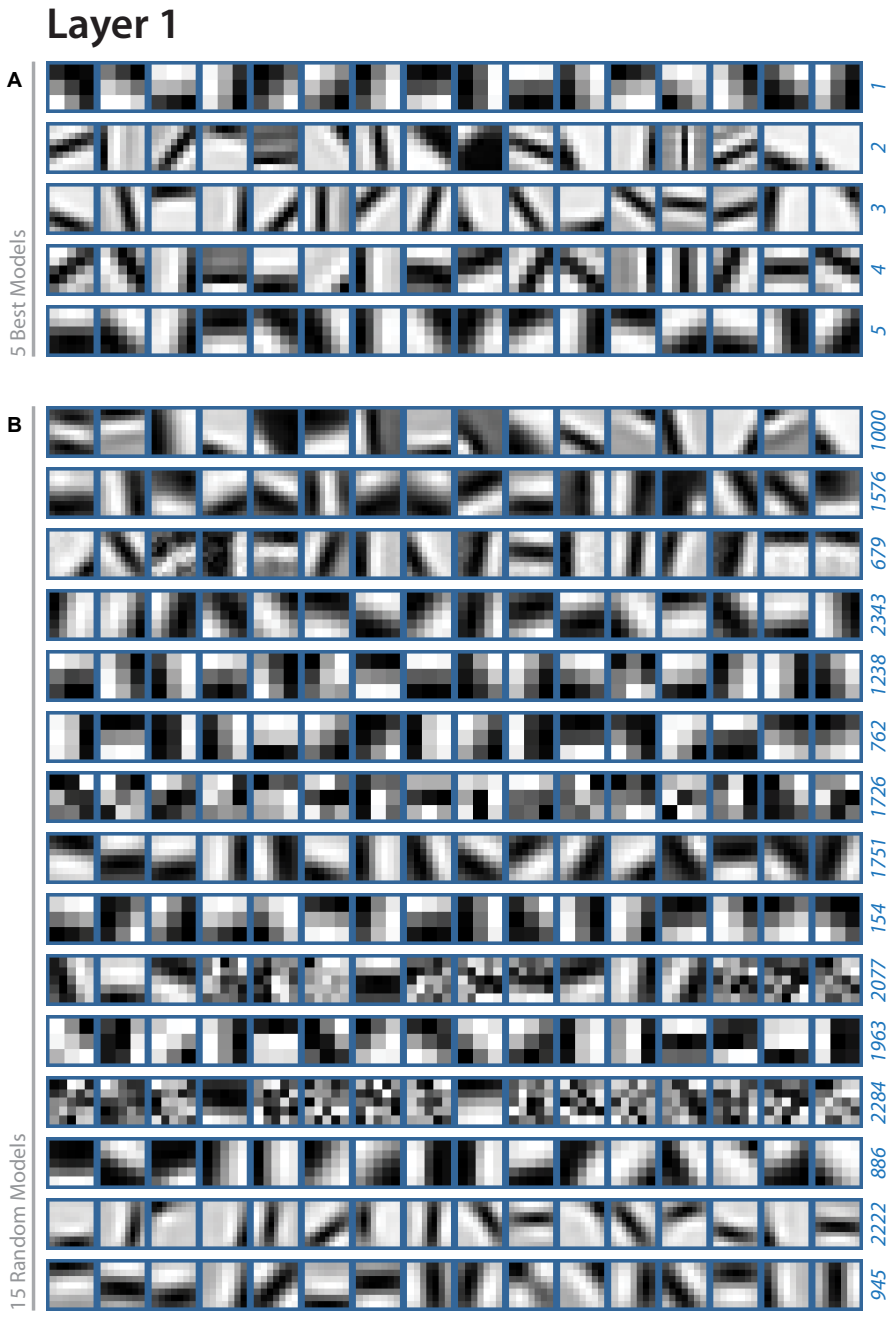



**Figure 9.10: Examples of Layer 1 filters taken from different models.** A random assortment of linear filter kernels taken from the first layers of the top five (a) and fifteen randomly chosen other model instantiations (b) taken from the "Law and Order" petri dish. Each square represents a single two-dimensional filter kernel, with the values of each filter element represented in gray scale (the gray-scale is assigned on a per-filter basis, such that black is the smallest value found in the kernel, and white is the largest). For purposes of comparison, a fixed number of filters were taken from each model's Layer 1, even though different models have differing number of filters in each layer. Filter kernels are initialized with random values and learn their structure during the *Unsupervised Learning Phase* of model generation. Interestingly, oriented structures are common in filter from both the top five models and from non-top-five models.

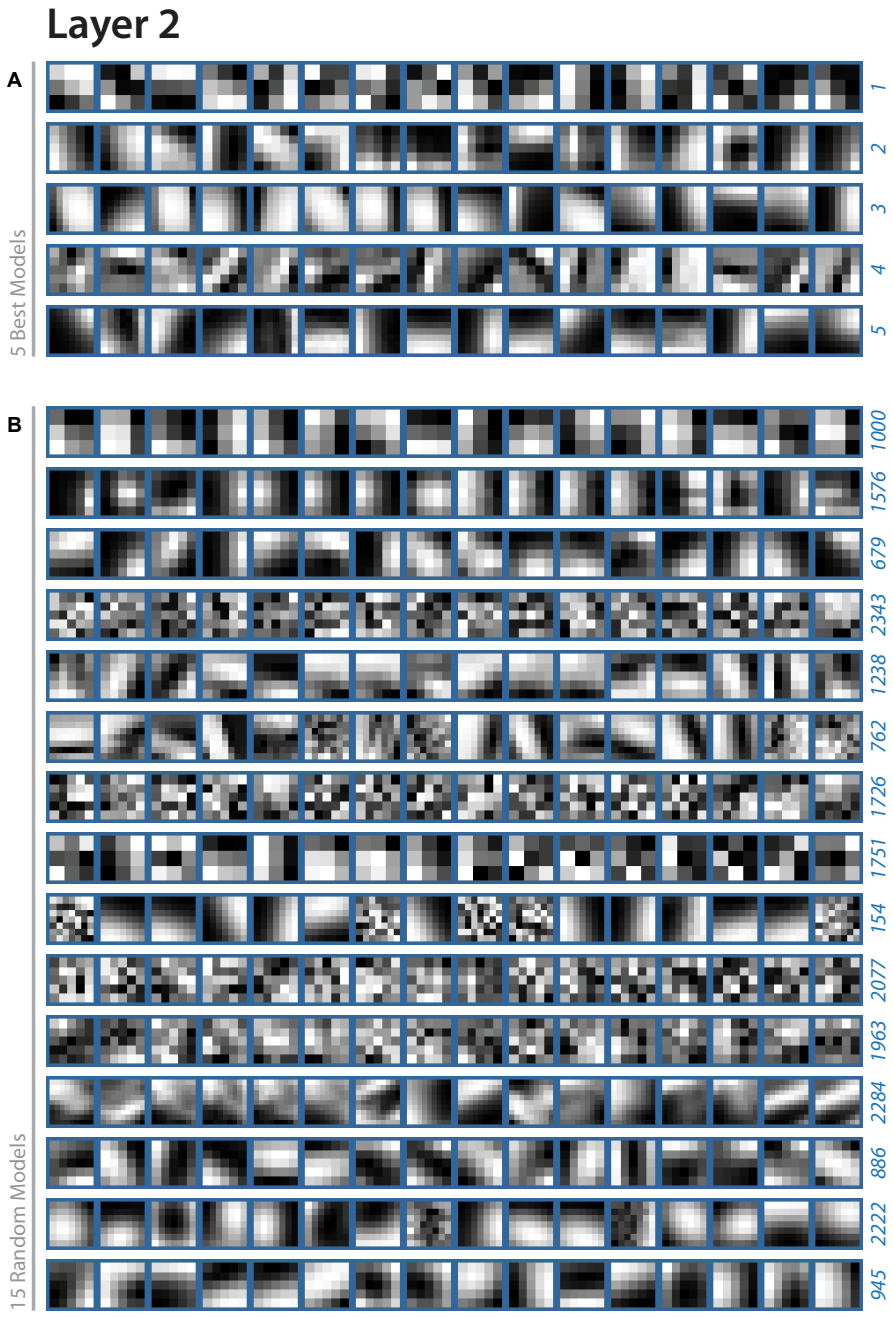



**Figure 9.11: Examples of Layer 2 filters taken from different models.** Following the same basic convention as in Supplemental Figure S3, a random assortment of portions of filter kernels from Layer 2 of the top five (a) and fifteen other randomly-chosen model instantiations (b) are shown in gray-scale to provide a qualitative sense of what the linear filters (produced as a result of the *Unsupervised Learning Phase*) look like. Note that since each Layer 1 is itself a stack of $k^{\ell=1}$ two-dimensional planes (or "feature maps") resulting from filtering with a stack of $k^{\ell=1}$ filters (see Supplemental Text S1 and Supplemental Figure S6, each Layer 2 filter is actually a $f_s^{\ell=2} \times f_s^{\ell=2} \times k^{\ell=1}$ kernel For the sake of visual clarity, we here present just one randomly-chosen $f_s^{\ell=2} \times f_s^{\ell=2}$ "slice" from each of the randomly-chosen filters. As in Supplemental Figure S3, there are signs of "structure" in the filters of both the top five and non-top-five models.

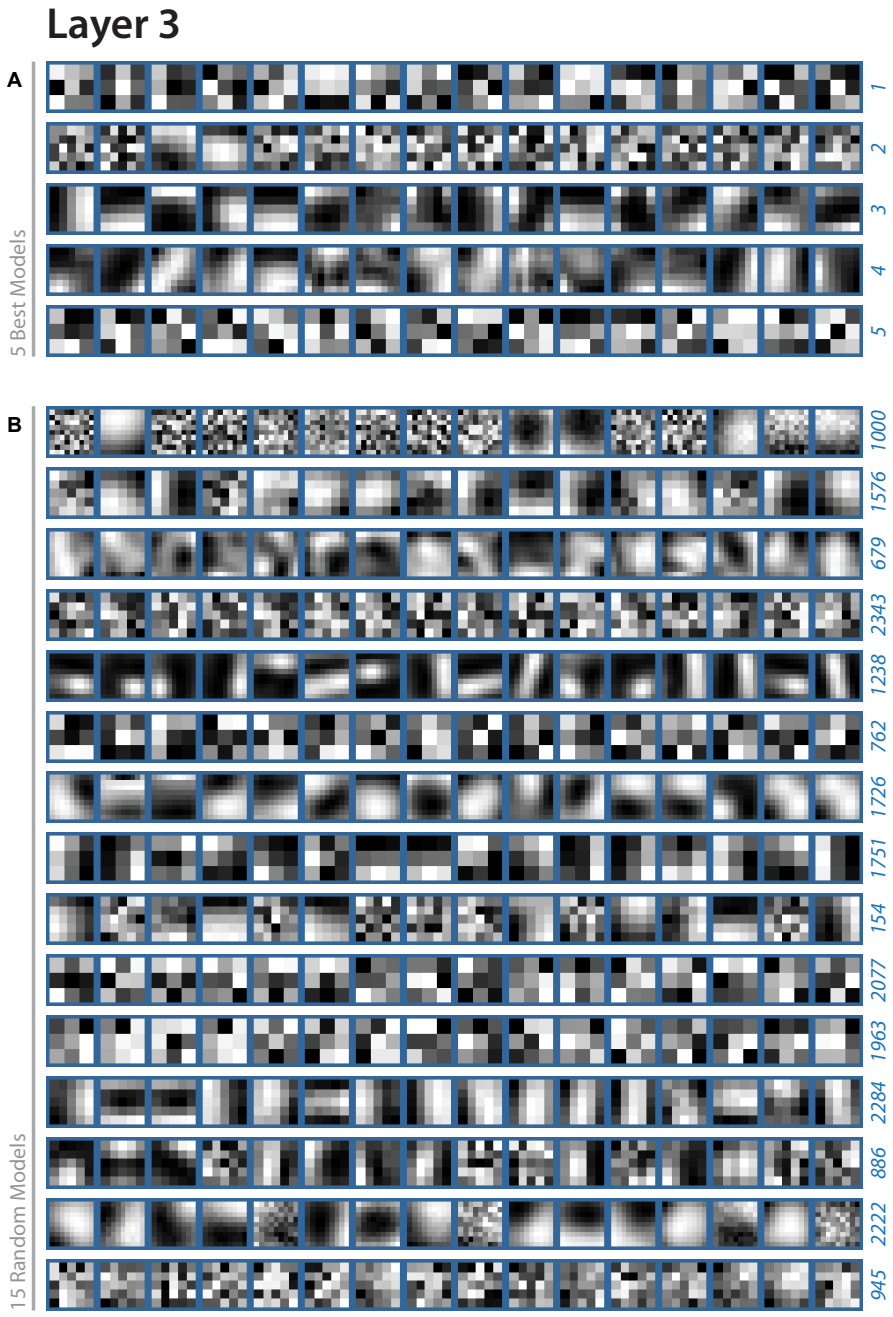



**Figure 9.12: Examples of Layer 3 filters taken from different models.** Following the same basic convention as in Supplemental Figures S3 and S4, a random assortment of portions of filter kernels from Layer 3 of the top five (a) and fifteen other randomly-chosen model instantiations (b) are shown in gray-scale to provide a qualitative sense of what the linear filters (produced as a result of the *Unsupervised Learning Phase*) look like. Note that since each Layer 2 is itself a stack of $k^{\ell=2}$ two-dimensional planes (or "feature maps") resulting from filtering with a stack of $k^{\ell=2}$ filters (see Supplemental Text S1 and Supplemental Figure S6), each Layer 3 filter is actually a $f_s^{\ell=3} \times f_s^{\ell=3} \times k^{\ell=2}$ kernel. For the sake of visual clarity, we here present just one randomly-chosen $f_s^{\ell=3} \times f_s^{\ell=3}$ "slice" from each of the randomly-chosen filters. As in Supplemental Figures S3 and S4, there are signs of "structure" in the filters of both the top five and non-top-five models.

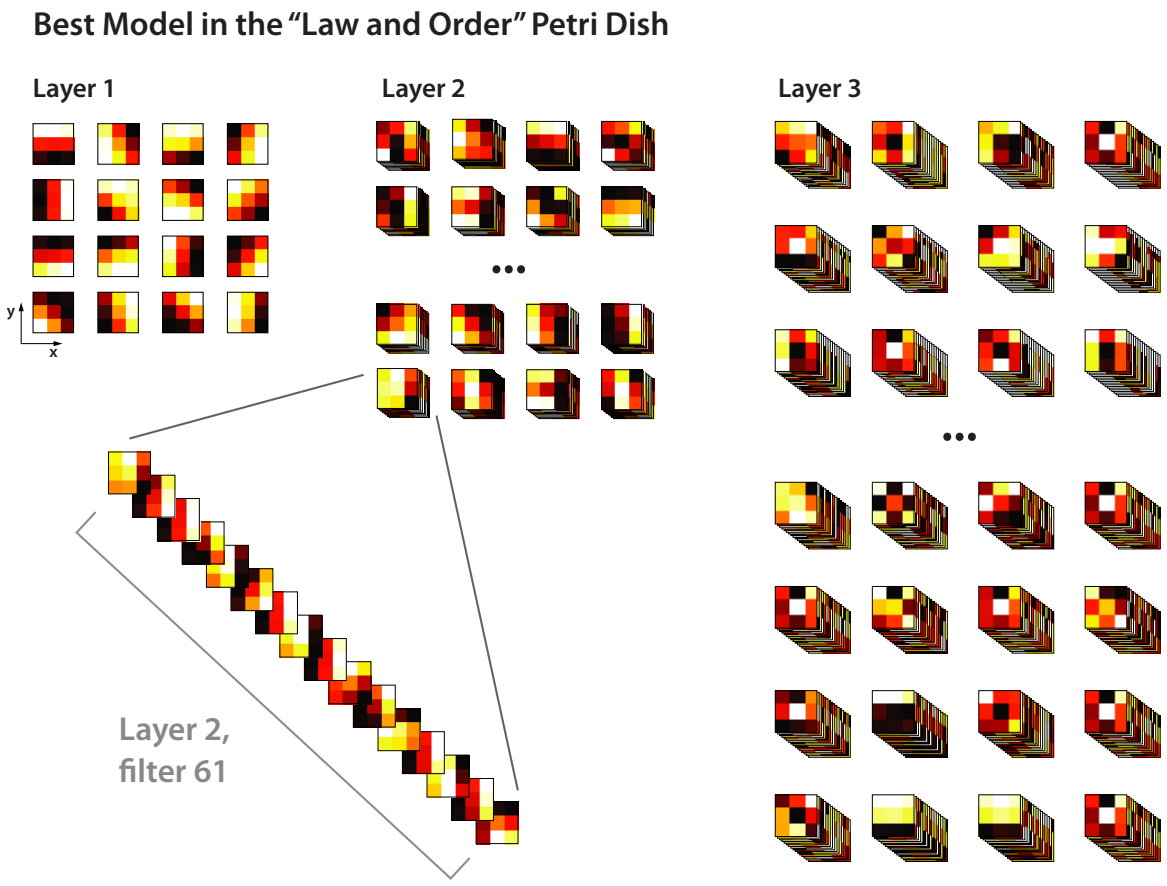



**Figure 9.13: Example filterbanks from the best model instantiation in the "Law and Order" Petri Dish.** Filter kernels were learned during the *Unsupervised Learning Phase*, after which filter weights were fixed. Colors indicate filter weights, and were individually normalized to make filter structure clearer (black-body color scale with black indicating the smallest filter weight, white representing the largest filter weight). The filter stack for each layer consists of $k^{\ell}$ filters, with size $f_s$. Because the Layer 1 filterbank for this model includes 16 filters, the Layer 1 output will have a feature "depth" of 16, and thus each Layer 2 filter is a stack of 16 $f_s \times f_s$ kernels. One filter (filter 61) is shown expanded for illustration purposes. Similarly, since the Layer 2 filterbank in this example model includes 64 filters, the output of Layer 2 will have a depth of 64, and thus each filter in Layer 3 filterbank must also be 64-deep.

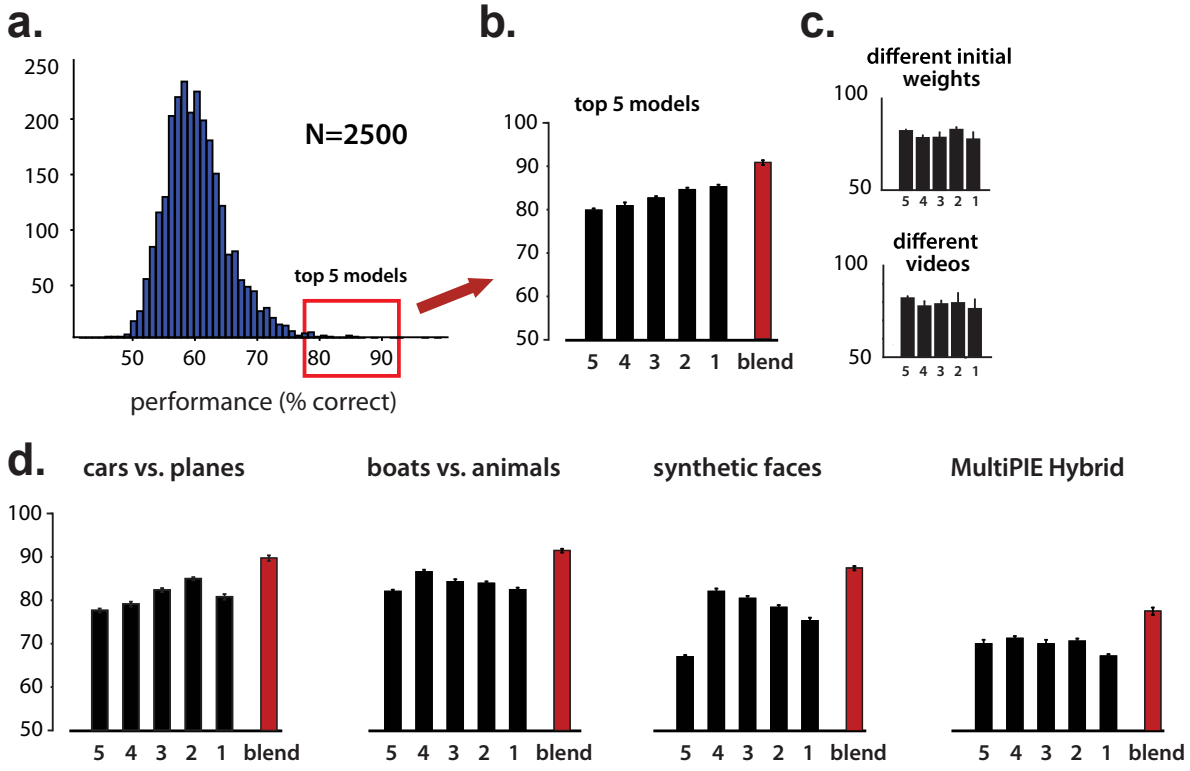



**Figure 9.14: High-throughput screening in the "Cars and Planes" Petri Dish.** Data are shown according to the same display convention set forth in the main paper. (a) Histogram of the performance of 2,500 models on the "Cars vs. Planes" screening task. The top five performing models were selected for further analysis. (b) Performance of the top five models (1-5). (c) Performance of the top five models when trained with a different random initialization of filter weights (top) or with a different set of video clips (bottom). (d) Performance of the top five models from the *Screening Phase* on a variety of other object recognition challenges.

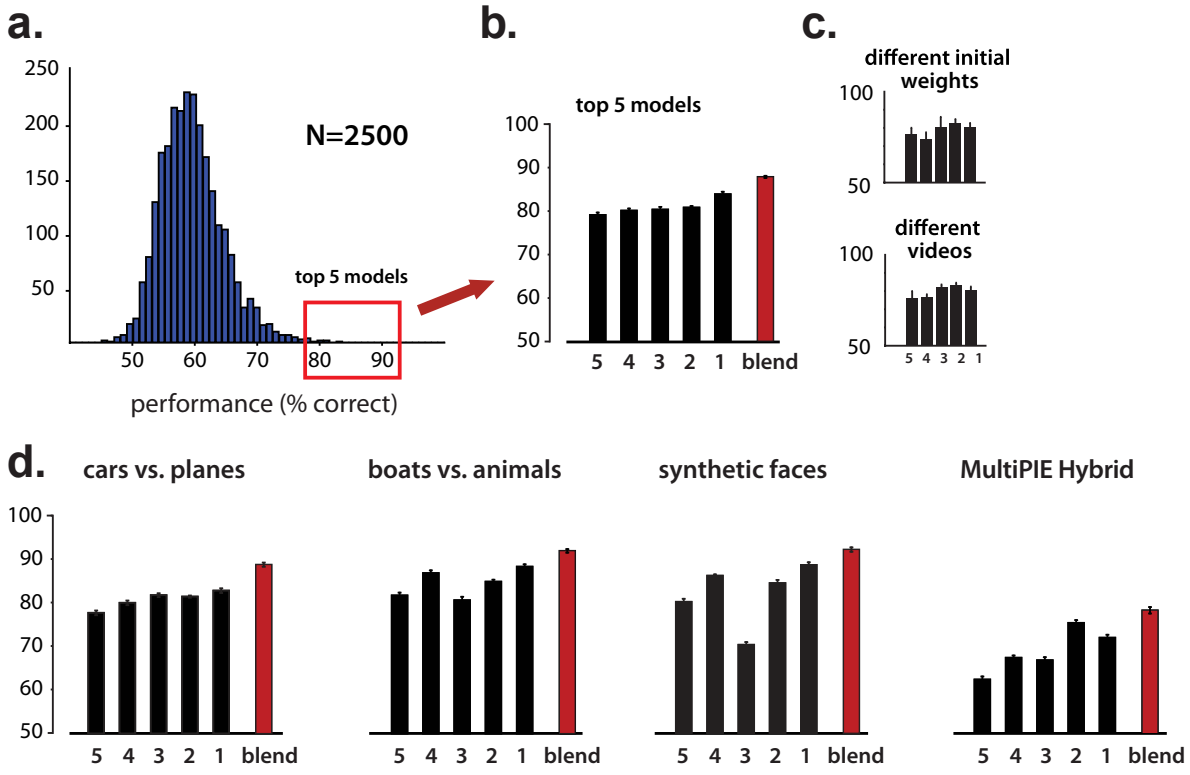



**Figure 9.15: High-throughput screening and validation in the "Boats" Petri Dish.** Data are shown according to the same display convention set forth in the main paper. (a) Histogram of the performance of 2,500 models on the "Cars vs. Planes" screening task. The top five performing models were selected for further analysis. (b) Performance of the top five models (1-5). (c) Performance of the top five models when trained with a different random initialization of filter weights (top) or with a different set of video clips (bottom). (d) Performance of the top five models from the *Screening Phase* on a variety of other object recognition challenges.

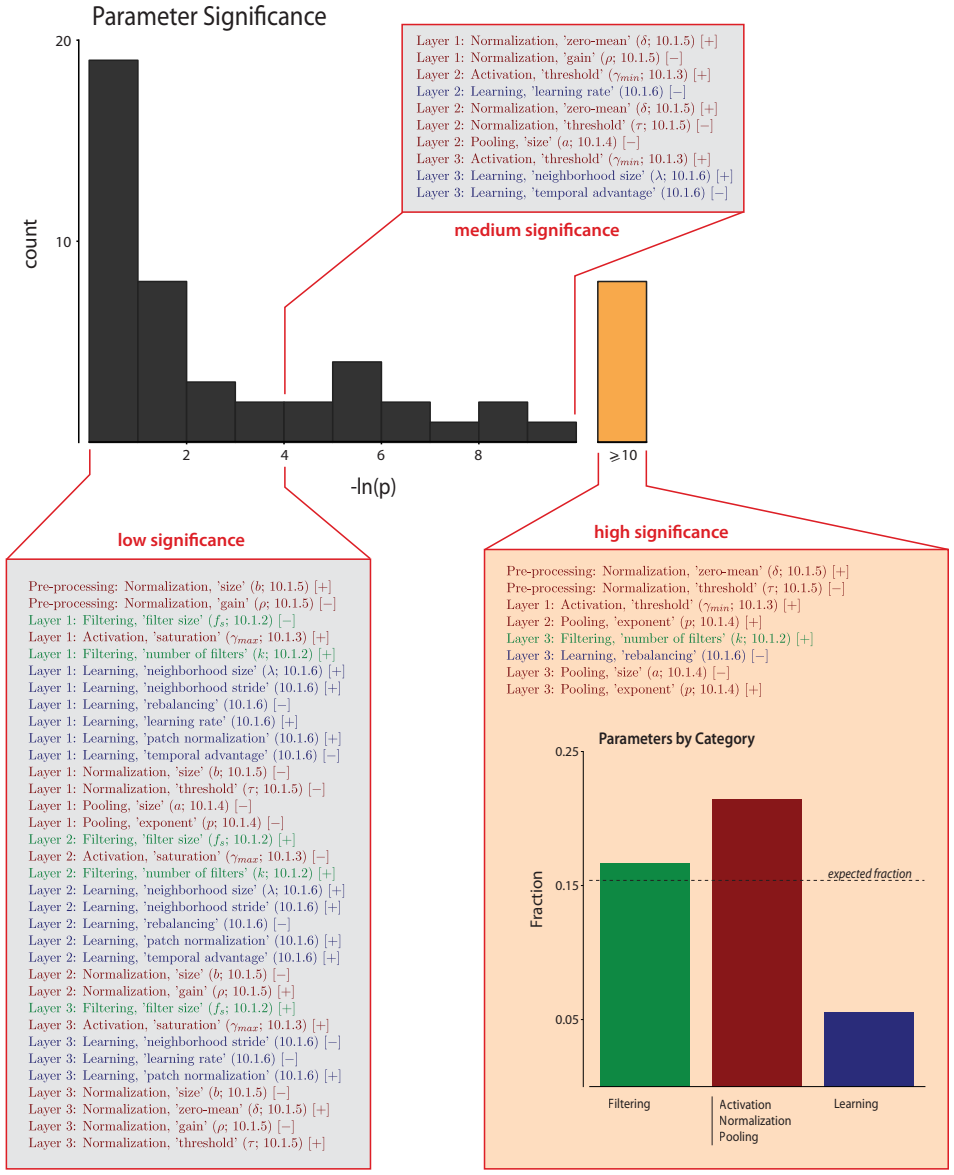



**Figure 9.16: Linear regression analysis of relationship between parameter values and model performance.** As a first-order analysis of the relationship between model parameters and model performance, we performed a linear regression analysis in which the values of each of the 52 parameters were included as predictors in a multiple linear regression analysis. Next, p-values were computed for the t statistic on each beta weight in the regression. A histogram of the negative natural log of the p-values is shown here, with the bin including significant p-values highlighted in orange (each count corresponds to one model parameter). For reference, the histogram is divided into three ranges (low-nonsignificant, medium-nonsignificant, and significant) and a listing of parameters included each significance range is printed below the histogram. Each parameter listing includes a 1) verbal description of the parameter, 2) its symbol according to the terminology in the Supplemental Methods, 3) the section number where it is referenced, and 4) whether it was positively ("+") or negatively ("-") correlated with performance. In addition, the parameters were divided into three rough conceptual groups and were color-coded accordingly: Filtering (green), Normalization/Activation/Pooling (red), and Learning (blue). Beneath the bin corresponding to significantly predictive parameters, a bar plot showing the fraction of each group found in the set of significant parameters. The expected fraction, if the parameters were distributed randomly, is shown as a dotted line. Activation/Normalization/Pooling parameters were slightly over-represented in the set of significantly-predictive parameters, but no group was found to be significantly over- or under-represented ($p = 0.338$; Fischer's exact test).

