Human-Inspired Methods for Extending Advances in Computer Vision to Data- and Compute-Constrained Environments

by

Laura E. Brandt

B.A., University of California at Berkeley (2019)
B.S., University of California at Berkeley (2019)
S.M., Massachusetts Institute of Technology (2021)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by:	Laura E. Brandt Department of Electrical Engineering and Computer Science May 23, 2024
Certified by:	Nicholas Roy Professor of Aeronautics and Astronautics Thesis Supervisor
Accepted by:	Leslie A. Kolodziejski Professor of Electrical Engineering and Computer Science Chair, Department Committee on Graduate Students

Human-Inspired Methods for Extending Advances in Computer Vision to Data- and Compute-Constrained Environments

by

Laura E. Brandt

Submitted to the Department of Electrical Engineering and Computer Science
on May 23, 2024 in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

ABSTRACT

Recent developments in computer vision have often relied on access to big data, powerful compute, or both. City-based systems, such as self-driving cars and airport checkpoints, have benefited greatly from these advances, so much so that automated cars and security checks are beginning see true deployment in modern society. In contrast, robots and autonomous systems in data- and compute-constrained environments, like remote wilderness regions or off-Earth, are still relying on pre-deep learning era computer vision algorithms. **Robots in the most challenging of environments — and, correspondingly, the environments that require the highest level of autonomy for robots — have been left behind by modern computer vision.**

In this dissertation, I discuss several **human-inspired methods** that I have developed during my time at MIT, with the goal of **closing the gap between the capabilities of modern computer vision and what is currently possible on robots operating in data- and compute- constrained environments**. I explore methods for collecting good — but small — datasets, improving visual estimates by upsampling the feature embeddings used to generate them, and identifying which image samples (and sample regions) a learned model is 'uncertain' about. Finally, I share sketches of several human-inspired paradigms for using these new methods to make vision models more efficient and generalisable, which I hope can serve as a starting point for future efforts to close the gap and bring modern vision to deployment in data- and compute- constrained environments.

Thesis supervisor: Nicholas Roy
Title: Professor of Aeronautics and Astronautics

Acknowledgements

First and foremost, I want to thank my collaborators, without whom this thesis would not exist. Stephanie Fu interned with myself and Mark Hamilton from Fall 2021 — Spring 2023, and that our FeatUp work came to fruition is truly due to her. Mark himself was amazing as a collaborator and cheerleader. Many thanks are also due to my other collaborators. Zhang Zhoutong, in particular, bricked a computer in his attempt to implement an improved JBU algorithm in CUDA — a sacrifice that must never be forgotten. Thank God we had Axel Feldmann to swoop in and save a second computer from the same fate.

Sunshine Jiang, similarly, has interned with myself and Dr. Sid Ancha starting in Fall 2023, and has been an absolute legend. She has helped Sid and myself turn our DiffUnc ideas and system sketches into an implemented reality. We will be forever grateful to her for it. Her support to myself, personally, in the lead-up to my defense, has been truly above and beyond — several times when I realised I needed some plot or visualisation or other to make a point, she'd volunteer "I can do that!" and come back within hours with not only what I'd asked for, but something better. All of this while working remotely, first from Oxford University where she was studying abroad, then from China where she was visiting her family. As for Sid — he has been a wonderful collaborator, mentor, cheerleader, and listener in this final stretch of my doctorate. This man better keep in touch, and I hope he brings his dog Tiggy 'home to Seattle' to visit.

I must also thank those who started me off on this doctoral journey at MIT — Prof. Bill Freeman, with whom I completed my master's degree on human and computer vision; Dr. Ruth Rosenholtz, who mentored me professionally during my early years and taught me about human peripheral vision; and Prof. Ted Adelson, whose group meetings I attended regularly for nearly a year. It was there that I realised the relevance and importance of robotics as an application of both human and computer vision science.

Of all my friends, I would particularly like to thank Dr. Shyan Akmal, my 'thesis buddy'. It is a truly special experience to have a good friend who's on the same timeline as you from the very start; we matriculated at MIT together in 2019, and now we'll graduate together in 2024. And he's supported me all the way. I must also thank my friend from 'the olden days', Suzannah Tebon, who has — despite being a remote friend for about a decade now — always been willing to drop everything to lend an ear over virtual tea, or even to fly across the country to be with me in times of crisis. Thanks are also due to Liza Tremsina, who has been with me since Berkeley; we graduated from Cal together in 2019 with the same dual degrees, then came straight to MIT to PhD in the same department together. I'll miss her dearly as our paths split — for now.

And there are a million other people to thank for their support, no matter how insignificant

or natural it may have seemed to them; including (in seed-randomised order) Yang Zhutian and Jeff Haas and Dr. Alex Miller and Tejas Jayashankar and Martina Stadler-Kurtz and Dr. Emilie Josephs and Tim Lloyd and Ethan Fahnestock and Dr. Thomas O'Connell and Veevee Cai and Imrathien A'Carise and Sandra Liu and John Niroula and Ken Jacobson and Dr. Bob Brandt and Yasmin Veys and Bonny Thiel and Erick Fuentes and Vanessa Wan Kwok Brandt and Kajetan Haas and Dr. Jake Arkin and Christina Ji and Kuan-Wei Huang and Dr. Prafull Sharma and Mike Noseworthy and... I cannot list them all.

Penultimately, I thank the family pets for loving and sustaining me. Jo March and Ezri Dax Brandt have been the perfect doctoral ESAs, and Jo is the bestest service dog-in-training. Arcadia and Benji Brandt, similarly, pretty much single-handedly got me through my master's degree back in 2021. And my father's dog, Raychel, has been the perfect playmate for Jo in her time off.

Finally, but no less importantly, I want to thank my committee — Profs. Leslie Kaelbling and Phillip Isola — and advisor — Prof. Nick Roy — and our EECS graduate student committee chair — Prof. Leslie Kolodziejski — for their support, advice, suggestions, hand-holding, constructive criticisms, time, honesty, energy, overall wonderfulness, and refusal to let me quit. I wouldn't be here without them. Thank you.

Jo March


Ezri Dax


Arcadia (-2021) & Benji


Benji (-2023)


Raychel & Jo March

To my grandparents, Sherry (-2022) and Ed (-2024) DeWan.

About the Author

Laura Brandt was born in Champaign-Urbana to a research scientist at the Beckman Institute and an businesswoman from Hainan and Hong Kong. Having lived in three countries and every region of the United States, she considers herself a mix between a nomad and a Midwestern Californian, but currently lives in a lovely Seattle WA community with her dog, her cat, five other humans, and two other pets. She has worked on electronics for Venus probes, built telescopes, designed sensors that keep rocket engines from exploding, and conceptualised vision systems for Lunar landers. Outside of work, she enjoys board games and bookstores and drinking tea and reading and painting and spending quality time with pets and animals.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The past decade in computer vision has been an exciting one. Since the advent of AlexNet in 2012 [1], deep learning in general and computer vision in particular have experienced amazing performance leaps, as first convolutional neural networks (CNNs), then generative adversarial networks (GANs) and implicit models, and now transformers have rocked the scene [2]. We now have transformer-based models that can accomplish tasks like semantic segmentation (*e.g.*, the Segment Anything Model (SAM) [3]) and depth estimation (*e.g.*, the Dense Prediction Transformer (DPT) [4]) with mean-Intersection over Union (mIoU) values approaching 80% for the former, and pixel-accuracy nearing 85% for the latter. Qualitatively, the results that come from these models would have it appear to the human eye that these vision challenges have in fact been solved for machines (Figure 1.1).

However, the training processes for these models rely on access to a massive amount of high-resolution data and compute. SAM, for example, is based on a Vision Transformer (ViT) architecture [5] which in itself has around 300M parameters, and must be trained using "11M diverse, high-resolution... images... [labelled with] 1.1B high-quality segmentation masks" (Figure 1.2). And while it is true that many of these images are labelled automatically using a special data engine [3], humans still have to painstakingly label *at least* 300K images with 10.2M masks. And those images *must* be high-resolution in the first place [3]. Similarly, the DPT depth estimator is also ViT-based and uses around 300M parameters, and trains on 1.4M images from a composite dataset compiled for the task — MIX6, the largest monocular depth dataset to-date [4].

Unfortunately, we do not always have access to large amounts of cleanly labelled and high-resolution imagery, and the computation available on real platforms is often limited by size and power constraints. In robotics we often find that modern computer vision models either (a) cannot run on our platforms at all, or (b) cannot be trained on datasets of sufficient quality to yield results of the caliber of Figure 1.1. As a result, roboticists often rely instead on *sparse* depth estimates from LiDAR scans (Figure 1.3) or stereo vision algorithms (Figure 1.4), and semantic perception is often limited to bounding box object detection (Figure 1.5). The advances of modern computer vision have so far passed robotics by.

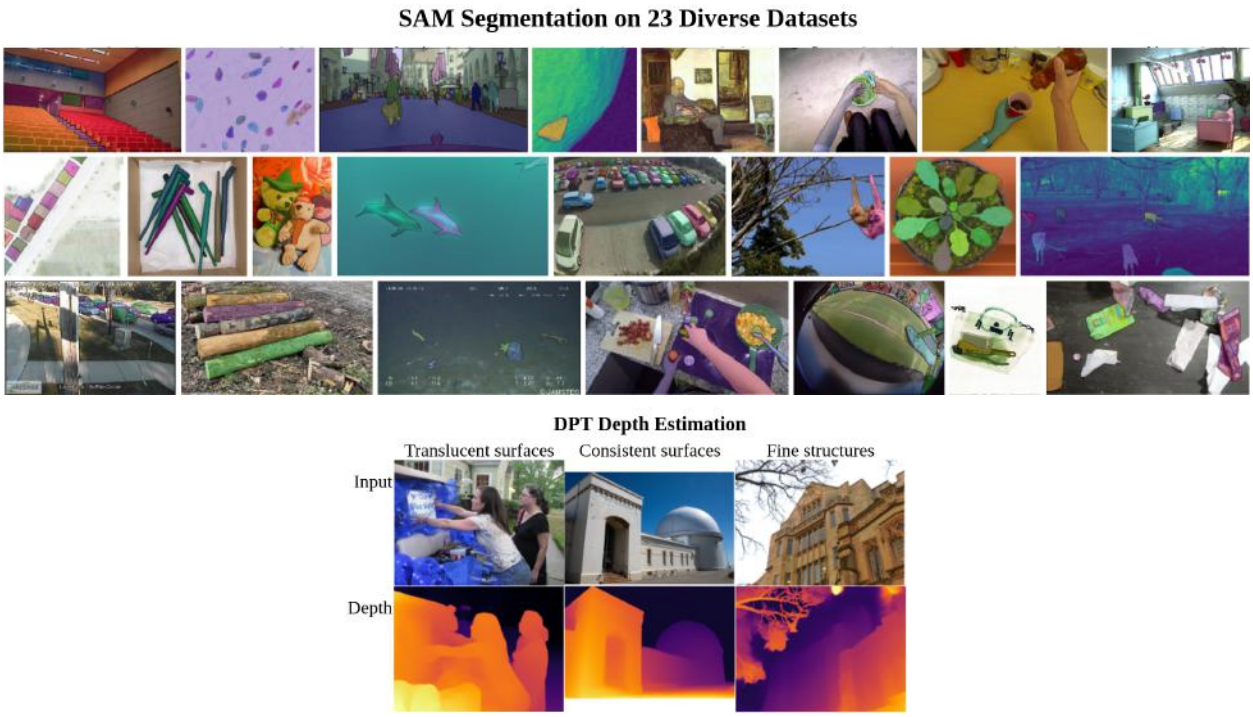

Figure 1.1: Qualitative results from large, modern, computer vision models. Top: The Segment Anything Model (SAM)'s zero-shot transfer capabilities [3], which seem able to handle just about any data on Earth one might throw at them. Bottom: The Dense Prediction Transformer (DPT) successfully estimates accurate and consistent depth, even on translucent surfaces, large surfaces, and fine structures [4].

The work I present in this dissertation is therefore, in general, some combination of:

1. **Compute-constrained algorithms**. Specifically, I have aimed to develop algorithms for computer vision that either:

   (a) use on the order of 50-75M parameters. (This range is what can currently be deployed on remote and mobile systems like those pictured in Figure 1.6, and is unlikely to grow substantially in the near future. High-end GPUs require 150-300W to power [6], compared to the max-120W GPUs being used by mobile robots now [7].)

   (b) contribute significantly to reducing the amount of compute used by existing, high-performance computer vision algorithms.

2. **Data-constrained algorithms**. The data available is sparse (*e.g.*, point clouds) and/or not bountifully available during training.

Additionally, my work often derives either its goal or its approach from knowledge and hypotheses about how human vision works or what it functionally does, because humans are our favourite example of mobile agents capable of perceiving and navigating the world in a data- and compute-constrained manner. Thus the title of this doctoral work and dissertation: Human-inspired Methods for Extending Advances in Computer vision to Data- and Compute-Constrained Environments.

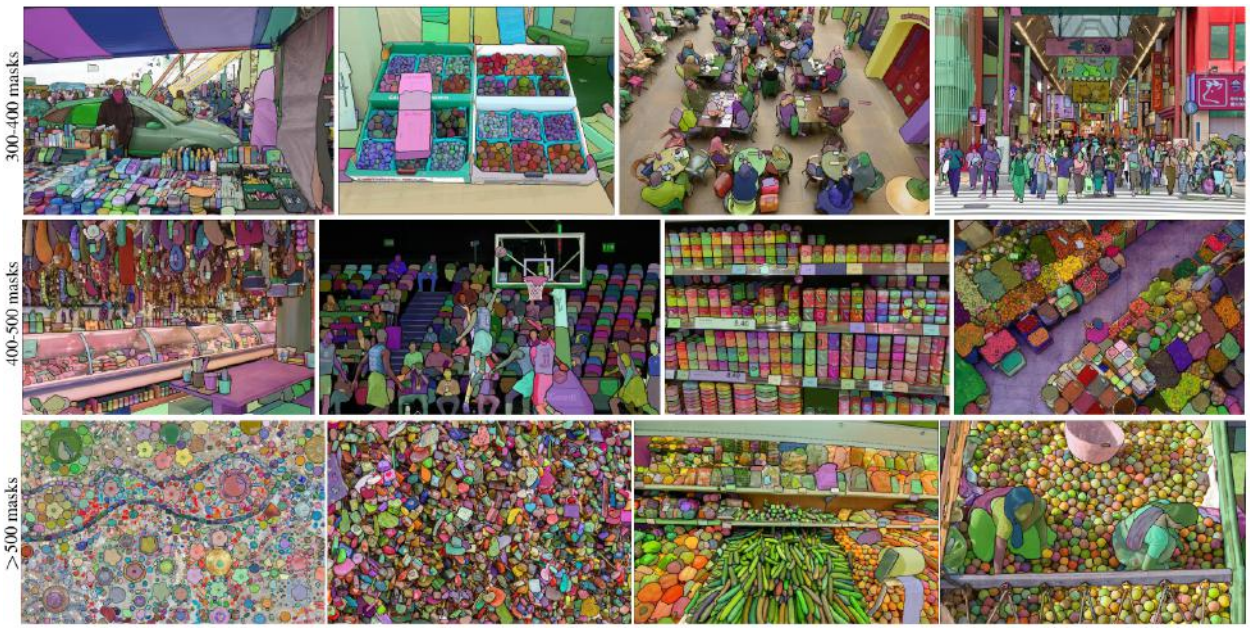

Figure 1.2: Example images with overlaid masks from the SA-1B dataset used to train SAM. SA-1B contains 11M diverse and high-resolution images, and 1.1B high-quality segmentation masks [3].

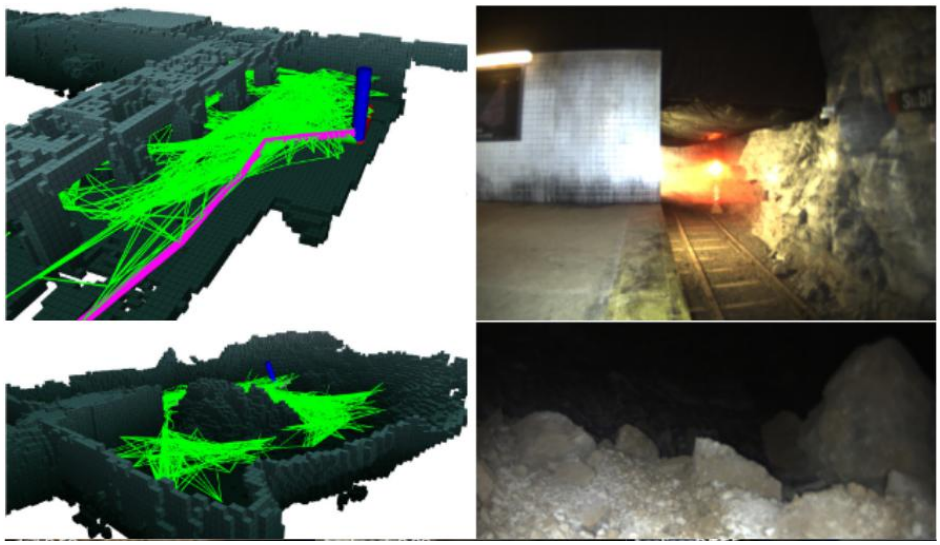

Figure 1.3: Examples of the low-resolution imagery (right) and occupancy grids (left) *actually* in use in real-world robots [7].

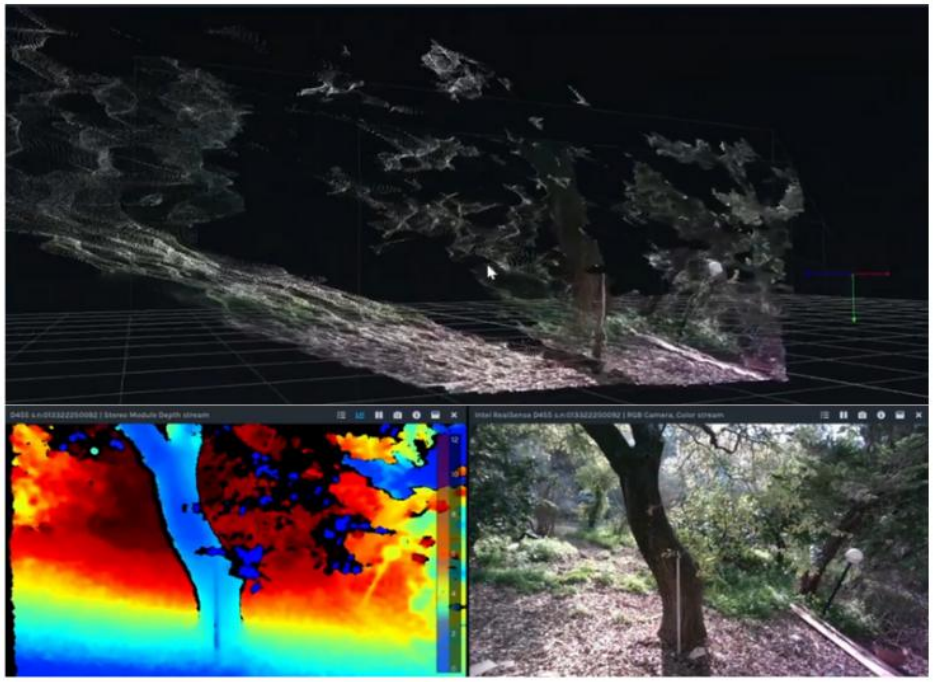

Figure 1.4: Examples of point clouds (top) as collected by the Intel RealSense stereo cameras commonly used in robotics, and the depth map computed by their stereo depth algorithm (bottom-left) [8].

Figure 1.5: Examples of object detection and labels as produced by YOLOv3 Tiny [9], an approximately 63M-parameter [10] object detection model commonly used on mobile robotic platforms for its efficiency [7].

Figure 1.6: Mobile systems like the Husky [7] and Spot [11] often have limited compute due to payload and power supply constraints.

My master's thesis in Prof. Bill Freeman's group [12] tackled the challenge of teaching machines how to perceive shape from surface contour markings. Such markings are commonly used in striped clothing, data visualisations, and other person-made constructs to give the impression (or illusion) of shape. Humans have an apparently natural ability to interpret them, but machines and computer vision algorithms do not. We approached this problem by synthesising a new dataset of surface grid- and line-marked 3D surfaces (SurfaceGrid, Figure 1.7) and training a deep neural network to estimate their shape (Figure 1.8). Our algorithm successfully reconstructed shape from synthetic 3D surfaces rendered with a variety of grid- and line-contour markings with less than 0.5% mean-squared relative error, and generalised reasonably well to 3D mesh models and real-world wireframe objects (Figure 1.9).

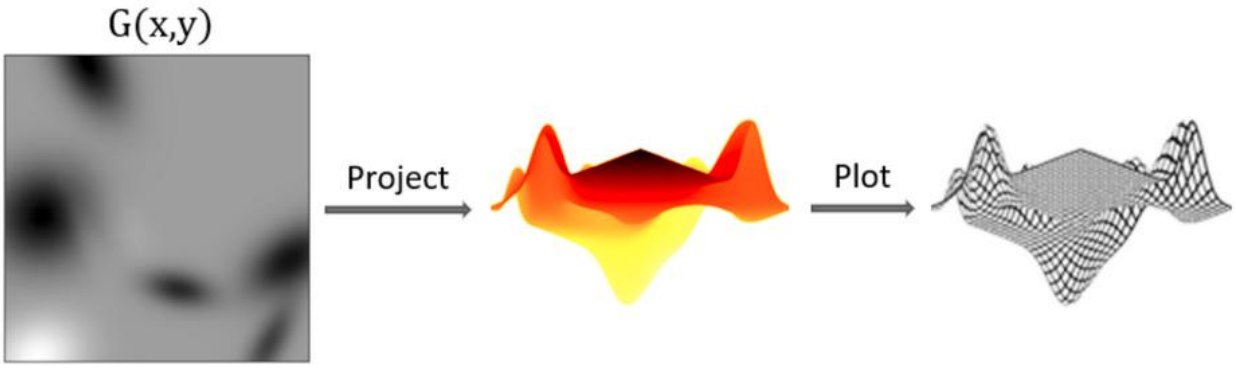

Figure 1.7: SurfaceGrid is a dataset of contour-marked surfaces generated by summing $N \leq 10$ randomly-generated 2D Gaussians to generate functions $G(x, y)$, which were then visualised, and their depth maps computed, from a variety of viewpoints [13].

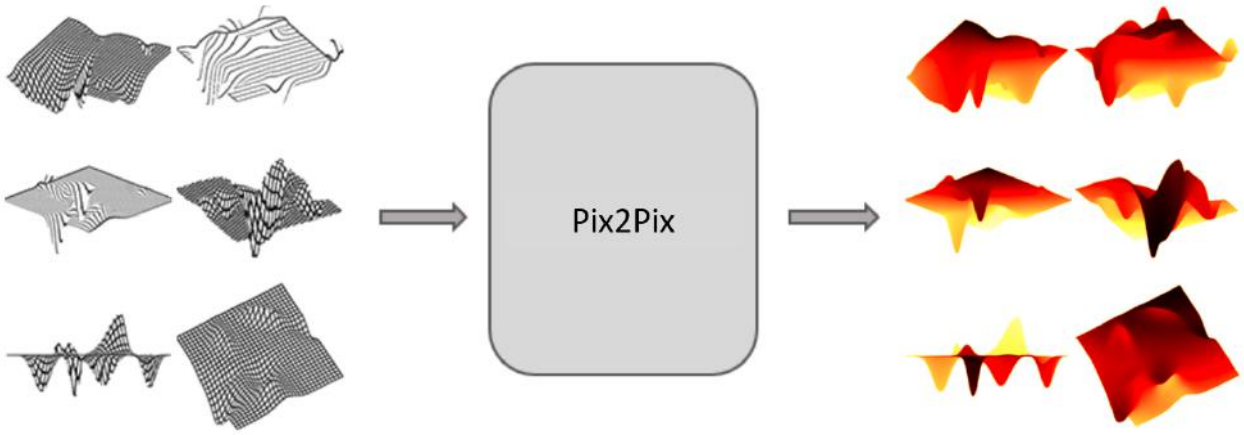

Figure 1.8: We trained a Pix2Pix generative adversarial model (GAN) [14] to take as input contour-marked surfaces from SurfaceGrid, and output estimates of their corresponding depth maps. The model required only 1.2k image pairs to train [13], far fewer than the 1.4M of [4] and 11M of [3].

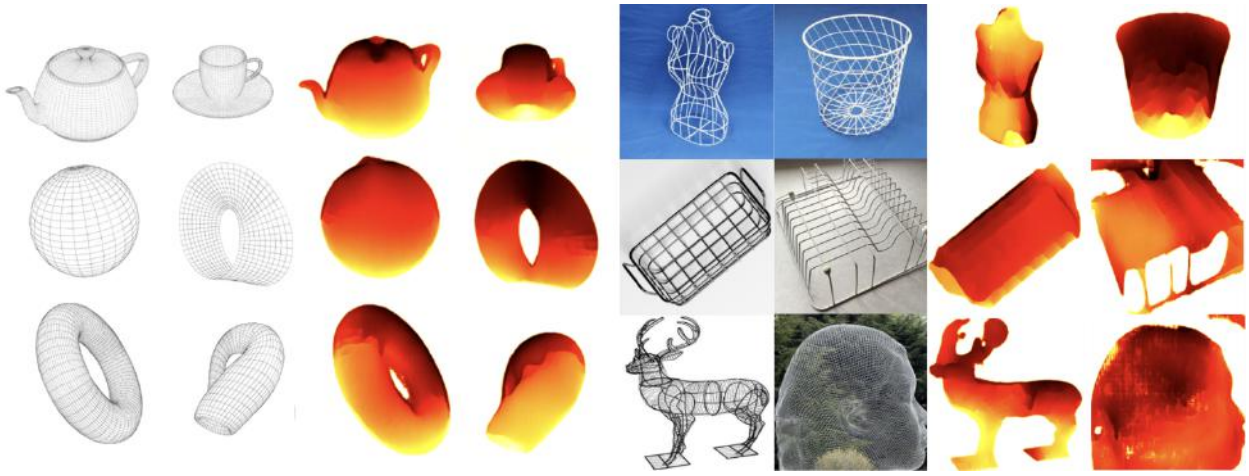

Figure 1.9: The Pix2Pix model trained in Figure 1.8 [13] generalised surprisingly well to data that was nothing like what it had seen before (Figures 1.7, 1.8), including real-world wireframe data [12]. (It is worth noting, however, that the numerical scaling of the estimates were calibrated to that of the training set, rather than to those of the various test samples, which is not surprising.)

For more detail, I refer the reader to my master's thesis [12], and note that it is not the focus of this dissertation, which is distinct. But it was the starting point! From my master's work, I learned two interesting (and, in my view, important) things: First, that even relatively small neural networks, like the approximately 55M-parameter Pix2Pix network [14][15], are surprisingly powerful when trained with the 'right data'. We believed that the good generalisation of Figure 1.9 was due to our intentional choice of mixtures of Gaussians

as a 'basis set' for approximating the space of all 3D shapes — though, similar to the results shown in Figure 1.7, we approximated with only at most 10 terms at a time, which would limit the power of the expansion [12]. Second, I learned that one way of picking the 'right data' was to construct a good 'basis set' of training examples. From these two realisations, I gained two new **questions** that have lain at the core of my research:

1. How do we choose the 'right data' in the real world? (We usually cannot just synthesise what we want, as I did for my master's research [12][13].)

2. How can we get good visual estimates out of compute-constrained systems for more complex and 'interesting' real-world data?

These were the questions that motivated my doctoral work, and I look forward to sharing this research with you in the following chapters.

I follow this introduction with an exposition of two sizable projects I undertook with my collaborators, aiming to answer these two questions. FeatUp (Chapter 2, [16]) is a fast, lightweight, and guided method for upsampling features extracted from *any* backbone architecture. It can be used to improve the performance of linear probes trained for *any* vision task that takes as input a feature embedding, by upsampling the embedding as a preprocessing step. This opens the door to the future development of multi-task computer vision *systems* consisting of several linear probes operating on a single upsampled feature embedding from a single shared backbone, thereby drastically reducing the number of entirely separate vision models that must be trained and stored if one wishes to accomplish several vision tasks.

DiffUnc (Chapter 3, [17]) is a diffusion-based method for estimating a model's uncertainty regarding a sample image, which is due to how similar that sample is to the images seen by the model during training. In other terms, we seek to estimate the portion of the 'epistemic' uncertainty due to the degree to which an image is 'out-of-distribution' (OOD) relative to the distribution of the images present in the training set. Our assumption, common in uncertainty estimation for neural networks [18], is that 'OODness' is a primary influencer of a learned model's epistemic uncertainty.

DiffUnc, *localises* uncertainty within sample images, where preceding diffusion-based models only estimated an image-level metric. In localising the uncertainty, we make the uncertainty human-interpretable, in that a human looking at an "uncertain" image can tell *where* (and make an informed hypothesis as to *why*) it is uncertain. This is important in, for example, situations where a remote supervisor is trying to understand why an autonomous robot has failed, based on a low-resolution image that the robot has beamed back through low-bandwidth communications. Improving our ability to estimate which data are out-of-distribution (and why) also allows us to select the 'right data' for training a maximally-generalisable yet minimally data-guzzling vision algorithm, by intentionally crafting the diversity of our training (or finetuning) dataset, rather than taking the currently prevalent approach of simply gathering as much data as possible.

Finally, having described these two projects in detail, and how they each tie into the two core questions of my research, I will discuss what I view to be the *next* question: How to *use* the 'right data' for improved vision model performance? Although I give several partial answers throughout this dissertation, I never answer it extensively or with dedicated focus,

and I hope with Chapter 4 to encourage others to pursue it for themselves, and to do so with a human-inspired lens.

Concisely: I discuss our peripheral vision-inspired FeatUp method for feature upsampling in Chapter 2. In Chapter 3, I share our uncertainty estimation work, DiffUnc, which was inspired by the human ability to estimate and use uncertainty to identify and learn from mistakes. Then, in Chapter 4, I sketch out several ideas for how these contributions can be used to improve vision models in data- and compute- constrained environments. I conclude in Chapter 5 by making a final case for other researchers to develop vision models for robotics through a human-inspired lens.

Welcome to my dissertation. Let's get started!

Chapter 2

FeatUp: A Model-Agnostic Framework for Features at Any Resolution

This work was published at ICLR 2024 under the same title — FeatUp: A Model-Agnostic Framework for Features at Any Resolution — *along with Stephanie Fu and Mark Hamilton (joint first authors), Axel Feldmann, Dr. Zhang Zhoutong, and Prof. Bill Freeman* [16].

How can we process images to get good visual estimates — *e.g.*, of shape, distance, or semantic content — on compute-constrained systems? I tackled this question with my master's thesis, but the world is generally far more complex and 'interesting' than those contour-marked surfaces (Figures 1.8 and 1.9). Computationally-efficient yet high-quality visual navigation should be possible; humans, after all, are simultaneously compute-constrained relative to the vast number of simultaneous computations and processes they undertake *and* effective navigators of the real world.

When it comes to human vision, the first and foremost method our bodies and brain employ to reduce our computational needs is actually a fundamental limitation of our visual sensor — the eye [19]. Most of our visual field is measured by the 'peripheral' retina, which has a drastically reduced concentration of cone-type photoreceptors compared our 'fovea', which aligns with the optic axis (that is, the direct line of gaze or fixation, Figure 2.1) [20].

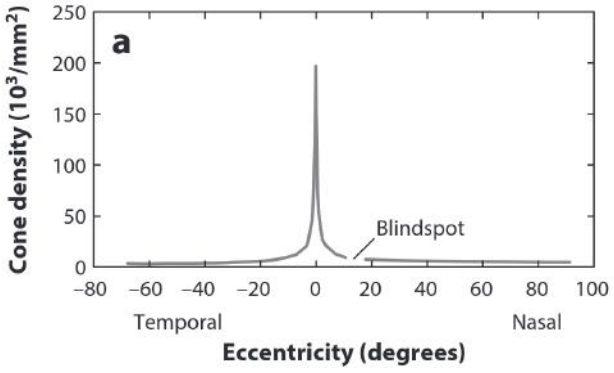

Figure 2.1: Density of colour-sensitive photoreceptors in the human retina as a function of eccentricity [20]. Most cone cells in the retina lie within ±2° of the optic axis.