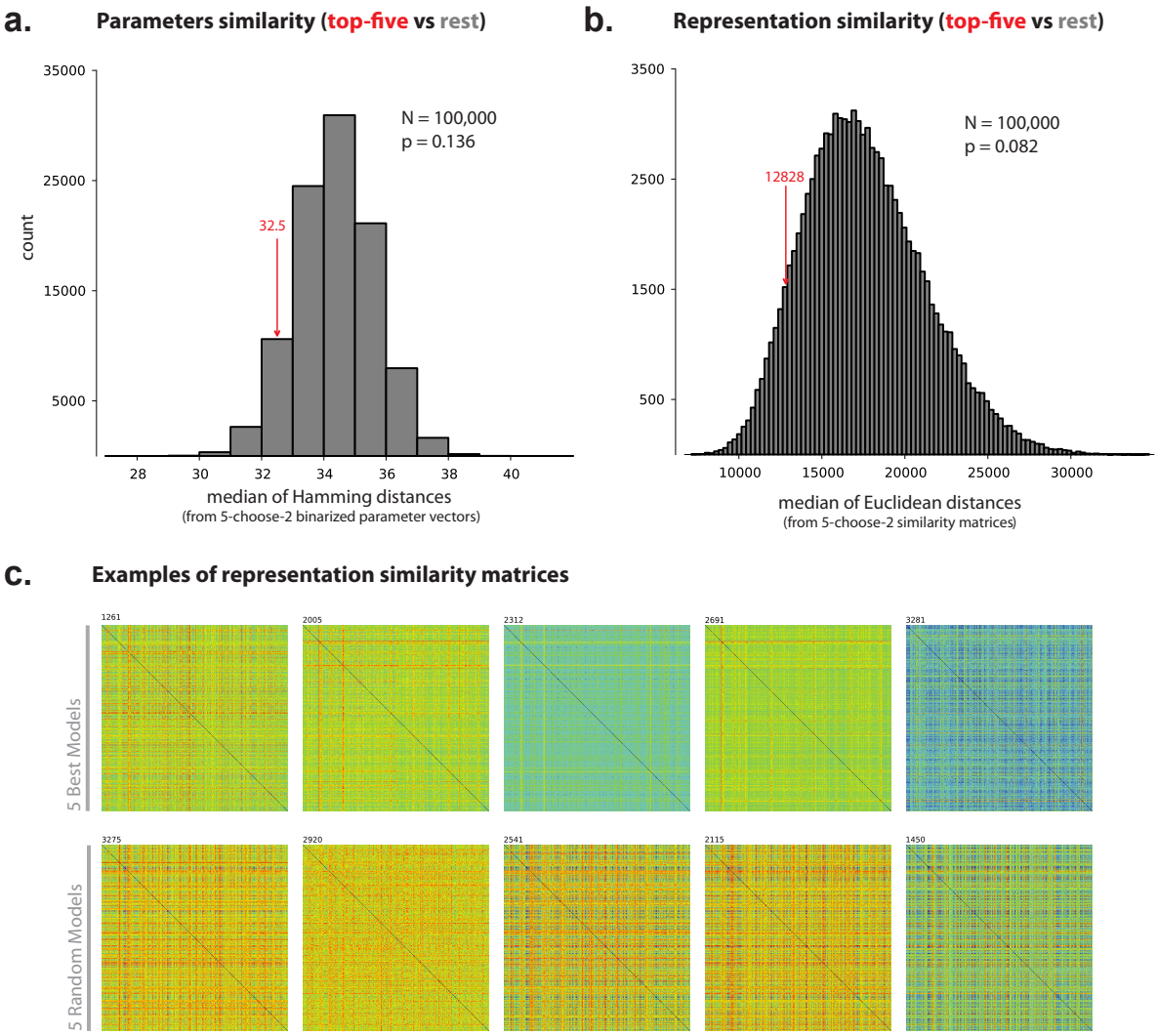



**Figure 9.17: How similar are the top models?** (a) *Model similarity on the basis of parameter values ($\ell_0$ or Hamming Distance).* Each model is specified by a vector of 52 parameter values. As a first attempt at comparing models, we generated an expanded binary parameter vector in which every possible parameter/value combination was represented as a separate variable (e.g. a parameter $\omega$ that can take on values 3, 5, and 7 would be included in the expanded vector as three binary values [$\omega = 3$], [$\omega = 5$], and [$\omega = 7$]). The Hamming distance distance between any two vectors can then serve as a metric of the similarity between any two models. In order to determine if the top five models taken from the "Law and Order" petri dish were more similar to each than would be expected of five randomly selected models, we computed the median pairwise Hamming distance between the top five models, and between a random sampling of 100,000 sets of five models taken from the remaining (non-top-five) models. The distribution of randomly selected model pairs is shown in (a), and the observed median distance amongst the top five models is indicated by an arrow. The top-five models tended to be more similar to one another than to a random selection of models from the full population, but this effect was not significant ($p = 0.136$; permutation test). (b) *Model similarity on the basis of output ("Representation" similarity).* As another way to compare model similarity, for each model we computed model output vectors for a selection of 600 images taken from the Screening task image sets. We then computed the $\ell_2$ (Euclidean) distance matrix between these "re-represented" image vectors as a proxy for the structure of the output space of each model. A distance metric between any two models was then defined as the $\ell_2$ distance between the unrolled upper-diagonal portion of the two models' similarity matrices (this distance metric is similar to the Frobenius norm). Finally, as in (a), the median distances between the top five models and between a collection of 10,000 randomly drawn sets of five models were computed. The histogram in (b) shows the distribution of median distances from randomly drawn sets of five models, and the arrow indicates the median distance observed in the top-five set. As in (a), the top-five models tended to be more similar to one another (lower distance), but this effect was not significant ($p = 0.082$; permutation test).

# Acknowledgments

We would like to thank Tomaso Poggio and Thomas Serre for helpful discussions; Roman Stanchak, Youssef Barhomi and Jennie Deutsch for technical assistance, and Andreas Klöckner for supporting PyCUDA.

This study was funded in part by the NVIDIA Graduate Fellowship, The National Institutes of Health (NEI R01-EY014970), The McKnight Endowment for Neuroscience, Dr. Gerald Burnett and Marjorie Burnett and The Rowland Institute of Harvard. Hardware support was generously provided by the NVIDIA Corporation.

Chapter 10

# GPU Meta-Programming and Auto-Tuning: A Case Study in Biologically-Inspired Computer Vision*

"Premature optimization is the root of all evil."

*Donald Knuth*

We here describe a tutorial on ways that meta-programming techniques – dynamically generating specialized code at runtime and compiling it just-in-time – can be used to greatly accelerate an implementation. We use filterbank convolution, a key component of the biologically-inspired machine vision systems that form the core of our research program, as a case study to illustrate these techniques.

We present an overview of several key themes in template meta-programming, and culminate in a full example of GPU auto-tuning in which an instrumented GPU kernel template is built and the space of all possible instantiations of this kernel is automatically grid-searched to find the best implementation on various hardware/software

*This chapter presents work done in collaboration with David D. Cox and will appear in the book "GPU Computing Gems Vol.2" [Pinto and Cox, 2011b].

platforms. We show that this method yields significant speed-ups over hand-tuned GPU kernels.

## 10.1 Problem Statement and Context

In recent years, digital cameras have become increasingly inexpensive and ubiquitous, and cameras are now embedded in a wide array of devices, from cellphones to cars. This explosion in imaging has led to enormous opportunity in the field of computer vision, as the need grows for algorithms that can automatically analyze, organize, and react to the new torrent of digital imagery.

While traditional machine vision algorithms achieve modest success in certain task (e.g. detecting the presence of a face in an image), many other visual tasks that come easily for humans remain extremely challenging for computers (e.g. recognizing the identity of a particular face).

Inspired by the ease with which the human brain is able to solve visual tasks, many so-called "biologically-inspired" vision approaches have emerged. The basic architecture of the brain-inspired vision systems that we work with is shown in schematic form in Figures 9.2 and 9.8. Briefly, this architecture consists of a cascade of multiple layers of linear filtering operations and static nonlinearities. Learning algorithms adjust the parameters of these operations and adapt the network to a given kind of input. Models from this class have been shown to achieve state-of-the-art performance in variety of object and face recognition domains (see Parts II and IV, and Chapter 9).

However, there are two major challenges associated with the construction of biologically-inspired vision systems. First, the brain is a massively parallel computer, with millions of processing elements; mimicking this level of computational power is not a trivial undertaking. Second, while biology has given us some hints about how to organize a biologically-inspired visual system, neuroscience experiments have provided us to date with relatively few constraints on the details and parameters to be used. As a result, we are often left exploring the *space* of possible models, rather than evaluating any one model, *per se*.

The combination of these two challenges raises a unique problem in high-performance

computing. The scale of the models to be studied demands enormous compute power. Properly utilized, GPUs begin to provide the level of power needed to undertake serious large-scale visual system modeling. However, the second challenge — the need to explore a wide range of different kinds of biologically-inspired models — poses a serious challenge for optimization. Optimization is often an exercise in specialization: an algorithm is carefully matched to set of hardware/software resources, exploiting as much regularity in the underlying problem and inputs as possible. For instance, if an input image is always known to be small, or of a power-of-two in width and height, such information can be exploited to craft an optimal implementation. However, for our problem, we must build algorithms that can tolerate widely varying inputs, since a major drive of our work is to find model parameters that can provide high levels of generic object recognition performance.

Here, we focus on one key sub-operation of our models: filterbank convolution (i.e. the application of a number of filters to an incoming image, in parallel). We show how this operation can be performed efficiently, in spite of widely varying conditions (e.g. different sizes and number of filters) using a dynamically-specialized template meta-programming approach.

## 10.2 Core Method

Optimization is often an exercise in specialization — increasing the performance of a given algorithm on a given piece of hardware requires taking advantage of specific details of the algorithm, the kinds of inputs it will see, and the resources available in the hardware (and software stack) on which the algorithm will run. In many contexts, however, we must operate within a indeterminate, broadly-defined, or changing parameter space, where we cannot be guaranteed that all inputs will look similar. In our own research, a key computational bottleneck is a compute-intensive filterbank 3D convolution operation (which mimics one part of the processing that is thought to be done by biological neurons). However, while such operations are straightforward to optimize using standard techniques when all of the relevant dimensions of the input and filters are known, in our research, we must search a wide range of possible neural network ar-

chitectures to find those that are best at specific tasks (see Chapter 9). Thus, we need high-performance solutions not just to one particular problem, but a family of possible instantiations.

We employ a meta-programming approach in which we use the high-level language Python, to generate parameterized CUDA kernel code using a string template engine. These generated kernels are then just-in-time compiled and run with the "behind-the-scenes" help of the PyCUDA toolkit [Klöckner *et al.*, 2009]. Meta-kernels such as these allow the developer to achieve a high degree of flexibility in exploring many different optimization strategies while avoiding unnecessary hand-coding or combinatorial issues when multiple strategic decisions interact. Finally, we show how meta-parameter selection can be turned over to an automated process, allowing for auto-tuning of kernels across different inputs (where resource requirements vary wildly) and different hardware generations (where resource availability varies).

While Python's strength in string processing (e.g. Cheetah, Mako, Jinja), general scientific computing (e.g. Numpy, Scipy, Matplotlib), and GPU programming (e.g. PyCUDA, PyOpenCL) provides a particularly convenient platform for template meta-programming, the techniques described here could just as well be applied in any programming language. Indeed, the only language facilities one needs to apply meta-programming are a means of transforming a template string (available in one form another in effectively every language), and a means to compile and run a kernel (which is readily available in both CUDA and OpenCL).

In this case study, we describe several basic regimes where meta-programming can lead to cleaner code, improved understanding, and ultimately better performance. We discuss how a meta-programming philosophy can lower the cost of trying out new ideas, since these ideas can be merged into an existing common code base, and how some relatively simple template-driven features (such as full/partial loop unrolling) can produce significant speed-ups without cluttering code. Finally, we show how meta-parameter auto-tuning can yield significant speed-ups, by generating many dynamically specialized kernels from a single, understandable kernel template. A sample of our approach to this problem is presented below.

## 10.3 Algorithms, Implementations, and Evaluations

### 10.3.1 Towards General, Optimized Code

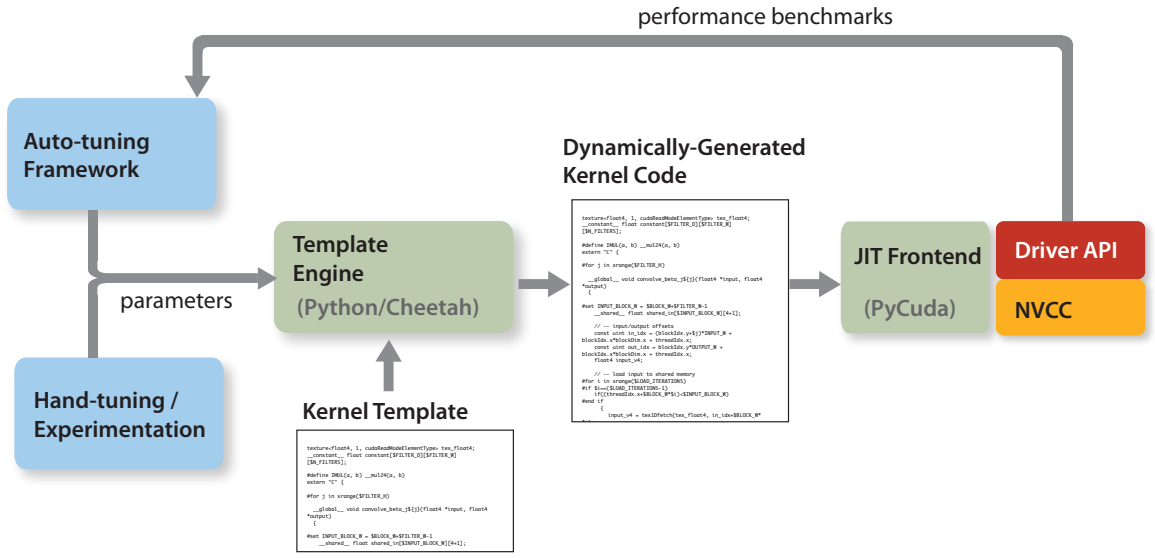



**Figure 10.1: A schematic diagram of a basic GPU meta-programming system.**

Figure 10.1 shows the basic components of a GPU meta-programming approach. The central insight is that rather than writing GPU kernels, one can write kernel *templates*, with multiple optimization strategies and optional paths embedded in a single piece of code. With an appropriate template transformation module, a dynamically-specialized kernel is generated based on given set of parameters, and this code is just-in-time compiled. The choice of parameters can either be done by manually, or it can be turned over to an auto-tuning algorithm. We will begin by discussion general strategies for template meta-programming, and will close with a complete auto-tuning example.

In the following examples, we use Python, a high-level language, in combination with PyCUDA, a package that seamlessly wraps the CUDA Driver Application Programming Interface (API). While the use of an interpreted language might initially seem at odds with the goals of high-performance computing, the fraction of time spent in the "outer loop" of an High-Performance Computing (HPC) application is typically small relative

to that spent in a handful of critical sections.

A key advantage of just-in-time compilation of a kernel from a string is that new kernels can easily be generated on the fly. Below, using a standard Python string templating package, Cheetah[1], we can substitute values into a kernel, evaluate conditionals, and generate iterated structures using loops. As an example (the details of which are not important, for the moment):

```
...
#for nk in xrange($N_KERNELS)
  __global__
  void cudafilter_kernel_${nk}
  (
   float4 *input
#for o in xrange($N_OUTPUT4S)
   , float4 *output$o
#end for
   )
  {
    // -- Shared-memory buffer for the input tiles
    __shared__ float shared_in                \
      [$BLOCK_H]                              \
      [$N_FILTER_ROWS]                        \
      [$INPUT_D]                              \
      [$INPUT_BLOCK_W + ${int($PAD_SHARED)}];
...
  }
```

At run time, instances of `$N_KERNELS`, for instance, are replaced by a value passed into the template engine in a dictionary, and conditionals (`#if`) and loops (`#for`) are evaluated and expanded. Note that the template allows for the manipulation of structures that are not easily accessible in C code: variable length argument lists, run-time generation of new functions and variable names. The resulting kernel code is then com-

[1] `http://www.cheetahtemplate.org`

piled and called entirely from Python. Such dynamic code generation opens up a wide range of possibilities, some of which are described below.

### 10.3.2 Syntax-Level Code Control

One major advantage of a template meta-programming approach is that it make it easy to produce CUDA kernels that would otherwise be tedious or error-prone to produce by hand. One such area is fine-controlled loop unrolling: performance gains can often be achieved by avoiding `for` loops and unrolling a loop manually. In a template meta-programming context, this is easy. For example, in the inner loop of our filtering operation, we could write:

```
// Loop unrolling example
#for d in xrange($FILTER_D)
    #for i in xrange($FILTER_W)
        v = shared_in[threadIdx.x+$i][$d];
        #for n in xrange($N_FILTERS)
            w = constant[$d][$i][$n];
            sum$n += v*w;
        #end for
    #end for
#end for
```

Which would at runtime generate the following code, dependent on the values of `FILTER_D`, `FILTER_W` and `N_FILTERS`:

```
...
v = shared_in[threadIdx.x+0][0];
w = constant[0][0][0];
sum0 += v*w;
w = constant[0][0][1];
sum1 += v*w;
w = constant[0][0][2];
sum2 += v*w;
w = constant[0][0][3];
sum3 += v*w;
```

```
v = shared_in[threadIdx.x+1][0];
w = constant[0][1][0];
sum0 += v*w;
w = constant[0][1][1];
...
```

One consequence of the above template-based approach, is that we can generate distinct, specialized versions of kernels with very fine control for, say, a Nx4x4x8 filterbank and for a Nx8x8x4 filterbank. Creating such specialized version by hand would be tedious and/or error-prone.

Of course, loop unrolling isn't universally beneficial; unrolling a loop consumes a larger number of registers, which, depending on what register usage in the rest of the kernel, can lead to dramatic decreases in performance if the number of available registers is exceeded. In addition to allowing a straightforward framework for turning on and off unrolling, it is easy to imagine how the above example could be updated to support *partial* loop unrolling, balancing register usage against branching costs.

Control of a kernel's code at a syntactic level also enables one to have a much more fluid relationship with memory resources. Since templates operate at the level of syntax, it is, for instance, possible to use registers as an "indexable" resource (e.g. `register_masquerading_as_array$index`) without overly complexifying code. Conversely, if one is register-limited, one can instead use shared memory resources as if they were registers (a technique known as "register spilling").

### 10.3.3 Exploring Design Decision Space More Freely

Another important area where meta-programming can be highly valuable is in exploring a space of design decisions. A GPU developer is typically confronted with a multitude of decisions when developing a CUDA implementation: what kind of memory to use and how to access it (e.g. linear memory, tex1D, tex2D, etc.), how to layout data in memory, etc.

In some cases, subtle decisions can produce large differences in performance. Below we illustrate one such design decision, wherein a given filter weight is stored in constant

memory (organized as filter x height x width), and the kernel is either configured to either index the filter to be computed by thread index, or by a constant (with multiple filter responses being computed through multiple kernel executions). A templated example is shown below, wherein the inner loop is conditioned (at template-time) by the variable `USE_THREAD_PER_FILTER`:

```
# -- Constant memory usage example
#for i in xrange($FILTER_W)
    #for k in xrange($FILTER_D)
        v = shared_in[(threadIdx.x+$i)][$k];
        #if $USE_THREAD_PER_FILTER
            w = constant[threadIdx.y][$i][$k];
        #else
            w = constant[$FILTER_ID][$i][$k];
        #end if
        sum += v*w;
    #end for
#end for
```

Benchmarking the code, we find that our performance is cut in half when `USE_THREAD_PER_FILTER` is true. Here, inspection of the disassembled cubin (using the *decuda* disassembler) is instructive. A snippet from the disassembled cubin when `USE_THREAD_PER_FILTER` is false is shown below:

```
...
mad.rn.f32 $r0, s[$ofs2+0x0000], c0[$ofs2+0x0000], $r0
mad.rn.f32 $r0, s[$ofs2+0x0008], c0[$ofs2+0x0008], $r0
mad.rn.f32 $r0, s[$ofs2+0x000c], c0[$ofs2+0x000c], $r0
mad.rn.f32 $r0, s[$ofs2+0x0010], c0[$ofs2+0x0010], $r0
...
```

Here, each `mad` is a multiply-add instruction, and the cubin is dominated by a long, back-to-back stream of arithmetic. If, on the other hand, `USE_THREAD_PER_FILTER` is true, we see a very different pattern of instructions:

```
...
```

```
mad.rn.f32 $r4, s[$ofs3+0x0000], $r4, $r1
mov.b32 $r1, c0[$ofs2+0x0008]
mad.rn.f32 $r4, s[$ofs3+0x0008], $r1, $r4
mov.b32 $r1, c0[$ofs2+0x000c]
...
```

Here, multiply-adds are interleaved with "move" instructions, resulting in a computation that is vastly less efficient.

It should be noted that, strictly speaking, this issue has *now* been properly documented in the CUDA developers guide – indexing constant memory by a `threadIdx` leads to sub-optimal constant memory access patterns. However, the issues here are subtle, and easy to miss. Meta-programming provides a mechanism for flexibly exploring multiple optimization paths at the same time, which provides a powerful tool for understanding the hardware at a deeper level. Importantly, not all aspects of the hardware contributing to performance are (or reasonably can be, e.g. for proprietary/-competitive reasons) documented. In such cases, template meta-programming lowers the cost of trying different strategies, allowing one to exploit the hardware much more fully.

There is also no guarantee that the many design choices that one must make are independent. That is, at given stage of optimization, a particular path may legitimately represent the best one, given the current context of the rest of the program. However, as other aspects of the design are tweaked and iterated, there is no guarantee that these earlier insights still hold, especially as different approaches differential tax the scarce resources of the hardware (e.g. registers, shared memory, memory bandwidth). Meta-programming yields the significant advantage that intermediate design decisions can be made explicit and both "forks" in the path can be kept in place, without incurring actual, performance-eroding if/then branches in the kernel itself. This approach frees up the developer to revisit past choices, without incurring a combinatorial explosion of separate pieces of code. Retesting sets of assumptions can be done frequently and programmatically from the "outer" framework of code.

### 10.3.4 Auto-Tuning

Above, we have shown how meta-programming can assist in the manual optimization of an implementation in a regime where the parameters of the input (e.g. input size, filter size, number of filters, etc.) are fixed. Our optimization process also implicitly assumes a given hardware context, unless we explicitly tweak our designs on different generations and grades of GPUs. In reality, our particular problem-of-interest (like many problems) demands that our implementation must work over a wide range of input parameters, and on a variety of different kinds of hardware. Ideally, we'd like to have the best possible implementation in each context, without having to undertake a massive effort in hand-tuning. A powerful extension of template meta-programming is *auto-tuning*: allowing software to choose the best set of meta-parameters for a given set of inputs and hardware / software stack.

Here, we take the simplest possible approach to auto-tuning, performing a coarse grid search across a range of possible meta-parameter values.

The meta-parameters to be tuned can include:

1. Degree of Loop Unrolling
2. Register Spilling
3. Memory structure type – e.g. linear memory, tex1D, tex2D, etc.
4. Block and Grid dimensions
5. Number of filters to compute per kernel invocation (i.e. thread work size)
6. Shared-memory padding
7. etc.

Pseudo-code demonstrating a basic auto-tuning algorithm is shown below; code for a proof-of-concept auto-tuned implementation of filterbank convolution is available upon request.

```
"""
=======================================================
Auto-Tuning Pseudo-Code (Filterbank Convolution)
=======================================================

Parameters
----------
 arr:    input array (height x width x depth)
 fb:     filter-bank (nfilters x fsize x fsize x depth)
"""

# -- Get the set of meta-programming / templating parameters
# to explore during auto-tuning
mp_set = get_metaprog_parameter_set()

# -- Get informations about the hardware
# (may include GPU architecture, host CPU, memory, etc.)
hw = get_hardware_specs()

# -- Get informations about the software stack
# (may include CUDA SDK version, CUDA Driver version, etc.)
sw = get_software_specs()

db = autotuning_db() # the auto-tuning database

# -- Has this combination of input array, filter-bank,
# hardware and software already been tuned ?
if (arr, fb, hw, sw) not in db:

    # -- If not, we'll loop over each element in the
    # meta-programming parameter set, gather timing
    # informations and select the fastest code
    n_warmups, n_runs = get_n_trials()
    best_func, best_time = None, inf
    for mp in mp_set:

        tmpl = get_gpu_src_template(arr, fb)

        # Render the code template (using e.g. Cheetah).
        # Note that only a subset of 'mp' will be used here
        # (e.g. unrolling factor, register spilling, etc.)
        gpu_src = render_gpu_src_template(tmpl, mp)

        # Compile the source code or retrieve it from a cache
        # (using NVCC through e.g. PyCUDA)
        # Note that only a subset of 'mp' will be used here
```

```
            # (e.g. using fast math, constraining the number
            # of registers through the compiler, etc.)
            gpu_bin = compile_and_cache_gpu_src(hw, sw, gpu_src, mp)

            # Load the GPU binary code and prepare the device
            # for execution (using the Driver API through e.g. PyCUDA)
            func = load_and_prepare_gpu_execution(gpu_bin, mp)

            # Warm up the GPU
            for _ in n_warmups: func()

            # Collect timings
            timings = list()
            for _ in n_runs:
                start = time()
                func()
                end = time()
                timings.append(end-start)

            # Is this version of the code faster?
            if median(timings) < best_time:
                best_time = median(timings)
                best_func = func

        # -- Add the result to the database
        db.add(arr, fb, hw, sw, best_func)

    else:

        best_func = db.get(arr, fb, hw, sw)

    # -- Return the best performing code
    return best_func
```

## 10.4 Final Evaluation

For the purposes of demonstration, we chose set of 73 meta-parameter configurations (i.e. 73 unique combinations values of the above meta-parameters) and auto-tuned for four different input parameter sets, which roughly bracket the range of possible input shapes and sizes that the are encountered in our experiments. The results of auto-tuning, on different NVIDIA GPUs spanning multiple generations of graphics hardware, multiple end-user markets (gaming versus professional), and a wide range of

variation in hardware-level resources available, are shown below:

| *GPU / SDK* | *Input* | *Filterbank* | *Default (gflops)* | *Auto-tuned (gflops)* | *Boost* |
|---|---|---|---|---|---|
| 8600GT CUDA2.3 | 256x256x8 | 64x9x9x8 | 5.493 ± 0.019 | 33.881 ± 0.068 | 516.8 % |
| | 512x512x4 | 32x13x13x4 | 11.619 ± 0.007 | 33.456 ± 0.045 | 187.9 % |
| | 1024x1024x8 | 16x5x5x8 | 19.056 ± 0.017 | 33.109 ± 0.632 | 73.7 % |
| | 2048x2048x4 | 4x8x8x4 | 23.824 ± 0.055 | 38.867 ± 0.118 | 63.1 % |
| 9400M CUDA3.1 | 256x256x8 | 64x9x9x8 | 2.177 ± 0.013 | 15.796 ± 0.049 | 625.6 % |
| | 512x512x4 | 32x13x13x4 | 5.562 ± 0.001 | 15.331 ± 0.004 | 175.6 % |
| | 1024x1024x8 | 16x5x5x8 | 2.309 ± 0.022 | 4.571 ± 0.015 | 98.0 % |
| 9600M GT CUDA3.1 | 256x256x8 | 64x9x9x8 | 6.710 ± 0.005 | 36.584 ± 0.023 | 445.2 % |
| | 512x512x4 | 32x13x13x4 | 13.606 ± 0.002 | 35.582 ± 0.003 | 161.5 % |
| | 1024x1024x8 | 16x5x5x8 | 20.034 ± 0.113 | 26.084 ± 6.243 | 30.2 % |
| | 2048x2048x4 | 4x8x8x4 | 25.781 ± 0.044 | 46.945 ± 0.100 | 82.1 % |
| C1060 CUDA2.3 | 256x256x8 | 64x9x9x8 | 104.188 ± 0.051 | 168.083 ± 0.372 | 61.3 % |
| | 512x512x4 | 32x13x13x4 | 125.739 ± 0.109 | 234.053 ± 0.266 | 86.1 % |
| | 1024x1024x8 | 16x5x5x8 | 144.279 ± 0.764 | 243.697 ± 0.346 | 68.9 % |
| | 2048x2048x4 | 4x8x8x4 | 180.060 ± 0.018 | 322.328 ± 0.348 | 79.0 % |
| GTX295 CUDA2.3 | 256x256x8 | 64x9x9x8 | 126.563 ± 0.590 | 262.848 ± 0.176 | 107.7 % |
| | 512x512x4 | 32x13x13x4 | 172.701 ± 0.014 | 317.108 ± 0.056 | 83.6 % |
| | 1024x1024x8 | 16x5x5x8 | 104.972 ± 0.011 | 168.298 ± 0.174 | 60.3 % |
| | 2048x2048x4 | 4x8x8x4 | 120.693 ± 0.020 | 226.534 ± 0.195 | 87.7 % |
| GTX285 CUDA2.3 | 256x256x8 | 64x9x9x8 | 123.396 ± 0.016 | 197.006 ± 0.219 | 59.7 % |
| | 512x512x4 | 32x13x13x4 | 143.277 ± 0.044 | 270.206 ± 0.209 | 88.6 % |
| | 1024x1024x8 | 16x5x5x8 | 148.841 ± 0.465 | 310.276 ± 0.538 | 108.5 % |
| | 2048x2048x4 | 4x8x8x4 | 205.152 ± 0.015 | 376.685 ± 0.070 | 83.6 % |
| GTX480 CUDA3.1 | 256x256x8 | 64x9x9x8 | 467.631 ± 19.100 | 471.902 ± 11.419 | 0.9 % |
| | 512x512x4 | 32x13x13x4 | 834.838 ± 8.275 | 974.266 ± 3.809 | 16.7 % |
| | 1024x1024x8 | 16x5x5x8 | 542.808 ± 1.135 | 614.019 ± 0.904 | 13.1 % |
| | 2048x2048x4 | 4x8x8x4 | 378.165 ± 0.537 | 806.628 ± 0.168 | 113.3 % |

Large performance gains are observed for the auto-tuned meta-kernels as compared

to the "default" parameter set, which was hand-picked to allow correct execution of all input ranges on all GPUs – without running up against hardware limitations.

Interestingly, we note that a different peak-performing meta-parameter set was chosen for each input size, and for different hardware platforms. Given the many demands on system resources that trade-off against each other, a different "sweet-spot" implementation exists for different incoming inputs and for different constellations of hardware resources. To illustrate this point, in Table 10.2 we show the performance with best auto-tuned parameters for two different hardware platforms (a 9400M laptop-grade GPU, and a GTX480 high-end desktop GPU), as well as the performance for each if the parameter sets were swapped (i.e. if we tuned on the 9400M and ran on the GTX480, and vice versa). In all cases, best parameter sets were chosen using half of the time trials, and the median performances shown in the table were computed using the remaining trials. We see large differences in performance (in some cases over 100%) when a custom hardware auto-tuned kernel is used, as compared to when an optimal kernel for a different platform is used. Such performance differences are particularly important when development is done on a different machine (e.g. a laptop) than where the code will be run in production mode. Similarly, for applications that are widely deployed on a variety of user hardware, optimal performance can be achieved by either optimizing *in situ* or shipping with a database of parameter sets for different platforms.

Similarly, in Table 10.3 we show the effect of tuning on one input configuration and running on another. Again, significant speed-ups are obtained using kernels tailored to a specific input configuration, as opposed to generic kernels optimized under different conditions. Without meta-programming, hand-tuning for each of the many hardware configuration in existence and for many different input configurations would be a tedious and error-prone process. By contrast, template meta-programming in combination with a simple auto-tuning scheme allows optimal implementations to be chosen for any platform and input size.

| | optimized for: | | |
|---|---|---|---|
| run on: | 9400M | GTX480 | tuning speedup |
| 9400M | 0.32s | 2.52s | 675% |

| | | | |
|---|---|---|---|
| GTX480 | 0.016s | 0.011s | 52% |

**Table 10.2:** Performance of auto-tuned implementations on two hardware platforms, including performance tuned on one platform and run on the other.

| | optimized for: | | |
|---|---|---|---|
| run on: | config1 | config2 | tuning speedup |
| config1 | 11.1ms | 15.7ms | 41% |
| config2 | fails | 10.8ms | not comparable |

**Table 10.3:** Performance of auto-tuned implementations on two input configurations, including performance tuned for one configuration and run with the other.

## 10.5 Future Directions

Above, we have demonstrated how writing kernel *templates*, rather than kernels *per se* can result in cleaner, more readable code, and can provide a coherent framework for exploring the interactions of many implementation decisions. In our auto-tuning example code, we show a straightforward implementation of a brute-force auto-tuning approach, in which we grid search a large number of combinations and permutations of template parameters and auto-benchmark. While this brute-force search procedure leads to surprisingly good results despite its simplicity, it clearly becomes suboptimal as the number of template parameters increases. Thus, an important future direction is the application of more intelligent optimization algorithms – e.g. decision trees, simplex search, simulated annealing, genetic algorithms, or derivative-free de-randomized methods (such as Covariance Matrix Adaptation) – to more efficient search the space of possible implementations.

# Acknowledgements

We would like Andreas Klöckner, Paul Ivanov and the anonymous reviewers for helpful discussions; Wen-Mei Hwu, Volodymyr Kindratenko and Jeremy Enos as well as Hanspeter Pfister, Robert Parrott, Seppo Sahrakorpi and Matthew Miller for making additional GPU clusters available for this work.

This study was funded in part by the NVIDIA Graduate Fellowship, the Singleton Fellowship, the Rowland Institute of Harvard and the National Science Foundation. Hardware support was generously provided by the NVIDIA Corporation.