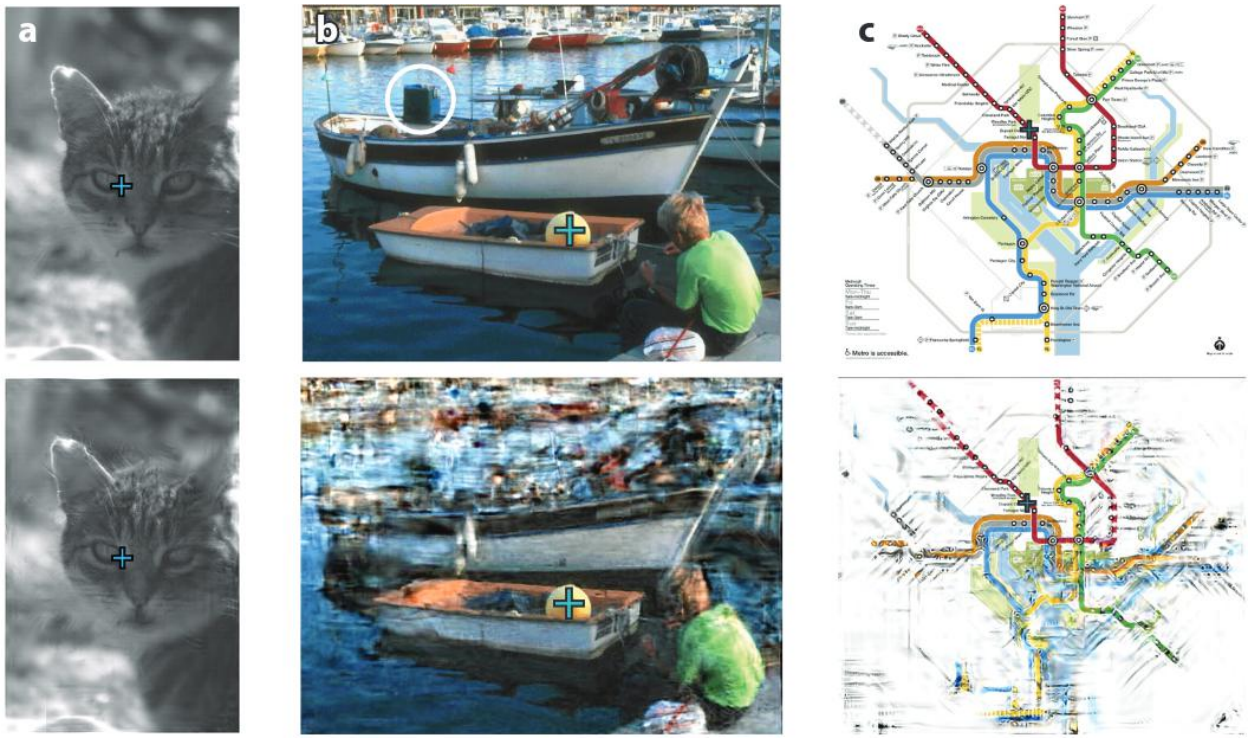

Figure 2.2: A visualised comparison between foveal (top) and peripheral (bottom) vision, according to the Texture Tiling Model (TTM) of [21]. The peripheral visualisations are designed such that they are approximately indistinguishable from the original image, if viewed in the human periphery. Test it out for yourself by sitting with eyes approximately 2ft from the page. Fixate on the midpoint of the line between subway maps (c), while bringing your cognitive attention to the two boat pictures (b). Do they look roughly the same? (The proper distance from the page will differ between people; if your answer is 'no', try changing your distance.)

Given this physiological difference, vision scientists have put a lot of effort into understanding how peripheral and foveal vision differ from one another. Figure 2.2 visualises the difference between the two as far as what data is believed to be both collected by the eyes and available for cognitive use by the brain.

A consequence of the degradation of the majority of visual data available to our visual systems is that human often generate initial visual estimates at a fairly low resolution, and only refine those estimates as needed. We are capable of refining both cognitively (by redirecting our cognitive attention and undertaking abstracted processing in the 'higher' levels of visual cortex) and physically (by redirecting our gaze to capture higher-resolution data with our fovea) [19]. For example, a human performing visual search for a salamander may at first catch sight of something 'salamander-y' from the corner of their eye, which they process at low-resolution before deciding to redirect their gaze in order to refine their visual estimate (Figure 2.3). I and my collaborators wondered if we replicate this ability for computer vision models, in a manner that would allow the majority of estimates and computations to be undertaken at low resolution (thus reducing memory and computational needs), without forcing us to compromise on overall model performance.

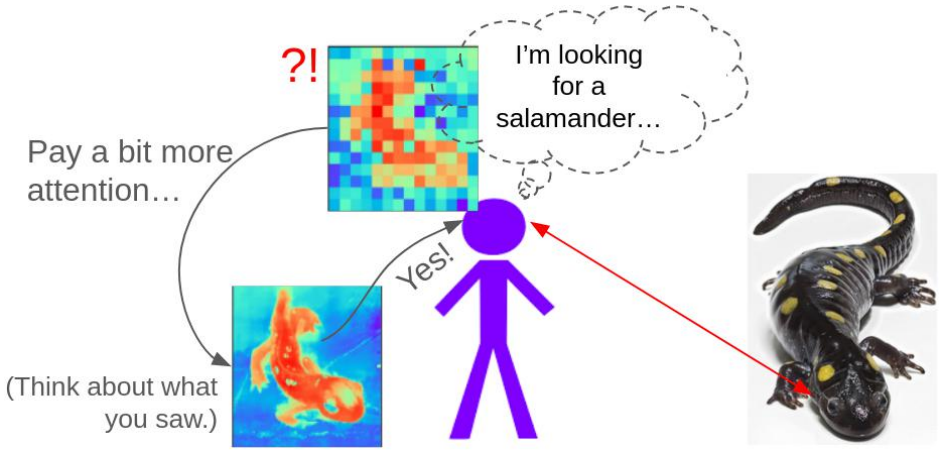

Figure 2.3: Humans often create initial estimates at low- (*e.g.*, peripheral) resolution, and only refine them if needed [19]. We wondered if we could replicate this in machines.

Latent network features are a cornerstone of computer vision research, capturing image semantics and abstractions and enabling the community to solve downstream tasks even in the zero- or few-shot regime. However, these features often lack the spatial resolution to directly perform dense estimation tasks like segmentation and depth estimation because models aggressively pool information over large areas. In this chapter, I will discuss FeatUp, a task- and model-agnostic framework to restore lost spatial information in deep features (Figure 2.4). Further, I will explain the two variants of FeatUp: one that guides features using a high-resolution signal in a single forward pass, and one that fits an implicit model to a single image to reconstruct features at any resolution. Both approaches use a multi-view consistency loss with deep analogies to the loss used in training neural radiance fields (NeRFs) [22].

Our upsampled features retain their original semantics and can be swapped into existing applications to yield resolution and performance gains even without re-training. We show that FeatUp significantly outperforms other feature upsampling and image super-resolution approaches in class activation map generation, transfer learning for segmentation and depth estimation, and end-to-end training for semantic segmentation.

2.1 Introduction

Recently, considerable effort has been made to develop methods to extract features from data modalities such as vision [23]–[27], text [28]–[30], and audio [31][32]. These features often form the backbone of different methods, including classification [33], weakly-supervised learning [34][35], semantic segmentation [36], optical flow [37][38], neural rendering [39], and more recently, image generation [40]. Despite their immense success, deep neural networks are usually forced to sacrifice spatial resolution for semantic quality. This is because semantic quality is generally proportional to feature vector length C, and if one were to encode a $H \times W \times 3$ image to a $H \times W \times C \gg 3$ latent feature map, the memory and computational requirements of the neural network would explode. So it is standard to downsample spatially. For example, ResNet-50 [41] produces 7×7 deep features from a

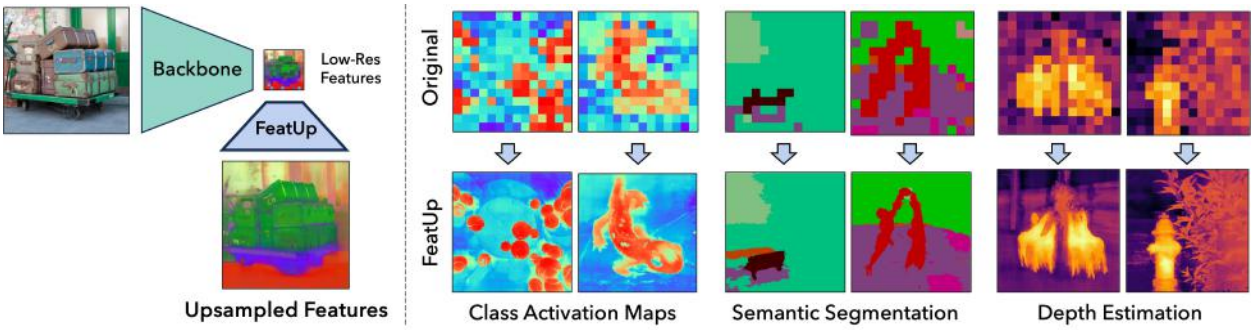

Figure 2.4: FeatUp upsamples image features from any model backbone, adding spatial resolution to existing semantics. High-resolution features can be learned either as a per-image implicit network or a general-purpose upsampling operation; the latter is an independent module to improve downstream dense estimation tasks.

224×224 pixel input (a $32\times$ reduction in spatial resolution). Even Vision Transformers (ViTs) [5] incur a significant resolution reduction, making it challenging to perform dense prediction tasks such as segmentation or depth estimation using these features alone.

To mitigate these issues, we propose FeatUp: a new framework for improving the resolution of any vision model's features without changing their original 'meaning' or orientation. Our primary insight, inspired by 3D reconstruction frameworks like neural radiance fields (NeRF) [22], is that enforced multiview consistency of low-resolution signals can supervise the reconstruction of high-resolution signals. More specifically, we learn to produce high-resolution information by aggregating low-resolution views from a model's outputs across multiple 'jittered' (*e.g.*, flipped, padded, cropped) images. We aggregate this information by learning an upsampling network with a multiview consistency loss. Our work explores two architectures for upsampling: a single guided upsampling feedforward network that generalises across images, and an implicit representation overfit to a single image.

This feedforward upsampler is a parameterised generalisation of a Joint Bilateral Upsampling (JBU) filter [42] powered by a CUDA kernel that is orders of magnitude faster and less memory-intensive than existing implementations. This upsampler can produce high quality features aligned to object edges at a computational cost comparable to only a few convolutions. Meanwhile, our implicit upsampler draws a direct parallel to NeRFs and overfits a deep implicit network to a signal. This model can be queried for features of arbitrary resolution, with low storage costs. Both architectures maintain correct semantic meaning, and can therefore be used to upsample features for use in pre-trained downstream applications. We show that the resulting upsampled features can significantly improve a variety of downstream tasks including semantic segmentation and depth estimation. Additionally, we show that model explanation methods such as class activation mapping (CAM) can be made higher-resolution using our upsampled features. In particular, one can study a model's behavior with much greater detail without the need for complex methods based on relevance and information propagation [43][44]. In summary, we include a short video describing FeatUp at https://aka.ms/featup and make the following contributions:

- FeatUp: a new method to significantly improve the spatial resolution of any model's

features, parameterised as either a fast feedforward upsampling network or an implicit network.

- A fast CUDA implementation of Joint Bilateral Upsampling that is orders of magnitude more efficient than a standard PyTorch implementation, and which allows guided upsampling in large-scale models.
- We show that FeatUp features can be used as drop-in replacements for ordinary features to improve performance on dense estimation tasks and model explainability.

2.2 Related Work

Image-adaptive filtering. Adaptive filters are commonly used to enhance images while preserving their underlying structure and content. For example, bilateral filters [45]–[47] apply a spatial filter to a low-resolution signal and an intensity filter to a high-resolution guidance to blend information from the two. Joint Bilateral Upsampling (JBU) [42] uses this technique to upsample a low-resolution signal with a high-resolution guidance. JBU has been used successfully for efficient image enhancement and other applications, but is not equipped to enhance high-dimensional signals like latent neural features, and existing implementations have prohibitive computational requirements. Recently, some works embed bilateral filtering approaches [48] and nonlocal means [49] into convolutional networks [50]–[53] and vision transformers [54][55]. Shape Recipes [56] learn the local relationship between signals to create upsampled target signals. Pixel-adaptive convolutional (PAC) networks [57] adapt a convolution operation to input data and has been used to advance performance in segmentation [58][59] and monocular depth estimation [60]–[62]. The Spatially-Adaptive Convolution (SAC) in [63] factors the adaptive filter into an attention map and convolution kernel. [64] extends bilateral filtering to superpixels and embeds this operation inside of a deep network to improve semantic segmentation. This class of methods, effective across a variety of applications, directly incorporates spatial information into the task while still allowing for flexibility in learning a network.

Image super-resolution. One of the earliest deep unsupervised super-resolution methods was Zero-Shot Super-resolution (ZSSR) [65], which learns a single-image network at test time. Local implicit models [66] use locally-adaptive models to interpolate information, and have been shown to improve the performance of super-resolution networks. Deep Image Priors [67] show that CNNs provide inductive biases for inverse problems such as zero-shot image denoising and super-resolution. While there is extensive literature on image super-resolution, these methods are not well-adapted to handle ultra-low resolution, yet high-dimensional deep features as we show in Section 2.6.2.

General-purpose feature upsampling. A widely-used approach for upsampling deep feature maps is bilinear interpolation. Though efficient, this method blurs information and is insensitive to the semantic content and high-resolution structure in the original image. Nearest neighbor and bicubic interpolation [68] have similar drawbacks. Encoding a higher resolution

input image yields correspondingly high-resolution features, but at a steep computational cost. Furthermore, this often degrades model performance and semantics due to the decreased relative receptive field size. Finally, for deep convolutional networks, one popular technique is to set final convolution strides to 1 [69][44]. However, this approach yields blurry features, as the model's receptive field is still large. Recent works using vision transformers (ViTs) [70][71] perform a similar modification on input patch strides, and interpolate positional encodings. Though simple and reasonably effective, this approach incurs a steep increase in computational footprint for every doubling in resolution, making it impossible to use in practice for larger upsampling factors. This approach can also distort features because of the previously mentioned fixed receptive field of the patches.

Image-adaptive feature upsampling. Many different operations exist in the literature to create features at higher resolutions. Deconvolutions [72]–[75] and transposed convolutions [76] use a learned kernel to transform features into a new space with a larger resolution. The resize-convolution [77] appends a learned convolution to a deterministic upsampling procedure, which reduces the checkerboard artifacts that plague deconvolutions [77]–[79]. The resize-convolution is now a common component of image decoders such as the U-Net [80] and has been applied to semantic segmentation [81]–[83] and super-resolution [84]–[86]. Other methods such as IndexNet [87] and Affinity-Aware Upsampling (A2U) [88] are effective on image matting but fall short on other dense estimation tasks [89]. Methods such as Pixel-Adaptive Convolutions [57], CARAFE [90], SAPA [91], and DGF [92] use learned input-adaptive operators to transform features. Though PAC is flexible, it does not upsample *existing* feature maps faithfully and instead is used to *transform* features for downstream tasks. DGF approximates the JBU operation with learned pointwise convolutions and linear maps, but does not fully implement JBU because it is computationally intractable. This is precisely the problem we solve, without approximation, with our new efficient CUDA kernel. FADE [89] introduces a new semi-shift operator and uses decoder features to produce a joint feature upsampling module. Implicit feature alignment (IFA) [93] takes a different tack, focusing on a nearest-neighbors approach to align feature maps in encoder-decoder architectures. While IFA performs well on the specific semantic segmentation benchmarks, it does not take advantage of image guidance and fails to learn high quality representations outside of the encode-decoder framework, as we show in Figure 2.12.

2.3 Problem statement

Given an 8-bit, 3-channel RGB image x of shape $H \times W \times 3$ and a pre-trained 'backbone' feature encoder $f(\cdot)$ that maps from the $H \times W \times 3$ image space to the spatially-downsampled $H_\downarrow \times W_\downarrow \times C$ latent space, where C is the length of the feature vectors $f(x)_{ij}$, one can obtain a feature map $f(x)$. We aim to train a guided upsampler $\sigma_\uparrow(f(x), x)$ that is capable of upsampling the spatial dimensions of a low-resolution $f(x)$ from $H_\downarrow \times W_\downarrow$ back to $H \times W$, potentially using information from x a spatial 'guidance' signal, while also retaining the semantics encoded in the C-dimensional feature vectors $f(x)_{ij}$, for all $i \in [0, H_\downarrow)$ and $j \in [0, W_\downarrow)$. We label the upsampled $H \times W \times C$ feature map $F_{hr} \equiv \sigma_\uparrow(f(x), x)$.

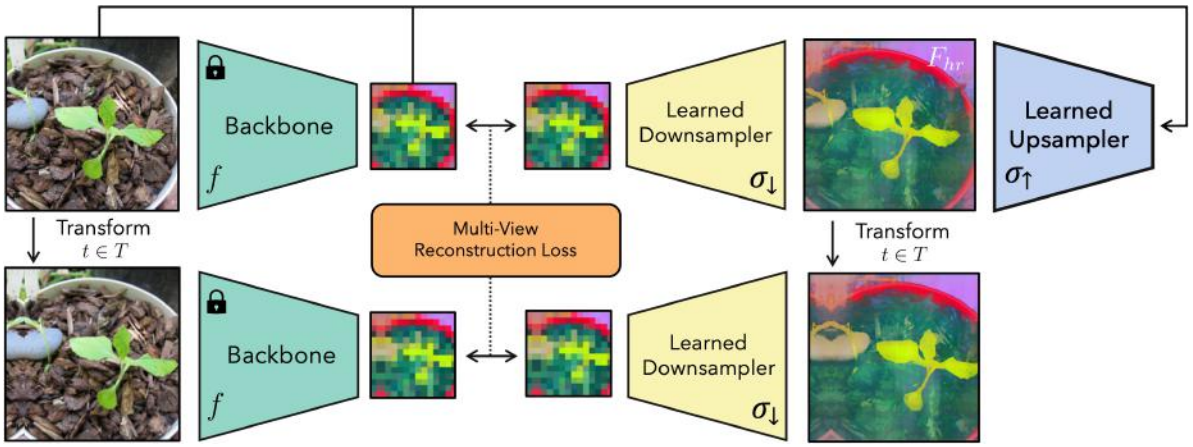

Figure 2.5: The FeatUp training architecture. FeatUp learns to upsample features through a consistency loss on low resolution 'views' of a model's features that arise from small perturbations of the input image.

2.4 Methods

The core intuition behind FeatUp is that one can compute high-resolution features by observing multiple different 'views' of low-resolution features. We draw a comparison with 3D scene reconstruction models such as NeRF [22]; in the same way that NeRF builds an implicit representation [94][95] of a 3D scene by enforcing consistency across many 2D projections of the scene, FeatUp builds an upsampler by enforcing consistency across many low-resolution feature maps. As in the broader NeRF literature, a variety of methods can arise from this basic idea. In this work, we introduce a lightweight, forward-pass upsampler based on Joint Bilateral Upsampling [42] as well as an implicit network-based upsampler. The latter is learned per-image and query-able at arbitrary resolution. We provide an overview of the general FeatUp architecture in Figure 2.5.

The first step in our pipeline is to generate low-resolution feature views to refine into a single high-resolution output. To this end, we perturb the input image with small pads, scales, and horizontal flips, and apply a 'backbone' encoder model to each transformed image, in addition to the original image, to extract a collection of low-resolution feature maps. These small image jitters allow us to observe tiny differences in the output features and provide sub-feature information to train the upsampler.

Next, we construct a consistent high-resolution feature map from these views. We postulate that we can learn a latent high-resolution feature map that, when downsampled, reproduces our low-resolution jittered features (see Figure 2.5). FeatUp's downsampling is a direct analog to ray-marching; just as 3D data is projected down to 2D in this NeRF step, our downsampler projects high-resolution features to low-resolution features. Unlike NeRF, we do not need to estimate the parameters that generate each view. Instead, we track the parameters used to 'jitter' each image and apply *the same* transformation to our learned high-resolution features prior to downsampling. We then compare downsampled features to the true model outputs using a Gaussian likelihood loss [96]. A good high-resolution feature map should reconstruct the observed features across all the different views.

More formally, let $t \in T$ be from a collection of image perturbation transforms such as pads, zooms, crops, horizontal flips, and their compositions. Let x be an input image, $f(\cdot)$ be our model backbone (such that $f(x)$ is our low-resolution feature map), $\sigma_\uparrow(\cdot,\cdot)$ be a learned upsampler that yields the high-resolution feature map $F_{hr} \equiv \sigma_\uparrow(f(x), x)$ when applied to $f(x)$ and x, and $\sigma_\downarrow(\cdot)$ be a learned downsampler that maps $H \times W \times C \to H_\downarrow \times W_\downarrow \times C$. We note that this parameterisation allows $\sigma_\uparrow$ to be a guided upsampler (which depends on both $f(x)$ and x), an unguided upsampler (which depends on only $f(x)$), or an implicit network (which depends on only x).

Our goal is to minimise, across all of the $t \in T$, the difference between the low-resolution feature map $f(t(x))$ that results from perturbing the input image x by $t(\cdot)$, and the downsampled feature map that $\sigma_\downarrow(\cdot)$ produces from a perturbed high-resolution feature map $t(F_{hr})$ (this is the *same* perturbation transform $t(\cdot)$), *i.e.*, $\sigma_\downarrow(t(F_{hr}))$. The expectation is that these two perturbed, low-resolution feature maps — one obtained by directly encoding, the other from downsampling — are equivalent to one another, and should ideally be the same. We therefore construct our main multi-view reconstruction loss term as follows:

$$\mathcal{L}_{rec} = \frac{1}{|T|} \sum_{t \in T} \frac{1}{2s^2} \| f\left(t\left(x\right)\right) - \sigma_\downarrow\left(t\left(F_{hr}\right)\right) \|_2^2 + \log(s) \tag{2.1}$$

averaging over all $t \in T$, where $\|\cdot\|_2$ is the standard L2 norm and $s = \mathcal{M}(f\left(t\left(x\right)\right))$ is a spatially-varying normalisation term parameterised by a small linear network $\mathcal{M}$ [96]. The normalisation s turns the mean-squared error (MSE) loss into a proper likelihood. This extra flexibility allows the network to learn when certain outlier features fundamentally cannot be upsampled. In Figure 2.18, we visualise this adaptive normalisation. In Figure 2.13, we show its effectiveness in an ablation study.

2.4.1 Choosing a Downsampler

Given the loss function described above, we need to choose the architecture of our learned downsampler $\sigma_\downarrow$. We introduce two options: a fast and simple learned blur kernel, and a more flexible attention-based downsampler. Both proposed modules do not change the 'space' or 'semantics' of the features with nontrivial transformations, but rather only interpolate features within a small neighborhood. We diagram both choices in Figure 2.6 and demonstrate the effectiveness of the attention downsampler in Figure 2.13.

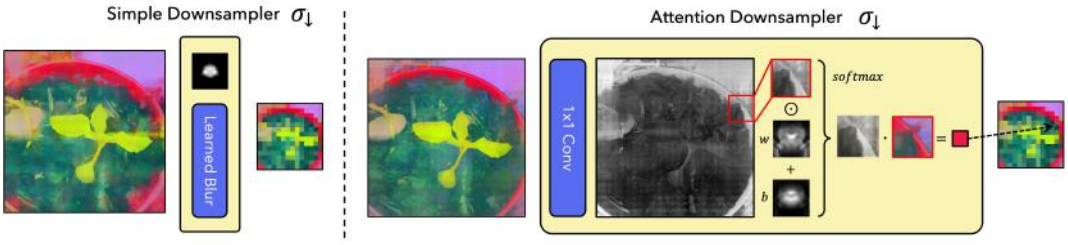

Figure 2.6: We introduce two learned downsamplers. The simple downsampler (Left) is a fast, learned blur kernel. The attention downsampler (right) combines a predicted salience map with spatially invariant weight and bias kernels. The attention downsampler adapts better to networks with nonlinear and dynamic receptive fields.

Our simple downsampler blurs the features with a learned blur kernel and can be implemented as a convolution applied independently to each channel. The learned kernel is normalised to be non-negative and to sum to 1 to ensure the features remain in the same space.

Though this blur-based downsampler is efficient, it cannot capture dynamic receptive fields, object salience, or other nonlinear effects. To this end, we also introduce a more flexible attention downsampler that spatially adapts the downsampling kernel. In short, this component uses a 1x1 convolution to predict a saliency map from the high-dimensional, high-resolution features. Then, it combines this saliency map with learned but spatially-invariant weight and bias kernels and normalises the result to create a spatially-varying blur kernel that interpolates the features. More formally:

$$\sigma_{\downarrow}(F_{hr})_{ij} = \text{softmax}(w \odot \text{Conv}(F_{hr}[\Omega_{ij}]) + b) \cdot F_{hr}[\Omega_{ij}] \tag{2.2}$$

where $\sigma_{\downarrow}(F_{hr})_{ij}$ is the ijth component of the resulting feature map and $F_{hr}[\Omega_{ij}]$ refers to a patch of high resolution features corresponding to the ij location in the downsampled features. $\text{Conv}(\cdot)$ is the 1x1 convolution. $\odot$ and $\cdot$ refer to the elementwise and inner products respectively, and w and b are learned weight and bias kernels shared across all patches. Our main hyperparameter for both downsamplers is the kernel size, which should be larger for models with larger receptive fields such as convolutional nets. We defer discussion of model-specific hyperparameters to Section 2.6.14.

2.4.2 Choosing an Upsampler

A central choice in our architecture is the parameterisation of $\sigma_{\uparrow}$, and we introduce two variants. First, 'JBU' FeatUp parameterises $\sigma_{\uparrow}$ with a guided upsampler based on a stack of Joint Bilateral Upsamplers (JBU) [42]. This architecture learns an upsampling strategy that generalises across a corpus of images. The second method, 'Implicit' FeatUp, uses an implicit network to parameterise $\sigma_{\uparrow}$ and can yield remarkably crisp features, but is overfit to a single image. Both methods are trained using the same broader architecture and multiview consistency loss. We illustrate both strategies in Figure 2.7.

Figure 2.7: Our Implicit version of FeatUp learns an implicit network to upsample a single image's features. Our JBU FeatUp learns a stack of JBUs that learns to quickly upsample features from a large image corpus.

Joint Bilateral Upsampler. Our feedforward upsampler uses a stack of parameterised joint bilateral upsamplers (JBU) [42] to obtain the high resolution feature map F_{hr}:

$$F_{hr} \equiv \sigma_{\uparrow}(f(x), x) = \text{JBU}(\cdot, x) \circ \text{JBU}(\cdot, x) \circ \ldots \circ \text{JBU}(f(x), x) \tag{2.3}$$

where $\circ$ is function composition, $f(x)$ is the low-resolution feature map, and x is the original image. This architecture is fast, directly incorporates high-frequency details from the input image into the upsampling process, and is independent of the architecture of f.

In joint bilateral upsampling a high-resolution signal, G, is used as guidance for the low-resolution features F_{lr}. In FeatUp, G is the input imageWe let Ω be a neighborhood of each pixel in the guidance. In practice, we use a 3×3 square centered at each pixel. Let $k(\cdot, \cdot)$ be a similarity kernel that measures how 'close' two vectors are. We can then form our joint bilateral filter:

$$\hat{F}_{hr}[i, j] = \frac{1}{Z} \sum_{(a,b)\in\Omega} \left(F_{lr}[a, b] \cdot k_{range}(G[i, j], G[a, b]) \cdot k_{spatial}([i, j], [a, b]) \right) \tag{2.4}$$

where Z is a normalisation factor to ensure that the kernel sums to 1. Here, $k_{spatial}$ is a learnable Gaussian kernel on the Euclidean distance between coordinate vectors of width $\sigma_{spatial}$:

$$k_{spatial}(x, y) = \exp\left(\frac{-\|x - y\|_2^2}{2\sigma_{spatial}^2} \right) \tag{2.5}$$

Furthermore, k_{range} is a temperature-weighted softmax [96] applied to the inner products from a multi-layer perceptron (MLP) that operates on the guidance signal G:

$$k_{range}(x, y) = \text{softmax}_{(a,b)\in\Omega} \left(\frac{1}{\sigma_{range}^2} \text{MLP}(G[i, j]) \cdot \text{MLP}(G[a, b]) \right) \tag{2.6}$$

where σ_{range}^2 acts as the temperature. We note that the original JBU used a fixed Gaussian kernel on the guidance signal, G. Our formulation generalises the original JBU [42] implementation to high-dimensional signals and makes this operation learnable. In our experiments, we use a two-layer GeLU [97] MLP with 30-dimensional hidden and output vectors. To evaluate $F_{lr}[a, b]$ we follow the original JBU formulation and use bilinear-interpolated features if the guidance pixel does not directly align with a low-resolution feature. For resolution independence, we use coordinate distances normalized to $[-1, 1]$ in the spatial kernel.

One challenge we faced was the poor speed and memory performance of existing JBU implementations, which could explain why this simple and apparently effective approach is not used more widely. To address this challenge, we contribute an efficient CUDA implementation of the spatially adaptive kernel used in the JBU. Compared to a naive PyTorch implementation with the `torch.nn.Unfold` operator, our operation uses up to two orders of magnitude less memory and speeds inference by up to $10\times$. We demonstrate its significant performance improvements in Section Table 2.6.

Implicit Our second upsampler architecture draws a direct analogy with NeRF by parameterising the high-resolution features of a *single* image with an implicit function $F_{hr} = \text{MLP}(z)$. Several existing upsampling solutions also take this inference-time training approach, including DIP [67] and LIIF [66]. We use a small MLP to map image coordinates and intensities to a high-dimensional feature for the given location. We follow the guidance of prior works [22][94][98] and use Fourier features to improve the spatial resolution of our implicit representations. In addition to standard Fourier positional features, we show that adding Fourier colour features allows the network to use high-frequency colour information from the original image. This significantly speeds convergence and enables elegant usage of high-resolution image information without requiring the complication of additional techniques like Conditional Random Fields (CRFs, [99]). We illustrate the profound effect of Fourier colour features in Section 2.6.3.

More formally, let $h(z, \hat{\omega})$ represent the component-wise discrete Fourier transform of an input signal z, with a vector of frequencies $\hat{\omega}$. Let e_i and e_j represent the two-dimensional pixel coordinate fields ranging in the interval $[-1, 1]$. Let : represent concatenation along the channel dimension. We can now express our high-resolution feature map as:

$$F_{hr} = \text{MLP}(h(e_i : e_j : x, \hat{\omega})) \tag{2.7}$$

Our MLP is a small 3-layer ReLU [100] network with dropout $p = 0.1$ [101] and layer normalization [102]. We note that, at test time, we can query the pixel coordinate field to yield features F_{hr} at *any* resolution. The number of parameters in our implicit representation is over two orders of magnitude smaller than a (224×224) explicit representation while being more expressive, significantly reducing convergence time and storage size.

2.4.3 Additional Method Details

Accelerated Training with Feature Compression To reduce the memory footprint and further speed up the training of FeatUp's implicit network, we first compress the spatially-varying features to their top $k = 128$ principal components. This operation is approximately lossless as the top 128 components explain $\sim 96\%$ of the variance across a single image's features. This compression improves training time by a factor of $60\times$ for ResNet-50, reduces the memory footprint, enables larger batches, and does not have any observable effect on learned feature quality. When training the JBU upsampler, we sample random projection matrices in each batch to avoid computing a principal component analysis (PCA) in the inner loop. This sampling achieves the same effect as inner-loop PCA, thanks to the Johnson–Lindenstrauss lemma [103].

Total Variation Prior To avoid spurious noise in the high resolution features, we add a small ($\lambda_{tv} = 0.05$) total variation smoothness prior [104] on the implicit feature magnitudes:

$$\mathcal{L}_{tv} = \sum_{i,j} \left((||F_{hr}[i,j]|| - ||F_{hr}[i-1,j]||)^2 + (||F_{hr}[i,j]|| - ||F_{hr}[i,j-1]||)^2 \right) \tag{2.8}$$

This prior is faster than regularising full features and avoids overprescribing how the individual components should organise. We do not use this in the JBU upsampler because the

upsampler does not suffer from overfitting. We demonstrate the importance of this regulariser in Section 2.6.3.

2.5 Key Experiments

We compare both our JBU and Implicit variants of FeatUp against several key upsampling baselines from the literature, in particular: Bilinear upsampling, Resize-conv, Strided (*i.e.*, reducing the stride of the backbone's patch extractor), Large Image (*i.e.*, using a larger input image), CARAFE [90], SAPA [91], and FADE [89]. We upsample ViT [5] features sixteenfold (to the resolution of the input image) with every method except the strided and large-image baselines, which are computationally infeasible above eightfold upsampling. For additional details on the strided implementation, please refer to Section 2.6.1.

2.5.1 Qualitative Comparisons

Visualising upsampling methods Figure 2.8 demonstrates the dramatic qualitative improvement FeatUp achieves compared to several baselines. Our visualisations fit a 3-dimensional PCA on each image's low-resolution ViT features and use this PCA to map upsampled features into the same RGB space. We also show that this high-fidelity upsampling extends to higher PCA components in Figure 2.16, and that FeatUp can improve small object retrieval in Figure 2.22.