# Part IV

# Large-Scale Applications

Chapter 11

# Beyond Simple Features: A Large-Scale Neuromorphic Feature Search Approach to Unconstrained Face Recognition*

"In the beginner's mind there are many possibilities, but in the expert's mind there are few."

*Shunryu Suzuki*

Many modern computer vision algorithms are built atop of a set of low-level feature operators (such as SIFT [Lowe, 2004; Luo *et al.*, 2007]; HOG [Dalal and Triggs, 2005; Albiol *et al.*, 2008]; or LBP [Ahonen *et al.*, 2004, 2006]) that transform raw pixel values into a representation better suited to subsequent processing and classification. While the choice of feature representation is often not central to the logic of a given algorithm, the quality of the feature representation can have critically

*This chapter presents preliminary work done in collaboration with David D. Cox [Pinto and Cox, 2011a].

important implications for performance. Here, we demonstrate a large-scale feature search approach to generating new, more powerful feature representations in which a multitude of nonlinear, multilayer neuromorphic feature representations are randomly generated and screened to find those best suited for the task at hand.

In particular, we show that this approach can generate representations that, in combination with standard machine learning blending techniques, achieve state-of-the-art performance on the *Labeled Faces in the Wild (LFW)* [Huang *et al.*, 2007, 2008] unconstrained face recognition challenge set. These representations outperform previous state-of-the-art approaches, in spite of requiring less training data and using a conceptually much simpler machine learning backend. We argue that such large-scale-search-derived feature sets can play a synergistic role with other computer vision approaches by providing a richer base of features with which to work.

At the same time, we present an analysis of the errors made by our various models, and show that each of them makes appreciably the same errors, and that a large fraction of errors can be qualitatively explained by variation in the view of the targets. We argue that seriously tackling such image variation, and building sets that contain more real-world variation, will be an essential component of future research in unconstrained face recognition.

## 11.1 Introduction

Face recognition has long been, and continues to be, a highly active area of research [Belhumeur *et al.*, 2002; Yang, 2002; Vasilescu and Terzopoulos, 2002; Zhao *et al.*, 2003; He *et al.*, 2005; Zou *et al.*, 2007; Hua *et al.*, 2007; Hua and Akbarzadeh, 2009; Guillaumin *et al.*, 2009; Wright and Hua, 2009]. In recent years, interest in the problem of *unconstrained* face recognition has grown in the community, driven in large part by the creation of the *Labeled Faces in the Wild (LFW)* [Huang *et al.*, 2007, 2008] test set, which has provided a standardized benchmark against which to measure progress. While face recognition research *per se* has a long and rich history, much work prior to the last decade was focused on face recognition in relatively constrained environments (e.g. posed photographs, under controlled lighting conditions [Olivetti Research Labo-

ratory, 1994; Yale Center for Computational Vision and Control, 1997; Martinez and Benavente, 1998; Phillips *et al.*, 1998; Computer Vision Lab at the University of Ljubljana, 1999; Sim *et al.*, 2003; Gao *et al.*, 2007; Gross *et al.*, 2010]). More recently, thanks in large part to the rise of the internet, it has become possible to assemble large collections of face images "in the wild" in the sense that they come from a wide variety of sources and were not posed for the purpose of research. While this set has proven to be quite challenging, large strides have been in made in recent years towards higher performance (e.g. [Taigman *et al.*, 2009; Wolf *et al.*, 2009; Kumar *et al.*, 2009; Cao *et al.*, 2010] – see also Chapters 5 and 6).

While a variety of different approaches to the *LFW* set have been taken, a common feature of most approaches is the use of some low-level visual feature set, such as SIFT [Lowe, 2004; Luo *et al.*, 2007]; HOG [Dalal and Triggs, 2005; Albiol *et al.*, 2008]; or LBP [Ahonen *et al.*, 2004, 2006] that transforms raw pixels values into a better form for subsequent processing. While individual algorithms often do not depend critically on the choice of a particular feature representation used, the choice of features used does frequently play a key role in determining performance. Meanwhile, there are only a handful of visual feature representations in common use, and arguably less attention has been paid to developing new or better features.

One potentially promising source for new, more complex visual feature representations is the class of "biologically-inspired" representations. Biologically-inspired approaches seek to build artificial visual systems that capture aspects of the computational architecture of the brain, in the hope of eventually mimicking its computational abilities. Such efforts to model visual computations done by the brain have a long history, at least dating back to Fukushima's Neocognitron (1980; [Fukushima, 1980]). More recent experiments with biologically-inspired models have shown them to be highly competitive in a variety of different face and object recognition contexts (e.g. [LeCun *et al.*, 2004; Chopra *et al.*, 2005; Serre *et al.*, 2007c; Mutch and Lowe, 2008; Jarrett *et al.*, 2009; Kavukcuoglu *et al.*, 2009; Boureau *et al.*, 2010a] – see also Chapters 4, 5 and 6).

However, the range of possible feature representations that would count as "biologically-inspired" is broad, and it is not clear which particular instantiations of biologically-inspired ideas are best for a given task. In Chapter 9, we previously demonstrated a con-

ceptually simple high-throughput screening approach for model selection of biologically-inspired algorithms, wherein a large number of possible candidate models from an inclusive model family are considered, and the best performing models are "skimmed off the top" and evaluated further. However, while that work showed success with synthetic test images, it has not been known to date whether models from this class are competitive with current state-of-the-art approaches on standard face and object recognition test sets.

Here we present a modified large-scale feature search procedure that simplifies and accelerates the search procedure described in Chapter 9, with the goal of generating feature representations tailored for unconstrained face recognition, as embodied by the *LFW* test set. Multiple complimentary representations are further derived through training set augmentation, alternative face comparison functions, and feature set searches with a varying number of model layers. These individual feature representations are then combined using kernel techniques to achieve even better performance. We show that our approach yields multiple feature sets that outperform previous state-of-the-art approaches on the *LFW* set, even while requiring less training data and using simpler machine learning backends. In addition to providing evidence for the utility of large-scale feature search for standard "real world" test sets, these results emphasize the value of good underlying representations and point a path forward in the generation of new, more powerful visual features.

## 11.2 Methods

### 11.2.1 Large-scale feature search framework

The large-scale feature search approach used here consists of four basic components:

1. a parametric family of feature representation, wherein key aspects of the behavior of the features are controlled by a fixed set of parameters,

2. a generation procedure for choosing models from the larger family to evaluate,

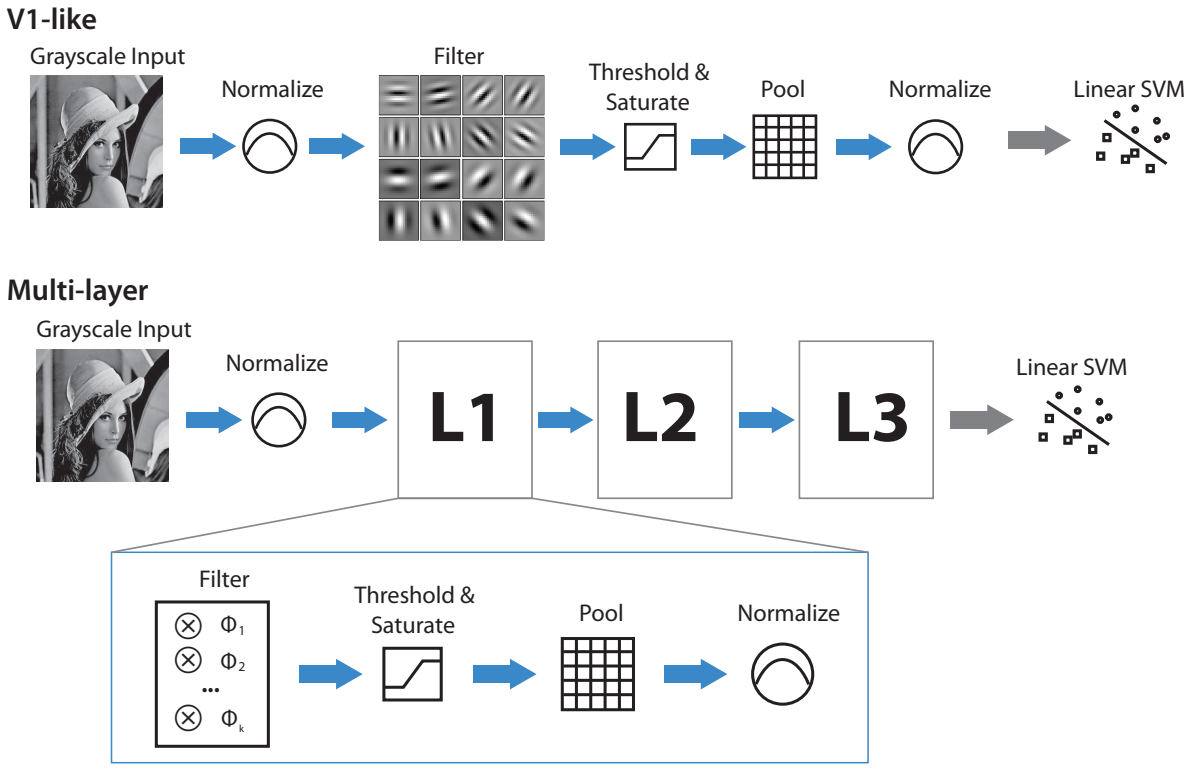



**Figure 11.1: A schematic diagram of the system architecture of the family of models considered.** Each model consists of one to three feed-forward filtering layers, with the filters in each layer being applied across the previous layer.

3. a screening procedure, run on each candidate feature representation, to determine which models to evaluate further and

4. a validation procedure, using independent data, to evaluate the utility of representations found during the screening procedure.

The approach we follow here is similar to that described in Chapter 9, with two important differences, which we describe briefly here, and detail in depth below. First, in Chapter 9, we used an unsupervised learning procedure in order to learn certain model parameters from a pre-training video set (see Methods Sections 9.2.3 and 9.5.7). Here, we dispense with this unsupervised learning procedure, instead opting for greatly speeded model generation, allowing more model architectures to be evaluated per unit time. Second, we used the *LFW View 1* subset as a screening set. Details of the

model family considered, and generation, screening and validation procedures used are described below.

### 11.2.2 Biologically-Inspired Visual Representations

In our experiments, we used two basic classes of biologically-inspired visual representations, shown in Figure 11.1.

First, as a control, we used *V1-like*, a one-layer model characterized by a cascade of linear and nonlinear processing steps and designed to encapsulate some of the known properties of the first cortical processing stage in the primate brain. Our *V1-like* implementation was taken without modification from Chapters 4, 5 and 6.

Second, we used two and three layer models following the basic multi-layer model scheme described in Chapter 9. Briefly, these models consist of multiple stacked layers of linear-nonlinear processing stages, similar to those in the *V1-like* model. Importantly, in order to speed the processing of these models, we disabled the unsupervised learning mechanisms described in Chapter 9 and instead used *random* filter kernels drawn from a uniform distribution. Prior experience of our group and others [Jarrett *et al.*, 2009] has suggested that random filters can in many cases function surprisingly well for models belonging to this general class.

Details of each model class follow.

### 11.2.3 "V1-like" Visual Representation

In the *V1-like* representation, features were taken without additional optimization from Chapter 4's *V1S+*. This visual representation is based on a first-order description of primary visual cortex V1 and consists of a collection of locally-normalized, thresholded Gabor wavelet functions spanning a range of orientations and spatial frequencies.

We have proposed these *V1-like* features as a neuroscientist "null" model (a baseline against which performance can be compared) for object and face recognition since they do not contain a particularly sophisticated representation of shape or appearance, nor do they possess any explicit mechanism designed to tolerate image variation (e.g. changes in view, lighting, position, etc. [DiCarlo and Cox, 2007]). Here, this model serves as

a lower bound on the level of performance that can be achieved by only relying on relatively low-level regularities that exist in the test set. To be considered a promising face recognition system in unconstrained settings, a model should minimally exceed the performance of the *V1-like* model.

In spite of their simplicity, these features have been shown to be among the best-performing non-blended features set on standard natural face and object recognition benchmarks (i.e. *Caltech101* [Fei-Fei *et al.*, 2004a], *Caltech256* [Griffin *et al.*, 2007], *ORL* [Olivetti Research Laboratory, 1994], *Yale* [Yale Center for Computational Vision and Control, 1997], *CVL* [Computer Vision Lab at the University of Ljubljana, 1999], *AR* [Martinez and Benavente, 1998], *LFW* [Huang *et al.*, 2007]) and they are a key component of the best blended solutions for some of these same benchmarks [Gehler and Nowozin, 2009]. We used the publicly available source code to generate these features and followed the same basic read-out/classification procedure as detailed in Chapter 4, with two minor modifications. Specifically, no PCA dimensionality reduction was performed prior to classification (the full vector was used), and a different SVM regularization parameter was used ($C = 10^5$ instead of $C = 10$, see below).

For a detailed description of the *V1-like* visual representation, we refer the interested reader to the methods of Section 4.4, and the publicly available open-source code[1] [2].

### 11.2.4 High-Throughput-Derived Multilayer Visual Representations: *HT-L2* and *HT-L3*

In this study, we considered the five best two- and three-layer models generated from a high-throughput feature search procedure (model selection) for a total of 10 multilayer visual representations. An important feature of the generation of these representations, according to the basic scheme set forth in Chapter 9, is the use of a massively parallel, high-throughput search over the parameter space of all possible instances of a large class of biologically-inspired models. This model class and the high-throughput screening (model selection) procedure are modified from Chapter 9, as described below.

[1] `http://pinto.scripts.mit.edu/Code`
[2] `https://github.com/npinto/v1like`

#### Model Architecture

Candidate models were composed of a hierarchy of two (*HT-L2*) or three layers (*HT-L3*), with each layer including a cascade of linear and nonlinear operations that produce successively elaborated nonlinear feature-map representations of the original image. A diagram detailing the flow of operations is shown in Figure 11.1.

#### Input and Pre-processing

The input of the *HT-L2* and *HT-L3* models were 100x100 and 200x200 pixel images, respectively. In the pre-processing stage, this input was converted to grayscale and locally normalized as in Section 9.5.1.

#### Linear Filtering

All filter kernels were fixed to random values drawn from a uniform distribution.

### 11.2.5 Final Model Output Dimensionality

The output dimensionality of each candidate model was determined by the number of filters in the final layer, and the x-y "footprint" of the layer (which, in turn, depends on the subsampling at each previous layer). In the model space explored here, the possible output dimensionality ranged from 256 to 73,984.

### 11.2.6 Screening (Model Selection)

A total of 5,915 *HT-L2* and 6,917 *HT-L3* models were screened on the *LFW* View 1 "aligned" set [Taigman *et al.*, 2009]. We selected the best five models from each "pool" for further analysis on the *LFW* View 2 set (Restricted Protocol). Note that *LFW* View 1 and View 2 do not contain the same individuals and are thus mutually exclusive sets. View 1 was designed as a model selection set while View 2 is used as an independent validation set for the purpose of comparing different methods.

Examples of the screening procedure for *HT-L2* and *HT-L3* models on the *LFW* View 1 task screening task are shown in Figure 11.2. Performance of randomly gener-

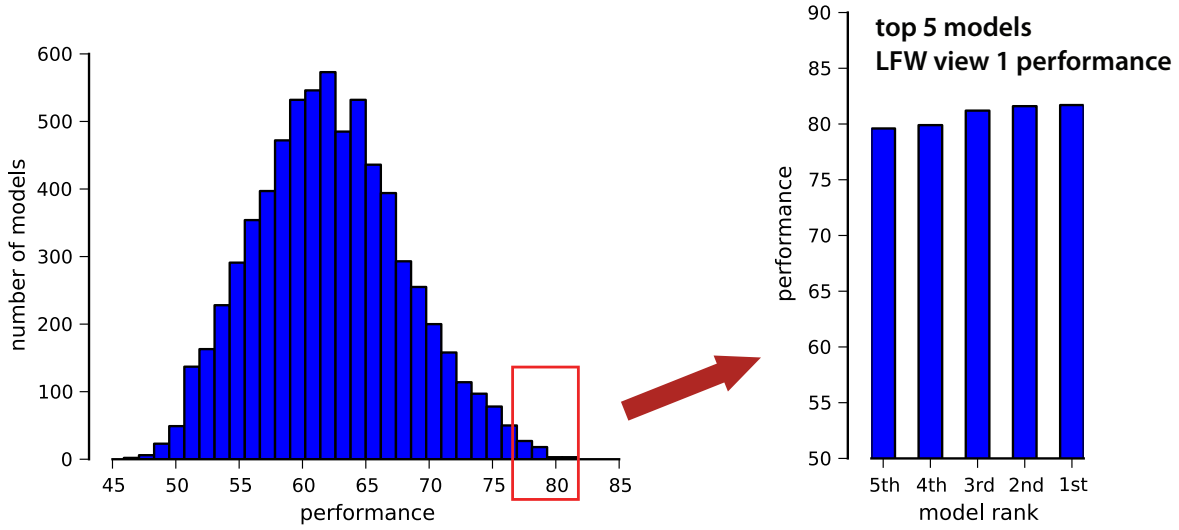



**Figure 11.2: The high-throughput screening process used to find good representations.** Here, data is shown for the screening of *HT-L3* models. A distribution of the performance of 6,917 randomly generated models is shown on the left, with the top five high-performing models replotted on the right. Following screening, the models were evaluated exclusively with sets that do not overlap with the screening set.

ated *HT-L3* models ranged from chance performance (50%) to better than 80% correct; the best five models were drawn from this set and are denoted *HT-L3-1st*, *HT-L3-2nd*, and so on. An analogous procedure was undertaken to generate five two-layer models, denoted *HT-L2-1st*, *HT-L2-2nd*, etc.

### 11.2.7 Evaluation Protocol

To evaluate the performance of our biologically-inspired representations, we followed the standard *LFW* face verification "Restricted View 2" protocol. 6,000 different face image pairs (half "same", half "different") were drawn randomly from the sets and divided into 10-fold cross validation splits with 5,400 training and 600 testing examples each.

Because the biologically-inspired representations used here generate one feature vector per image, comparison functions were used to generate a new feature vector for each pair, and these "comparison" features were used to train binary ("same" / "different")

hard-margin linear SVM classifiers. Following what was presented in Chapter 6, we used the following element-wise comparison functions:
$|F_1 - F_2|$, $\sqrt{|F_1 - F_2|}$, and $(F_1 - F_2)^2$, where $F_1$ and $F_2$ are the feature vectors generated from the first and the second image of the pair, respectively. We additionally added the comparison function $(F_1 \cdot F_2)$, which was not used in Chapter 6, under the logic that it serves as a soft "AND-like" function (i.e. it primarily results in a large response for elements where both $F_1$ and $F_2$ are large). We hypothesized that such a function would be valuable since our representations are all quite sparse, and thus a coincidence of high feature values in common between the two test images is likely to provide meaningful evidence of similarity.

### 11.2.8 Kernel Combinations And Data-Set Augmentation

While the high-throughput search techniques described above are capable of yielding relatively high-performing individual representations for *LFW* by themselves, effectively all of the top-performing face recognition systems on *LFW* employ some form of more advanced machine learning backend to enhance their performance [Taigman *et al.*, 2009; Wolf *et al.*, 2009; Kumar *et al.*, 2009; Cao *et al.*, 2010]. One common approach in this regard is to blend together a large number of weak learners to produce a blended classifier.

To explore what performance enhancement can be gained with modest amounts of blending on top of our feature representations, we pursued a progressive strategy of layering on additional kernels to produce successively larger and higher performing blends. Two basic strategies were used for generating new kernels:

1. feature augmentation, performing operations on the input image, such as cropping and rescaling to produce alternate kernels using the same representation,

2. representation blending, that is, combining together kernels derived from multiple separate feature representations (e.g. blending over the five *HT-L2* top models, or combining the top five *HT-L2* and *HT-L3* models).

The progression of these additional elaborations is described next:

### Multiple rescaled crops

As in Chapter 6, we augmented the dataset by computing features on three different centered crops of the image: 250x250 (original), 150x150 and 125x75. Each of these crops was resized to the standard input size of each representation, and SVMs were trained separately for each crop size. Blending of the resulting kernels was done by simple kernel addition, with each kernel being trace-normalized (by the training kernel trace) prior to summation. More sophisticated blending, for example Multi (or Infinite) Kernel Learning (MKL/IKL) [Sonnenburg *et al.*, 2006], or LP-Boost [Gehler and Nowozin, 2009] were not used at this stage.

### Blending of the Top 5 Models Within Class

While the top five models found by our high-throughput search all yield similar levels of performance, they achieve this performance with different parameter sets. Consequently, to the extent that the top five models represent a diversity of different ways to achieve good performance, we would expect that blending these models would yield further enhancement of performance. At this stage, we combined all of the Stage 1 kernels above (multiple rescaled crops) from each of the top five models within each model-class (e.g. *HT-L2* and *HT-L3*).

### Hierarchical (weighted) blends across model class

Finally, we also explored a more principled way to blend the representations from each model class. Following [Lazebnik *et al.*, 2009] we assigned exponentially larger weight to higher-level representation (*V1-like* < *HT-L2* < *HT-L3*) resulting in the following kernel:

$$K(\cdot,\cdot) = \sum_{\ell} (2^{\ell-1}) k_{\ell}(\cdot,\cdot) \tag{11.1}$$

where $\ell = 1$ for *V1-like* (one layer), $\ell = 2$ for the top five *HT-L2* (two layers) and $\ell = 3$ for the top five *HT-L3* (three layers).

We note that the choice of blending strategies to consider on the View 2 set was

| | alone | +crops | within blend | V1+L2+L3 | V1+L2+L3 (weighted) |
|---|---|---|---|---|---|
| ***V1-like*** | 77.0 ± 0.5 | 82.4 ± 0.5 | | | |
| ***HT-L2*** | | | | | |
| 5th | 77.8 ± 0.4 | 82.8 ± 0.5 | | | |
| 4th | 81.3 ± 0.4 | 85.4 ± 0.6 | | | |
| 3rd | 81.5 ± 0.6 | 85.1 ± 0.5 | 87.5 ± 0.5 | | |
| 2nd | 80.8 ± 0.4 | 83.6 ± 0.5 | | | |
| 1st | 81.0 ± 0.3 | 83.3 ± 0.5 | | 87.6 ± 0.6 | 88.1 ± 0.6 |
| ***HT-L3*** | | | | | |
| 5th | 82.8 ± 0.6 | 84.5 ± 0.6 | | | |
| 4th | 82.3 ± 0.3 | 82.7 ± 0.5 | | | |
| 3rd | 83.3 ± 0.4 | 85.6 ± 0.6 | 87.8 ± 0.4 | | |
| 2nd | 83.9 ± 0.3 | 86.8 ± 0.4 | | | |
| 1st | 84.1 ± 0.3 | 86.8 ± 0.3 | | | |

**Table 11.1:** Performance (*LFW* Restricted View 2) of the family of biologically-inspired models and blends thereof.

driven by performance on the View 1 set, thereby avoiding selection bias artifacts.

## 11.3 Results

### 11.3.1 High-throughput screening with *LFW* View 1

Figure 11.2 shows the results of high-throughput screening to select model instantiations that are well-suited to the *LFW* verification task. For each model class, a multitude of models were randomly generated and evaluated on the *LFW* view 1 set, and the best five were selected for further analysis.

### 11.3.2 Performance on *LFW* Restricted View 2

Performance of individual models and model blends are shown in Table 11.1. Performance ranging from **77.1%** for the simplest *V1-like* model to **88.1%** for the largest blend were observed. Taken together, these results show that state-of-the-art level performance is possible within the model family, and there exist multiple paths (e.g. based purely on *V1-like* models, and based on high-throughput, multi-layer models) to achieving high levels of performance. Figure 11.3 shows receiver-operator characteristic

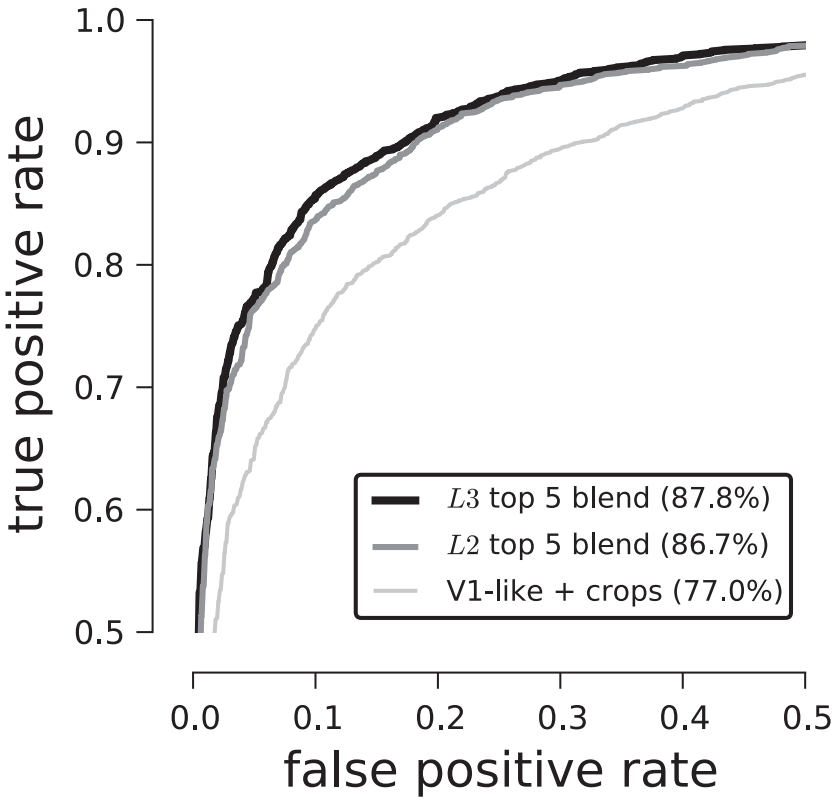



**Figure 11.3: ROC curves for various model sub-families on *LFW* Restricted View 2.** Plots are zoomed-in to facilitate comparison.

(ROC) curves for each of these models.

Interestingly, the inclusion of a single additional comparison function to the *V1-like* model blend described in Chapter 6 brings an additional 3% performance, placing it close to the last reported best performance on this set, even without extensive blending. Furthermore, we see that individual *HT-L3* models also perform surprisingly well – coming to within a few percent correct of the previous state-of-the-art.

A major advantage of our high-throughput approach is that it produces not one, but a diversity of models, and this situation is ideally suited to kernel blending approaches. Once blending is added, especially when coupled with an intelligent algorithm for weighting blended kernels, several different blends achieved performance exceeding previously reported state-of-the-art values (see Figure 11.4 and Table 11.2). ROC curves for various blend groupings are shown in Figure 11.3.

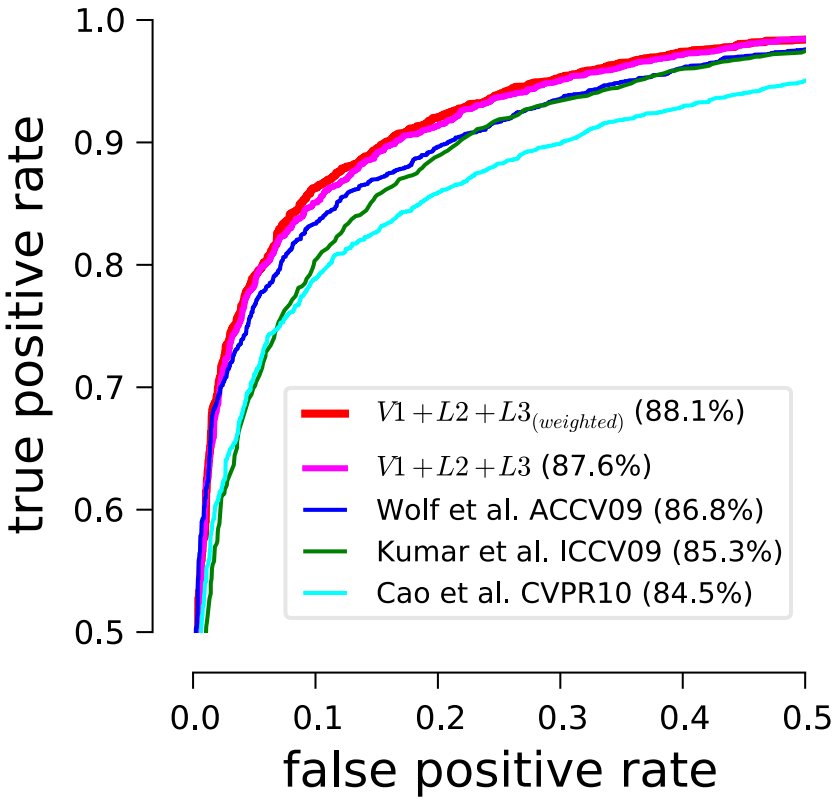



**Figure 11.4: Comparison with the state-of-the-art on *LFW* Restricted View 2.** ROC curves for [Wolf *et al.*, 2009], [Kumar *et al.*, 2009] and [Cao *et al.*, 2010] are retrieved from the official *LFW* website. Plots are zoomed-in to facilitate comparison.

| Reference | [Kumar *et al.*, 2009] (ICCV09) | [Wolf *et al.*, 2009] (ACCV09) | [Cao *et al.*, 2010] (CVPR10) | Ours |
|---|---|---|---|---|
| Mean classification error | 14.7% ±1.2 | 13.2% ±0.3 | 15.5% ±0.5 | **11.9% ±0.6** |

**Table 11.2: Mean classification errors for different state-of-the-art methods.**

### 11.3.3 Analysis of Errors

To understand better where room for improvement lies, we examined the error trials (misses and false alarms) produced by each model for quantitative and qualitative trends. To determine whether different models were primarily making the same or

(a) Misses (b) False Positives

**Figure 11.5: Examples of common errors across models.** Misses tend to be dominated by differences in view, while false positives frequently occur when different individuals share a common view or expression.

different errors, we segregated the responses of the *V1-like* and *HT-L3* models (rescaled-crop augmented variants, see Methods) into four categories: hits, misses, false positives, and correct rejections. We then computed the fraction of errors that these two models held in common and found 84.3% of false positives were the same across the two models, and that 87.3% of misses were missed by both models. This high level of consistency between error cases across the two models led us to ask whether a subset of "hard" images within the larger *LFW* set could be driving errors and capping performance.

Figure 11.5 shows examples of misses and false positives held in common for both models. While developing a quantitative framework within which to analyze these errors is beyond the scope of this paper, several patterns are evident, even upon casual inspec-

tion. First, misses are dominated by situations where the individual-to-be-matched is seen in non-frontal view in at least one of the images. Second, false positives appear to occur more often in cases where different individuals appear in a very similar view, or with a similar expression.

## 11.4 Discussion

Our results provide more evidence that biologically-inspired models represent a promising and powerful direction in face recognition research. Individual models from this class are able to achieve good performance (e.g. around 77% for *V1-like* models, 84% for *HT-L3*), and blends of these models achieve more than 88% correct performance, beating previously reported state-of-the-art values.

Consistent with expectations, progressively more complex, multi-layer models are able to outperform the simpler *V1-like* model. Whether this higher performance is due to a greater ability to tolerate image variation – one of the original purposes for the construction of the *HT-L3* model class (see Chapter 9) – or some other factor remains to be seen (Chapter 13). It should be noted that the *HT-L2* and *HT-L3* models used here were substantially simplified from those presented in Chapter 9, in that they did not have structured filter kernels, nor were they subjected to any unsupervised learning. Whether adding these features back will result in higher levels of performance is an important future research question.

While there still remains substantial room for improvement, concerns that the *LFW* set does not necessarily accurately reflect the "full" problem of unconstrained face recognition remain. *LFW* includes only a handful of examples per individual, and these photographs were often taken in the same setting and at the same event. Furthermore, Kumar et al. [Kumar *et al.*, 2009] showed that human observers were able to perform at greater-than-90% correct even when the faces themselves were masked out of the test images, indicating that the backgrounds in the *LFW* are more than sufficient for solving the task at a level higher than the current machine state of the art.

An analysis of the errors made by our models provides some clues about which parts of the *LFW* set are difficult and which ones are not. Our models failed on remarkably

similar sets of face pairs, indicating that a common core of "hard" images may exist within the larger *LFW* set. A striking, albeit anecdotal, observation is that common error cases are dominated by misses when the same individual is shown in differing views and by false positives when two different individuals are compared while viewed from a similar angle (e.g. Figure 11.5). An important feature of the *LFW* set is that faces must be detected by a Viola-Jones face detector [Viola and Jones, 2001] in order to be included in the set [Huang *et al.*, 2007], and this effectively restricts the range of face views that enter into the set (i.e. there is a bias towards frontal views). We hypothesize that those more off-axis views that do manage to pass the face detection filter will present a particularly difficult challenge for a system trained on the *LFW* set. The low-level (e.g. pixel-level) difference between two different views of the same individual can easily be larger than the low-level differences between two individuals in a similar pose. A system that is not specially designed to tolerate this kind of variation will have a high false alarm rate on trials where two different individuals are seen in the same pose and a high miss rate where the same individual is compared across different poses. At the same time, if the *LFW* set contains a relatively small fraction of these off-axis faces, then a system trained exclusively on the *LFW* set will face difficulty learning to tolerate these cases, even if that system has the capability to learn such tolerance in principle.

As continued research manages to chip away at the remaining "performance gap" between human and machines on the *LFW* set, increased attention will need to be paid to whether *LFW* truly represents the problem of interest. On one hand, as long as some performance gap exists, the set is obviously valid at a basic level. However, the question remains whether a "fuller" formulation of the problem (i.e. more natural, less filtered) might lead to faster progress.

## Acknowledgments

We would like to thank Zak Stone, Todd Zickler and David Luebke for helpful discussions; and Hanspeter Pfister, Wen-Mei Hwu, Volodymyr Kindratenko, Jeremy Enos, Robert Parrott, Seppo Sahrakorpi and Matthew Miller for making additional GPU

clusters available for this work.

This study was funded in part by the NVIDIA Graduate Fellowship, the Singleton Fellowship, the Rowland Institute of Harvard and the National Science Foundation. Hardware support was generously provided by the NVIDIA Corporation.

Chapter 12

# From Face Verification to Large-Scale Face Identification: A Case Study with Biologically-Inspired Visual Representations*

> "Academic approaches to solutions tend not to be useful in the real world. [...] Finding the right balance between hackery and not ignoring the many years of academic research is what is needed."
>
> *Con Kolivas*

The problem of unconstrained face recognition has attracted increasing interest from the community in recent years. However, while much work has focused

*This chapter is modified from a preliminary study done in collaboration with Zak Stone, Todd Zickler and David D. Cox [Pinto et al., 2011b].

on face *verification* (asking whether two faces are the same) "in the wild", relatively less attention has been paid to face *identification* (labeling a face from a set of examples) in realistic photographs. While face verification is interesting in its own right, real-world face identification is increasingly relevant, given the rapid growth of personal digital photo and video collections. To explore a large-scale identification regime, we created two new face sets: *Facebook100*, a set gathered from the Facebook social network site, and *PubFig83*, a subset of the publicly-available *PubFig* data adapted for identification tasks. To benchmark these sets, we present experiments with a family of biologically-inspired models – which have previously achieved state-of-the-art performance on the *LFW* face verification set – and show that they yield high levels of face-identification performance even when large numbers of individuals are considered; this performance increases steadily as more examples are used. Finally, we discuss current limitations and future opportunities associated with datasets such as these and argue that careful creation of large sets that support both verification and identification is an important future direction.

## 12.1 Introduction

In recent years, several serious efforts have emerged to move face recognition research towards more and more unconstrained, "real-world" settings. A major driver of this push has been the *Labeled Faces in the Wild (LFW)* data set, which brings together thousands of face images of public figures from the internet. While some concerns have been raised about whether this set is an ideal surrogate for the "full" problem of real-world face recognition (see Chapters 5 and 6), it nonetheless has served to focus the efforts of the community. More recently, a set in the same vein called *PubFig* [Kumar *et al.*, 2009] has been introduced to help facilitate larger-scale explorations in real-world face recognition.

One property that *LFW* and *PubFig* have in common (at least in their usage to date) is that they are designed primarily as tests of face *verification* – deciding whether two faces represent the same person – rather than face *identification*, which requires matching an unknown face or face set against a gallery of labeled face samples. Clearly,

a continuum exists between these two, and, within limits, a system built under one regime can be reconfigured to perform in the other. That is, if a system has previously learned identities, it can perform verification by identifying each individual first and comparing the labels to determine whether they are the same or different. Similarly, a face verification system can be the primary component of an identification system, with verification used to compare a test image pairwise with many training images; the resulting similarity scores can then inform the identification choice. In practice, however, a system that does identification by verification may be very sensitive to verification errors, and it may not fully utilize the advantages of having a large, labeled training set per individual.

Verification is a natural paradigm in many contexts (e.g. biometric authentication), and it is obviously desirable to have face recognition systems that can function even without a large amount of training data. But experiments in a large-scale face identification regime have become increasingly practical and relevant. The explosion in usage of digital cameras has greatly increased the number of real-world photos that are captured, and photo-sharing software and services (e.g. Facebook, Flickr, iPhoto, and Picasa) have aggregated and organized these photos. Today, it is not uncommon for individuals to have large personal databases of photos of familiar faces, with hundreds or even thousands of images per individual. The ubiquity of personal and shared photo databases presents opportunities both in terms of assembling data sets to guide face recognition research and for potential use cases for working face recognition systems. Already, several available software applications attempt to perform automatic face tagging, with varying levels of success.

The problem of face identification is also intimately related to familiar face recognition in humans. While humans are able in many contexts to identify a face with only a single prior exposure (e.g. [Clutterbuck and Johnston, 2002]), day-to-day visual experience is dominated by repeated exposure to a much smaller group of familiar individuals. Numerous human psychophysical studies have shown that processing of familiar faces is enhanced relative to unfamiliar face recognition (e.g. [Bruce, 1986]), even for tasks that don't depend on identity (e.g. gender recognition; [Rossion, 2002; Balas *et al.*, 2007]).

Research on face identification requires datasets for evaluation; unfortunately, however, it is not always possible to convert existing verification datasets into identification datasets. The *LFW* data set, for example, contains few face samples for most individuals and few individuals with large numbers of face samples. To address this problem, we introduce two new datasets for identification research, both of which are derived from images taken "in the wild", and both of which include many face samples per individual.

The first dataset we created is a set of face samples drawn from photos shared online through the Facebook social network; images were collected in the manner of [Stone *et al.*, 2008, 2010]. Due to the vast size of the network, we were able to extract many labeled samples of many distinct individuals, and it will be straightforward to expand our benchmark set to increase the difficulty of the identification problem. As a public complement to this private set of Facebook photos, we assembled a subset of the *PubFig* dataset with an emphasis on removing near-duplicate images, which are commonly encountered when seeking images of celebrities online.

To benchmark these sets, we used a family of biologically-inspired visual models. These models seek to instantiate biologically-plausible neural-network-style computational elements organized either into a single- (Chapters 5 and 6) or multi-layer (Chapter 9) architecture. These models have been shown to excel in a standard face verification task (*LFW*), previously achieving state-of-the-art performance on that set (Chapter 11).

## 12.2 Datasets

### 12.2.1 The "Facebook100" Dataset

The *Facebook100* data set used in this study contains 100 distinct person categories, each of which is represented by 100 cropped face samples. These labeled face samples were extracted from a set of shared Facebook photos and their associated "tags", which identify the locations of particular people in specific photographs. Figure 12.1 shows a typical Facebook photo with its manually-applied tag locations superimposed in white.

Facebook users tag themselves and their friends in photos for a variety of social purposes [Nov *et al.*, 2008; Marlow *et al.*, 2006; Ames and Naaman, 2007], and they typically manually assign a tag to a photograph by clicking somewhere on the photo and entering a name. The coordinates of the click are used to place the tag, and these coordinates are often conveniently centered on faces [Stone *et al.*, 2008, 2010]. At present, the Facebook interface does not allow users to specify the size of a tagged region, so the tags are assumed to label square regions of a standard size as shown in Figure 12.1. Because the act of assigning a tag to a photo typically triggers a notification to the person tagged and all of the friends of that person and the photographer (at least), the identities assigned with tags tend to be extremely accurate.

Given a photo and its associated tags, we ran the OpenCV face detector to identify actual face locations (shown as green circles in Figure 12.1), and we associated the detected face regions with nearby identity tags using a conservative distance threshold. In the photo shown in Figure 12.1, the detected foreground face is successfully matched with a tag to yield a labeled face sample; the face detected in the background is correctly *not* matched with the remaining tag, which is too spatially distant. More intensive and sophisticated face detection techniques would allow us to harvest the tagged, rotated foreground face that OpenCV missed in addition to more challenging tagged face images throughout the Facebook dataset.

The face samples used in the *Facebook100* dataset were drawn from the user-tagged photos of approximately 50 college-age volunteers and their friends on the Facebook social network; each volunteer authorized a Facebook application to allow us to collect this data. Face samples were extracted and labeled as described above and then grouped by individual, and face samples with OpenCV detection diameters less than 80 pixels were discarded for the purposes of this study. The 500 individuals with the largest number of remaining face samples were selected to create a larger database of individuals, and 100 of those individuals were chosen at random to form the dataset used here. Each individual is represented by 100 face samples chosen at random from the set of their available samples. We reserve the full set of 500 individuals for ongoing work.

### 12.2.2 The "PubFig83" Data set

The main disadvantage of the Facebook data set is that the images it contains are currently private. Facebook has recently made it easier for users to share their photographs with "Everyone", and, as a consequence, we expect that many numbers of tagged photographs and videos of an extremely large number of individuals will eventually be available to the public from Facebook and other sources. In an effort to facilitate academic research on familiar face recognition in the wild at the current time, however, we have created a data set of public face images culled from the web that we call *PubFig83*. Our hope is that recognition performance on *PubFig83* will be broadly predictive of recognition performance on more realistic face images from personal photos such as those shared on Facebook, and we can then use the much larger repository of Facebook images to explore how various algorithms perform with increasingly difficult images and much larger databases of people.

To create the *PubFig83* dataset, we began with the recently released *PubFig* dataset [Kumar *et al.*, 2009], which consists of a set of nearly 60,000 image URLs that depict 200 people, most of whom are well-known celebrities. In a series of processing steps, we selected a subset of *PubFig* that we hope will provide a stable foundation for face recognition research. First, we downloaded all of the images that were still available from the original image lists for both the *development* and *evaluation* sets, and we obtained roughly 89% of the original images. We then ran the OpenCV face detector on all downloaded images and treated the provided face label locations as "tags"; we proceeded to match the face detections with identity labels just as we did for the Facebook data set. This OpenCV filtering step left us with 90.6% of the readable images (80.9% of the original *PubFig* set).

Upon examination of the remaining images, we noticed several sets of near-duplicate images in many individual identity categories. These near-duplicate copies of a single image varied from the original in many ways: some were scaled, cropped, and compressed differently, some had their color spaces altered, and some had been digitally edited more substantially, with whole backgrounds replaced or overlays added. With millions of within-class image pairs to consider, we could not evaluate each pair manu-

ally. To remove the vast majority of images that could be duplicates and produce the final *PubFig83* dataset, we applied a simple but coarse method: we globally ranked all within-class image pairs by the similarity of their labeled face samples, and we treated a portion of the most similar image pairs as duplicates. We compared images on the basis of their face samples to avoid the effects of extreme cropping, and we scored each pair by the maximum correlation of the central region of the face sample in one image to the central region of the face sample in the other. After browsing the globally-ranked list of image pairs manually, we determined that most obvious near-duplicates landed in the top 4% of the list, so we treated all pairs in that range as duplicates. Manual inspection of the remaining images suggested that this technique eliminated the majority of the near-duplicate image pairs, but it also eliminated pairs of images in which the same individual makes nearly identical expressions on different occasions. This artificial culling of similar facial expressions makes this dataset more challenging than it would have been if we could have removed only the true duplicate images.

For this study, we further selected all of the individuals in both the development and evaluation sets for whom 100 or more face samples remained. This yielded a final dataset of 83 individuals suitable for large-scale face identification testing.

## 12.3 Biologically-Inspired Visual Representations

In order to produce benchmarks for identification performance on the *Facebook100* and *PubFig83* data sets, we relied on the family of biologically-inspired visual representations described in previous chapters and designed to model various stages of visual cortex in the brain.

As in Chapter 11, two basic model classes were used (see Figure 11.1). First, we used "V1-like," a simple one-layer model characterized by a cascade of linear and nonlinear processing steps and designed to encapsulate some of the known properties of the first cortical processing stage in the primate brain. Again, our *V1-like* implementation was taken without modification from Chapters 4, 5 and 6.

Second, we used two- and three-layer models following the basic multi-layer model scheme described in Chapters 9 and 11. Briefly, these models consist of multiple stacked

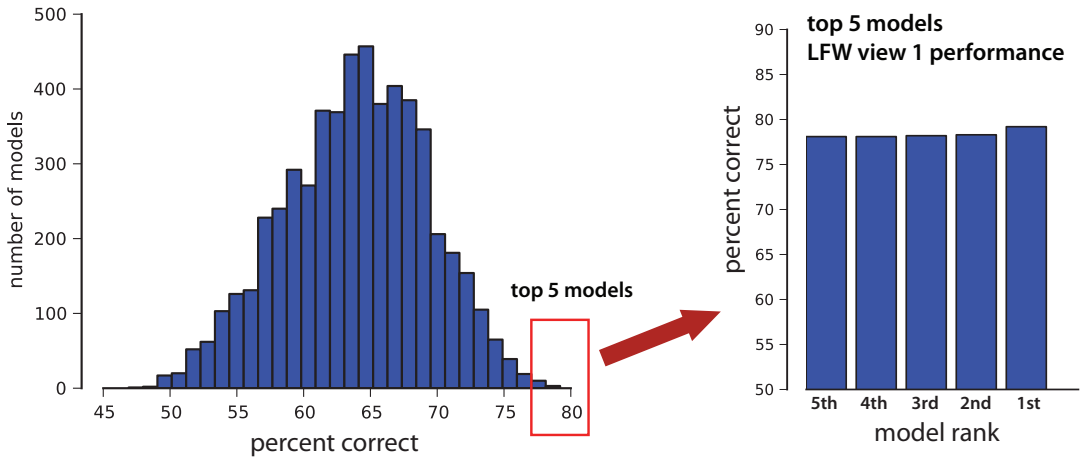



**Figure 12.2: An example of the high-throughput screening process used to find *HT-L2* and *HT-L3* representations.** Here, data is shown for the screening of *HT-L2* models. A distribution of the performance of 5,915 randomly generated models is shown on the left, with the top five high-performing models replotted on the right. Following screening, the models were evaluated exclusively with sets that do not overlap with the screening set.

layers of linear-nonlinear processing stages, similar to those in V1-like. Importantly, in order to speed the processing of these models, the learning mechanisms described in Chapter 9 was disabled and instead used random filter kernels drawn from a uniform distribution.

For a detailed description of these visual representations, we refer the interested reader to the methods in Section 11.2.2.

### 12.3.1 Screening (Model Selection)

A total of 5,915 *HT-L2* and 6,917 *HT-L3* models were screened on the *LFW* View 1 "aligned" set [Taigman *et al.*, 2009]. Following Chapter 11, we selected the best five models from each "pool" for further analysis on the *Facebook100*, *PubFig83* and *LFW Restricted View 2* sets. Note that *LFW* View 1 and View 2 do not contain the same individuals and are thus fully mutually exclusive sets. View 1 was designed as a model selection set while View 2 is used as an independent validation set for the purpose of comparing different methods. Importantly, no special optimization of these models was

done for either *Facebook100* or *PubFig83*.

An example of the screening procedure for the HT-L2 models on the *LFW* View 1 task screening task is shown in Figure 12.2. Performance of randomly generated HT-L2 models ranged from chance performance (50%) to 80% correct; the best five models were drawn from this set and are denoted *HT-L2-1st*, *HT-L2-2nd*, and so on. An analogous procedure was undertaken to generate five three-layer models, denoted *HT-L3-1st*, *HT-L3-2nd*, etc.

### 12.3.2 Identification

To test in an *identification* mode for a given feature representation and data set, we first generated feature vectors for each image in the set. These feature vectors were then used to train a binary linear support vector machine (SVM) [Scholkopf and Smola, 2002] per individual in a one-versus-all configuration [Rifkin and Klautau, 2004] using the Shogun Toolbox [Sonnenburg *et al.*, 2006] with the LIBSVM solver [Chang and Lin, 2001]. To avoid the computational cost of fitting the SVM's regularization hyperparameter $C$, we fixed $C$ to a very high value ($10^5$), allowing no slack and thus resulting in a quasi-parameter-free hard-margin SVM.

Final performance values were computed as the average of ten random test/train splits of the data, with a variable number of training examples (see Figures 12.3 and 12.4) and ten testing examples per individual. In the case of the *Facebook100* set, all performance values presented here were the results of 100-way classification. For the *PubFig83* set, 83-way classifiers were used.

### 12.3.3 Verification

To explore the relationship between identification and verification, we also used the *Facebook100* and *PubFig83* sets in a verification mode, following the structure of the *LFW* face verification protocol (*Restricted View 2*) as closely as possible. 6,000 different face image pairs (half "same", half "different") were drawn randomly from the sets and divided into 10-fold cross validation splits with 5,400 training and 600 testing examples each.

Because the biologically-inspired representations used here generate one feature vector per image, comparison functions were used to generate a new feature vector for each pair, and these "comparison" features were used to train binary ("same"/"different") hard-margin linear SVM classifiers. As in Chapter 11, we used four comparison functions: $|F_1 - F_2|$, $\sqrt{|F_1 - F_2|}$, $(F_1 - F_2)^2$, and $(F_1 \cdot F_2)$, where $F_1$ and $F_2$ are the feature vectors generated from the first and the second image of the pair, respectively.

As an additional point of reference, we also include verification performance on the *LFW* set. Verification performance was derived for the *Restricted View 2* portion of the set. Performance of the selected *V1-like*, *HT-L2*, and *HT-L3* models on *LFW* was also reported in Chapter 11. While that work showed that relatively simple blended combinations of multiple models belonging to this class were able to significantly outperform the state-of-the-art on the *LFW* set ($> 88\%$ performance), here we opted to use each model individually for the sake of simplicity (a total of 11 models were evaluated: one from *V1-like*, five from *HT-L2*, and five from *HT-L3*). Also, in contrast with Chapters 6 and 11, we restricted ourselves to grayscale versions of the *original* image crops.

## 12.4 Results

### 12.4.1 Facebook100

Performance using our biologically-inspired feature representations on the *Facebook100* followed the same basic pattern as had been previously observed for *Labeled Faces in the Wild* (in Chapter 11), with progressively more complex models (those with more layers) yielding progressively higher performance (i.e. *HT-L3* > *HT-L2* > *V1-like*). Figure 12.3 shows performance as a function of number of training examples per individual for the *V1-like*, *HT-L2-1st* (i.e. the best-ranked two-layer model, as ranked by its performance on the *Labeled Faces in the Wild* view-1 set), and the *HT-L3-1st* models. Interestingly, we find that relatively high levels of performance (up to 86%) are possible on this 100-way identification task, especially as the number of training examples increases to 90.

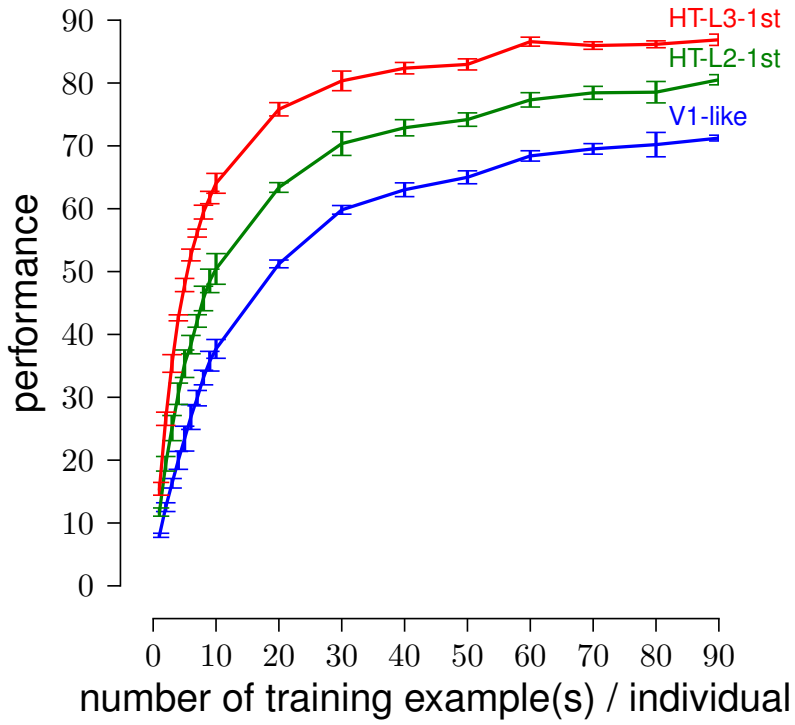



**Figure 12.3: Facebook100.** Performance of three models as a function of the number of training examples per individual.

### 12.4.2 Pubfig83

Performance on the *PubFig83* set followed appreciably the same trend as for the *Facebook100 set*. Figure 12.4 shows performance of *V1-like*, *HT-L2-1st*, and *HT-L3-2nd* as a function of the number of training examples per individual.

Asymptotic performance on the *PubFig83* set was lower for all feature representations as compared to performance on the *Facebook100* set. This is consistent with the fact that the creation of *PubFig83* involved an aggressive screening process designed to remove duplicates, which also removed many legitimate faces from the set that were similar to other faces in the set. We hypothesize that these "typical" faces would be easier to classify, because their presence increases the odds that, for each test face, one or more similar faces would normally exist in the training set. Figure 12.5 shows a

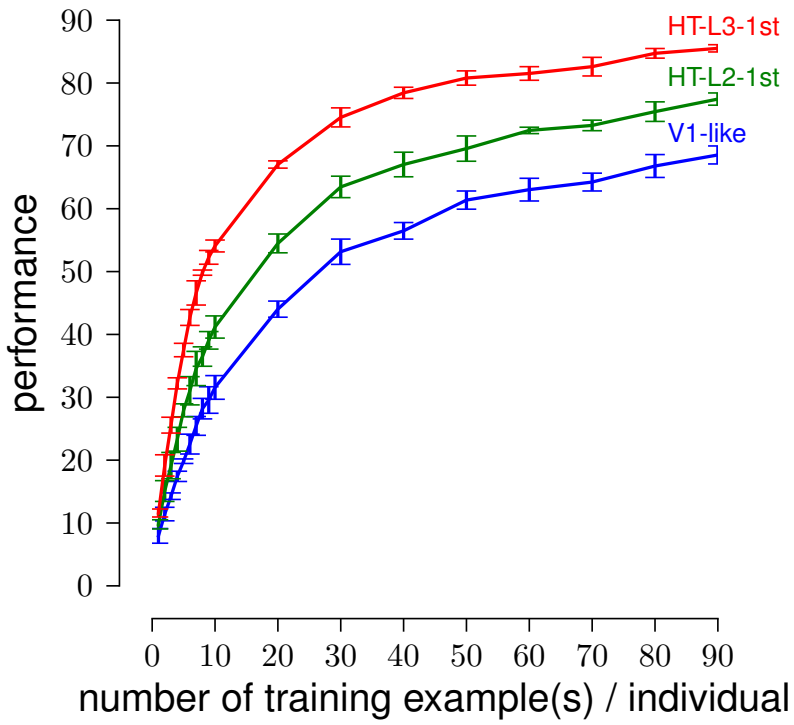



**Figure 12.4: PubFig83.** Performance of three models as a function of the number of training examples per individual.

scatter plot of the relative performance on these two sets for each of the 11 models considered here (V1-like, five HT-L2 models, and five HT-L3 models). While the performance on the *PubFig83* set is displaced downward for all models, the relationship between performance on the *PubFig83* and *Facebook100* sets is remarkably linear.

### 12.4.3 Comparing Verification and Identification Paradigms

To explore the relationship between face verification and identification paradigms, we ran verification-mode experiments (in the style of *Labeled Faces in the Wild*) using the *Facebook100* and *PubFig83* sets. Verification performance on the *Facebook100* set ranged from 62.45%, with the *V1-like* model, to 69.5% for the best HT-L3 model. Verification performance on the *PubFig83* set followed a similar range, with the *V1-like*

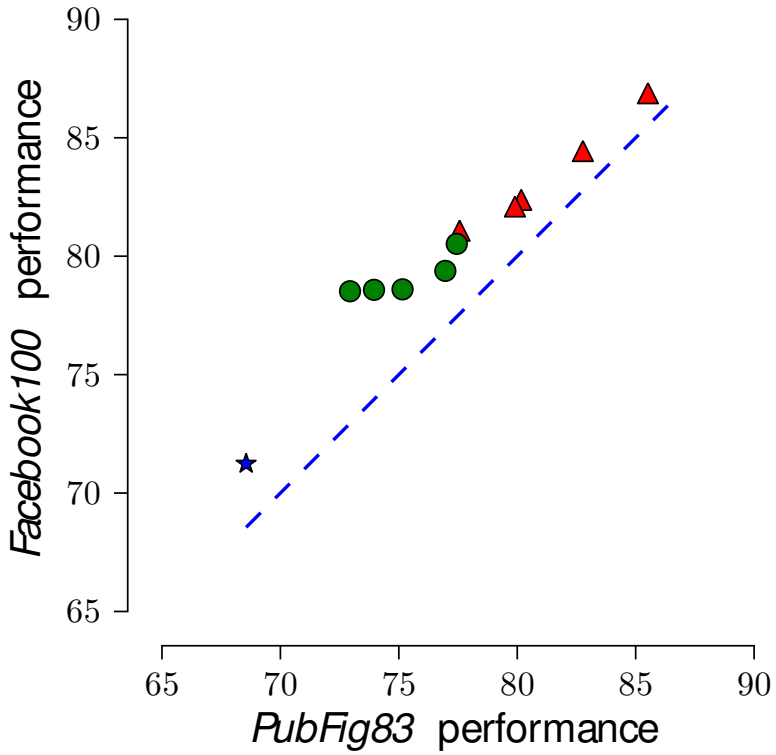



**Figure 12.5: Performance comparisons across models and data sets.** Comparison of identification performance on the *PubFig83* and *Facebook100* data sets. Red triangles indicate *HT-L3* models, green circles indicate *HT-L2* models, and the blue star indicates *V1-like*.

model achieving 63.4% and the best HT-L3 achieving 70.2%. Figure 12.6 shows the verification-mode performance of each of the 11 models considered here, plotted against their identification-mode performance. Interestingly, the rough rank order of models (from *V1-like* to *HT-L2* to *HT-L3*) is preserved in both verification and identification modes, and the approximately linear relationship between verification and identification in the *Facebook100* and *PubFig83* is quite similar, despite these sets' substantially different provenance.

Finally, Figures 12.6(c) and 12.6(d) show the relationship between verification on the *Labeled Faces in the Wild* set and identification-mode performance on *Facebook100* and *PubFig83*. Again, a roughly similar, albeit shifted, relationship between verification and identification performance is observed.

### 12.4.4 Comparing Verification Paradigms Across Sets

For the sake of completeness, one final area that our experiments enable us to explore is the relationship between verification-mode tasks across different sets. Figures 12.6(e) and 12.6(f) show the performance of our model family on *LFW* versus verification performance on each of our new sets. Here we see that while *Facebook100* and *PubFig83* continue to behave in a highly similar manner, both sets are substantially more difficult in a verification task than the *LFW* view 2 set. It should be noted that the models used here were originally screened from a larger set of models using the *LFW* view 1 set (see Section 12.3.1), so to the extent that *LFW* view 1 is somehow more similar to *LFW* view 2 than it is to either *PubFig83* or *Facebook100*, these models might be better tailored to this set. In any event, this result underscores the fact that not all verification challenges are created equal, and argues that one should be careful in comparing results across sets, even when the sets were ostensibly produced by a similar process.

## 12.5 Discussion

Here we have presented experiments with face-identification in real-world settings. We introduced two new large-scale face identification sets: *Facebook100*, a naturalistic set of face images from users of the Facebook social networking website, and *PubFig83*, a filtered subset of the original *PubFig* data set with many near-duplicate images removed. While the *Facebook100* cannot be shared due to privacy concerns, our results indicate that, at least for the set of representations considered here, performance on *PubFig83* is highly predictive of performance on the *Facebook100* set, and we will make the *PubFig83* dataset available online. As privacy norms continue to evolve on Facebook, we anticipate that much larger face sets (more individuals, more examples per individual) will eventually become available for research purposes.

The methods used to collect our datasets are samples from a larger space of possibilities. The original *PubFig* dataset leveraged text-image co-occurrence on the web to harvest facial images of famous individuals, and similar results can be obtained by exploiting captions in news feeds and videos [Berg *et al.*, 2004; Everingham *et al.*, 2006]

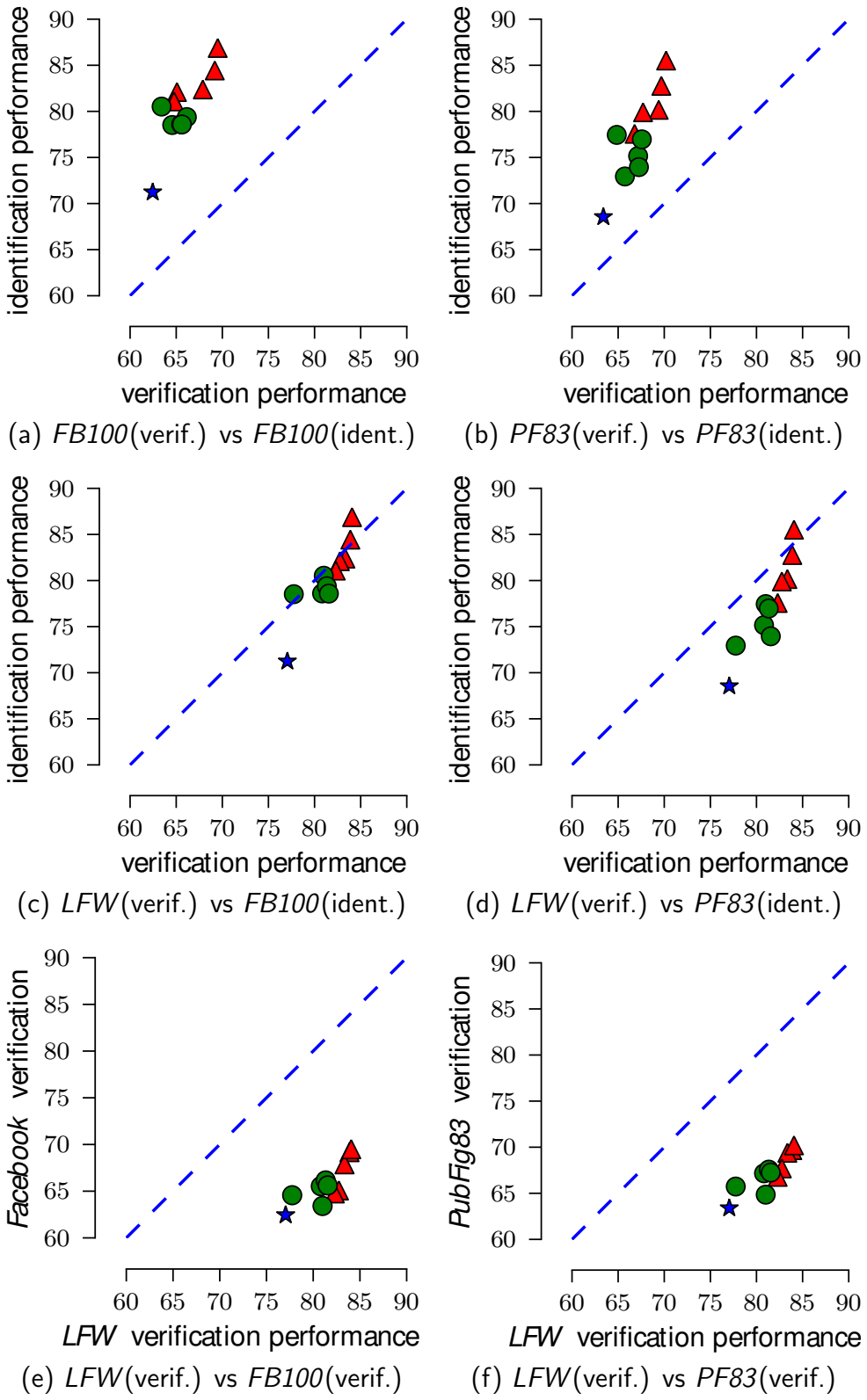



**Figure 12.6: Comparison of face verification and identification for 11 biologically-inspired models.** Symbols and colors follow the same conventions as in Figures 12.3 and 12.4.

or by combining image and video data [Zhao *et al.*, 2008]. In fact, because clothing and hair features allow faces in videos to be tracked through partial occlusion and drastic pose changes, face datasets harvested from video can more easily be built to include these large-scale effects [Ramanan *et al.*, 2007]. In contrast, the faces in our datasets are currently filtered by a frontal face detector and therefore include only limited variations in pose.