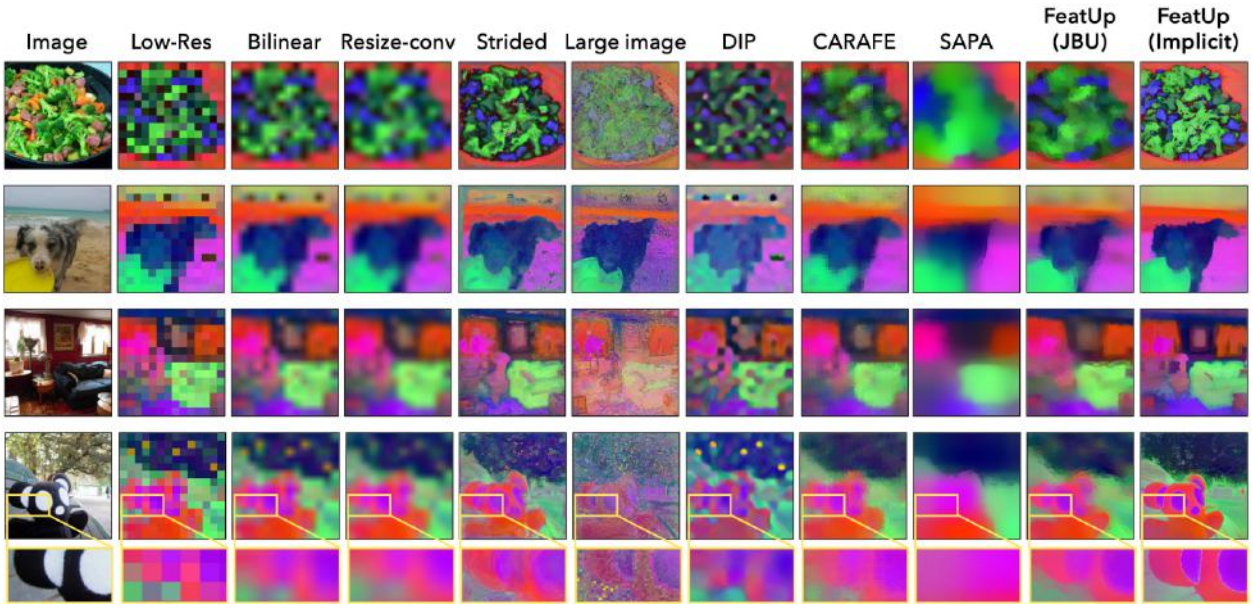

Figure 2.8: Low-resolution ViT features (14×14) from the COCO-Stuff validation set are upsampled by $16\times$. FeatUp produces clean, high-resolution results when compared with faster methods like bilinear, resize-conv, and SAPA. Other methods like strided and CARAFE perform similarly well on many samples, but run substantially slower (Figure 2.24).

Robustness across vision backbones Figure 2.9 demonstrates that FeatUp can upsample a variety of modern vision backbones. In particular, we show the implicit FeatUp features across a variety of backbones spanning transformers, convolutional nets, and both supervised and self-supervised models. Even though backbones like ResNet-50 do not precisely localise objects due to their large receptive fields, FeatUp can reasonably associate features to the correct object.

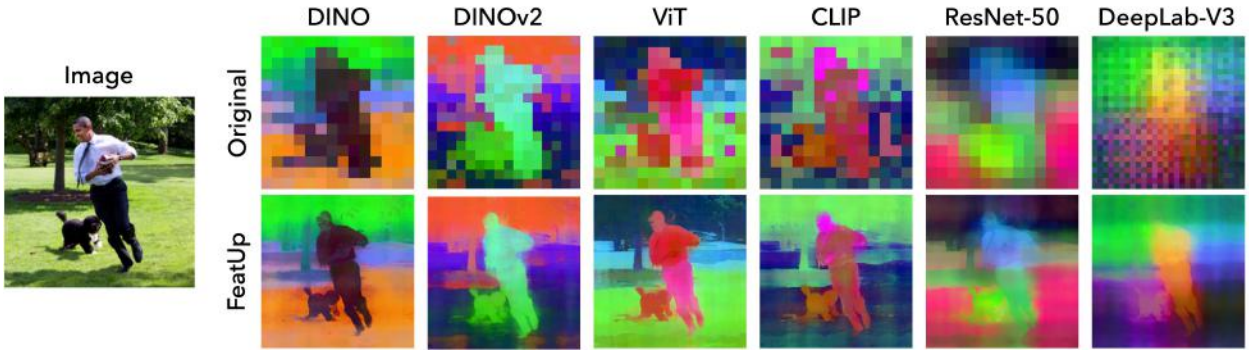

Figure 2.9: FeatUp can upsample the features of any backbone, even convolutional networks with aggressive nonlinear pooling. In all cases, the feature maps yielded are high-resolution, object-aligned, and semantically meaningful.

2.5.2 Transfer Learning for Semantic Segmentation and Depth Estimation

Next, we demonstrate that FeatUp can serve as a drop-in replacement for existing features in downstream applications. To demonstrate this, we adopt the widely used experimental procedure of using linear probe transfer learning to evaluate representation quality [105]. More specifically, we train linear probes on top of low-resolution features for both semantic segmentation and depth estimation. We then freeze and apply these probes to upsampled features to measure performance improvement. If the upsampled features are valid improvements, existing probes should work well without adaptation. For all experiments, we use a frozen pre-trained ViT-S/16 as the featuriser, upsample the features (14x14 $\rightarrow$ 224x224), and extract maps by applying a linear layer on the features.

For semantic segmentation, we follow the experimental setting of [106][35] and train a linear projection to predict the coarse classes of the COCO-Stuff (27 classes) training dataset using a cross-entropy loss. We report mIoU and accuracy on the validation set in Table 2.1. For depth estimation we train on pseudo-labels from the MiDaS [107] depth estimation network using their scale- and shift-invariant MSE. We report root mean-squared error (RMSE) and the $\delta > 1.25$ metric which is common in monocular depth estimation literature. More specifically this metric is defined as the percentage of pixels with $\delta = \max(\frac{y}{y^*}, \frac{y^*}{y}) > 1.25$ where y is the depth prediction and y^* is the ground truth.

	CAM Score		Semantic Seg.		Depth Estimation	
	↓ A.D.	↑ A.I.	↑ Acc.	↑ mIoU	↓ RMSE	↑ $\delta > 1.25$
Low-res	10.69	4.81	65.17	40.65	1.25	0.894
Bilinear	10.24	4.91	66.95	42.40	1.19	0.910
Resize-conv	11.02	4.95	67.72	42.95	1.14	0.917
DIP	10.57	5.16	63.78	39.86	1.19	0.907
Strided	11.48	4.97	64.44	40.54	2.62	0.900
Large image	13.66	3.95	58.98	36.44	2.33	0.896
CARAFE	10.24	4.96	67.1	42.39	<u>1.09</u>	0.920
SAPA	10.62	4.85	65.69	41.17	1.19	0.917
FeatUp (JBU)	<u>9.83</u>	<u>5.24</u>	<u>68.77</u>	<u>43.41</u>	<u>1.09</u>	**0.938**
FeatUp (Implicit)	**8.84**	**5.60**	**71.58**	**47.37**	**1.04**	<u>0.927</u>

Table 2.1: Comparison of feature upsamplers across metrics on CAM faithfulness, linear probe semantic segmentation, and linear probe depth estimation. Both FeatUp variants consistently outperform other approaches, including other forward-pass upsamplers (CARAFE, SAPA) and features optimised at inference-time (DIP). **Bold** is best. <u>Underlined</u> is second-best.

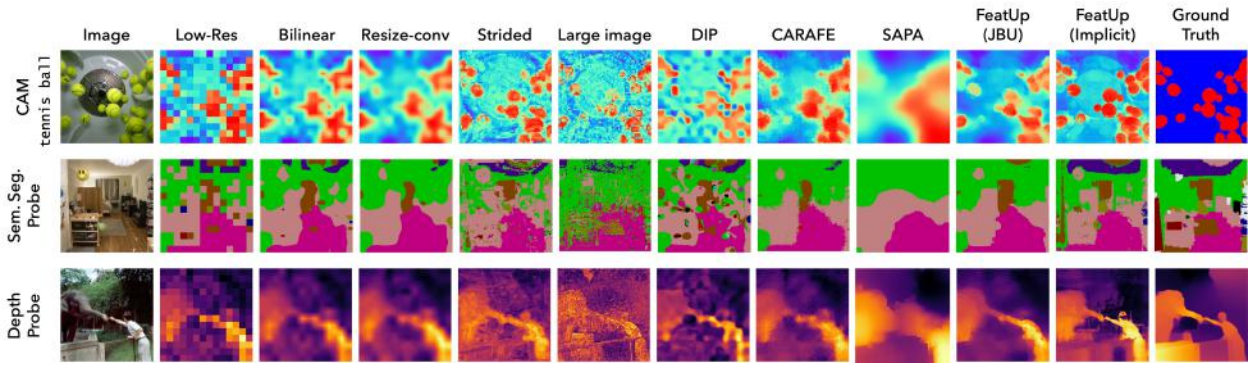

Figure 2.10: A comparison of different upsampling methods across each of the tasks considered in our analysis. FeatUp achieves significant improvements in resolution across each task.

We stress that these linear probe evaluations show that FeatUp features can improve downstream tasks *without* re-training models. These analyses do not aim to create SOTA segmentation or depth networks. Both FeatUp variants outperform all baselines across all experiments, showing that either variant can be used as a drop-in replacement for existing features. Qualitatively, Figure 2.10 and Figures 2.20 - 2.21 show cleaner, more object-aligned predictions across all tasks.

2.5.3 Class Activation Map Quality

Attributing a model's predictions to specific pixels is crucial for diagnosing failures and understanding a model's behavior. Unfortunately, common interpretation methods like Class

Activation Maps (CAM) are limited by the low resolution of the deep feature maps and cannot resolve small objects. We show that FeatUp features can be incorporated into existing CAM analyses to yield stronger and more precise explanations. More specifically, we use the literature's established metrics, Average Drop (A.D.) and Average Increase (A.I.), that measure CAM quality (refer to Section 2.6.9 for a detailed description of these metrics). Intuitively, A.D. and A.I. capture how much an image's most salient region changes the classification output. A good CAM should highlight regions with the greatest effect on the classifier's predictions, so censoring these regions will have the largest impact on the model's predictions (lower A.D., higher A.I.). Upsamplers are trained on the ImageNet training set for 2,000 steps, and we compute metrics across 2,000 random images from the validation set. We use a frozen pre-trained ViT-S/16 as the featuriser, and extract CAMs by applying a linear classifier after max-pooling. Upsampling is done (14x14 $\rightarrow$ 224x224) on the features themselves, and CAMs are obtained from these high-resolution maps. We report results in Table 2.1 and Figures 2.10 and 2.19.

2.5.4 End-to-end Semantic Segmentation

FeatUp not only improves the resolution of pre-trained features but can also improve models learned end-to-end. We adopt the experimental setting of [91][89] to show that our JBU upsampler improves end-to-end performance on ADE20K semantic segmentation using the SegFormer [108] architecture. Specifically, we train SegFormer on ADE20k [109][110] (20,210 training and 2,000 validation images) for 160k steps. To validate that our setup matches that of the existing literature despite numerical discrepancies, we also compute FLOPs for SegFormer with various upsamplers in Table 2.2. These counts are comparable with those in [111], confirming our architectural setup. We report mean intersection over union (mIoU), mean class accuracy (mAcc), and all-pixel accuracy (aAcc) against several recent baselines in Table 2.2 including IndexNet [87], A2U [112], CARAFE [90], SAPA [91], and FADE [89], in addition to more standard bilinear and resize-conv operators. Figure 2.11 shows examples of segmentation predictions across these methods. FeatUp consistently outperforms baselines with fewer added parameters, showing that FeatUp can also improve a broader, jointly trained architecture.

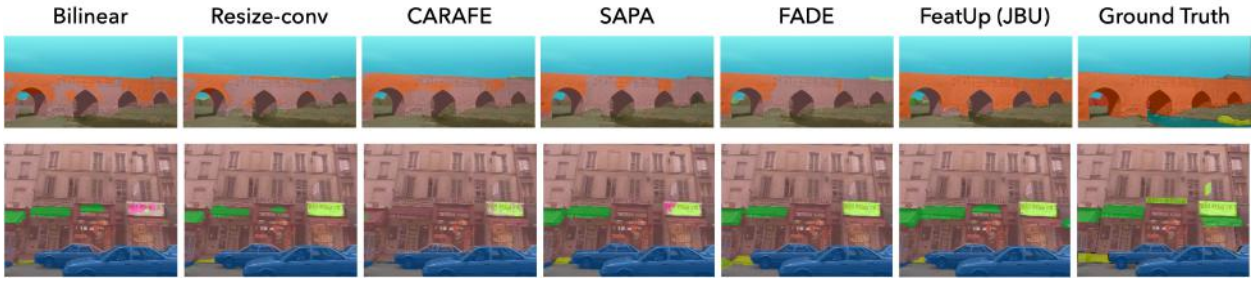

Figure 2.11: End-to-end training performance of different upsampling methods from our SegFormer-based semantic segmentation experiments. These results do not use linear probes, but instead train the architecture jointly.

Metric	Bilinear	Resize-conv	IndexNet	A2U	CARAFE	SAPA	FADE	FeatUp (JBU)
↑ mIoU	39.7	41.1	41.5	41.5	42.4	41.6	<u>43.6</u>	**44.2**
↑ mAcc	51.6	51.9	52.2	52.3	53.2	<u>55.3</u>	54.8	**55.8**
↑ aAcc	78.7	79.8	<u>80.2</u>	79.9	80.1	79.8	**80.7**	**80.7**
↓ Params (M)	13.7	+3.54	+12.6	+0.12	+0.78	+0.20	+0.29	+0.16
↓ GFLOPs	16.0	+34.40	+30.90	+0.51	+1.66	+1.15	+2.95	+1.70

Table 2.2: Semantic segmentation results with the SegFormer [108] architecture trained on the ADE20k train set and evaluated on the val set. FeatUp (JBU) outperforms the standard bilinear and resize-conv upsamplers in U-Net architectures, IndexNet [87], A2U [88], and other task-agnostic upsamplers (CARAFE [90], SAPA [91], FADE [89]). Additionally, our upsampler is competitive in parameter and floating-point operation (FLOP) count. **Bold** is best. <u>Underlined</u> is second-best.

2.6 Additional Results

2.6.1 Strided baseline implementation

For the DINO and ViT backbones, we extract patches with a stride of $\frac{16}{\text{upsample factor}}$ to produce a higher density of feature vectors and thus increase feature resolution. We point out that the upsampling factor is limited with this method (as the stride is lower bounded by one), so this approach can only upsample up to 16x for ViT-S/16. Practically however, these maximum upsampling factors are impractical as they require far more memory than current GPUs provide (see Figure 2.24).

2.6.2 Comparison to Image-Upsampling Methods

A variety of methods have been proposed for image super-resolution. Among the learning-based approaches, deep image prior (DIP) [67] has been used succesfully for enhancing images without additional training data. Figure 2.12 shows that DIP poorly upsamples features, introducing artifacts and 'blob' patterns in the features and downstream outputs. [65] introduced Zero-Shot Super-Resolution, a method that learns an image-specific CNN at test time without additional training data. Additionally, images can be represented as Local Implicit Image Functions (LIIF) [66] which can be queried at arbitrary resolution. While similar to FeatUp's implicit network, LIIF trained to continuously represent a feature map does not produce sharp outputs like FeatUp (Figure 2.12) Despite these methods' successes in the image super-resolution problem space, they are not equipped to upsample high-dimensional features.