All of these approaches ([Berg *et al.*, 2004; Everingham *et al.*, 2006; Ramanan *et al.*, 2007; Zhao *et al.*, 2008]) to dataset construction exploit metadata that is associated with public figures, and the resulting datasets are consequently limited in the number and diversity of individuals they contain. Furthermore, while many face samples can be extracted from polished video sources such as movies and television shows, the appearance variations of the people they represent are typically artificial and tightly controlled. We are entering an era in which repositories of "user-generated content" will grow vastly larger than collections of professionally-produced photos and videos, and the models and the *Facebook100* dataset we present here are intended for this new era. We expect that most users of the Web will appear in an increasing number of photos and videos online that are captured under natural, uncontrolled conditions over larger and larger timespans, and it will become feasible to combine several of the data collection techniques above to collect high-quality labeled face samples and tracks that richly represent millions of individuals with extreme variations in appearance, pose, expression, age, and other variables. Furthermore, the behavior of existing Facebook users suggests that future user-generated content will also be annotated with socially-incentivized metadata. In addition to enabling the harvesting of test sets, this metadata can be used as another channel of information to further improve image-based face identification rates at run-time [Stone *et al.*, 2008, 2010].

Our experiments provide some indication of how a given set of representations will perform across verification and identification tasks. It is not obvious *a priori* how performance with a given model on a verification task ought to relate to performance on an identification task. It is also difficult to reason about which task ought to be "harder." On one hand, verification-mode decisions are made with relatively impoverished data (only a pair of images) and can require judgments to be made about two individuals

whose faces do not occur anywhere in the training set. On the other hand, verification tasks are binary decisions, so performance values will naturally be numerically higher because chance performance is 50%. Ultimately, given the diversity in different sets being used in face recognition research, it does not make sense to expect any sort of predictable relationship between performance in various contexts, and thus this relationship is a largely empirical question for each set. Our results in this regard are reassuring in that the rough ranking of models we considered is well preserved throughout, suggesting that there is some consistency across verification and identification. Our data also provide hints at trends that may exist in the relationship between verification and identification, though the evaluation of more systems, including systems from other families, would be required to say anything definitive about these trends.

Another important finding from this study is that high levels of performance (<86%) are achievable when reasonable quantities of training data are available. We note that we did not attempt to optimize any of the representations used here for face identification, nor did we pursue any blending strategies to combine together multiple representation (such strategies have been demonstrated to yield even better performance, see Chapters 6 and 11). Consequently, the performance numbers presented here likely serve as a lower bound on performance that might be possible. Similarly, as even larger numbers of examples per individual are included (Facebook users are routinely tagged in hundreds if not thousands of photos), we anticipate higher performance still.

## Acknowledgments

We would like to thank Hanspeter Pfister, Wen-Mei Hwu, Volodymyr Kindratenko and Jeremy Enos for making additional GPU clusters available for this work.

This study was funded in part by the NVIDIA Graduate Fellowship, the Singleton Fellowship, the Rowland Institute of Harvard and the National Science Foundation. Hardware support was generously provided by the NVIDIA Corporation.

**Figure 12.7:** Examples of near-duplicate images in the original *PubFig* set [Kumar *et al.*, 2009] before duplicate removal (top) and after (bottom). In this case, our correlation-based filtering process correctly retains only a single unique example. Figure from Zak Stone.

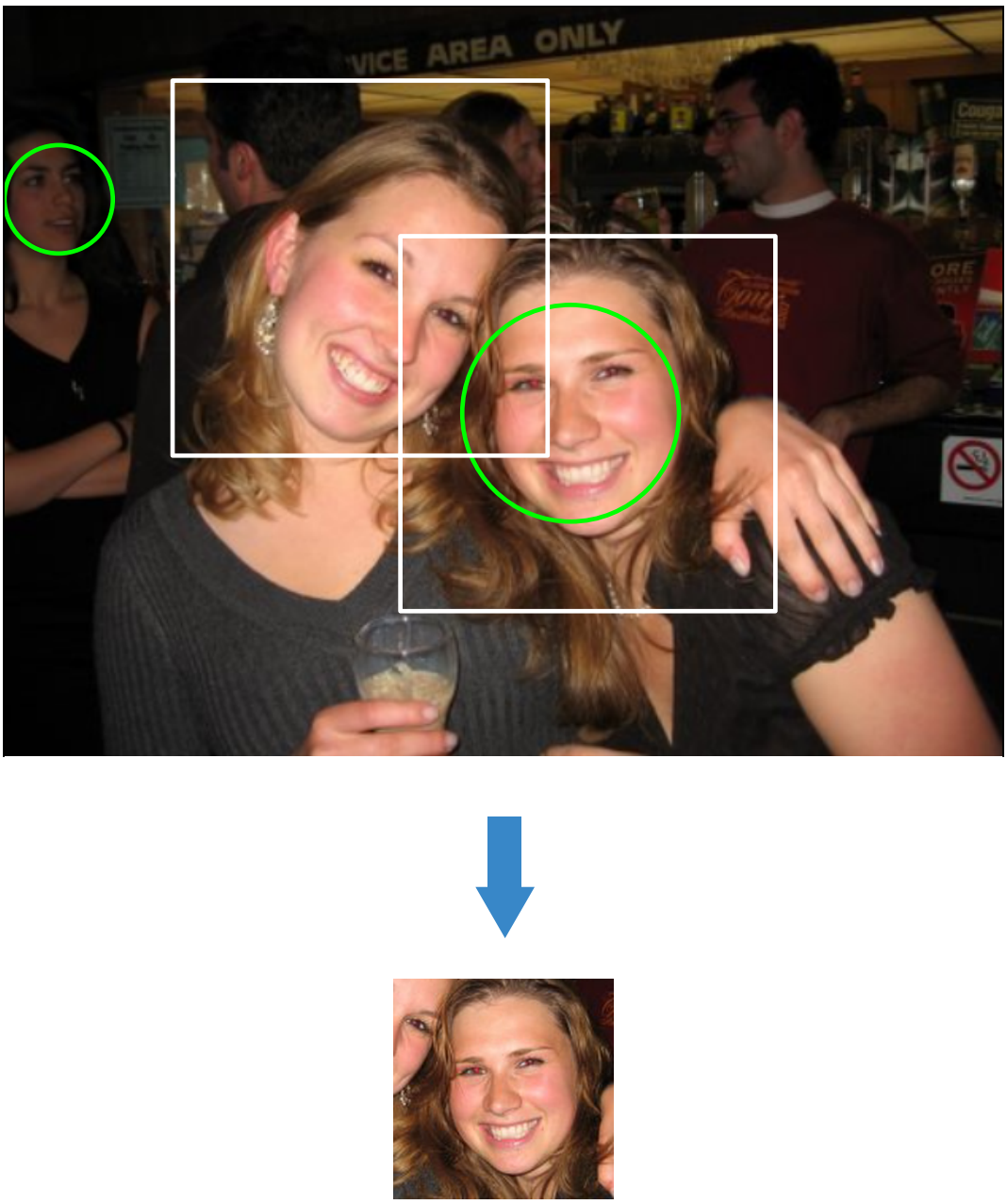



**Figure 12.1:** A representative Facebook photo with manually-applied "tags" (white square outlines) and OpenCV face detections (green circles) superimposed. Detected face regions are matched with nearby tags to yield labeled face samples, as shown above. Note that the untagged face detected in the background is correctly not matched with the unassigned tag. Because tags carry social meaning and can trigger notifications to hundreds of people when assigned, the identities that they specify are typically extremely accurate. In the photo above, the leftmost foreground face is rotated too far to be detected by OpenCV; more sophisticated face detection techniques would make it possible to harvest this labeled face sample as well. Figure from Zak Stone.

Chapter 13

# Evaluating the Invariance Properties of Successful Biologically-Inspired Face Recognition Systems*

"If there is no solution, it is because there is no problem."

*Les Shadoks*

A key challenge in building face recognition systems – biologically-inspired or otherwise – is evaluating performance. While much of face recognition research has traditionally used posed photographs for evaluation, recent efforts have emerged to build more naturalistic, unconstrained test sets by collecting large numbers of face images from the internet (e.g. the "Labeled Faces in the Wild" (LFW) test set [Huang *et al.*, 2007]). While such efforts represent a large step forward in the direction of realism, the nature of posed photographs from the internet arguably represents an

** This chapter is modified from a study that will be published in the proceedings of the International ICST Conference on Bio-Inspired Models of Network, Information, and Computing Systems (BIONETICS) in collaboration with David D. Cox [Pinto et al., 2011a].*

incomplete sampling of the range of variation in view, lighting, etc. found in the real world.

Here, we evaluate further the family of large-scale biologically-inspired vision algorithms that has previously proven to outperform the current state-of-the-art approaches on a variety of object and face recognition test sets, and specifically on the challenging LFW, PubFig83 and Facebook100 face benchmarks (see Chapters 11 and 12).

As a counterpoint to internet-photo based approaches, we use synthetic (rendered) face images where the amount of view variation is controllable and known by design (as in Chapters 5, 6, 7 and 9). We show that while there is gross agreement between the LFW benchmark and synthetic benchmarks, the synthetic benchmarks reveal a substantially greater degree of tolerance to view variation than is apparent from the LFW benchmark in models containing deeper hierarchies. Furthermore, such an approach yields important insights into which axes of variation are most challenging.

These results underscore, again, that parametric synthetic benchmarks can play a critical role in guiding and monitoring the progress of biologically-inspired vision systems.

## 13.1 Introduction

In Chapter 9, we described a very simple but large-scale feature search approach in which thousands of candidate biologically-inspired feature sets are rapidly "screened" to find model architectures that are well suited to a given problem domain. In Chapter 11, we applied this method to the LFW face *verification* challenge, and find that it achieves high levels of performance, on par with state-of-the-art methods, even without using any particularly sophisticated machine-learning backend. In Chapter 12, we showed that the models screened achieved excellent performance on PubFig83 and Facebook100 face *identification* benchmarks.

However, face sets like LFW provide little direct insight into why one model performs better than another, and the extent to which the LFW set – which is primarily composed of posed photographs of celebrities – is reflective of the "real" problem of unconstrained face recognition is not entirely clear. In particular, it is not clear that this set contains an

accurate sampling of the range of view variation found in the real world (see Chapters 5 and 6) since most images are frontal views, and some of the examples of a given individual are taken on the same day, at the same event (e.g. multiple photos of Halle Berry taken from the academy awards ceremony). Thus, while the LFW challenge is clearly useful, and an improvement over more controlled sets, it does not provide an obvious path to the full evaluation of a vision model, nor is it clear how performance on the LFW sets will transfer to other real-world scenarios.

As an complement to the LFW set, we here draw upon carefully-crafted synthetic image sets. While synthetic images have fallen out of favor in the computer vision community in recent years, advances in 3D rendering software have increasingly narrowed the gap between real and synthetic imagery, and rendered images offer several critical advantages over collected photographs. In particular, rendered images allow for complete knowledge and control over the view, position, scale, lighting, presence of other objects etc. in a scene. As a result, synthetic test sets that span whatever range of variation the experimenter desires can be easily generated, and tasks of parametrically variable difficulty can be constructed. Importantly, such data sets also allow one to specifically test the performance of a model as a function of variation in view, lighting, etc. The ability of a model to tolerate such variation – referred to as "invariance" in the parlance of neuroscience – is a critical property of natural vision systems, and is a key stumbling block in the creation of artificial systems.

## 13.2 Methods

In the experiments presented below, we studied the biologically-inspired visual representations used and described in previous chapters: *V1-like*, multi-layer "High-Throughput" (HT) models (see Methods Section 11.2). The *HT* models having either two or three layers (referred to hereafter as *HT-L2* and *HT-L3*, respectively).

| | *5th* | *4th* | *3rd* | *2nd* | *1st* |
|---|---|---|---|---|---|
| *V1-like* | 77.0 ± 0.5 | | | | |
| *HT-L2* | 77.8 ± 0.4 | 81.3 ± 0.4 | 81.5 ± 0.6 | 80.8 ± 0.4 | 81.0 ± 0.3 |
| *HT-L3* | 82.8 ± 0.6 | 82.3 ± 0.4 | 83.3 ± 0.4 | 83.9 ± 0.3 | 84.1 ± 0.3 |

**Table 13.1: Performance of the family of biologically-inspired models on the *LFW* challenge set (Restricted View 2).** For the *HT-L2* and *HT-L3* models, the cross-validated performance of the top 5 randomly-generated models is shown (e.g. 1st, 2nd, etc.). The performance of the simpler single layer *V1-like* model is provided for comparison.

### 13.2.1 Synthetic Face Images

In order to assess model performance on an image set with a known amount of variation, we generated a set of 3D-rendered face images as in Chapters 5, 6 and 7. For the experiments presented here, rotation, size, and position were combined into a single composite "variation level" wherein the variation in the pixel-level euclidean norm was equalized for each kind of variation (e.g. one "unit" of rotation variation produced an equivalent pixel-level change as one "unit" of position variation). Examples of several variation "levels" are shown in Figure 13.1(a).

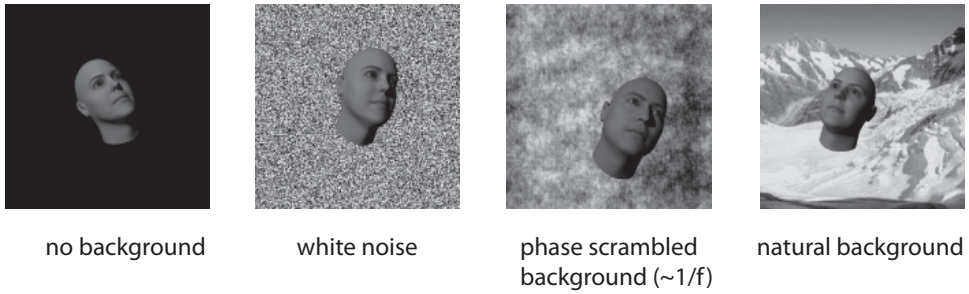
The rendered face/head was next composited onto one of four kinds of backgrounds: no background, a white noise background, a phase-scrambled natural background (approximately equivalent to $\frac{1}{f}$ "pink" noise), and a randomly chosen natural background, chosen from a large pool of outdoor background images (Figure 13.1(b)). Care was taken to ensure that the same background image was never used in more than one final image.

### 13.2.2 Classification and Performance Evaluation

To evaluate the performance of a given model with a given stimulus set, we trained a multi-class support vector machine (SVM) classifier [Scholkopf and Smola, 2002] using a one-vs-all configuration [Rifkin and Klautau, 2004] for each target class.

Training and test data were strictly segregated, and performance was evaluated using five 250 train / 50 test random folds of the data. Error bars in all plots show the


(a) View, position an scale variation




(b) Background variation

**Figure 13.1: Synthetic face stimuli.**

standard deviation of performance across these five folds.

## 13.3 Results

### 13.3.1 LFW performance

Performance on the LFW data set for these models is presented in Table 13.1. Performance ranged as high as 84.1% percent correct for the best HT-L3 model, achieving performance within a few percent of state-of-the-art methods [Kumar *et al.*, 2009; Wolf *et al.*, 2009]. While more sophisticated kernel blending techniques have previously been

used to achieve better performance on the LFW challenge set by leveraging multiple feature representations (as in Chapters 6 and 11), we here restrict ourselves here to unblended model performance for the sake of clarity. Further, for simplicity, we also here only consider the best-performing model from each group (i.e. HT-L2-1st and HT-L3-1st).

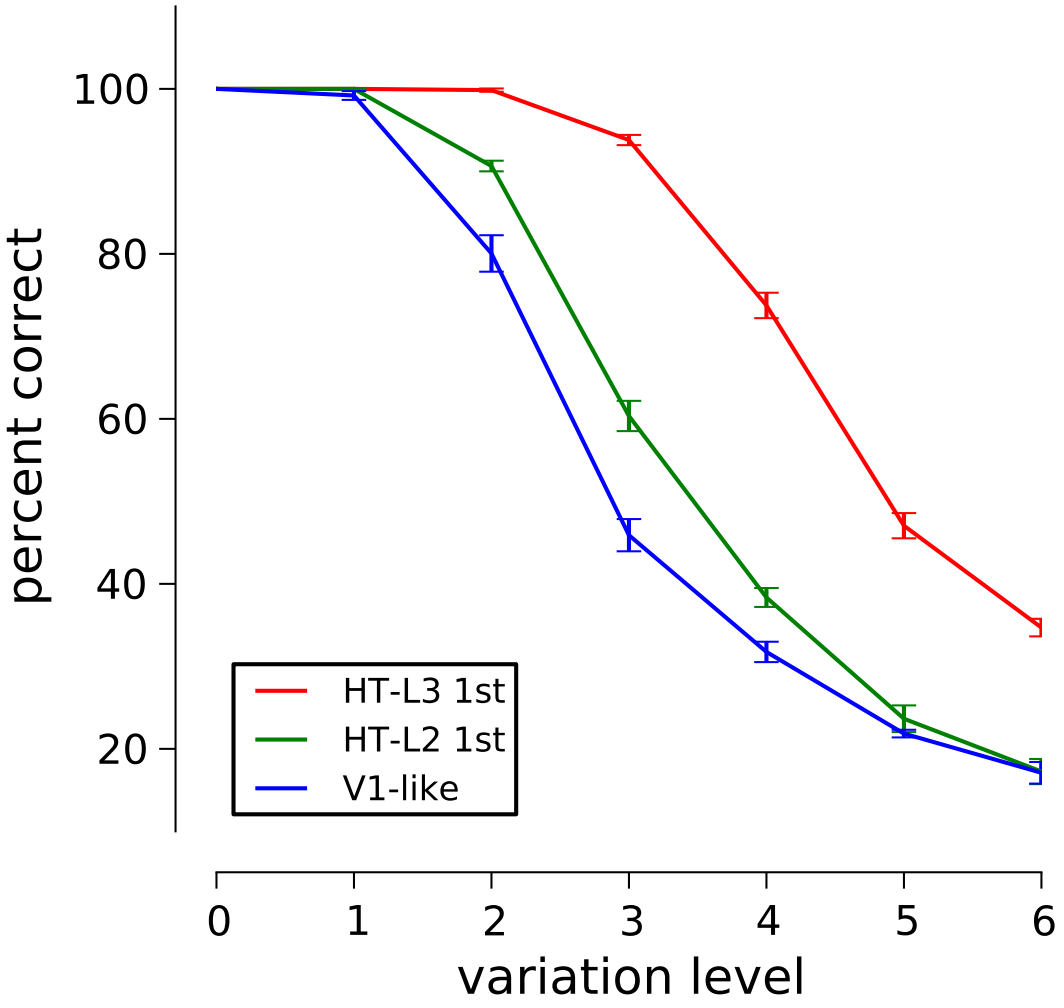



**Figure 13.2: Model performance on synthetic faces as a function of level of variation.**

### 13.3.2 Performance as a function of variation level

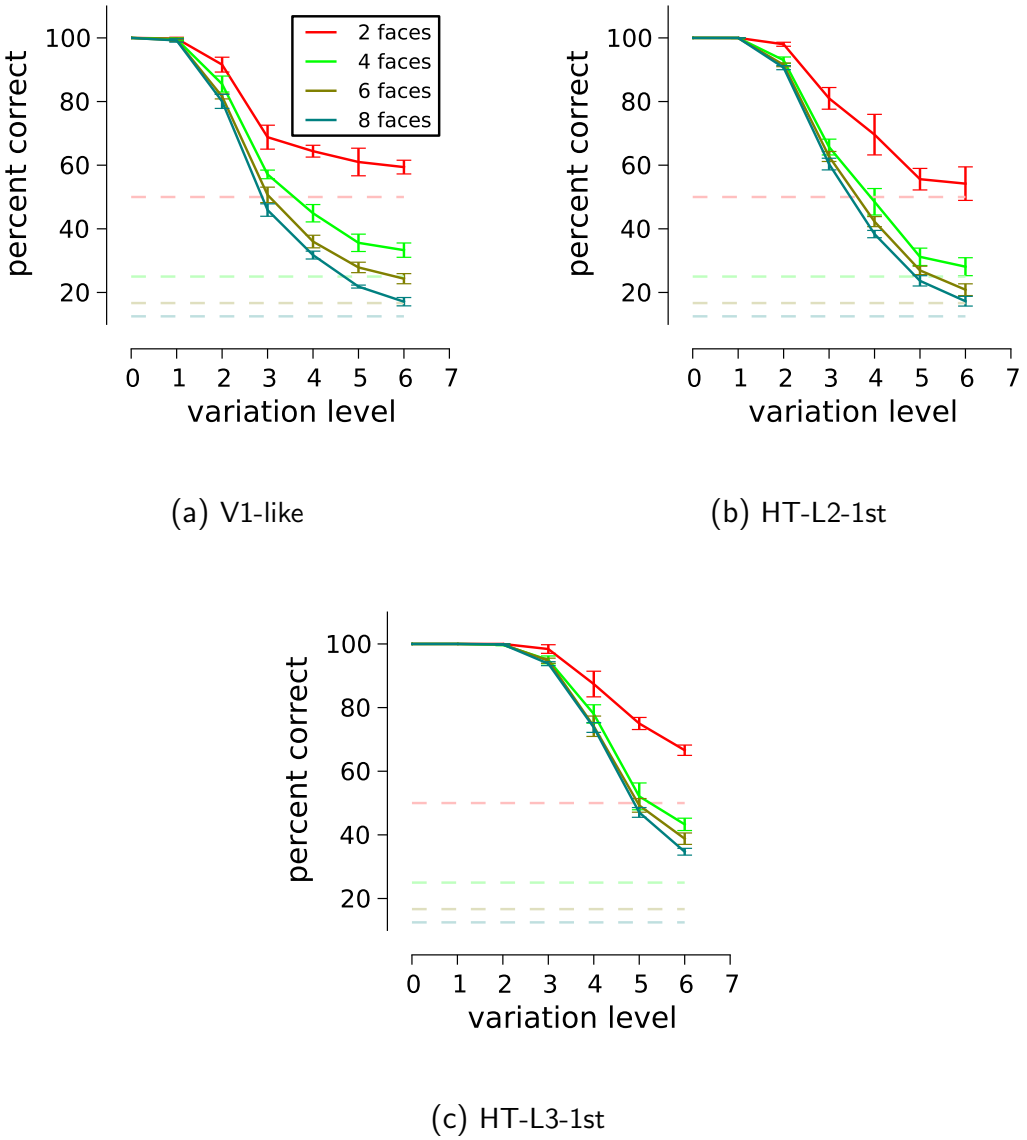
The synthetic face evaluation sets used here provide us with the ability to parametrically control the level of rotation, position and scale variation that our models are required to tolerate. Figure 13.2 shows the performance the best models from each model class (V1-like, HT-L2, HT-L3) as a function of (composite) variation level for an eight-way face classification task.

### 13.3.3 Effect of number of faces to be discriminated

To further explore the behavior of our models with a controlled stimulus, we examined model performance as a function of the number of faces to be discriminated. In particular, we considered cases with two, four, six, and eight faces. Performance, grouped by model is shown in Figure 13.3, and is shown grouped by variation level in Figure 13.4. Predictably, absolute performance level is depressed as a larger number of faces is considered, as is the chance performance level (dotted line). Interestingly, the rate at which performance falls off varies between models as a function of both number of faces to be discriminated, and as a function of variation level. The stability of the performance of the largest/deepest model – HT-L3-1st – is most pronounced when large number of faces and large amounts of variation are considered. Differences between models are far less pronounced with smaller numbers of faces and lesser degrees of variation.

### 13.3.4 Effect of background

To explore the role of background variation, we evaluated model performance with four different background conditions: no background, white-noise background, phase-scrambled natural backgrounds (i.e. $\frac{1}{f}$ "pink" noise), and natural backgrounds. Performance as a function of background and variation level is shown Figure 13.5. Choice of background was found to have a profound effect on model performance. In the absence of a background, the performance for most models remained high, even at relatively high levels of variation in view, position, and scale (e.g. greater than 90% performance at variation level 4 for the HT-L3-1st and V1-like models). However, the inclusion of




(a) V1-like

(b) HT-L2-1st

(c) HT-L3-1st

**Figure 13.3: Effect of number of synthetic faces to be discriminated, sorted by model.**

any background resulted in a precipitous drop-off in performance for all models, except for the HT-L3-1st model, whose performance degraded gradually. In general, progressively more realistic backgrounds proved increasingly difficult for all models.

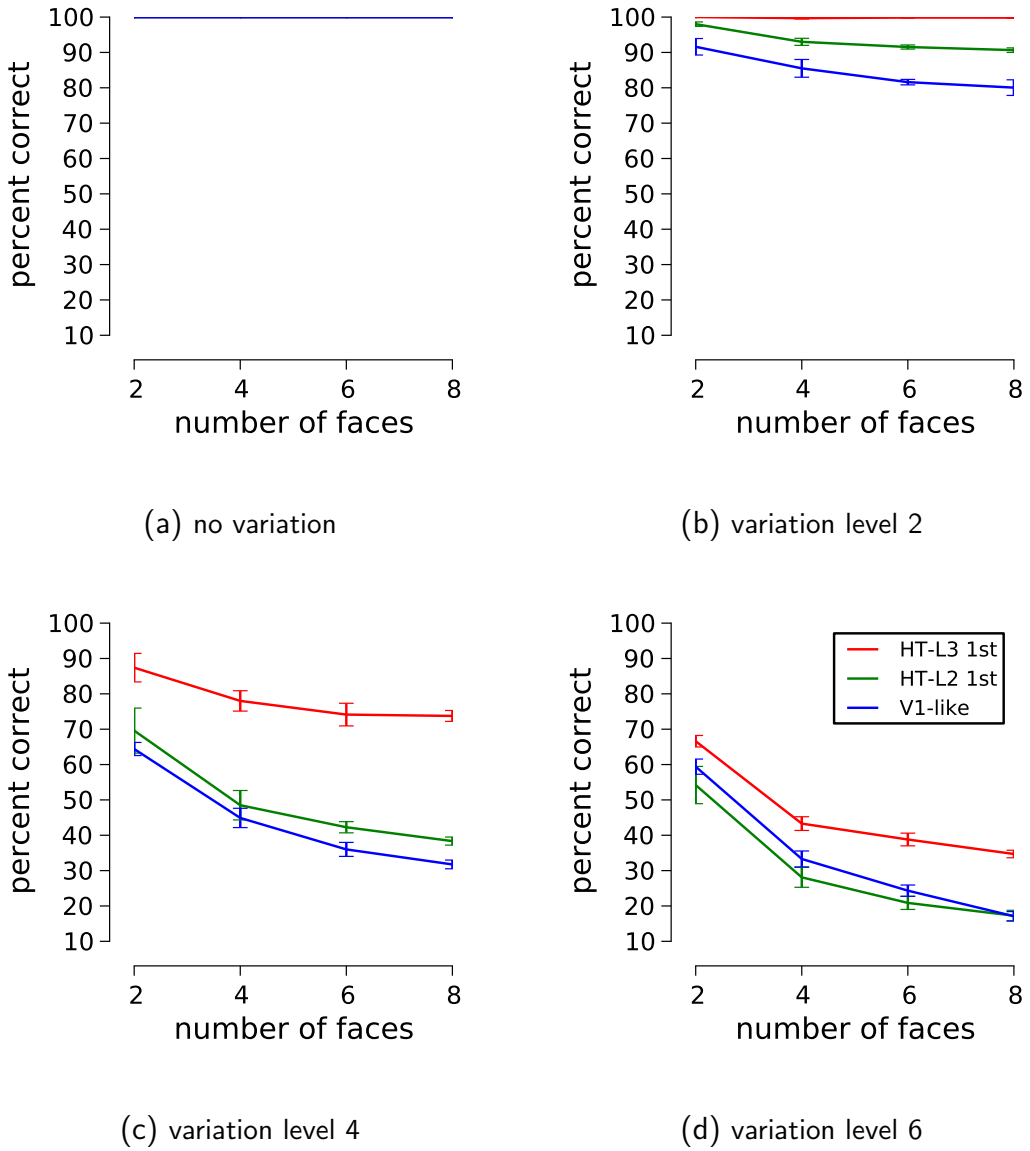



**Figure 13.4: Effect of number of synthetic faces to be discriminated, sorted by variation level.** Note that the performance was 100% in all cases for the zero variation.