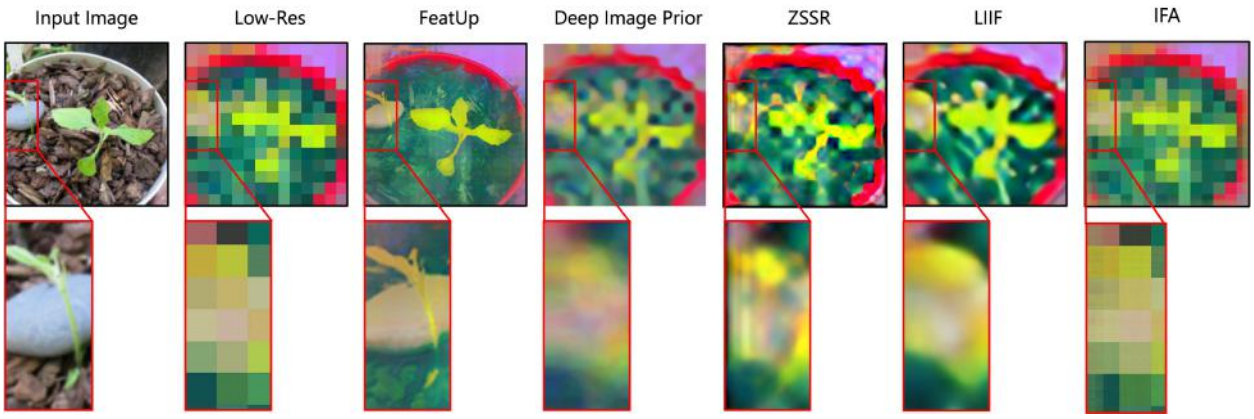

Figure 2.12: Comparison of image super-resolution methods using Deep Image Prior, Zero-Shot Super-Resolution (ZSSR), and Local Implicit Image Function (LIIF). We also include a visualisation on Implicit Feature Alignment (IFA). As shown in the whole feature map and zoomed-in section, these image upsampling methods do not effectively upsample the low-resolution and high-dimensional feature maps by the large upsampling factors that FeatUp is able to handle.

2.6.3 Ablation Studies

We show the effects of each design decision for FeatUp in Figure 2.13. Without the spatially varying normalisation, our upsampler blurs ResNet features; possibly because it cannot ignore certain nonlinear artifacts or resolve the large pooling window present in ResNet-50. The magnitude regulariser provides smoothing and regularisation benefits. Our choice to include Fourier colour features dramatically improves resolution and high-frequency details. Finally, the attention downsampler helps the system avoid odd edge and halo effects by learning kernels more focused on salient parts of the signal. Using an explicit buffer of features instead of an implicit network yields significant artifacts, though we note that the artifacts are significantly less dramatic if the simple downsampler is also used.

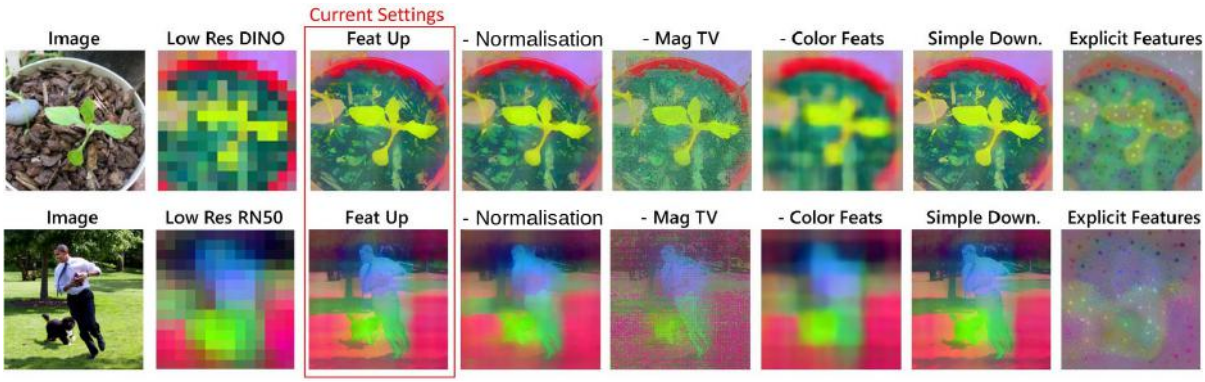

Figure 2.13: Qualitative ablation study across both DINO and Resnet50 Backbones. The biggest improvements arise from the implicit featuriser, colour features, and the magnitude TV regulariser. Including the spatially-dependent normalisation has little effect in the first row, but appears to improve feature map alignment with objects and shadows in the second.

We also provide an ablation study of FeatUp's jitter hyperparameters in Figure 2.14, and of the total variation and magnitude regularisers in Figure 2.15 and Table 2.3. Our regulariser is fairly robust to different settings as shown by the 2x multiplication for both terms in the 3rd column. However, there still exists an optimal λ range that provide important smoothing properties; larger values can interfere with the main reconstruction objective as shown in the final column.

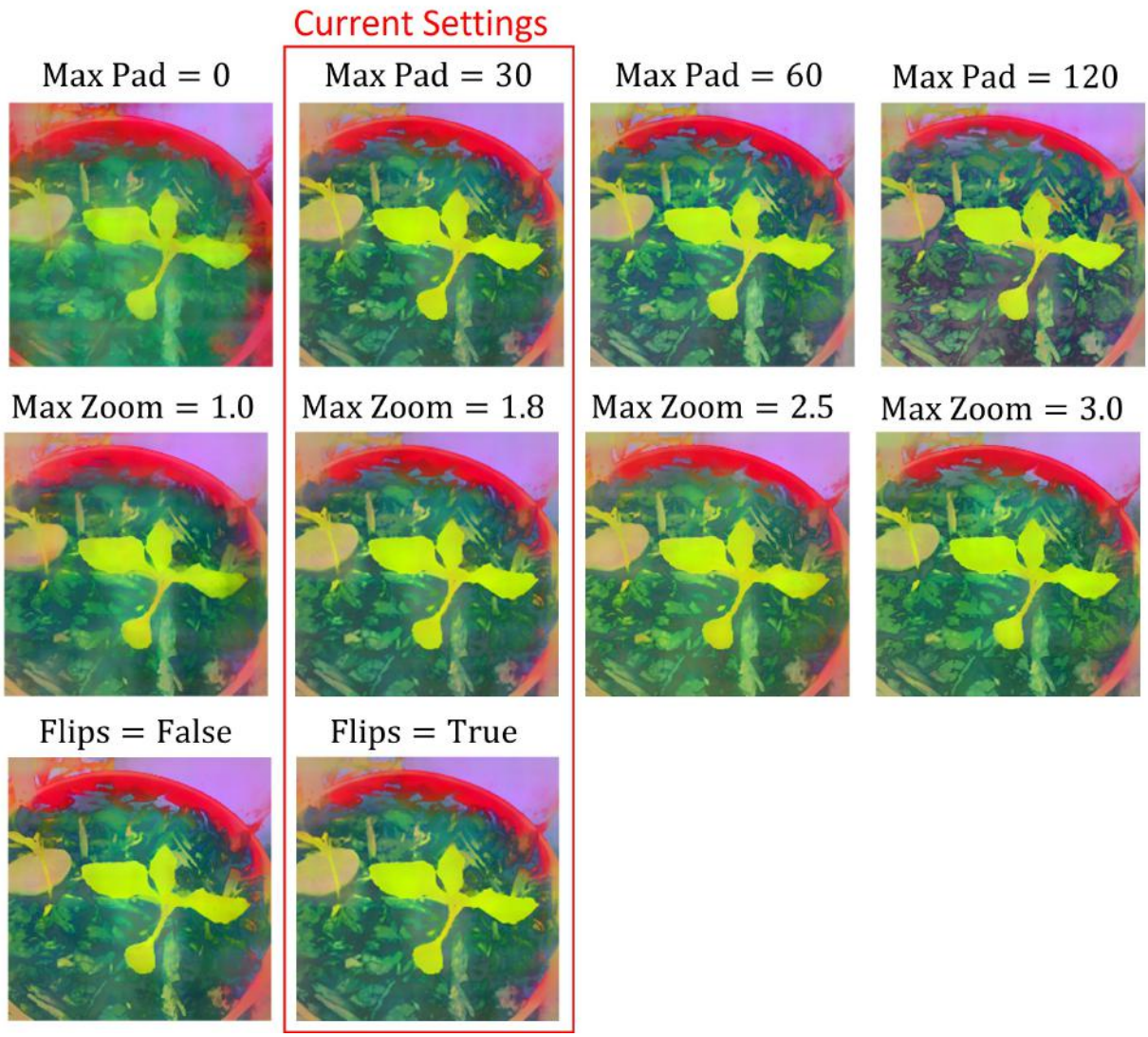

Figure 2.14: Ablation of FeatUp's jitter hyperparameters. We are robust to a range of pad, zoom, and flip values, though features degrade with large changes in max pad.

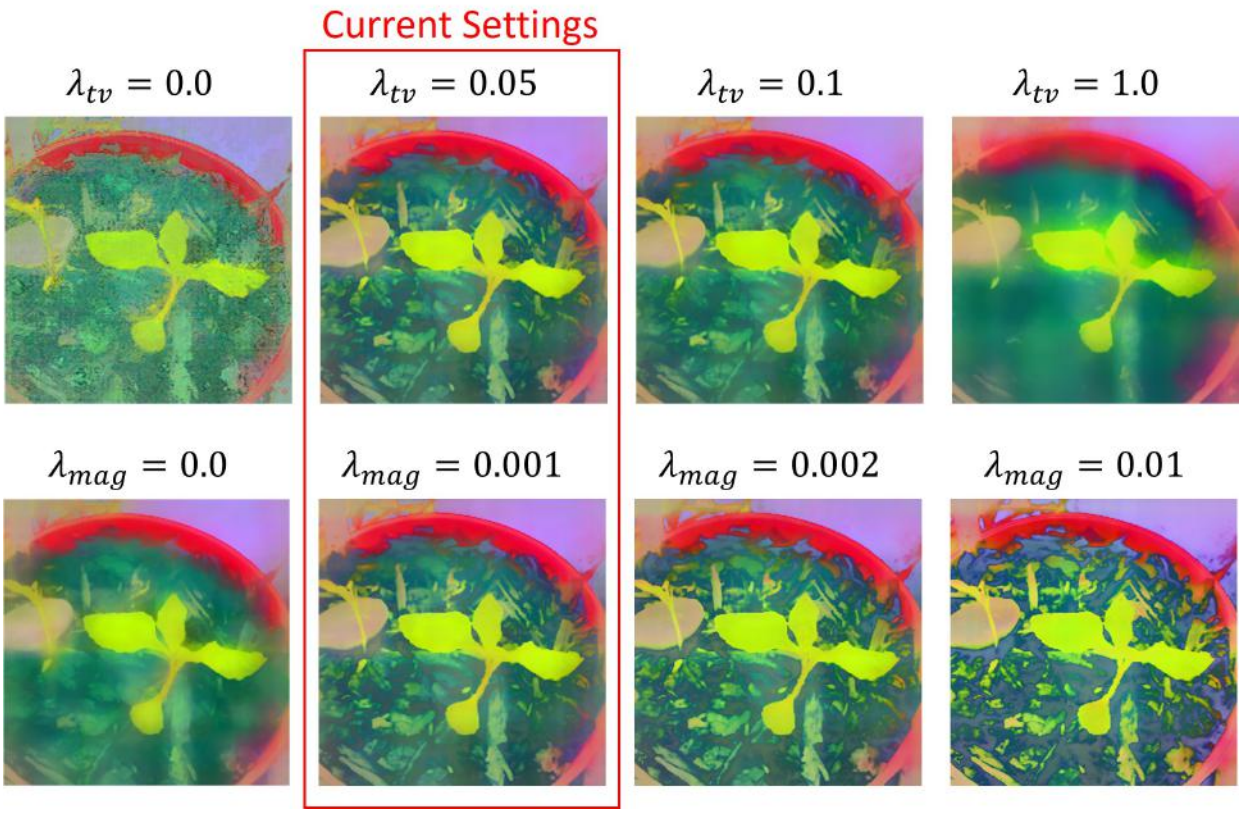

Figure 2.15: Qualitative ablation study of the TV and magnitude regularisers. FeatUp is fairly robust to the settng of these parameters.

			CAM Score		Semantic Seg.		Depth Estimation	
Attn DS.	O.D.	TV Reg.	↓ A.D.	↑ A.I.	↑ Acc.	↑ mIoU	↓ RMSE	↑ δ >1.25
✓	✓	✓	**8.84**	**5.60**	**71.58**	**47.37**	**1.04**	**0.927**
✗	✓	✓	9.07	5.06	70.95	46.79	1.11	0.916
✓	✗	✓	8.91	5.55	71.26	46.89	1.08	0.920
✓	✓	✗	9.10	5.00	68.06	44.36	1.11	0.913

Table 2.3: Ablation study for implicit FeatUp features with varied downsampler (attention = ✓, simple = ✗), outlier detection, and λ_{TV} (0.05 = ✓, 0.0 = ✗). **Bold** is best.

To further justify our design decisions in the context of an end-to-end trained architecture, we evaluate JBU FeatUp's performance both in use for linear probe transfer to class activation mapping (CAM), semantic segmentation, and depth estimation (Table 2.4), and in the SegFormer [108] trained end-to-end for semantic segmentation (Table 2.5). We (1) remove the MLP (see Equation 2.6) on the guidance signal, (2) remove the temperature-weighted softmax and replacing it with Euclidean distance between the central feature and its neighborhood, and (3) remove the softmax and replacing it with cosine distance. Each ablation degrades segmentation performance, with the MLP exclusion being the most detrimental.

	CAM Score		Semantic Seg.		Depth Estimation	
Ablation	↓ A.D.	↑ A.I.	↑ Acc.	↑ mIoU	↓ RMSE	↑ δ >1.25
Original	**9.83**	**5.24**	**68.77**	**43.41**	**1.09**	**0.938**
- MLP	10.04	5.10	68.12	42.99	1.14	0.917
- Softmax + Euclidean	9.98	5.19	68.68	43.16	1.10	0.928
- Softmax + Cosine	9.97	5.21	68.49	43.15	1.12	0.924

Table 2.4: FeatUp (JBU) performance with ablated architectural components: removing the MLP, replacing softmax with a Gaussian kernel w.r.t. Euclidean or cosine distance. Across all metrics, each ablation degrades performance. **Bold** is best.

	FeatUp (JBU)			
	Original	- MLP	- Softmax + Euclidean Dist.	- Softmax + Cosine Dist.
↑ mIoU	44.2	42.9	43.8	43.7
↑ mAcc	55.8	54.7	54.5	55.3
↑ aAcc	80.7	79.4	80.0	80.4

Table 2.5: Semantic segmentation performance with the SegFormer architecture trained on the ADE20k training set and evaluated on their validation set. Ablated FeatUp (JBU) replaces the original feature upsampling in the SegFormer decoder.

2.6.4 Visualising Additional PCA Components

Throughout this chapter, we share the principal component analysis (PCA) using only the three most dominant components, which can be mapped to RGB channel values for visualisation purposes. In Figure 2.16, we demonstrate that it is not only these top-3 components that are upsampled precisely; higher-order principal components also benefit from FeatUp's guided upsampling methodology.

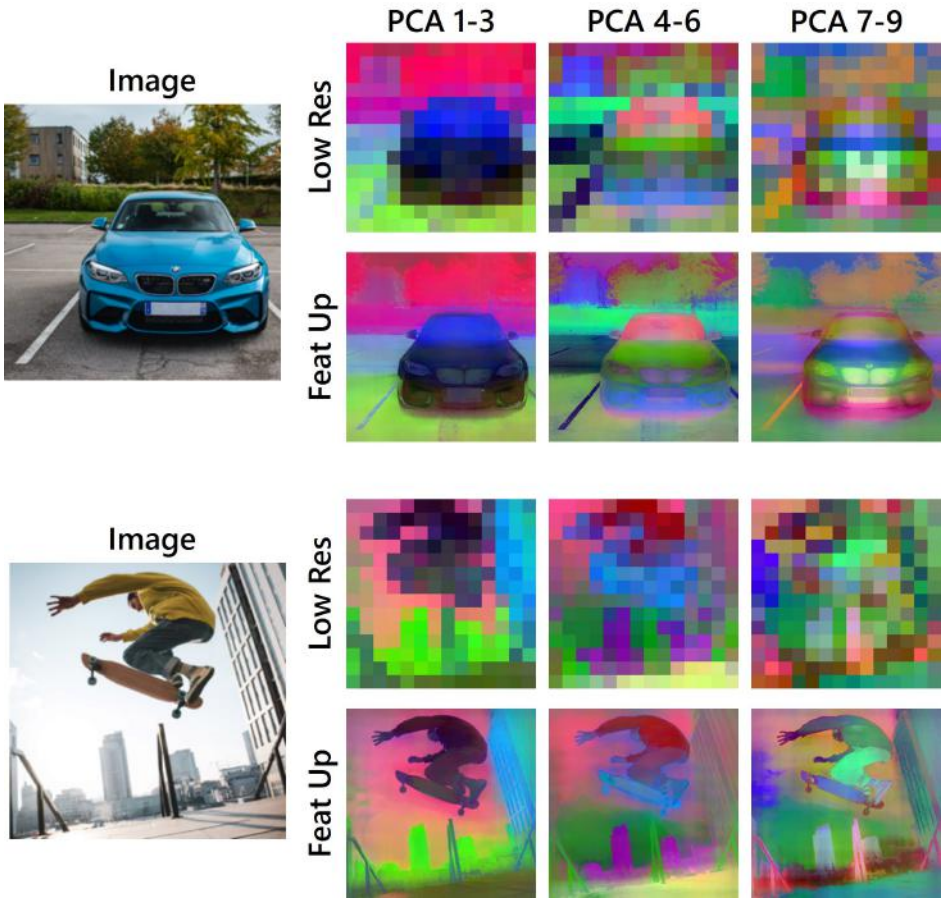

Figure 2.16: Visualising higher PCA components with FeatUp. FeatUp upsamples entire feature maps, so their higher-order principal components also remain in the same space as the original features and are upsampled precisely. Higher components tend to separate more fine-grained object categories like the skater from the skateboard, and the trees from the background, and the clouds from the sky. Note that each subobject's features are upsampled precisely to the object it represents.