## 13.4 Discussion

While it is standard practice to test computer vision algorithms with standardized "natural" image test sets such as the LFW set, the performance obtained on such a set provides a relatively narrow window onto behavior of a given system. Here, we used

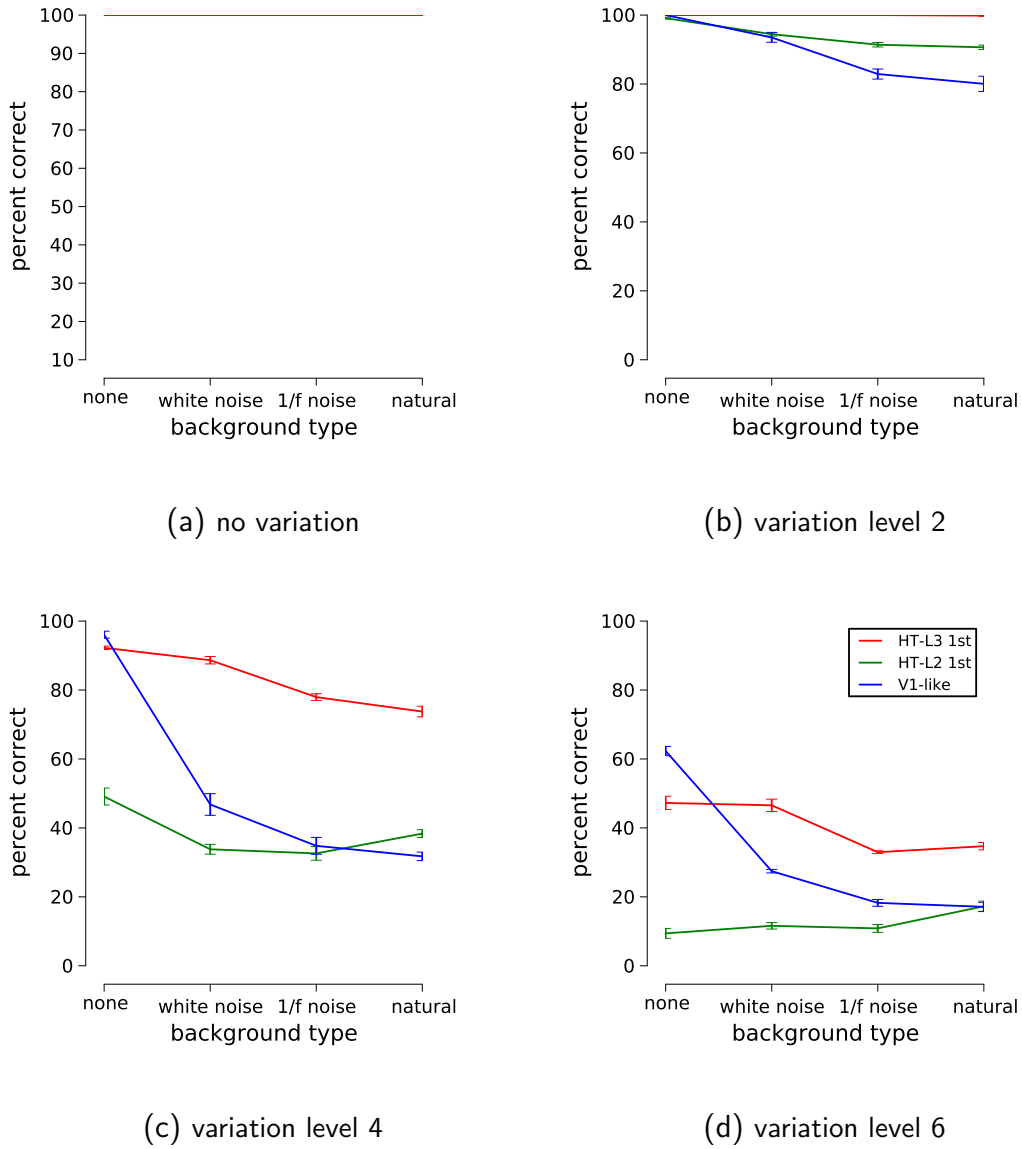



**Figure 13.5: Effect of background type on performance with synthetic faces.** Note that the performance was 100% in all cases for the zero variation condition.

synthetic test images, rendered with known amounts of variation, to provide a much richer multidimensional assessment of the invariance properties of a class of models that have achieved high levels of performance on the LFW set.

While the ordinal performance of the one-, two- and three-layer models considered

here is roughly the same as is observed for the LFW set (i.e. V1-like < HT-L2 < HT-L3), tests with synthetic sets reveal that the model with the deepest hierarchy (HT-L3) is substantially better able to tolerate variation in view, position, scale and background as compared to the other models considered here. This dramatic difference was not at all apparent from the LFW performance, where the best HT-L3 model performed only a few percent higher than its nearest rivals. While there is no hard evidence one way or another, we speculate that the relatively compressed range of performance between the various models on the LFW set is reflective of the relatively limited range of view variation found in that set. Indeed, when we examine a relatively low level of variation with our synthetic faces, we see a similarly compressed range of performance variation across the models.

More broadly, our results suggest that the level of variation present in a set, both in terms of view and in terms of background can have a large effect on the "dynamic range" within which one has the ability to distinguish between models. Indeed, without any background, and at low levels of variation, the differences between models can become vanishing small, and in some cases can even reverse. These results underscore the importance of building sets, be they synthetic or natural, that contain more realistic ranges of variation.

# Acknowledgments

This study was funded in part by the NVIDIA Graduate Fellowship, the Singleton Fellowship, the Rowland Institute of Harvard and the National Science Foundation.

# Part V

# Discussion

Chapter 14

# Summary and Key Contributions

This chapter highlights the key contributions of this thesis and outlines their implications.

## 14.1 New Baselines and Benchmarks

In Part II, we introduced clear and measurable indicators of progress to support efforts in developing visual object recognition models.

We found that caution is warranted when using large databases of ostensibly "natural" images (collected from the web) by showing that very simple neuroscience "toy" models (capable only of extracting trivial regularities from a set of images) were able to outperform most state-of-the-art object and face recognition systems on many standard "natural" image benchmarks.

At the same time, these rudimentary models were easily defeated by apparently "simpler" synthetic (ray-traced) recognition tasks that we designed to efficiently capture many of the key parameters that make visual object recognition challenging, and thus to better span the range of real world image variation (object pose, position, scale, etc.) in a tightly controlled way (Chapters 4, 5 and 6).

We also compared and contrasted a variety of state-of-the-art visual recognition systems on the same controlled tasks and reported that most of them performed very

poorly on invariant recognition (with the exception of a single high-level biologically-inspired model), despite the expectation from leaders in the field that they should be capable of dealing "fairly well" with simpler synthetic invariance tests, even though some of these tests remain trivially easy for human observers. With a relatively small image set and minimal effort, we showed how this approach can more deeply illuminate the strengths and weaknesses of different visual systems (Chapters 7 and 8).

Taken together, our experiments demonstrated that even though current uncontrolled image sets that look "natural" to human observers are laudable because they encourage systematic comparison of various algorithms. Furthermore, they can be dangerous when hidden confounds exist in the sets, or misleading when it is not clear *why* the sets are difficult – hence potentially guiding progress in the wrong direction. In addition to tempering claims of success in the computer vision literature, these results (1) suggest that care must be taken to establish appropriate baselines against which performance can be judged, (2) call for a reexamination of what it means for images to be natural and point the way forward by renewing our focus on image variation as a central computational challenge in visual recognition, and (3) underscore the importance of building *efficient* parametric tests focused on capturing important insights into which axes of variation are most challenging.

## 14.2 High-Throughput Solution Discovery

In Part III, we showed that a high-throughput exploration of the large hypothesis space of possible computational vision models inspired by the visual cortex was attainable by harnessing the power of massively parallel ubiquitous graphics hardware (e.g. Sony's PlayStation 3 and NVIDIA GPUs).

We demonstrated that this conceptually straightforward proof-of-concept (which is no more and no less than a pure *brute-force* controlled model selection procedure or "fishing expedition") was extremely effective at identifying promising models within a relatively restricted "first-order" family of semi-supervised hierarchical feature-based models composed of three basic layers. The models we found consistently outperformed the experimentally-motivated baseline models set forth in Part II as well as a crop of

state-of-the-art computer vision systems that have been hand-tuned for many years.

Furthermore, we validated that the use of small synthetic sets proposed in Part II was an efficient way of screening and selecting models that generalize well since the representations of visual space instantiated by these models were found to be useful *generally* across a variety of controlled object and face recognition tasks, even without any specific optimization (which are usually performed by other approaches). This reinforces our belief that parametric sets can capture the essence of the invariant object recognition problem. Another critical advantage of this parametric screening approach is that task difficulty can be increased on demand – that is, as models are found to succeed at a given level of image variation, the level of variation (and therefore the level of task difficulty), can be "ratcheted up" as well, maintaining evolutionary "pressure" towards better and better models.

Our computational framework – based on flexible open-source tools with no commercial dependency (e.g. Python), simple automated code optimization techniques (see Section 9.6 and Chapter 10), and large computing resources (e.g. "do-it-yourself" custom-built clusters of GPUs) – played an essential role in making our experiments possible. Instead of spending many years and/or hundreds of thousand dollars with conventional methods (e.g. MATLAB with CPUs), our approach can be applied in a couple of weeks at a low cost. We believe that our models are among the first to achieve state-of-the-art performance while being applied at scales approaching that of high-level biological visual systems, both in terms of input dimensionality and the amount of experience obtained during development.

From the "first-order" family used here, we observed that only a handful of model instantiations performed substantially better than the rest, with these "good" models occurring at a low rate. The relative rarity of these models illustrates the importance of performing large-scale experiments with many instantiations, since they would otherwise be easy to miss in traditional "one-off" modes of exploration.

In sum, these results point a new way forward in the creation of high-performing computer vision systems and underscore the importance of directly tackling one of the primary obstacles to understanding the computational underpinnings of biological vision: its sheer scale.

## 14.3 Large-Scale Applications

In Part IV, we applied models harvested from our high-throughput approach to new large-scale application domains such as face recognition "in the wild". The primary purpose was to validate the *generality* and *scalability* of our simple method.

We screened thousands of stripped-down[1] two- and three-layer models on the Labeled Faces in the Wild (LFW) unconstrained face *verification*[2] training set. Consistent with expectations, progressively deeper multi-layer models achieved increasingly better performance and were able to outperform the simpler one-layer neuroscientist "null" model proposed as a baseline in Part II. Interestingly, straightforward blends of the best models defeated other state-of-the-art approaches on the LFW test set, in spite of requiring less training data and using a conceptually simpler machine learning backend (Chapter 11).

In addition, we showed that the same models were capable of dealing fairly well with large-scale face *identification*[3] as it is not obvious *a priori* how performance on a verification task ought to relate to performance on an identification task. To this end, we introduced two new sets: (1) *Facebook100*: a naturalistic set of face images gather from the Facebook social networking website, and (2) *PubFig83*: a filtered subset of the publicly-available *PubFig* data adapted for identification tasks with many near-duplicate images removed. Interestingly, our models yielded high levels of face-identification performance even when large numbers of individuals were considered and the accuracy increased steadily as more training examples were used (Chapter 12).

We also addressed the concern raised in Part II that high performance on uncontrolled "natural" sets may *not* come from the ability to tolerate image variation, even though it was one of the original purposes for the construction of the model class (Chapter 9). First, we presented an analysis of the errors made by our various models (Chapter 11), and observed that each of them makes appreciably the same errors, and that a large fraction of errors can be qualitatively explained by variation in the view

[1] For the sake of simplicity, and to be able to screen more models per unit time, these models did not have any structure in their filter kernels, nor were they subjected to the unsupervised learning stage described in Part III

[2] Asking whether two faces are from the same person.

[3] Labeling the face, i.e. asking "which person is this?"

of the targets (i.e. common error cases were dominated by misses when the same individual is shown in differing views and by false positives when two different individuals are compared while viewed from a similar angle). Then, we evaluated the best performing models using the controlled synthetic procedure described in Part II and we showed that while there is gross agreement between the "natural" face benchmarks and the synthetic benchmarks, the synthetic benchmarks reveal a substantially greater degree of tolerance to image variation (in view, position, scale and background) than the "natural" benchmarks in models containing deeper hierarchies. Finally, we discovered that the deepest (three-layer) screened model was substantially better able to tolerate variation as compared to the other models considered (Chapter 13).

Collectively, these results provide convincing evidence that high-throughput screening of biologically-inspired models represent a promising and powerful direction in the development of large-scale "real-world" applications, especially because the performance numbers reported only serve as a lower bound on the performance that might be possible. In addition, these results underscore, again, the importance of building benchmarks, be they synthetic or natural, that contain more realistic ranges of variation.

## 14.4 Expectations

As the scale of the available computational power provided by massively parallel technology continues to expand, we believe that this research holds great potential for rapidly advancing our understanding of the principles underlying visual perception, accelerating progress in computer vision, and, most importantly, to generate new experimentally testable hypotheses for the study of biological vision.

In addition to guide future experiments aimed at "reverse-engineering" the brain, we hope this thesis will contribute to the development of new intelligent visual technologies.

# Bibliography


Adee, S. 2008. Reverse engineering the brain. *Spectrum, IEEE*, **45**(6), 51–53.

Afraz, S.R., Kiani, R., and Esteky, H. 2006. Microstimulation of inferotemporal cortex influences face categorization. *Nature*, **442**(7103), 692–695.

Ahissar, M., and Hochstein, S. 2004. The reverse hierarchy theory of visual perceptual learning. *Trends in Cognitive Sciences*, **8**(10), 457–464.

Ahonen, T., Hadid, A., and Pietikainen, M. 2004. Face recognition with local binary patterns. *ECCV*.

Ahonen, T., Hadid, A., *et al.*. 2006. Face description with local binary patterns: Application to face recognition. *PAMI*.

Aizenstein, H.J., MacDonald, A.W., Stenger, V.A., Nebes, R.D., Larson, J.K., Ursu, S., and Carter, C.S. 2000. Complementary category learning systems identified using event-related functional MRI. *Journal of Cognitive Neuroscience*, **12**(6), 977–987.

Albiol, A., Monzo, D., Martin, A., Sastre, J., and Albiol, A. 2008. Face recognition using HOG-EBGM. *Pattern Recognition Letters*.

Alonso, J.M., Usrey, W.M., and Reid, R.C. 1996. Precisely correlated firing in cells of the lateral geniculate nucleus. *Nature*, **383**(6603), 815–819.

Ames, M., and Naaman, M. 2007. Why We Tag: Motivations for Annotation in Mobile and Online Media. *SIGCHI Conference on Human Factors in Computing Systems*.

Amit, Y., and Geman, D. 1999. A computational model for visual selection. *Neural computation*, **11**(7), 1691–1715.

Anderson, C.H., and Van Essen, D.C. 1987. Shifter circuits: a computational strategy for dynamic aspects of visual processing. *Proceedings of the National Academy of Sciences of the United States of America*, **84**(17), 6297.

Anzai, A., Peng, X., and Van Essen, D.C. 2007. Neurons in monkey visual area V2 encode combinations of orientations. *Nature Neuroscience*, **10**(10), 1313–1321.

Arathorn, D.W. 2002. *Map-seeking circuits in visual cognition: A computational mechanism for biological and machine vision*. Stanford University Press.

Bach, F.R., Lanckriet, G.R.G., and Jordan, M.I. 2004. Multiple kernel learning, conic duality, and the SMO algorithm. *Proceedings of the International Conference on Machine learning (ICML)*.

Baker, C.I., Behrmann, M., and Olson, C.R. 2002. Impact of learning on representation of parts and wholes in monkey inferotemporal cortex. *Nature Neuroscience*, **5**(11), 1210–1216.

Balas, B., Cox, D., and Conwell, E. 2007. The effect of real-world personal familiarity on the speed of face information processing. *PloS One*, **2**(11).

Barlow, H. 1994. What is the computational goal of the neocortex?

Barlow, HB. 1961. Possible principles underlying the transformation of sensory messages. *Sensory Communication*.

Barlow, H.B. 1972. Single units and sensation: a neuron doctrine for perceptual psychology. *Perception*, **1**(4), 371–394.

Barlow, HB. 1985a. Cerebral cortex as model builder. *Models of the visual cortex*, 37–46.

Barlow, HB. 1985b. The twelfth Bartlett memorial lecture: The role of single neurons in the psychology of perception. *The Quarterly Journal of Experimental Psychology Section A*, **37**(2), 121–145.

Barlow, H.B. 1989. Unsupervised learning. *Neural Computation*, **1**(3), 295–311.

Barone, P., Batardiere, A., Knoblauch, K., and Kennedy, H. 2000. Laminar distribution of neurons in extrastriate areas projecting to visual areas V1 and V4 correlates with the hierarchical rank and indicates the operation of a distance rule. *Journal of Neuroscience*, **20**(9), 3263.

Becker, S. 1993. Learning to categorize objects using temporal coherence. *Advances in neural information processing systems*, 361–361.

Becker, S., and Hinton, G.E. 1992. Self-organizing neural network that discovers surfaces in random-dot stereograms. *Nature*, **355**(6356), 161–163.

Belhumeur, P.N., Hespanha, J.P., and Kriegman, D.J. 2002. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *PAMI*.

Bell, A.J., and Sejnowski, T.J. 1997. The "independent components" of natural scenes are edge filters. *Vision research*, **37**(23), 3327.

Bengio, Y. 2009. *Learning deep architectures for AI*. Now Publishers Inc.

Bengio, Y., and LeCun, Y. 2007. Scaling learning algorithms towards AI. *Large-Scale Kernel Machines*, **34**.

Bengio, Y., Lamblin, P., Popovici, D., and Larochelle, H. 2007. Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems 19: Proceedings of the 2006 Conference*, 153.

Berg, A. C., and Malik, J. 2001. Geometric blur and template matching. *CVPR*.

Berg, T., Berg, A., Edwards, J., Maire, M., White, R., Teh, Y.W., Learned-Miller, E., and Forsyth, D. 2004. Names and faces in the news. *Computer Vision and Pattern Recognition Conference (CVPR)*.

Berkes, P., and Wiskott, L. 2005. Slow feature analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, **5**(6).

Bertero, M., Poggio, T.A., and Torre, V. 1988. Ill-posed problems in early vision. *Proceedings of the IEEE*, **76**(8), 869–889.

Biederman, I. 1972. Perceiving real-world scenes. *Science*, **177**(43), 77–80.

Biederman, I. 1987. Recognition-by-components: A theory of human image interpretation. *Psychological Review*, **94**(115-148), 32–33.

Biederman, I., and Cooper, E.E. 1991. Priming contour-deleted images: Evidence for intermediate representations in visual object recognition* 1. *Cognitive Psychology*, **23**(3), 393–419.

Biederman, I., Rabinowitz, J.C., Glass, A.L., and Stacy, E.W. 1974. On the information extracted from a glance at a scene. *Journal of Experimental Psychology*, **103**(3), 597–600.

Bileschi, S.M. 2006. StreetScenes: Towards scene understanding in still images. *MIT CSAIL PhD Thesis*.

Binford, T.O. 1971. Visual perception by computer. *In: IEEE conference on Systems and Control*, vol. 313.

Bishop, C.M. 1995. *Neural networks for pattern recognition*. Oxford University Press, USA.

Blaschke, T., Berkes, P., and Wiskott, L. 2006. What is the relation between slow feature analysis and independent component analysis? *Neural Computation*, **18**(10), 2495–2508.

Booth, MC, and Rolls, E.T. 1998. View-invariant representations of familiar objects by neurons in the inferior temporal visual cortex. *Cerebral Cortex*, **8**(6), 510.

Bosch, A., Zisserman, A., and Munoz, X. 2007. Representing shape with a spatial pyramid kernel. *Proceedings of the International Conference on Image and Video Retrieval (CIVR)*.

Bottou, L., and Bengio, Y. 1995. Convergence Properties of the KMeans Algorithm. *Advances in Neural Information Processing Systems (NIPS)*.

Boureau, Y-Lan, Bach, Francis, LeCun, Yann, and Ponce, Jean. 2010a. Learning Mid-Level Features for Recognition. *International Conference on Computer Vision and Pattern Recognition (CVPR)*.

Boureau, Y-Lan, Ponce, Jean, and LeCun, Yann. 2010b. A theoretical analysis of feature pooling in vision algorithms. *International Conference on Machine learning (ICML)*.

Boussaoud, D., Ungerleider, L.G., and Desimone, R. 1990. Pathways for motion analysis: cortical connections of the medial superior temporal and fundus of the superior temporal visual areas in the macaque. *The Journal of comparative neurology*, **296**(3), 462–495.

Bouvrie, J., Rosasco, L., and Poggio, T. 2009. On Invariance in Hierarchical Models. *Advances in Neural Information Processing Systems (NIPS)*, **22**.

Boynton, G.M., and Hegdé, J. 2004. Visual cortex: The continuing puzzle of area V2. *Current Biology*, **14**(13), R523–R524.

Brady, M.J., and Kersten, D. 2003. Bootstrapped learning of novel objects. *Journal of Vision*, **3**(6).

Brady, T.F., Konkle, T., Alvarez, G.A., and Oliva, A. 2008. Visual long-term memory has a massive storage capacity for object details. *Proceedings of the National Academy of Sciences*, **105**(38), 14325.

Brincat, S.L., and Connor, C.E. 2004. Underlying principles of visual shape selectivity in posterior inferotemporal cortex. *Nature Neuroscience*, **7**(8), 880–886.

Bruce, V. 1986. Influences of familiarity on the processing of faces. *Perception*, **15**, 387–397.

Bulthoff, H.H., Edelman, S.Y., and Tarr, M.J. 1995. How are three-dimensional objects represented in the brain? *Cerebral Cortex*, **5**(3), 247.

Burl, MC, and Perona, P. 1996. Recognition of Planar Object Classes. *Computer Vision and Pattern Recognition, Proceedings of the IEEE Computer Society Conference on*.

Cadieu, C., Kouh, M., Pasupathy, A., Connor, C.E., Riesenhuber, M., and Poggio, T. 2007. A model of V4 shape selectivity and invariance. *Journal of Neurophysiology*, **98**(3), 1733.

Callaway, E.M. 1998. Local circuits in primary visual cortex of the macaque monkey. *Annual Review of Neuroscience*, **21**(1), 47–74.

Cao, Z., Yin, Q., Tang, X., and Sun, J. 2010. Face recognition with learning-based descriptor. *CVPR*.

Carandini, M., Heeger, D.J., and Movshon, J.A. 1997. Linearity and normalization in simple cells of the macaque primary visual cortex. *Journal of Neuroscience*, **17**(21), 8621.

Carandini, M., Demb, J.B., Mante, V., Tolhurst, D.J., Dan, Y., Olshausen, B.A., Gallant, J.L., and Rust, N.C. 2005. Do we know what the early visual system does? *Journal of Neuroscience*, **25**(46), 10577.

Cardoso-Leite, P., and Gorea, A. 2010. On the Perceptual/Motor Dissociation: A Review of Concepts, Theory, Experimental Paradigms and Data Interpretations. *Seeing and Perceiving*, **23**(2), 89–151.

Chang, C.C., and Lin, C.J. 2001. LIBSVM: a library for support vector machines. *Software available at* `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chelazzi, L., Duncan, J., Miller, E.K., and Desimone, R. 1998. Responses of neurons in inferior temporal cortex during memory-guided visual search. *Journal of Neurophysiology*, **80**(6), 2918.

Chikkerur, S.S., Serre, T., Tan, C., and Poggio, T. 2010. What and where: A Bayesian inference theory of attention. *Vision Research*.

Chin, Gilbert. 2010. Neuroscience: The Next Top Model. *Science*, **327**(5961), 13.

Chopra, Sumit, Hadsell, Raia, and LeCun, Yann. 2005. Learning a Similarity Metric Discriminatively, with Application to Face Verification.

Christensen, H.I., and Phillips, P.J. 2002. *Empirical evaluation methods in computer vision*.

Clutterbuck, R., and Johnston, R.A. 2002. Exploring levels of face familiarity by using an indirect face-matching measure. *Perception*, **31**(8), 985–994.

Collins, B., Deng, J., Kai, L., and L., L. Fei-Fei. 2008. Towards scalable dataset construction: An active learning approach. *Proceedings of the European Conference on Computer Vision (ECCV)*.

Comon, P. 1994. Independent component analysis, a new concept? *Signal processing*, **36**(3), 287–314.

Computer Vision Lab at the University of Ljubljana. 1999. CVL Face Set. *Available at* `http://www.lrv.fri.uni-lj.si/facedb.html`.

Cox, D.D. 2007. *Reverse engineering object recognition*. Ph.D. thesis, Massachusetts Institute of Technology.

Cox, D.D., Meier, P., Oertelt, N., and DiCarlo, J.J. 2005. 'Breaking'position-invariant object recognition. *Nature neuroscience*, **8**(9), 1145–1147.

Crandall, D., and Huttenlocher, D. 2006. Weakly supervised learning of part-based spatial models for visual object recognition. *ECCV*, 16–29.

Crandall, D., Felzenszwalb, P., and Huttenlocher, D. 2005. Spatial priors for part-based recognition using statistical models. *IEEE Conference on Computer Vision and Pattern Recognition*.

Creutzig, F., and Sprekeler, H. 2008. Predictive coding and the slowness principle: An information-theoretic approach. *Neural computation*, **20**(4), 1026–1041.

Crist, R.E., Li, W., and Gilbert, C.D. 2001. Learning to see: experience and attention in primary visual cortex. *Nature Neuroscience*, **4**(5), 519–525.

Dalal, N., and Triggs, B. 2005. Histograms of oriented gradients for human detection. *CVPR*.

de Beeck, H.O., Wagemans, J., and Vogels, R. 2001. Inferotemporal neurons represent low-dimensional configurations of parameterized shapes. *Nature Neuroscience*, **4**(12), 1244–1252.

De Sa, V.R., and Ballard, D.H. 1998. Category learning through multimodality sensing. *Neural Computation*, **10**(5), 1097–1117.

Dean, P. 1976. Effects of inferotemporal lesions on the behavior of monkeys. *Psychol. Bull*, **83**(1), 41–71.

Dean, P. 1982. Visual behavior in monkeys with inferotemporal lesions. *Analysis of visual behavior*, 587–628.

DeAngelis, G.C., Anzai, A., Ohzawa, I., and Freeman, R.D. 1995. Receptive field structure in the visual cortex: does selective stimulation induce plasticity? *Proceedings of the National Academy of Sciences of the United States of America*, **92**(21), 9682.

Deb, K. 2001. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley.

Deco, G., and Rolls, E.T. 2004. A neurodynamical cortical model of visual attention and invariant object recognition. *Vision research*, **44**(6), 621–642.

DeGutis, J., and D'Esposito, M. 2007. Distinct mechanisms in visual category learning. *Cognitive, affective & behavioral neuroscience*, **7**(3), 251.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 248–255.

Desimone, R. 1991. Face-selective cells in the temporal cortex of monkeys. *Journal of Cognitive Neuroscience*, **3**(1), 1–8.

Desimone, R., and Schein, S.J. 1987. Visual properties of neurons in area V4 of the macaque: sensitivity to stimulus form. *Journal of Neurophysiology*, **57**(3), 835.

Desimone, R., Albright, T.D., Gross, C.G., and Bruce, C. 1984. Stimulus-selective properties of inferior temporal neurons in the macaque. *Journal of Neuroscience*, **4**(8), 2051.

DeYoe, E.A., and Van Essen, D.C. 1988. Concurrent processing streams in monkey visual cortex. *Trends in Neurosciences*, **11**(5), 219–226.

DiCarlo, J.J., and Cox, D.D. 2007. Untangling invariant object recognition. *Trends in Cognitive Sciences*, **11**(8), 333–341.

DiCarlo, J.J., and Maunsell, J.H.R. 2000. Form representation in monkey inferotemporal cortex is virtually unaltered by free viewing. *nature neuroscience*, **3**(8), 814–821.

Dickinson, S. 2009. The evolution of object categorization and the challenge of image abstraction. *Object Categorization: Computer and Human Vision Perspectives.*

Dickinson, S.J., Pentland, A.P., and Rosenfeld, A. 1992. From volumes to views: An approach to 3-D object recognition. *CVGIP: Image Understanding*, **55**(2), 130–154.

Dolan, RJ, Fink, GR, Rolls, E., Booth, M., Holmes, A., Frackowiak, RSJ, and Friston, KJ. 1997. How the brain learns to see objects and faces in an impoverished context. *Nature*, **389**(6651), 596–598.

Dollár, P., Wojek, C., Schiele, B., and Perona, P. 2009. Pedestrian Detection: A Benchmark. *CVPR*.

Donoho, D.L. 2006. Compressed sensing. *Information Theory, IEEE Transactions on*, **52**(4), 1289–1306.

Edelman, S. 1999. *Representation and recognition in vision*. The MIT Press.

Edelman, S., and Duvdevani-Bar, S. 1997. A model of visual recognition and categorization. *Philosophical Transactions of the Royal Society B: Biological Sciences*, **352**(1358), 1191.

Eggert, J., and Korner, E. 2004. Sparse coding and NMF. *IEEE International Joint Conference on Neural Networks*, 2529–2533.

Einhauser, W., Kayser, C., Konig, P., and Kording, K.P. 2002. Learning the invariance properties of complex cells from their responses to natural stimuli. *European Journal of Neuroscience*, **15**(3), 475–486.

Einhäuser, W., Hipp, J., Eggert, J., Körner, E., and König, P. 2005. Learning viewpoint invariant object representations using a temporal coherence principle. *Biological Cybernetics*, **93**(1), 79–90.

Einhäuser, W., Koch, C., and Makeig, S. 2007. The duration of the attentional blink in natural scenes depends on stimulus category. *Vision research*, **47**(5), 597–607.

Elliffe, MCM, Rolls, ET, and Stringer, SM. 2002. Invariant recognition of feature combinations in the visual system. *Biological Cybernetics*, **86**(1), 59–71.

Erickson, C.A., Jagadeesh, B., and Desimone, R. 2000. Clustering of perirhinal neurons with similar properties following visual experience in adult monkeys. *nature neuroscience*, **3**(11), 1143–1148.

Everingham, M., Sivic, J., and Zisserman, A. 2006. Hello! my name is... Buffy – automatic naming of characters in tv video. *British Machine Vision Conference (BMVC)*.

Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., and Zisserman, A. 2010. The PASCAL visual object classes (VOC) challenge. *International journal of computer vision*, **88**(2), 303–338.

Farivar, R. 2009. Dorsal-ventral integration in object recognition. *Brain Research Reviews*, **61**(2), 144–153.

Fei-Fei, L., and Perona, P. 2005. A bayesian hierarchical model for learning natural scene categories. *Computer Vision and Pattern Recognition Conference (CVPR)*.

Fei-Fei, L., Fergus, R., and Perona, P. 2004a. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. *CVPR Workshop on Generative-Model Based Vision*.

Fei-Fei, Li, Fergus, Rob, and Perona, Pietro. 2003. A Bayesian Approach to Unsupervised One-Shot Learning of Object Categories. *Computer Vision, IEEE International Conference on*, **2**, 1134.

Fei-Fei, Li, Fergus, Rob, and Perona, Pietro. 2004b. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. *Computer Vision and Pattern Recognition Workshop*, **12**, 178.

Feldman, J.A. 1982. Dynamic connections in neural networks. *Biological Cybernetics*, **46**(1), 27–39.

Felleman, D.J., and Van Essen, D.C. 1991. Distributed Hierarchical Processing in the Primate Cerebral Cortex. *Cerebral Cortex*, **1**(1), 1.

Felsen, G., and Dan, Y. 2005. A natural approach to studying vision. *Nature Neuroscience*, **8**(12), 1643–1646.

Felzenszwalb, P.F., and Huttenlocher, D.P. 2000. Efficient matching of pictorial structures. *IEEE Conference on Computer Vision and Pattern Recognition*.

Felzenszwalb, P.F., and Huttenlocher, D.P. 2005. Pictorial structures for object recognition. *International Journal of Computer Vision*, **61**(1), 55–79.

Felzenszwalb, P.F., Girshick, R.B., and McAllester, D. 2010a. Cascade object detection with deformable part models. *IEEE Conference on Computer Vision and Pattern Recognition*.

Felzenszwalb, P.F., Girshick, R.B., McAllester, D., and Ramanan, D. 2010b. Object detection with discriminatively trained part based models. *IEEE transactions on pattern analysis and machine intelligence*, **32**(9), 1627–1645.

Fergus, R., Perona, P., and Zisserman, A. 2003. Object Class Recognition by Unsupervised Scale-Invariant Learning. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, **2**, 264.

Field, D.J. 1987. Relations between the statistics of natural images and the response properties of cortical cells. *Journal of the Optical Society of America A*, **4**(12), 2379–2394.

Field, D.J. 1994. What is the goal of sensory coding? *Neural computation*, **6**(4), 559–601.

Fine, I., Wade, A.R., Brewer, A.A., May, M.G., Goodman, D.F., Boynton, G.M., Wandell, B.A., and MacLeod, D.I.A. 2003. Long-term deprivation affects visual perception and cortex. *Nature Neuroscience*, **6**(9), 915–916.

Fischler, MA, and Elschlager, RA. 1973. The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers*, **22**(1), 67–92.

Fiser, J., and Aslin, R.N. 2002. Statistical learning of new visual feature combinations by infants. *Proceedings of the National Academy of Sciences of the United States of America*, **99**(24), 15822.

Földiák, P. 1990. Forming sparse representations by local anti-Hebbian learning. *Biological cybernetics*, **64**(2), 165–170.

Földiak, P. 1991. Learning invariance from transformation sequences. *Neural Computation*, **3**(2), 194–200.

Franzius, Mathias, Wilbert, Niko, and Wiskott, Laurenz. 2008. Invariant Object Recognition with Slow Feature Analysis. *Proc. 18th Intl. Conf. on Artificial Neural Networks, ICANN'08, Prague".*

Freedman, D.J., Riesenhuber, M., Poggio, T., and Miller, E.K. 2001. Categorical representation of visual stimuli in the primate prefrontal cortex. *Science*, **291**(5502), 312.

Freedman, D.J., Riesenhuber, M., Poggio, T., and Miller, E.K. 2002. Visual categorization and the primate prefrontal cortex: neurophysiology and behavior. *Journal of Neurophysiology*, **88**(2), 929.

Freedman, D.J., Riesenhuber, M., Poggio, T., and Miller, E.K. 2003. A comparison of primate prefrontal and inferior temporal cortices during visual categorization. *Journal of Neuroscience*, **23**(12), 5235.

Freedman, D.J., Riesenhuber, M., Poggio, T., and Miller, E.K. 2006. Experience-dependent sharpening of visual shape selectivity in inferior temporal cortex. *Cerebral Cortex*, **16**(11), 1631.

Fukushima, K. 1969. Visual feature extraction by a multilayered network of analog threshold elements. *Systems Science and Cybernetics, IEEE Transactions on*, **5**(4), 322–333.

Fukushima, K. 1970. A feature extractor for curvilinear patterns: A design suggested by the mammalian visual system. *Biological Cybernetics*, **7**(4), 153–160.

Fukushima, K. 1975. Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics*, **20**(3), 121–136.

Fukushima, K. 1980. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, **36**(4), 193–202.

Fukushima, K. 1988. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural networks*, **1**(2), 119–130.

Fukushima, K. 1989. Analysis of the process of visual pattern recognition by the neocognitron. *Neural Networks*, **2**(6), 413–420.

Fukushima, K., and Miyake, S. 1982. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, **15**(6), 455–469.

Gabor, D. 1946. Theory of communication. *Electr. Eng*, **93**, 429–457.

Gallant, J.L., Braun, J., and Van Essen, D.C. 1993. Selectivity for polar, hyperbolic, and Cartesian gratings in macaque visual cortex. *Science*, **259**(5091), 100.

Gallant, J.L., Connor, C.E., Rakshit, S., Lewis, J.W., and Van Essen, D.C. 1996. Neural responses to polar, hyperbolic, and Cartesian gratings in area V4 of the macaque monkey. *Journal of Neurophysiology*, **76**(4), 2718.

Gallant, J.L., Connor, C.E., and Van Essen, D.C. 1998. Neural activity in areas V1, V2 and V4 during free viewing of natural scenes compared to controlled viewing. *NeuroReport*, **9**(1), 85–89.

Gao, W., Cao, B., Shan, S., Chen, X., Zhou, D., Zhang, X., and Zhao, D. 2007. The CAS-PEAL large-scale Chinese face database and baseline evaluations. *Systems, Man and Cybernetics*.

Gauthier, I., Tarr, M.J., Anderson, A.W., Skudlarski, P., and Gore, J.C. 1999. Activation of the middle fusiform "face area" increases with expertise in recognizing novel objects. *Nature Neuroscience*.

Gautrais, J., and Thorpe, S. 1998. Rate coding versus temporal order coding: a theoretical approach. *Biosystems*, **48**(1-3), 57–65.

Gehler, P., and Nowozin, S. 2009. On Feature Combination for Multiclass Object Classification. *ICCV*.

Geisler, WS, and Albrecht, DG. 1992. Cortical neurons: isolation of contrast gain control. *Vision Research*, **32**(8), 1409–10.

George, D. 2008. How the brain might work: A hierarchical and temporal model for learning and recognition. *PhD Thesis, Stanford University*.

George, D., and Hawkins, J. 2009. Towards a mathematical theory of cortical micro-circuits. *PLoS Comput Biol*, **5**(10), e1000532.

Ghose, G.M. 2004. Learning in mammalian sensory cortex. *Current opinion in neurobiology*, **14**(4), 513–518.

Ghose, G.M., Yang, T., and Maunsell, J.H.R. 2002. Physiological correlates of perceptual learning in monkey V1 and V2. *Journal of Neurophysiology*, **87**(4), 1867.

Giebel, H. 1971. Feature extraction and recognition of handwritten characters by homogeneous layers. *Pattern recognition in biological and technical systems. Griisser, O.-J., Klinke, R.(eds.)*, 162–169.

Goodale, M.A., and Milner, A.D. 1992. Separate visual pathways for perception and action. *Trends in neurosciences*, **15**(1), 20–25.

Grauman, K., and Darrell, T. 2006. Pyramid Match Kernels: Discriminative Classification with Sets of Image Features (version 2).

Gregor, K., and Lecun, Y. 2010. Learning Fast Approximations of Sparse Coding. *International Conference on Machine learning (ICML)*.

Gregory, R.L., and Wallace, J.G. 1963. *Recovery from early blindness: A case study*. [Pergamon Press], [Cambridge].

Griffin, G., Holub, A., and Perona, P. 2007. Caltech-256 object category dataset.

Gross, C.G. 1994. How inferior temporal cortex became a visual area. *Cereb Cortex*, **5**, 455–469.

Gross, C.G. 2002. Genealogy of the" Grandmother Cell". *The Neuroscientist*, **8**(5), 512.

Gross, C.G., Schiller, P.H., Wells, C., and Gerstein, G.L. 1967. Single-unit activity in temporal association cortex of the monkey. *Journal of Neurophysiology*, **30**(4), 833.

Gross, CG, Bender, DB, and Rocha-Miranda, CE. 1969. Visual Receptive Fields of Neurons in Inferotemporal Cortex of the Monkey. *Science*, **166**(3910), 1303.

Gross, C.G., Rocha-Miranda, CE, Bender, DB, *et al.*. 1972. Visual properties of neurons in inferotemporal cortex of the macaque. *Journal of Neurophysiology*, **35**(1), 96–111.

Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. 2007. *The CMU multi-pose, illumination, and expression (Multi-PIE) face database*. Tech. rept. Tech. rep., Robotics Institute, Carnegie Mellon University, 2007. TR-07-08.

Gross, R., Matthews, I., Cohn, J., Kanade, T., and Baker, S. 2010. Multi-PIE. *Image and Vision Computing*.

Guillaumin, M., Verbeek, J., and Schmid, C. 2009. Is that you? Metric learning approaches for face identification. *ICCV*.

Hadsell, Raia, Sermanet, Pierre, Scoffier, Marco, Erkan, Ayse, Kavackuoglu, Koray, Muller, Urs, and LeCun, Yann. 2009. Learning Long-Range Vision for Autonomous Off-Road Driving. *Journal of Field Robotics*, **26**(2), 120–144.

Hand, D.J. 2006. Classifier technology and the illusion of progress. *Statistical Science*, **21**(1), 1–14.

Hansen, N., and Ostermeier, A. 2001. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, **9**(2), 159–195.

Hapgood, F. 2006. *Reverse engineering the brain*.

Harel, A., Ullman, S., Epshtein, B., and Bentin, S. 2007. Mutual information of image fragments predicts categorization in humans: Electrophysiological and behavioral evidence. *Vision research*, **47**(15), 2010–2020.

Hasler, S., Wersing, H., and Körner, E. 2005. Class-specific sparse coding for learning of object representations. *Artificial Neural Networks: Biological Inspirations*, 475–480.

Haxby, J.V., Grady, C.L., Horwitz, B., Ungerleider, L.G., Mishkin, M., Carson, R.E., Herscovitch, P., Schapiro, M.B., and Rapoport, S.I. 1991. Dissociation of object and spatial visual processing pathways in human extrastriate cortex. *Proceedings of the National Academy of Sciences of the United States of America*, **88**(5), 1621.

Haykin, S. 1994. *Neural networks: a comprehensive foundation*. Prentice Hall.

He, X., Yan, S., Hu, Y., Niyogi, P., and Zhang, H.J. 2005. Face recognition using laplacianfaces. *PAMI*.

Hebb, D.O. 1949. *The organization of behavior: A neuropsychological approach*. Lawrence Erlbaum.

Heisele, B., Serre, T., Pontil, M., and Poggio, T. 2001. Component-based face detection. *Computer Vision and Pattern Recognition, Proceedings of the IEEE Computer Society Conference on*.

Hinkle, D.A., and Connor, C.E. 2002. Three-dimensional orientation tuning in macaque area V4. *nature neuroscience*, **5**(7), 665–670.

Hinton, G.E. 1989. Connectionist learning procedures. *Artificial intelligence*, **40**(1-3), 185–234.

Hinton, G.E. 2005. What kind of a graphical model is the brain? *Proc. of the Intl. Joint Conf. on Artificial Intelligence (IJCAI-05)*, 1765–1775.

Hinton, G.E. 2007a. Learning multiple layers of representation. *Trends in cognitive sciences*, **11**(10), 428–434.

Hinton, G.E. 2007b. To recognize shapes, first learn to generate images. *Progress in brain research*, **165**, 535–547.

Hinton, G.E. 2010. Learning to represent visual input. *Philosophical Transactions of the Royal Society B: Biological Sciences*, **365**(1537), 177.

Hinton, G.E., and Salakhutdinov, R.R. 2006. Reducing the dimensionality of data with neural networks. *Science*, **313**(5786), 504.

Hinton, G.E., Osindero, S., and Teh, Y.W. 2006. A fast learning algorithm for deep belief nets. *Neural computation*, **18**(7), 1527–1554.

Hochstein, S., and Ahissar, M. 2002. View from the Top:: Hierarchies and Reverse Hierarchies in the Visual System. *Neuron*, **36**(5), 791–804.

Hoffman, KL, and Logothetis, NK. 2009. Cortical mechanisms of sensory learning and object recognition. *Philosophical Transactions of the Royal Society B: Biological Sciences*, **364**(1515), 321.

Holmes, E.J., and Gross, C.G. 1984. Effects of inferior temporal lesions on discrimination of stimuli differing in orientation. *Journal of Neuroscience*, **4**(12), 3063.

Holub, A., and Perona, P. 2005. A discriminative framework for modelling object classes. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, **1**, 664–671.

Horng, S.H., and Sur, M. 2006. Visual activity and cortical rewiring: activity-dependent plasticity of cortical networks. *Progress in Brain Research*, **157**, 3–11.

Hoyer, P.O., and Hyvärinen, A. 2002. A multi-layer sparse coding network learns contour coding from natural images* 1. *Vision Research*, **42**(12), 1593–1605.

Hua, G., and Akbarzadeh, A. 2009. A robust elastic and partial matching metric for face recognition. *ICCV*.

Hua, G., Viola, P.A., and Drucker, S.M. 2007. Face recognition using discriminatively trained orthogonal rank one tensor projections. *CVPR*.

Huang, G.B., Ramesh, M., Berg, T., and Learned-Miller, E. 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *University of Massachusetts, Amherst, Technical Report 07*.

Huang, G.B., Jones, M.J., and Learned-Miller, E. 2008. LFW Results Using a Combined Nowak Plus MERL Recognizer. *European Conference on Computer Vision (ECCV)*.

Hubel, D.H., and Wiesel, T.N. 1959. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, **148**(3), 574.

Hubel, D.H., and Wiesel, T.N. 1962. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol*, **160**, 106–54.

Hubel, D.H., and Wiesel, T.N. 1965. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *Journal of Neurophysiology*, **28**(2), 229.

Hubel, D.H., and Wiesel, T.N. 1968. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, **195**(1), 215–243.

Hubel, D.H., and Wiesel, T.N. 1977. Ferrier lecture: Functional architecture of macaque monkey visual cortex. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, **198**(1130), 1.

Hubel, D.H., and Wiesel, T.N. 1998. Early exploration of the visual cortex. *Neuron*, **20**(3), 401.

Hummel, J.E., and Biederman, I. 1992. Dynamic Binding in a Neural Network for Shape Recognition. *Psychological Review*, **99**(3,480-517).

Hung, C.P., Kreiman, G., Poggio, T., and DiCarlo, J.J. 2005. Fast readout of object identity from macaque inferior temporal cortex. *Science*, **310**(5749), 863.

Hurri, J., and Hyvärinen, A. 2003a. Simple-cell-like receptive fields maximize temporal coherence in natural video. *Neural Computation*, **15**(3), 663–691.

Hurri, J., and Hyvärinen, A. 2003b. Temporal and spatiotemporal coherence in simple-cell responses: a generative model of natural image sequences. *Network: Computation in Neural Systems*, **14**(3), 527–551.

Hyvärinen, A. 2010. Statistical models of natural images and cortical visual representation. *Topics in Cognitive Science*, **2**(2), 251–264.

Hyvärinen, A., and Hoyer, P.O. 2001. A two-layer sparse coding model learns simple and complex cell receptive fields and topography from natural images. *Vision Research*, **41**(18), 2413–2423.

Hyvärinen, A., and Oja, E. 2000. Independent component analysis: algorithms and applications. *Neural networks*, **13**(4-5), 411–430.

Hyvärinen, Aapo, Hoyer, Patrik O., and Inki, Mika O. 2001. Topographic Independent Component Analysis. *Neural Computation*, **13**(7), 1527–1558.

Igel, C., Hansen, N., and Roth, S. 2007. Covariance matrix adaptation for multi-objective optimization. *Evolutionary Computation*, **15**(1), 1–28.

Ito, M., and Komatsu, H. 2004. Representation of angles embedded within contour stimuli in area V2 of macaque monkeys. *Journal of Neuroscience*, **24**(13), 3313.

Ito, M., Tamura, H., Fujita, I., and Tanaka, K. 1995. Size and position invariance of neuronal responses in monkey inferotemporal cortex. *Journal of Neurophysiology*, **73**(1), 218.

Jagadeesh, B., Chelazzi, L., Mishkin, M., and Desimone, R. 2001. Learning increases stimulus salience in anterior inferior temporal cortex of the macaque. *Journal of Neurophysiology*, **86**(1), 290.

Jarrett, Kevin, Kavukcuoglu, Koray, Ranzato, Marc'Aurelio, and LeCun, Yann. 2009. What is the Best Multi-Stage Architecture for Object Recognition? IEEE.

Jhuang, H., Serre, T., Wolf, L., and Poggio, T. 2007. A biologically inspired system for action recognition. *International Conference on Computer Vision*.

Jiang, X., Bradley, E., Rini, R.A., Zeffiro, T., VanMeter, J., and Riesenhuber, M. 2007. Categorization training results in shape-and category-selective human neural plasticity. *Neuron*, **53**(6), 891–903.

Johnson, M.H. 2005. Subcortical face processing. *Nature Reviews Neuroscience*, **6**(10), 766–774.

Johnson, M.H., Dziurawiec, S., Ellis, H., and Morton, J. 1991. Newborns' preferential tracking of face-like stimuli and its subsequent decline. *Cognition*, **40**(1-2), 1–19.

Johnson, S.P., and Aslin, R.N. 1996. Perception of object unity in young infants: The roles of motion, depth, and orientation* 1. *Cognitive Development*, **11**(2), 161–180.

Jones, J.P., and Palmer, L.A. 1987. An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophysiology*, **58**(6), 1233.

Karni, A., and Sagi, D. 1991. Where practice makes perfect in texture discrimination: evidence for primary visual cortex plasticity. *Proceedings of the National Academy of Sciences of the United States of America*, **88**(11), 4966.

Kavukcuoglu, Koray, Ranzato, Marc Aurelio, Fergus, Rob, and LeCun, Yann. 2009. Learning Invariant Features through Topographic Filter Maps. *In: Proc. International Conference on Computer Vision and Pattern Recognition (CVPR'09)*. IEEE.

Kawasaki, K., and Sheinberg, D.L. 2008. Learning to recognize visual objects with microstimulation in inferior temporal cortex. *Journal of neurophysiology*, **100**(1), 197.

Kayaert, G., Biederman, I., and Vogels, R. 2003. Shape tuning in macaque inferior temporal cortex. *Journal of Neuroscience*, **23**(7), 3016.

Kayser, C., Einhäuser, W., Dümmer, O., König, P., and Körding, K. 2001. Extracting slow subspaces from natural videos leads to complex cells. *Artificial Neural Networks (ICANN)*, 1075–1080.

Kellman, P.J., Spelke, E.S., and Short, K.R. 1986. Infant perception of object unity from translatory motion in depth and vertical translation. *Child Development*, **57**(1), 72–86.

Kennedy, J., and Eberhart, R. 1995. Particle swarm optimization. *IEEE International Conference on Neural Networks*.

Kim, S., and Kweon, I.S. 2006. Biologically motivated perceptual feature: Generalized robust invariant feature. *ACCV*.

Kirstein, S., Wersing, H., and Körner, E. 2008. A biologically motivated visual memory architecture for online learning of objects. *Neural Networks*, **21**(1), 65–77.

Kirstein, S., Denecke, A., Hasler, S., Wersing, H., Gross, H.M., and Körner, E. 2009. A vision architecture for unconstrained and incremental learning of multiple categories. *Memetic Computing*, **1**(4), 291–304.

Klöckner, A., Pinto, N., Lee, Y., Catanzaro, B., Ivanov, P., and Fasih, A. 2009. PyCUDA: GPU run-time code generation for high-performance computing. *arXiv.org e-Print*.

Kobatake, E., and Tanaka, K. 1994. Neuronal selectivities to complex object features in the ventral visual pathway of the macaque cerebral cortex. *Journal of Neurophysiology*, **71**(3), 856.

Kobatake, E., Wang, G., and Tanaka, K. 1998. Effects of shape-discrimination training on the selectivity of inferotemporal cells in adult monkeys. *Journal of Neurophysiology*, **80**(1), 324.

Koch, C., Poggio, T., and Torres, V. 1982. Retinal ganglion cells: a functional interpretation of dendritic morphology. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, **298**(1090), 227–263.

Kording, K.P., Kayser, C., Einhauser, W., and Konig, P. 2004. How are complex cell properties adapted to the statistics of natural stimuli? *Journal of Neurophysiology*, **91**(1), 206.

Kouh, M., and Poggio, T. 2008. A canonical neural circuit for cortical nonlinear operations. *Neural computation*, **20**(6), 1427–1451.

Kourtzi, Z. 2010. Visual learning for perceptual and categorical decisions in the human brain. *Vision research*, **50**(4), 433–440.

Kourtzi, Z., and DiCarlo, J.J. 2006. Learning and neural plasticity in visual object recognition. *Current opinion in neurobiology*, **16**(2), 152–158.

Kumar, N., Berg, A. C., Belhumeur, P. N., and Nayar, S. K. 2009. Attribute and Simile Classifiers for Face Verification. *ICCV*.

Kurzak, J., Buttari, A., Luszczek, P., and Dongarra, J. 2008. The PlayStation 3 for High-Performance Scientific Computing. *Computing in Science & Engineering*, **10**(3), 84–87.

Landecker, W., Brumby, S. P., Thomure, M., Kenyon, G. T., Bettencourt, L. M., and Mitchell, M. 2010. Visualizing Classifications of Hierarchical Models of Cortex. *Computational Systems Neuroscience Conference (COSYNE)*.

Lanitis, A., Taylor, C.J., and Cootes, T.F. 1995. Automatic face identification system using flexible appearance models. *Image and Vision Computing*, **13**(5), 393–401.

Lazebnik, S., Schmid, C., and Ponce, J. 2009. Spatial Pyramid Matching. *Object Categorization: Computer and Human Vision Perspectives*.

Lazebnik, Svetlana, Schmid, Cordelia, and Ponce, Jean. 2006. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

LeCun, Y. 1985. A Learning Scheme for Asymmetric Threshold Networks. *Proceedings of Cognitiva (Paris, France)*, 599–604.

LeCun, Y. 1988. A Theoretical Framework for Back-Propagation. *Pages 21–28 of: Proceedings of the 1988 Connectionist Models Summer School*. M. Kaufmann.

LeCun, Y., and Bengio, Y. 1998. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*.

LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., and Jackel, L.D. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, **1**(4), 541–551.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. 1998. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, **86**(11), 2278–2324.

LeCun, Y., Kavukcuoglu, K., and Farabet, C. 2010. Convolutional networks and applications in vision. *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, 253–256.

LeCun, Yann, Huang, Fu-Jie, and Bottou, Leon. 2004. Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting. *In: Proceedings of CVPR'04*. IEEE Press.

LeCun, Yann, Lowe, David G., Malik, Jitendra, Mutch, Jim, Perona, Pietro, and Poggio, Tomaso. 2008. Object Recognition, Computer Vision, and the Caltech 101: A Response to Pinto et al.

Lee, D.D., and Seung, H.S. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, **401**(6755), 788–791.

Lee, H., Grosse, R., Ranganath, R., and Ng, A.Y. 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. *Proceedings of the 26th Annual International Conference on Machine Learning*, 609–616.

Lee, T.S., Yang, C.F., Romero, R.D., and Mumford, D. 2002. Neural activity in early visual cortex reflects behavioral experience and higher-order perceptual saliency. *Nature Neuroscience*, **5**(6), 589–597.

Lerner, Y., Epshtein, B., Ullman, S., and Malach, R. 2008. Class information predicts activation by object fragments in human object areas. *Journal of Cognitive Neuroscience*, **20**(7), 1189–1206.

Lesica, N.A., and Stanley, G.B. 2004. Encoding of natural scene movies by tonic and burst spikes in the lateral geniculate nucleus. *Journal of Neuroscience*, **24**(47), 10731.

Li, N., and DiCarlo, J.J. 2008. Unsupervised natural experience rapidly alters invariant object representation in visual cortex. *Science*, **321**(5895), 1502.

Li, N., and DiCarlo, J.J. 2010. Unsupervised Natural Visual Experience Rapidly Reshapes Size-Invariant Object Representation in Inferior Temporal Cortex. *Neuron*, **67**(6), 1062–1075.

Lindholm, E., Nickolls, J., Oberman, S., and Montrym, J. 2008. NVIDIA Tesla: A unified graphics and computing architecture. *IEEE Micro*, **28**(2), 39–55.

Logothetis, N.K., and Sheinberg, D.L. 1996. Visual object recognition. *Annual Review of Neuroscience*, **19**(1), 577–621.

Logothetis, N.K., Pauls, J., and Poggio, T. 1995. Shape representation in the inferior temporal cortex of monkeys. *Current Biology*, **5**(5), 552–563.

Lowe, D. G. 2000. Towards a computational model for object recognition in IT cortex. *IEEE International Workshop on Biologically Motivated Computer Vision*.

Lowe, D.G. 1999. Object recognition from local scale-invariant features. *International Conference on Computer Vision*.

Lowe, D.G. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, **60**(2), 91–110.

Luo, J., Ma, Y., Takikawa, E., Lao, S., Kawade, M., and Lu, B.L. 2007. Person-specific SIFT features for face recognition. *ICASSP*.

Mairal, J., Bach, F., Ponce, J., Sapiro, G., and Zisserman, A. 2008. Discriminative learned dictionaries for local image analysis. *International Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mairal, J., Bach, F., Ponce, J., and Sapiro, G. 2010. Online learning for matrix factorization and sparse coding. *The Journal of Machine Learning Research*, **11**, 19–60.

Malebranche, N. 1997. *The Search After Truth: With Elucidations of The Search After Truth*. Cambridge University Press.

Malsburg, C. 1995. Binding in models of perception and brain function. *Current Opinion in Neurobiology*, **5**(4), 520–526.

Marc-Aurelio Ranzato, C.P., Chopra, S., and LeCun, Y. 2006. Efficient learning of sparse representations with an energy-based model. *Advances in Neural Information Processing Systems (NIPS)*, **19**.

Marc Aurelio Ranzato, Y., Boureau, L., and LeCun, Y. 2007. Sparse feature learning for deep belief networks. *Advances in Neural Information Processing Systems (NIPS)*.

Marko, H. 1974. A biological approach to pattern recognition. *Systems, Man and Cybernetics, IEEE Transactions on*, 34–39.

Marko, H., and Giebel, H. 1970. Recognition of handwritten characters with a system of homogeneous layers. *Nachrichtentec. Z*, **9**, 455.

Marlow, Cameron, Naaman, Mor, Boyd, Danah, and Davis, Marc. 2006. HT06, Tagging Paper, Taxonomy, Flickr, Academic Article, To Read. *Conference on Hypertext and Hypermedia*.

Marr, D. 1982. Vision: A computational investigation into the human representation and processing of visual information.

Marr, D., and Nishihara, H.K. 1978. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, **200**(1140), 269–294.

Marr, D., Lal, S., and Barlow, HB. 1980. Visual Information Processing: The Structure and Creation of Visual Representations [and Discussion]. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, **290**(1038), 199–218.

Martin, D.R., Fowlkes, C.C., and Malik, J. 2004. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **26**(5), 530–549.

Martinez, A.M., and Benavente, R. 1998. The AR face database. *CVC Technical Report*.

Masland, R.H., and Martin, P.R. 2007. The unsolved mystery of vision. *Current Biology*, **17**(15), R577–R582.

Masquelier, T., and Thorpe, S.J. 2007. Unsupervised learning of visual features through spike timing dependent plasticity. *PLoS Comput Biol*, **3**(2), e31.

Masquelier, T., Serre, T., Thorpe, S., and Poggio, T. 2007. Learning complex cell invariance from natural videos: A plausibility proof. *MIT CSAIL Technical Report*.

Maurer, D., Lewis, T.L., and Mondloch, C.J. 2005. Missing sights: consequences for visual cognitive development. *Trends in cognitive sciences*, **9**(3), 144–151.

McQueen, J. 1967. Some methods for classification and analysis of multivariate observations. *Statistics and Probability*, 281–298.

Meister, M. 1996. Multineuronal codes in retinal signaling. *Proceedings of the National Academy of Sciences of the United States of America*, **93**(2), 609.

Meister, M., Lagnado, L., and Baylor, D.A. 1995. Concerted signaling by retinal ganglion cells. *Science*, **270**(5239), 1207.

Mel, B.W. 1997. SEEMORE: Combining color, shape, and texture histogramming in a neurally inspired approach to visual object recognition. *Neural computation*, **9**(4), 777–804.

Mel, B.W., and Fiser, J. 2000. Minimizing binding errors using learned conjunctive features. *Neural Computation*, **12**(4), 731–762.

Mervis, CB, and Rosch, E. 1981. Categorization of natural objects. *Annual review of psychology*.

Mikolajczyk, K., and Schmid, C. 2005. A performance evaluation of local descriptors. *PAMI*.

Miller, E.K. 2000. The prefontral cortex and cognitive control. *Nature Reviews Neuroscience*, **1**(1), 59–65.

Miller, E.K., and Cohen, J.D. 2003. An integrative theory of prefrontal cortex function.

Miller, E.K., and Desimone, R. 1994. Parallel neuronal mechanisms for short-term memory. *Science-AAAS-Weekly Paper Edition-including Guide to Scientific Information*, **263**(5146), 520–522.

Minsky, M.L., and Papert, S.A. 1987. *Perceptrons: An introduction to computational geometry*. MIT Press, Cambridge MA.

Mishkin, M., Ungerleider, L.G., and Macko, K.A. 1983. Object vision and spatial vision: Two cortical pathways. *Trends in neurosciences*, **6**, 414–417.

Mitchison, G. 1991. Removing time variation with the anti-Hebbian differential synapse. *Neural Computation*, **3**(3), 312–320.

Miyashita, Y. 1988. Neuronal correlate of visual associative long-term memory in the primate temporal cortex. *Nature*, **335**(6193), 817–820.

Miyashita, Y. 1993. Inferior temporal cortex: where visual perception meets memory. *Annual Review of Neuroscience*, **16**(1), 245–263.

Mohan, A., Papageorgiou, C., and Poggio, T. 2001. Example-Based Object Detection in Images by Components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(4), 361.

Moore Gordon, E. 1965. Cramming more components onto integrated circuits. *Electronics*, **38**(8), 114–117.

Moreels, P., and Perona, P. 2007. Evaluation of features detectors and descriptors based on 3D objects. *ICCV*.

Morton, J., and Johnson, M.H. 1991. CONSPEC and CONLERN: A two-process theory of infant face recognition. *Psychological Review*, **98**(2), 164–181.

Murase, H., and Nayar, S.K. 1995. Visual learning and recognition of 3-D objects from appearance. *IJCV*.

Mutch, J., and Lowe, D.G. 2006. Multiclass Object Recognition with Sparse, Localized Features. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mutch, J., and Lowe, D.G. 2008. Object class recognition and localization using sparse features with limited receptive fields. *International Journal of Computer Vision*, **80**(1), 45–57.

Nair, V., and Hinton, G.E. 2009. 3-d object recognition with deep belief nets. *Advances in Neural Information Processing Systems 22*.

Nov, O., Naaman, M., and Ye, C. 2008. What drives content tagging: the case of photos on Flickr. *SIGCHI Conference on Human Factors in Computing Systems*.

Nowak, E., and Jurie, F. 2007. Learning visual similarity measures for comparing never seen objects. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Olivetti Research Laboratory. 1994. ORL Face Set. *Available at* `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`.

Olshausen, B.A., and Field, D.J. 1997. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision research*, **37**(23), 3311–3325.

Olshausen, B.A., and Field, D.J. 2004. Sparse coding of sensory inputs. *Current opinion in neurobiology*, **14**(4), 481–487.

Olshausen, B.A., and Field, D.J. 2005. How close are we to understanding V1? *Neural Computation*, **17**(8), 1665–1699.

Olshausen, B.A., Anderson, C.H., and Van Essen, D.C. 1993. A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *Journal of Neuroscience*, **13**(11), 4700.

Olshausen, B.A., Anderson, C.H., and Essen, D.C. 1995. A multiscale dynamic routing circuit for forming size-and position-invariant object representations. *Journal of Computational Neuroscience*, **2**(1), 45–62.

Olshausen, B.A., *et al.*. 1996. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, **381**(6583), 607–609.

Op De Beeck, H., and Vogels, R. 2000. Spatial sensitivity of macaque inferior temporal neurons. *The Journal of Comparative Neurology*, **426**(4), 505–518.

Op de Beeck, H.P., Baker, C.I., DiCarlo, J.J., and Kanwisher, N.G. 2006. Discrimination training alters object representations in human extrastriate cortex. *Journal of Neuroscience*, **26**(50), 13025.

Oram, MW, and Perrett, DI. 1992. Time course of neural responses discriminating different views of the face and head. *Journal of Neurophysiology*, **68**(1), 70.

Oram, M.W., and Perrett, D.I. 1994. Modeling visual recognition from neurobiological constraints. *Neural Networks*, **7**(6-7), 945–972.

Osadchy, R., Miller, M., and LeCun, Y. 2004. *Synergistic Face Detection and Pose Estimation with Energy-Based Model*.

Ostrovsky, Y., Andalman, A., and Sinha, P. 2006. Vision following extended congenital blindness. *Psychological Science*, **17**(12), 1009.

Ostrovsky, Y., Meyers, E., Ganesh, S., Mathur, U., and Sinha, P. 2009. Visual parsing after recovery from blindness. *Psychological Science*, **20**(12), 1484.

Owens, John D., Luebke, David, Govindaraju, Naga, Harris, Mark, Krüger, Jens, Lefohn, Aaron E., and Purcell, Tim. 2007. A Survey of General-Purpose Computation on Graphics Hardware. *Computer Graphics Forum*, **26**(1), 80–113.

Owens, John D., Houston, Mike, Luebke, David, Green, Simon, Stone, John E., and Phillips, James C. 2008. GPU Computing. *Proceedings of the IEEE*, **96**(5), 879–899.

Papert, S. 1966. The summer vision project. *MIT AI Memo #100*.