2.6.5 Saliency Map Details

Downsampling in FeatUp is analogous to ray-marching in NeRF, which approximates the physics of image formation. FeatUp's downsampler approximates a network's process of pooling information into features. As shown in Figure 2.8, most networks preserve the rough location of objects in their features (the objects just appear downsampled and blurred). This observation leads us to use blur/pooling operators.

The simplest of these is average pooling, but we can do better by generalising this operation to a learned blur/pooling kernel so the downsampler can better match a network's receptive field size. To continue the NeRF analogy, this is like adding learned camera lens distortion parameters to the ray-marcher so NeRF can better fit the data.

As shown in Figure 2.9 and described in Section 2.4.1, even a learned blur/pooling kernel cannot capture dynamic receptive fields or object salience. For example if a small

amount of an important object is in a transformer's patch, the whole feature changes. We capture effects like this by making the learned pool/blur kernel dependent on image content using a 1x1 convolution (we do not need anything bigger than this single layer). This generalises the learned blur/pool and allows the downsampler to pool adaptively based on image content. Figure 2.17 shows that the salience network focuses on certain attributes (*e.g.*, object boundaries, some important small objects). We also note that many common pooling strategies such as average pooling or nearest/bilinear/bicubic resizing are special cases of our learnable attention pooling strategy.

2.6.6 Visualising Downsampler Salience and Kernels

FeatUp's downsampler makes use of a learned, spatially varying 'salience' kernel, in addition to learned, spatially invariant weight and bias kernels. We visualise these in Figure 2.17. The salience is particularly important for our downsampler's performance, because it captures the 'salience' of each pixel to the backbone feature encoder.

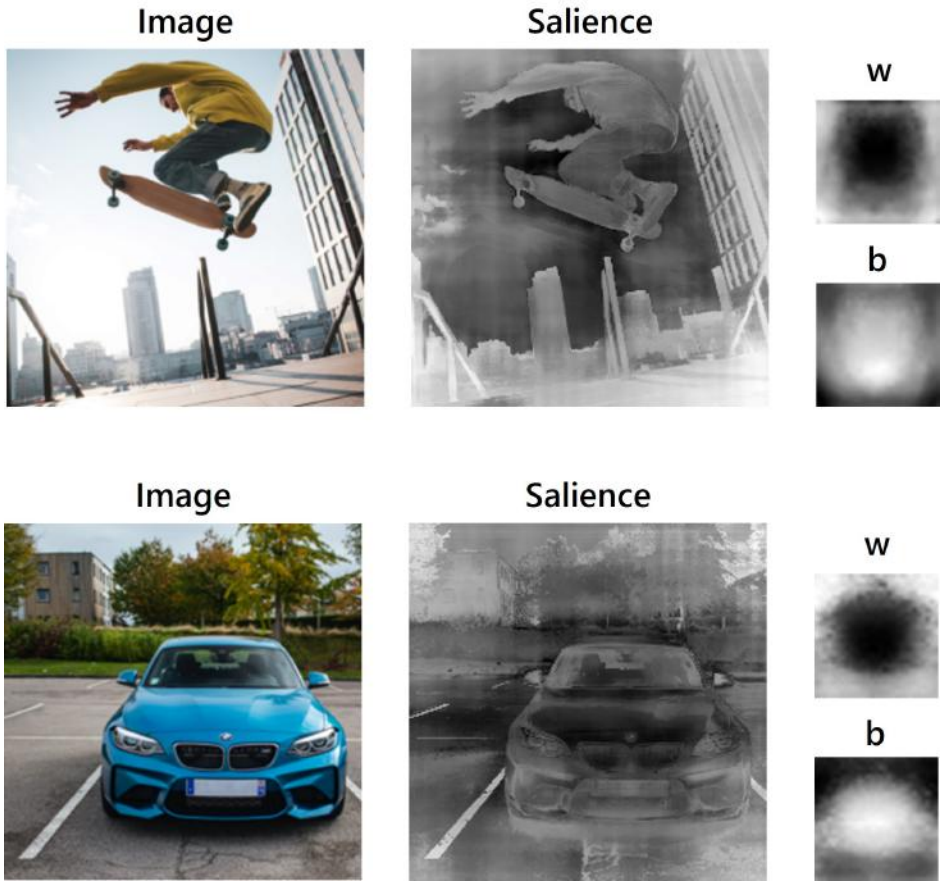

Figure 2.17: Visualisation of downsampler salience, weight, and bias kernels for two images. Note how fine-grained objects have higher salience and regions around important objects (like the sky between the hands and the skateboard) have lower salience. This allows the network to capture nonlinear behavior where embeddings from salient regions dominate the embeddings of other regions.

2.6.7 Visualising Predicted Uncertainty

[96] demonstrates a learned, spatially varying normalisation s that conditions the mean-squared error into a likelihood (Equation 2.1). They label this normalisation s and 'uncertainty', and in Figure 2.18 we share what that spatially varying uncertainty looks like over a map of ViT-encoded features.

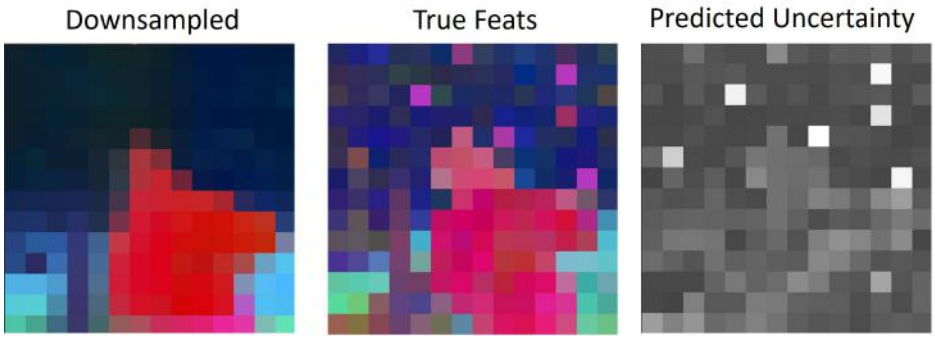

Figure 2.18: An example predicted uncertainty map for a set of ViT features. White areas have higher uncertainty. In this figure, we can see that nonlinear artifacts like the spurious pink tokens are marked with high uncertainty as they change location depending on the given evaluation. These tokens might serve some other role in the network, such as class-token-like information aggregation. We do not see these types of effects in DINO or convolutional networks.

2.6.8 Linear Probe details

In both linear probe tasks, one probe was trained on low-resolution (14x14) features from the COCO training set, and frozen for validation across all methods. FeatUp's performance improvements on this repurposed linear probe show that our methods increase resolution without compromising the original feature space. We highlight that these results are *not meant to improve state-of-the-art (SOTA) performance* on segmentation and depth estimation; they are meant to showcase *feature quality* across upsamplers. Because estimation for both tasks is done with a frozen backbone and a single trainable linear probe, the segmentation and depth maps are not meant for direct application in systems capable of running SOTA estimators. However, as previously mentioned, they show great promise for use in compute-constrained systems!

2.6.9 Average Drop and Average Increase Details

Average Drop is expressed as $\sum_{i=1}^{N} \frac{max(0, Y_i^c - O_i^c)}{Y_i^c} \cdot 100$, where Y_c^i is the classifier's softmax output (*i.e.*, confidence) on sample i for class c, and O_i^c is the classifier's softmax output on the CAM-masked sample i for class c. We generate O_i^c by keeping the top 50% of CAM values (and Gaussian blurring the remaining 50% of values with less explainability power). Though we generally expect classifiers to drop in confidence because even masking out less-salient

pixels can remove important image context, a high-quality CAM will target the explainable regions of an image more precisely and thus maintain a higher confidence. In the reverse direction, we measure the Average Increase to capture the instances where CAM-masked inputs increase model confidence. Specifically, we define Average Increase as $\sum_{i=1}^{N} \frac{\mathbb{1}_{Y_i^c < O_i^c}}{N} \cdot 100$ where $\mathbb{1}_{Y_i^c < O_i^c}$ is an indicator function equal to 1 when $Y_i^c < O_i^c$ (*i.e.*, when model confidence increases upon classifying a CAM-masked image).

Similar to the RelevanceCAM evaluation in [43], we randomly select 2000 images from the ImageNet validation set (limited to images where the label and model prediction match) on which to measure A.D. and A.I.

2.6.10 Additional Linear Probe Results

We provide additional CAM visualisations with supervised ViT features on the ImageNet validation set in Figure 2.19. As in Section 2.5.3, we upsample features from 14x14 to 224x224 output before extracting CAMs (except for the 'Low-Res' column, where the features are kept as-is). Both FeatUp (JBU)'s edge-preserving bilateral filters and the FeatUp (Implicit)'s feature representation allow resulting CAMs to highlight salient regions more accurately. Our CAMs combine the semantic advantages of low-resolution features with the spatial advantages of large images, producing refined versions of the original CAMs without the discontinuous patches present in the other upsampling schemes.

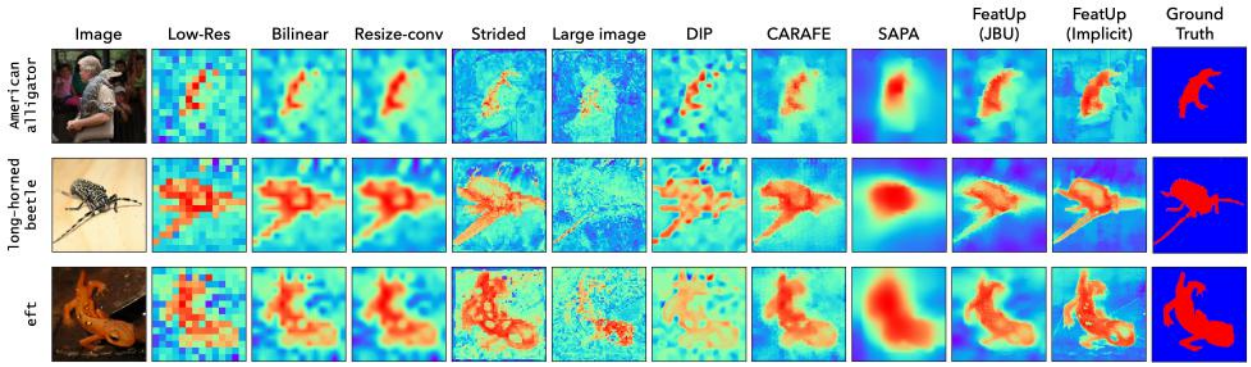

Figure 2.19: CAMs on the ImageNet validation set from a supervised ViT backbone and linear probe classifier. Both FeatUp variants produce features that are more precise with respect to the input image, allowing downstream CAMs to better align with object boundaries.

See Figure 2.20 for examples of linear probe transfer learning for semantic segmentation on the COCO-Stuff dataset. The 14x14 features output from a ViT backbone are upsampled with the following methods to achieve 224x224 resolution. Then, a linear probe is trained on the low-resolution features and frozen for evaluation on COCO-Stuff semantic class labels. Our methods recover more cohesive labels of objects and backgrounds.

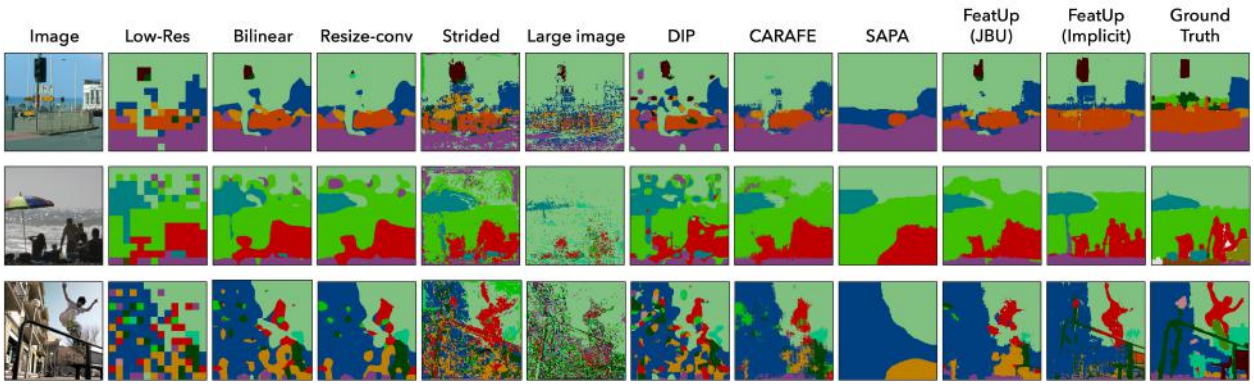

Figure 2.20: Examples of linear probe transfer learning for semantic segmentation on the COCO-Stuff dataset. Our methods more closely resemble ground-truth segmentation and smooth many of the artifacts present in the low-resolution feature results. Additionally, FeatUp (Implicit) recovers thin structures like the umbrella pole, which are not even present in the ground truth despite being semantically correct.

Figure 2.21 provides additional examples of linear probe transfer learning for depth estimation. The 14x14 features output from a ViT backbone are upsampled to achieve 224x224 resolution. Then, a linear probe is trained *directly on the features* to predict depth while supervised by a small MiDaS network. Our results show that both FeatUp variants result in high-quality features capable of transfer learning.

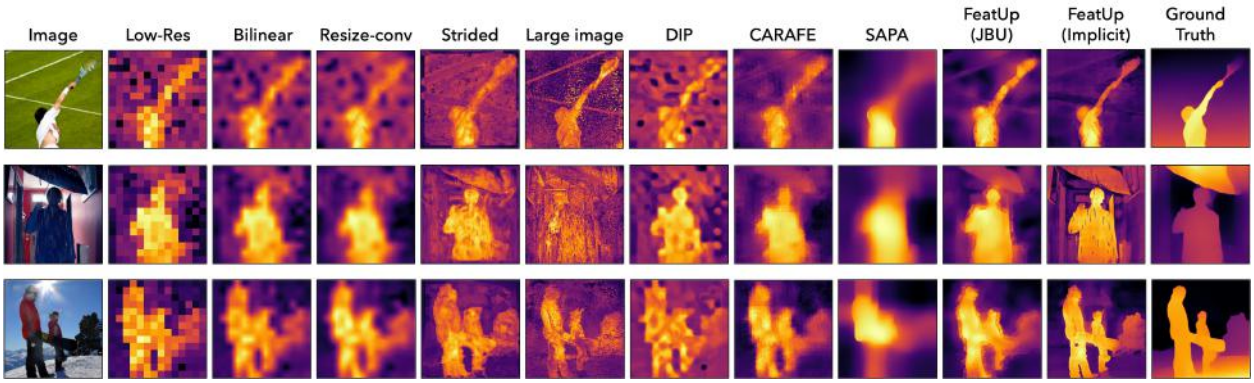

Figure 2.21: Examples of linear probe transfer learning for depth estimation. Our methods produce sharper object boundaries and smoother interiors that more closely align with true depth than other methods.

2.6.11 Improving Image Retrieval for Small Objects

FeatUp can be used to improve not only linear probe transfer-learning and end-to-end training (Sections 2.5 and 2.6.10), but also other feature-based tasks like point-query object retrieval (Figure 2.22). FeatUp's improved object-alignment enables improved localisation of similar objects in different image samples.

Figure 2.22: FeatUp can be used to improve the retrieval of small objects in cluttered scenes. A query image (left) is featurised with DINO and a query point is selected. In the two images on the right, we compare the pixel similarity (the overlaid heatmap — red is similar, blue is dissimilar) and object retrieval (the ×) produced by bilinear- vs. FeatUp-based algorithms. Because the scene is cluttered, bilinear interpolation blurs object features together and the resulting retrieval vector does not discriminate between ground and cones. FeatUp's features better align with objects, allowing only the traffic cones to be retrieved.