Parker, D.B. 1986. A COMPARISON OF ALGORITHMS FOR NEURON-LIKE CELLS. *Neural Networks for Computing*, 327.

Pasupathy, A., and Connor, C.E. 1999. Responses to contour features in macaque area V4. *Journal of Neurophysiology*, **82**(5), 2490.

Pasupathy, A., and Connor, C.E. 2001. Shape representation in area V4: position-specific tuning for boundary conformation. *Journal of Neurophysiology*, **86**(5), 2505.

Pasupathy, A., and Connor, C.E. 2002. Population coding of shape in area V4. *nature neuroscience*, **5**(12), 1332–1338.

Pasupathy, A., and Miller, BK. 2005. Different time courses of learning-related activity in the prefrental cortex and basal ganglia. *Nature*, **433**, 873–876.

Pentland, A.P. 1986. Parts: Structured descriptions of shape. *Proceedings of AAAI, Philadelphia, PA*, 695–701.

Pentland, A.P. 1987. Recognition by Parts. *IEEE International Conference on Computer Vision*.

Perrett, DI, and Oram, MW. 1993. Neurophysiology of shape processing. *Image and Vision Computing*, **11**(6), 317–333.

Perrett, DI, Smith, PA, Potter, DD, Mistlin, AJ, Head, AS, Milner, AD, and Jeeves, MA. 1984. Neurones responsive to faces in the temporal cortex: studies of functional organization, sensitivity to identity and relation to perception. *Human Neurobiology*, **3**(4), 197.

Perrett, DI, Smith, PAJ, Potter, DD, Mistlin, AJ, Head, AS, Milner, AD, and Jeeves, MA. 1985. Visual Cells in the Temporal Cortex Sensitive to Face View and Gaze Direction. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, **223**(1232), 293.

Perrett, DI, Mistlin, AJ, and Chitty, AJ. 1987. Visual cells responsive to faces. *Trends in Neurosciences*, **10**, 358–364.

Petrovic, N., Ivanovic, A., and Jojic, N. 2006. Recursive estimation of generative models of video. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, **1**, 79–86.

Phillips, P.J., Wechsler, H., Huang, J., and Rauss, P.J. 1998. The FERET database and evaluation procedure for face-recognition algorithms. *Image and Vision Computing*.

Pinto, N., and Cox, D.D. 2010. An Evaluation of the Invariance Properties of a Biologically-Inspired System for Unconstrained Face Recognition. *International ICST Conference on Bio-Inspired Models of Network, Information, and Computing Systems (BIONETICS)*.

Pinto, N., and Cox, D.D. 2011a. Beyond Simple Features: A Large-Scale Neuromorphic Feature Search Approach to Unconstrained Face Recognition. *Submitted*.

Pinto, N., and Cox, D.D. 2011b. GPU Meta-Programming: A Case Study in Biologically-Inspired Computer Vision. *GPU Computing Gems Vol.2*.

Pinto, N., DiCarlo, J.J., and Cox, D.D. 2008a. Establishing good benchmarks and baselines for face recognition. *European Conference on Computer Vision (ECCV)*.

Pinto, N., Cox, D.D., and DiCarlo, J.J. 2008b. Why is real-world visual object recognition hard. *PLoS computational biology*, **4**(1), 151–156.

Pinto, N., Doukhan, D., DiCarlo, J.J., and Cox, D.D. 2009a. A High-Throughput Screening Approach To Discovering Good Forms Of Biologically Inspired Visual Representation. *PLoS Computational Biology*, **5**(11), e1000579.

Pinto, N., DiCarlo, J.J., and Cox, D.D. 2009b. How far can you get with a modern face recognition test set using only simple features? *Computer Vision and Pattern Recognition Conference (CVPR)*.

Pinto, N., Majaj, N. J., Solomon, E. A., Cox, D. D., and J., DiCarlo J. 2010. Human versus machine: comparing visual object recognition systems on a level playing field. *Computational Systems Neuroscience Conference (COSYNE)*.

Pinto, N., Barhomi, Y., Cox, D.D., and Dicarlo, J.J. 2011a. Comparing State-of-the-Art Visual Features on Invariant Object Recognition Tasks. *IEEE Workshop on Applications of Computer Vision (WACV)*.

Pinto, N., Stone, Z, Zickler, T, and Cox, D.D. 2011b. From Face Verification to Large-Scale Face Identification: A Case Study with Biologically-Inspired Visual Representations. *Submitted*.

Pitts, W., and McCulloch, W.S. 1947. How we know universals the perception of auditory and visual forms. *Bulletin of Mathematical Biology*, **9**(3), 127–147.

Plebe, A. 2007. A model of angle selectivity development in visual area V2. *Neurocomputing*, **70**(10-12), 2060–2063.

Poggio, T., and Bizzi, E. 2004. Generalization in vision and motor control. *Nature*, **431**(7010), 768–774.

Poggio, T., and Edelman, S. 1990. A network that learns to recognize three-dimensional objects. *Nature*, **343**(6255), 263–266.

Poggio, T., and Girosi, F. 1990. Networks for Approximation and Learning. *Proceedings Of The IEEE*, **78**(9).

Poggio, T., and Smale, S. 2003. The Mathematics of Learning: Dealing with Data. *Notices of the American Mathematical Society (AMS)*, **50**(5), 537–544.

Pollen, D.A., Przybyszewski, A.W., Rubin, M.A., and Foote, W. 2002. Spatial receptive field organization of macaque V4 neurons. *Cerebral Cortex*, **12**(6), 601.

Ponce, J., Berg, T., Everingham, M., Forsyth, D., Hebert, M., Lazebnik, S., Marszalek, M., Schmid, C., Russell, B., Torralba, A., *et al.*. 2006. Dataset issues in object recognition. *Toward Category-Level Object Recognition*, 29–48.

Postma, E.O., Van Den Herik, H.J., and Hudson, P.T.W. 1997. SCAN: a scalable model of attentional selection. *Neural Networks*, **10**(6), 993–1015.

Potter, M.C. 1975. Meaning in visual search. *Science*, **187**(4180), 965–966.

Potter, M.C. 1976. Short-term conceptual memory for pictures. *Journal of Experimental Psychology: Human Learning and Memory*, **2**(5), 509–522.

Potter, M.C., and Levy, E.I. 1969. Recognition memory for a rapid sequence of pictures. *Journal of Experimental Psychology*, **81**(1), 10–15.

Potter, M.C., Staub, A., Rado, J., and O'Connor, D.H. 2002. Recognition memory for briefly presented pictures: The time course of rapid forgetting. *Journal of Experimental Psychology: Human Perception and Performance*, **28**(5), 1163–1175.

Quiroga, R.Q., Reddy, L., Kreiman, G., Koch, C., and Fried, I. 2005. Invariant visual representation by single neurons in the human brain. *Nature*, **435**(7045), 1102–1107.

Raina, R., Madhavan, A., and Ng, A.Y. 2009. Large-scale deep unsupervised learning using graphics processors. *Proceedings of the 26th Annual International Conference on Machine Learning*, 873–880.

Rainer, G., and Miller, E.K. 2000. Effects of visual experience on the representation of objects in the prefrontal cortex. *Neuron*, **27**(1), 179–189.

Rainer, G., Lee, H., and Logothetis, N.K. 2004. The effect of learning on the function of monkey extrastriate visual cortex. *PLoS Biology*, **2**(2), E44.

Rakotomamonjy, A., Bach, F., Canu, S., and Grandvalet, Y. 2007. More efficiency in multiple kernel learning. *Proceedings of the International Conference on Machine learning (ICML)*.

Ramanan, D., Baker, S., and Kakade, S. 2007. Leveraging archival video for building face datasets. *International Conference on Computer Vision (ICCV)*.

Reber, PJ, Stark, CEL, and Squire, LR. 1998. Cortical areas supporting category learning identified using functional MRI. *Proceedings of the National Academy of Sciences of the United States of America*, **95**(2), 747.

Reber, P.J., Gitelman, D.R., Parrish, T.B., and Mesulam, M.M. 2003. Dissociating explicit and implicit category knowledge with fMRI. *Journal of Cognitive Neuroscience*, **15**(4), 574–583.

Reinagel, P. 2001. How do visual neurons respond in the real world? *Current opinion in Neurobiology*, **11**(4), 437–442.

Reinagel, P., Godwin, D., Sherman, S.M., and Koch, C. 1999. Encoding of visual information by LGN bursts. *Journal of Neurophysiology*, **81**(5), 2558.

Riesenhuber, M., and Poggio, T. 1999a. Are Cortical Models Really Bound Review by the Binding Problem? *Neuron*, **24**, 87–93.

Riesenhuber, M., and Poggio, T. 1999b. Hierarchical models of object recognition in cortex. *nature neuroscience*, **2**(11), 1019.

Riesenhuber, M., and Poggio, T. 2000. Models of object recognition. *nature neuroscience*, **3**, 1199–1204.

Riesenhuber, M., and Poggio, T. 2002a. How visual cortex recognizes objects: The tale of the standard model. *The Visual Neurosciences*, **2**, 1640–1653.

Riesenhuber, M., and Poggio, T. 2002b. Neural mechanisms of object recognition. *Current Opinion in Neurobiology*, **12**(2), 162–168.

Rifkin, R., and Klautau, A. 2004. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, **5**, 101–141.

Rockland, K.S., and Van Hoesen, G.W. 1994. Direct temporal-occipital feedback connections to striate cortex (V1) in the macaque monkey. *Cerebral Cortex*, **4**(3), 300.

Rolls, E., and Deco, G. 2002. Computational neurosciences of vision.

Rolls, E.T. 1984. Neurons in the cortex of the temporal lobe and in the amygdala of the monkey with responses selective for faces. *Human neurobiology*, **3**(4), 209.

Rolls, E.T. 1991. Neural organization of higher visual functions. *Current Opinion in Neurobiology*, **1**(2), 274–278.

Rolls, E.T. 1995. Learning mechanisms in the temporal lobe visual cortex. *Behavioural Brain Research*, **66**(1-2), 177–185.

Rolls, E.T. 2000. Functions of the primate temporal review lobe cortical visual areas in invariant visual object and face recognition. *Neuron*, **27**, 205–218.

Rolls, E.T., and Milward, T. 2000. A model of invariant object recognition in the visual system: learning rules, activation functions, lateral inhibition, and information-based performance measures. *Neural Computation*, **12**(11), 2547–2572.

Rolls, ET, Perrett, DI, Caan, AW, and Wilson, FAW. 1982. Neuronal responses related to visual recognition. *Brain*, **105**(4), 611.

Rolls, E.T., Cowey, A., and Bruce, V. 1992. Neurophysiological mechanisms underlying face processing within and beyond the temporal cortical visual areas. *Philosophical Transactions: Biological Sciences*, 11–21.

Rosenblatt, F. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, **65**(6), 386–408.

Rosenblatt, F. 1961. *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*. Spartan Books, Washington DC.

Rossion, B. 2002. Is sex categorisation from faces really parallel to face recognition? *Visual Cognition*, **9**, 1003–1020.

Rumelhart, D.E., Hintont, G.E., and Williams, R.J. 1986. Learning representations by back-propagating errors. *NATURE*, **323**, 9.

Russell, B.C., Torralba, A., Murphy, K.P., and Freeman, W.T. 2008. LabelMe: a database and web-based tool for image annotation. *International Journal of Computer Vision*, **77**(1), 157–173.

Rust, N., and DiCarlo, J.J. 2008. Balanced increases in selectivity and invariance produce constant sparseness. *Computational Systems Neuroscience Conference (COSYNE)*.

Rust, N.C., and DiCarlo, J.J. 2010. Selectivity and Tolerance ("Invariance") Both Increase as Visual Information Propagates from Cortical Area V4 to IT. *Journal of Neuroscience*, **30**(39), 12978.

Rust, N.C., and Movshon, J.A. 2005. In praise of artifice. *Nature Neuroscience*, **8**(12), 1647–1650.

Rutenbar, R.A. 1989. Simulated annealing algorithms: An overview. *IEEE Circuits and Devices Magazine*, **5**(1), 19–26.

Sakai, K., and Miyashita, Y. 1991. Neural organization for the long-term memory of paired associates.

Salin, P.A., and Bullier, J. 1995. Corticocortical connections in the visual system: structure and function. *Physiological reviews*, **75**(1), 107.

Salinas, E., and Abbott, LF. 1997. Invariant visual responses from attentional gain fields. *Journal of Neurophysiology*, **77**(6), 3267.

Sary, G., Vogels, R., and Orban, GA. 1993. Cue-invariant shape selectivity of macaque inferior temporal neurons. *Science*, **260**(5110), 995.

Savin, Cristina, Joshi, Prashant, and Triesch, Jochen. 2010. Independent Component Analysis in Spiking Neurons. *PLoS Computational Biology*, **6**(4).

Schneider, G., Wersing, H., Sendhoff, B., and Korner, E. 2005. Evolutionary optimization of a hierarchical object recognition model. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, **35**(3), 426–437.

Scholkopf, B., and Smola, A.J. 2002. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT Press.

Schoups, A., Vogels, R., Qian, N., and Orban, G. 2001. Practising orientation identification improves orientation coding in V1 neurons. *Nature*, **412**(6846), 549–553.

Schuett, S., Bonhoeffer, T., and H
"ubener, M. 2001. Pairing-induced changes of orientation maps in cat visual cortex. *Neuron*, **32**(2), 325–337.

Schwartz, M.A. 2008. The importance of stupidity in scientific research. *Journal of Cell Science*, **121**(11), 1771–1772.

Selfridge, OG. 1966. Pandemonium: A Paradigm for Learning. *Pattern Recognition: Theory, Experiment, Computer Simulations, and Dynamic Models of Form Perception and Discovery*, 339.

Serre, T., Riesenhuber, M., Louie, J., and Poggio, T. 2002. On the role of object-specific features for real world object recognition in biological vision.

Serre, T., Kouh, M., Cadieu, C., Knoblich, U., Kreiman, G., and Poggio, T. 2005a. A theory of object recognition: computations and circuits in the feedforward path of the ventral stream in primate visual cortex. *MIT CSAIL Technical Report*.

Serre, T., Wolf, L., and Poggio, T. 2005b. Object recognition with features inspired by visual cortex. *Computer Vision and Pattern Recognition*, **2**.

Serre, T., Oliva, A., and Poggio, T. 2007a. A feedforward architecture accounts for rapid categorization. *Proceedings of the National Academy of Sciences*, **104**(15), 6424.

Serre, T., Kreiman, G., Kouh, M., Cadieu, C., Knoblich, U., and Poggio, T. 2007b. A quantitative theory of immediate visual recognition. *Progress in Brain Research*, **165**, 33–56.

Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., and Poggio, T. 2007c. Robust object recognition with cortex-like mechanisms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **29**(3), 411–426.

Shamir, L. 2008. Evaluation of face datasets as tools for assessing the performance of face recognition methods. *International journal of computer vision*, **79**(3), 225–230.

Shaw, J. 2005. Predictive Coding with Temporal Invariance. *University of Rochester Technical Report*.

Sheinberg, D.L., and Logothetis, N.K. 2001. Noticing familiar objects in real world scenes: the role of temporal cortical neurons in natural vision. *Journal of Neuroscience*, **21**(4), 1340.

Shepard, R.N. 1967. Recognition memory ftor words, sentences, and pictures1. *Journal of Verbal Learning and Verbal Behavior*, **6**(1), 156–163.

Sherman, S.M. 2001. Tonic and burst firing: dual modes of thalamocortical relay. *TRENDS in Neurosciences*, **24**(2), 122–126.

Sherman, SM, and Koch, C. 1990. The synaptic organization of the brain. *Thalamus. Oxford University Press, Oxford*, 246–278.

Shipp, S., and Zeki, S. 1989. The organization of connections between areas V5 and V2 in macaque monkey visual cortex. *European Journal of Neuroscience*, **1**(4), 333–354.

Sigala, N., and Logothetis, N.K. 2002. Visual categorization shapes feature selectivity in the primate temporal cortex. *Nature*, **415**(6869), 318–320.

Sigman, M., Pan, H., Yang, Y., Stern, E., Silbersweig, D., and Gilbert, C.D. 2005. Top-down reorganization of activity in the visual pathway after learning a shape identification task. *Neuron*, **46**(5), 823–835.

Sim, T., Baker, S., and Bsat, M. 2003. The CMU pose, illumination, and expression database. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*.

Simoncelli, E.P., and Olshausen, B.A. 2001. Natural Image Statistics and Neural Representation. *Annual Review of Neuroscience*, **24**(1), 1193–216.

Singer, W., Tretter, F., and Yinon, U. 1982. Evidence for long-term functional plasticity in the visual cortex of adult cats. *The Journal of Physiology*, **324**(1), 239.

Sinha, P., Balas, B., Ostrovsky, Y., and Russell, R. 2007. Face recognition by humans: Nineteen results all computer vision researchers should know about. *Proceedings of the IEEE*, **94**(11), 1948–1962.

Sivic, J., Russell, B. C., Efros, A. A., Zisserman, A., and Freeman, W. T. 2005. Discovering Object Categories in Image Collections. *International Conference on Computer Vision*.

Smale, S., Rosasco, L., Bouvrie, J., Caponnetto, A., and Poggio, T. 2009. Mathematics of the neural response. *Foundations of Computational Mathematics*, **10**(1), u–91.

Sonnenburg, S., Ratsch, G., and Schafer, C. 2005. Learning Interpretable SVMs for Biological Sequence Classification. *Proceedings of the Regulatory Genomics and Systems Biology 2008 Conference (RECOMB)*.

Sonnenburg, S., Rätsch, G., Schäfer, C., and Schölkopf, B. 2006. Large Scale Multiple Kernel Learning. *Journal of Machine Learning Research*, **7**, 1531–1565.

Spratling, M.W. 2005. Learning viewpoint invariant perceptual representations from cluttered images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **27**(5), 753–761.

Sprekeler, H., Michaelis, C., and Wiskott, L. 2007. Slowness: an objective for spike-timing-dependent plasticity. *PLoS Comput Biol*, **3**(6), e112.

Standing, L. 1973. Learning 10000 pictures. *The Quarterly Journal of Experimental Psychology*, **25**(2), 207–222.

Standing, L., Conezio, J., and Haber, R.N. 1970. Perception and memory for pictures: Single-trial learning of 2500 visual stimuli. *Psychonomic Science*.

Stone, J.V. 1996. Learning perceptually salient visual parameters using spatiotemporal smoothness constraints. *Neural Computation*, **8**(7), 1463–1492.

Stone, J.V. 2004. Independent Component Analysis: A Tutorial Introduction.

Stone, Z., Zickler, T., and Darrell, T. 2008. Autotagging Facebook: Social Network Context Improves Photo Annotation. *IEEE Workshop on Internet Vision*.

Stone, Z., Zickler, T., and Darrell, T. 2010. Toward Large-Scale Face Recognition Using Social Network Context. *IEEE Special Edition on Internet Vision*.

Stringer, S.M., and Rolls, E.T. 2002. Invariant object recognition in the visual system with novel views of 3D objects. *Neural Computation*, **14**(11), 2585–2596.

Sugita, Y. 2008. Face perception in monkeys reared with no exposure to faces. *Proceedings of the National Academy of Sciences*, **105**(1), 394.

Sutton, R.S. 1988. Learning to predict by the methods of temporal differences. *Machine learning*, **3**(1), 9–44.

Sutton, R.S., and Barto, A.G. 1981. Toward a modern theory of adaptive networks: Expectation and prediction. *Psychological review*, **88**(2), 135–170.

Taigman, Y., Wolf, L., Hassner, T., and Tel-Aviv, I. 2009. Multiple one-shots for utilizing class label information. *BMVC*.

Tanaka, K. 1996a. Inferotemporal cortex and object vision. *Annual Review of Neuroscience*, **19**(1), 109–139.

Tanaka, K. 1996b. Inferotemporal Cortex and Object Vision. *Annual Review of Neuroscience*, **19**(1), 109–139.

Te-Won, L. 1998. *Independent component analysis theory and applications*. Kluwer Academic Publishers, Boston.

Thorpe, S., Fize, D., and Marlot, C. 1996. Speed of processing in the human visual system. *nature*, **381**(6582), 520–522.

Thorpe, S., Delorme, A., and Van Rullen, R. 2001. Spike-based strategies for rapid processing. *Neural networks*, **14**(6-7), 715–725.

Thorpe, SJ. 2002. Ultra-rapid scene categorisation with a wave of spikes. . *Biologically Motivated Computer Vision*.

Thorpe, SJ, and Fabre-Thorpe, M. 2001. Perspectives: Neuroscience-Seeking categories in the brain. *Science*, **291**(5502), 260–262.

Thorpe, S.J., and Gautrais, J. 1997. Rapid visual processing using spike asynchrony. *Advances in neural information processing systems*, 901–907.

Thorpe, S.J., and Imbert, M. 1989. Biological constraints on connectionist modelling. *Connectionism in perspective*, 63–92.

Torralba, A., Fergus, R., and Freeman, W.T. 2008. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Tovee, M.J., Rolls, E.T., and Azzopardi, P. 1994. Translation invariance in the responses to faces of single neurons in the temporal visual cortical areas of the alert macaque. *Journal of Neurophysiology*, **72**(3), 1049.

Treisman, A. 1996. The binding problem. *Current opinion in neurobiology*, **6**(2), 171–178.

Triesch, J. 2007. Synergies Between Intrinsic and Synaptic Plasticity Mechanisms. *Neural computation*, **19**(4), 885–909.

Tsunoda, K., Yamane, Y., Nishizaki, M., and Tanifuji, M. 2001. Complex objects are represented in macaque inferotemporal cortex by the combination of feature columns. *nature neuroscience*, **4**(8), 832–838.

Turner, R., and Sahani, M. 2007. A maximum-likelihood interpretation for slow feature analysis. *Neural computation*, **19**(4), 1022–1038.

Ullman, S. 1989. Aligning pictorial descriptions: An approach to object recognition* 1. *Cognition*, **32**(3), 193–254.

Ullman, S. 1996. *High-level vision: Object recognition and visual cognition*. The MIT Press.

Ullman, S. 2007. Object recognition and segmentation by a fragment-based hierarchy. *Trends in Cognitive Sciences*, **11**(2), 58–64.

Ullman, S., and Soloviev, S. 1999. Computation of pattern invariance in brain-like structures. *Neural Networks*, **12**(7-8), 1021–1036.

Ullman, S., Vidal-Naquet, M., and Sali, E. 2002. Visual features of intermediate complexity and their use in classification. *Nature Neuroscience*, **5**(7), 682–687.

Ungerleider, L.G., and Haxby, J.V. 1994. "What" and "where" in the human brain. *Current Opinion in Neurobiology*, **4**(2), 157–165.

Ungerleider, L.G., Mishkin, M., *et al.*. 1982. Two cortical visual systems. *Analysis of visual behavior*, **549**, 586.

Valvo, A. 1968. Behavior patterns and visual rehabilitation after early and long-lasting blindness. *American journal of ophthalmology*, **65**(1), 19.

Valvo, A., Clark, L.L., and Jastrzembska, Z.Z. 1971. *Sight restoration after long-term blindness: The problems and behavior patterns of visual rehabilitation.* American Foundation for the Blind, New York, NY.

Van De Sande, K.E.A., Gevers, T., and Snoek, C.G.M. 2010. Evaluating color descriptors for object and scene recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **32**(9), 1582–1596.

van Hateren, J.H., and van der Schaaf, A. 1998. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society B: Biological Sciences*, **265**(1394), 359.

Van Rullen, R., Gautrais, J., Delorme, A., and Thorpe, S. 1998. Face processing using one spike per neurone. *Biosystems*, **48**(1-3), 229–239.

Vapnik, VN, and Chervonenkis, AY. 1971. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, **16**, 264–280.

Varma, M., and Ray, D. 2007. Learning The Discriminative Power-Invariance Trade-Off. *International Conference on Computer Vision (ICCV)*.

Vasilescu, M., and Terzopoulos, D. 2002. Multilinear image analysis for facial recognition. *ICPR*.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.A. 2008. Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25th international conference on Machine learning*, 1096–1103.

Viola, P., and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. *CVPR*.

Vogel, J., and Schiele, B. 2004. A semantic typicality measure for natural scene categorization. *Pattern Recognition*, 195–203.

Vogels, R., and Orban, G.A. 1994. Does practice in orientation discrimination lead to changes in the response properties of macaque inferior temporal neurons? *European Journal of Neuroscience*, **6**(11), 1680–1690.

Vogels, R., and Orban, G.A. 1996. Coding of stimulus invariances by inferior temporal neurons. *Progress in Brain Research*, **112**, 195–211.

Von Ahn, L., Liu, R., and Blum, M. 2006. Peekaboom: a game for locating objects in images. *Proceedings of the ACM SIGCHI conference on Human Factors in computing systems*.

Von der Heydt, R., Peterhans, E., and Baumgartner, G. 1982. Illusory contours and cortical neuron responses. *Ann. Inst. Oceanogr*, **58**, 297.

von der Malsburg, C. 1981. The correlation theory of brain function. *Models of neural networks II*.

von Senden, M. 1960. *Space and sight: The perception of space and shape in the congenitally blind before and after operation*. Methuen, London.

Wallis, G. 1996. Using spatio-temporal correlations to learn invariant object recognition. *Neural Networks*, **9**(9), 1513–1519.

Wallis, G., and Bülthoff, H.H. 2001. Effects of temporal association on recognition memory. *Proceedings of the National Academy of Sciences of the United States of America*, **98**(8), 4800.

Wallis, G., and Rolls, E.T. 1997. Invariant face and object recognition in the visual system. *Progress in Neurobiology*, **51**(2), 167–194.

Wallis, G., Rolls, E., and Foldiak, P. 1993. Learning invariant responses to the natural transformations of objects. *Pages 1087–1090 of: International Joint Conference on Neural Networks*, vol. 2. IEEE.

Walther, D.B., and Koch, C. 2007. Attention in hierarchical models of object recognition. *Progress in Brain Research*, **165**, 57–78.

Wandell, B.A., Brewer, A.A., and Dougherty, R.F. 2005. Visual field map clusters in human cortex. *Philosophical Transactions B*, **360**(1456), 693.

Wang, G., Zhang, Y., and Fei-Fei, L. 2006. Using dependent regions for object categorization in a generative framework. *Proc. IEEE Conf. Comp. Vision Patt. Recog.*

Weber, C., and Triesch, J. 2008. A sparse generative model of V1 simple cells with intrinsic plasticity. *Neural computation*, **20**(5), 1261–1284.

Weber, M., Welling, M., and Perona, P. 2000. Unsupervised learning of models for recognition. *Proc. ECCV*, **1**, 18–32.

Weiskrantz, L., and Saunders, RC. 1984. Impairments of visual object transforms in monkeys. *Brain*, **107**(4), 1033.

Wersing, H., and Körner, E. 2002. Unsupervised learning of combination features for hierarchical recognition models. *Artificial Neural Networks (ICANN)*, 137–137.

Wersing, H., and Körner, E. 2003. Learning optimized features for hierarchical models of invariant object recognition. *Neural computation*, **15**(7), 1559–1588.

Wibisono, A., Bouvrie, J., Rosasco, L., and Poggio, T. 2010. Learning and Invariance in a Family of Hierarchical Kernels. *MIT CSAIL Technical Report.*

Wiesel, T.N., and Hubel, D.H. 1963. Single-cell responses in striate cortex of kittens deprived of vision in one eye. *Journal of Neurophysiology*, **26**(6), 1003.

Wiskott, L. 1998. Learning invariance manifolds. *Symposium on Neural Computation*, 196–203.

Wiskott, L. 2003. Slow feature analysis: A theoretical analysis of optimal free responses. *Neural Computation*, **15**(9), 2147–2177.

Wiskott, L., and Sejnowski, T.J. 2002. Slow feature analysis: Unsupervised learning of invariances. *Neural computation*, **14**(4), 715–770.

Wiskott, L., Fellous, J.M., Krüger, N., and von der Malsburg, C. 1997. Face Recognition by Elastic Bunch Graph Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wolf, L., Hassner, T., and Taigman, Y. 2008. Descriptor Based Methods in the Wild. *European Conference on Computer Vision (ECCV)*.

Wolf, L., Hassner, T., and Taigman, Y. 2009. Similarity Scores based on Background Samples. *ACCV*.

Wright, J., and Hua, G. 2009. Implicit elastic matching with random projections for pose-variant face recognition. *CVPR*.

Wyss, R., König, P., and Verschure, P. 2003. Invariant representations of visual patterns in a temporal population code. *Proceedings of the National Academy of Sciences of the United States of America*, **100**(1), 324.

Wyss, R., König, P., and Verschure, P. 2006. A model of the ventral visual system based on temporal stability and local memory. *PLoS Biology*, **4**(5), 836.

Yale Center for Computational Vision and Control. 1997. Yale Face Set. *Available at* `http://cvc.yale.edu`.

Yamane, Y., Carlson, E.T., Bowman, K.C., Wang, Z., and Connor, C.E. 2008. A neural code for three-dimensional object shape in macaque inferotemporal cortex. *Nature neuroscience*, **11**(11), 1352–1360.

Yang, J., Yu, K., Gong, Y., and Huang, T. 2009. Linear spatial pyramid matching using sparse coding for image classification. *International Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yang, M.H. 2002. Kernel eigenfaces vs. kernel fisherfaces: Face recognition using kernel methods. *FG*.

Yang, R. 2004. Scientific Computing on Commodity Graphics Hardware. *Lecture Notes in Computer Science*, 1100–1105.

Yang, T., and Maunsell, J.H.R. 2004. The effect of perceptual learning on neuronal responses in monkey visual area V4. *Journal of Neuroscience*, **24**(7), 1617.

Yao, H., and Dan, Y. 2001. Stimulus timing-dependent plasticity in cortical processing of orientation. *Neuron*, **32**(2), 315–323.

Yuille, A., and Kersten, D. 2006. Vision as Bayesian inference: analysis by synthesis? *Trends in Cognitive Sciences*, **10**(7), 301–308.

Yuille, Alan L. 1991. Deformable Templates for Face Recognition. *Journal of Cognitive Neuroscience*, **3**(1), 59–70.

Zeki, S., and Shipp, S. 1989. Modular connections between areas V2 and V4 of macaque monkey visual cortex. *European Journal of Neuroscience*, **1**(5), 494–506.

Zhang, H., Berg, A.C., Maire, M., and Malik, J. 2006. SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. *Proc. CVPR*.

Zhang, J., Marszalek, M., Lazebnik, S., and Schmid, C. 2007. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*.

Zhao, M., Yagnik, J., Adam, H., and Bau, D. 2008. Large scale learning and recognition of faces in web videos. *Automatic Face and Gesture Recognition Conference*.

Zhao, W., Chellappa, R., Phillips, PJ, and Rosenfeld, A. 2003. Face recognition: A literature survey. *ACM Computing Surveys*.

Zhou, L., and Kambhamettu, C. 1999. Extending superquadrics with exponent functions: Modeling and reconstruction. *Pages 73–78 of: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2.

Zhou, L., and Kambhamettu, C. 2002. Representing and recognizing complete set of geons using extended superquadrics. *Pattern Recognition*, **3**, 30713.

Zoccolan, D., Cox, D.D., and DiCarlo, J.J. 2005. Multiple object response normalization in monkey inferotemporal cortex. *Journal of Neuroscience*, **25**(36), 8150.

Zoccolan, D., Kouh, M., Poggio, T., and DiCarlo, J.J. 2007. Trade-off between object selectivity and tolerance in monkey inferotemporal cortex. *Journal of Neuroscience*, **27**(45), 12292.

Zola-Morgan, S., and Squire, LR. 1993. Neuroanatomy of memory. *Annual Review of Neuroscience*, **16**(1), 547–563.

Zou, J., Ji, Q., and Nagy, G. 2007. A Comparative Study of Local Matching Approach for Face Recognition. *IEEE Transactions on Image Processing*.