2.6.12 Performance Benchmarking

We evaluate how various factors like spatial dimension, upsampling factor, and feature dimension impact the performance of our adaptive convolution CUDA kernal used in FeatUp (JBU). See Figure 2.23 for performance benchmarking for the full FeatUp algorithm compared with baselines, and Table 2.6 for performance benchmarking of our custom CUDA implementation compared with PyTorch- and TorchScript-based implementations

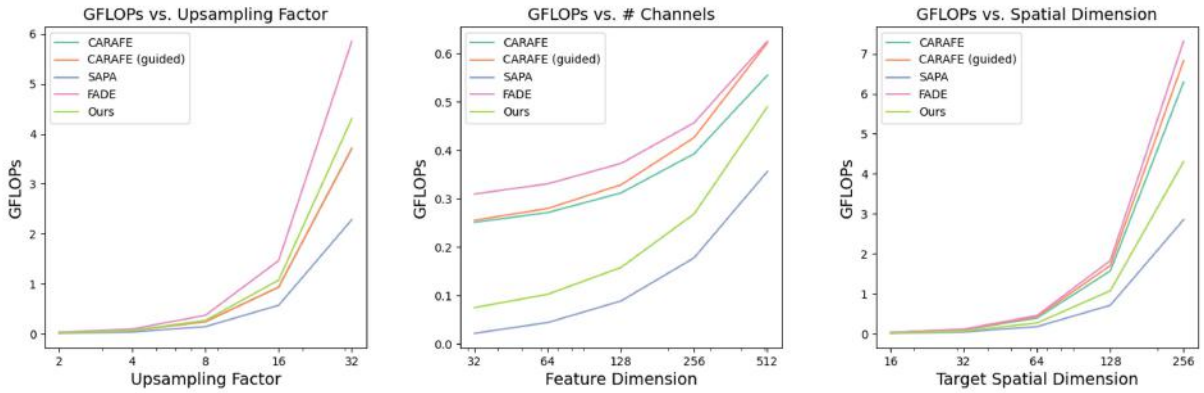

Figure 2.23: In varying the upsampling factor, feature dimension, and target spatial dimension, FeatUp (JBU) remains competitive in GFLOP usage. For each experiment, the attributes not studied are kept constant (upsampling factor = 2, feature dimension = 256, starting spatial dimension = 8x8).

We analyse peak memory usage and inference time for various upsampling methods. Specifically, we upsample ViT features from a $(1 \times 3 \times 224 \times 224)$ image (*i.e.*, low-resolution

Shape (B, H, W, C, F)	Method	↓ Forward (ms)	↓ Backward (ms)	↓ Peak Mem (Mb)
$1 \times 14 \times 14 \times 2048 \times 5$	Ours	**0.15**	**1.05**	**6.24**
	TorchScript	2455	69367	12.8
	Unfold	3.30	2.81	119.0
$1 \times 512 \times 512 \times 3 \times 5$	Ours	**0.55**	**2.10**	**10.2**
	TorchScript	147.0	520.0	24.3
	Unfold	3.47	4.85	231.0
$16 \times 32 \times 32 \times 2048 \times 5$	Ours	**8.43**	**90.8**	**372.**
	Unfold	118.0	218.0	6628.0
$32 \times 512 \times 512 \times 3 \times 5$	Ours	**17.7**	114.0	**326.**
	Unfold	36.0	**104.**	4901.0
$64 \times 14 \times 14 \times 2048 \times 5$	Ours	**6.12**	**61.1**	**400.**
	Unfold	57.5	170.0	5174.0
$64 \times 224 \times 224 \times 3 \times 5$	Ours	**6.27**	36.1	**128.**
	Unfold	16.7	**27.4**	1878.0
$64 \times 64 \times 64 \times 16 \times 5$	Ours	**1.06**	**8.99**	**44.5**
	Unfold	7.18	14.5	822.0
$64 \times 64 \times 64 \times 16 \times 7$	Ours	**2.00**	**8.36**	**52.6**
	Unfold	10.8	25.6	1596.0

Table 2.6: Comparing the performance of our CUDA JBU kernel with implementations based on PyTorch's `Unfold` operation and TorchScript. Our implementation dramatically reduces memory overhead and increases inference speed. Code for this operation is available in the provided link. **Bold** is best.

feature dimensions of $(1 \times 384 \times 14 \times 14)$) by factors of 2, 4, 8, and 16. Figure 2.24 shows that FeatUp (JBU)'s peak memory closely follows resize-conv and SAPA baselines and outperforms CARAFE. Additionally, FeatUp is as fast as yet outperforms baselines in all of our quantitative evaluations. We note that strided and large image baselines become computationally infeasible after 8× upsampling, even using a batch size of 1.

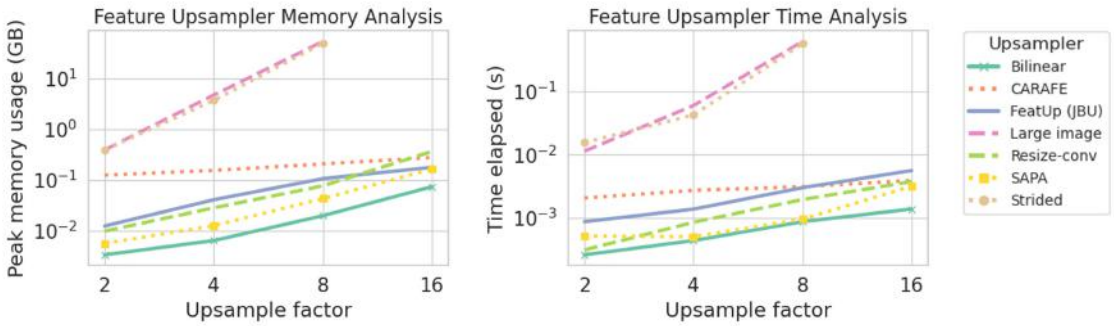

Figure 2.24: Analysis of peak memory usage (left) and inference time (right) for various forward-pass upsamplers. FeatUp (JBU) is competitive with SAPA and resize-conv across upsampling factors and is more efficient than CARAFE for smaller factors. The large image and strided approaches become infeasible at large upsampling factors, so we only show metrics for these methods up to 8× upsampling.

2.6.13 Limitations

The two FeatUp variants exhibit different limitations. More specifically, we find that Implicit FeatUp's NeRF-like training paradigm allows it to compute fine detail with high fidelity, sometime even exceeding the fidelity of the 'truth'. However, it can suffer from 'halo' effects near object boundaries that arise from the model not quite managing to enforce inter-sample consistency. Meanwhile, JBU FeatUp does not capture detail as finely.

For both variants, FeatUp's heavy usage of the input RGB image as guidance can lead to textures that have nothing to do with semantic content being injected into the upsampled features. Its reliance on the input also means that FeatUp simply cannot function in a system without access to the original image data.

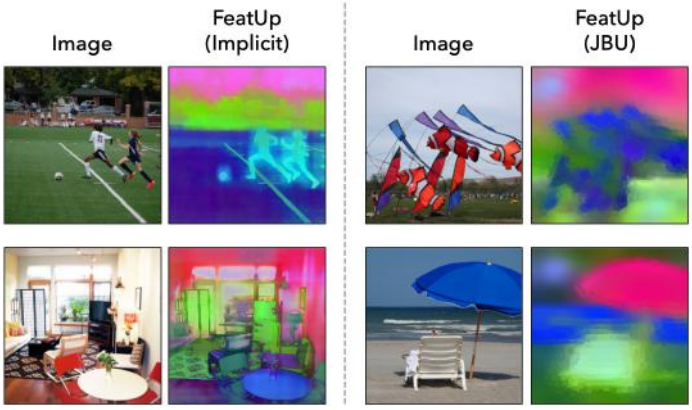

Figure 2.25: Left: Though FeatUp's implicit network can capture fine detail such as the soccer ball or window frame, it can still produce some halo effects (see soccer player). Additionally, because the method relies on the input image's spatial signal, certain patterns unrelated to object semantics can be transferred to the feature map (see rug pattern), though this is a rare occurrence. Right: FeatUp's JBU network is not as sensitive to fine detail as the implicit network, instead capturing broader contours.

2.6.14 Implementation Details

All backbones (DINO, DINOv2, ViT, ResNet-50, CLIP, and DeepLabV3) used to train FeatUp are frozen, pre-trained models obtained from the community. We outline the hyperparameters used to train FeatUp in table 2.7.

Hyperparameter	**FeatUp (Implicit)**	**FeatUp (JBU)**
Num Images	1	4
Num Jitters Per Image	10	2
Downsampler	Attention	Attention
Optimiser	NAdam	NAdam
Learning Rate	0.001	0.001
Image Load Size	224	224
Projection Dim	128	30
Training Steps	2000	2000
Max Transform Padding	30px	30px
Max Transform Zoom	1.8×	2×
Kernel Size	29	16
Total Variation Weight	0.05	0.0
Implicit Net Layers	3	n/a
Implicit Net Dropout	0.1	n/a
Implicit Net Activation	ReLU	n/a

Table 2.7: Hyperparameters used in training FeatUp.

.

2.6.15 Website, Video, and Code

We provide additional details and a short video explaining FeatUp at https://aka.ms/featup. Additionally, we provide our code at: https://tinyurl.com/28h3yppa.

2.7 Conclusion

FeatUp is a fast, lightweight, and guided method for upsampling features extracted from *any* backbone architecture. It can be used to improve the performance of linear probes trained for *any* vision task that takes as input a feature embedding, by upsampling the embedding as a preprocessing step.

Mobile systems like the Husky and Spot in Figure 1.6 are currently limited to low-performing vision models like YOLOv3 Tiny [9], but our work with FeatUp demonstrates that improved feature upsampling opens the door to the future development of multi-task models based on a single backbone with several linear probes attached (one per task). These would operate with a drastically reduced number of parameters compared with the several entirely separate vision models that must currently be trained and stored if one wishes to accomplish several vision tasks on a single platform (Figure 2.26).

More specifically, if M is the number of parameters in a backbone, $k \ll M$ is the number of parameters in each linear probe (to simplify, we take them all to be the same size regardless of task), and N is the number of tasks being undertaken, we can hypothetically expect to see a reduction from $O(NM)$ parameters to $O(M + kN) \to O(M)$ parameters.

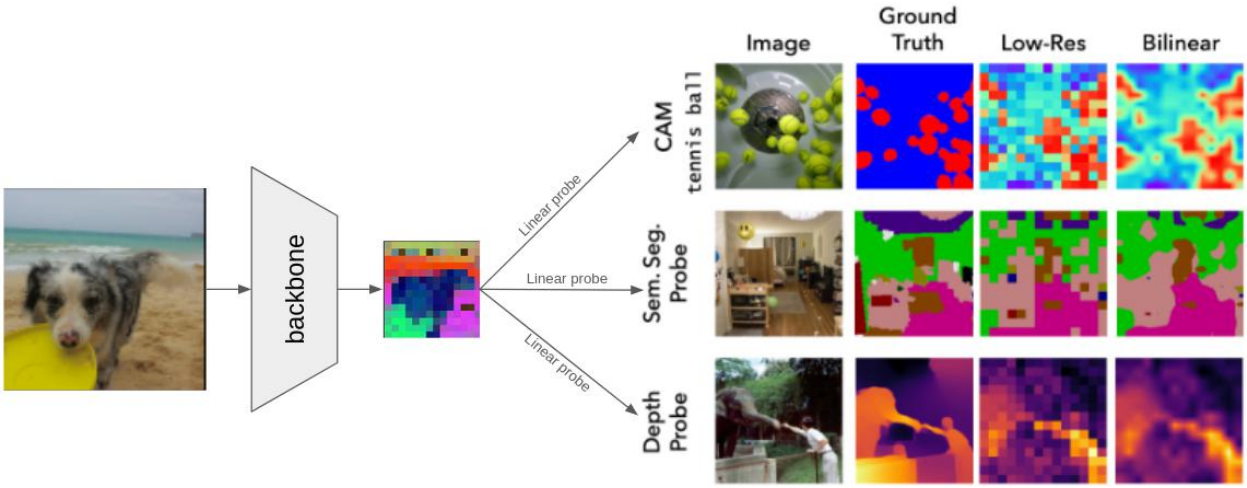

Figure 2.26: FeatUp's improved ability to upsample features opens the door for improved transfer learning from a single M-parameter backbone to a multitude of tasks via $k \ll M$-parameter linear probes [16]. This could drastically reduce the number of parameters required to simultaneously perform N visual tasks at once from $O(NM)$ to $O(M)$.

So it appears that FeatUp may be 'the' (or, at least, 'a') solution to the tricky challenge of vision for robots in compute-constrained environments. But how do we select the 'right data' with which to train these backbones and linear probes? We cannot always rely on synthesis, as I did in my master's thesis. And unlike with large models like SAM, we cannot always expect bounteous access to data. We need a smarter way of selecting the 'right data' to train with, and it is this challenge to which we turn in the next chapter.

Chapter 3

DiffUnc: Diffusion-based Uncertainty Estimation for Anomaly Detection

This work is in-progress, aiming for a June 2024 submission, and was pursued in collaboration with Sunshine Jiang, Dr. Siddharth Ancha, and Prof. Nicholas Roy [17]. Some background material is also copied with permission from [113].

How do we choose the 'right data' for training and finetuning vision models? We could take the approach of the Segment Anything Model (SAM) and train on millions of high-resolution images and billions of high-quality masks [3]. But in data-constrained environments (*e.g.*, marine, subterranean, nonterranean, and nighttime/low lighting) we are often limited by having little data to work with; and what data we *do* have is often low quality — lacking, for example, in resolution and/or contrast. Models like SAM can struggle to generalise to such environments, both because of this degradation in image quality, *and* because the environments themselves are often out-of-distribution, even relative to the massive dataset used to train SAM [114][115].

Humans — at least once they have crossed some threshold of competency — have a remarkable ability to recognise their own uncertainty when faced with new items and environments [116][pp. 112-130, 222-227 of [19]], and this is critical to our ability both to learn entirely new concepts, and to transfer what we know about familiar topics to sights unseen. More specifically, it helps us direct our cognitive attention appropriately toward closer examination or consultation with others [pp. 203-221 of [19]] (Figure 3.1). In many situations (*e.g.*, unfamiliar beasts), it allows us to know when we should activate our flight mechanism and **run**.

Visual anomaly detection is an increasingly important learning task that enables the vision and robotics communities to predict, detect, and prevent model failure. However, current methods for estimating uncertainty and predicting anomalous performance are often uncalibrated and unlocalised. In this chapter, I discuss DiffUnc, a diffusion-based method for estimating and *localising* meaningful uncertainty. I will also present two improvements to existing guided diffusion methods: a new approximation for the energy term, and an alternative penalty function. We make use of both improvements in our final results.

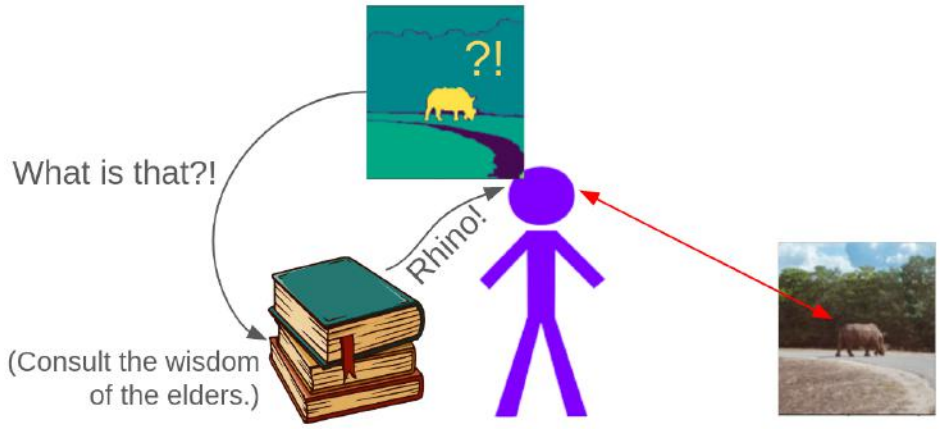

Figure 3.1: Humans learn to recognise their own uncertainty about what they see, and use it to choose which data to pay attention to and train on [19]. Can we leverage a similar principle to help machines choose the 'right data' more intelligently?

Our method produces uncertainty estimates at the pixel- and mask-level, enabling anomaly detection at both resolutions (Figure 3.2). We show that DiffUnc performs well on existing benchmarks for anomaly detection, and that its uncertainty estimates are meaningful and human-interpretable. Finally, we demonstrate that DiffUnc — especially in conjunction with our SAM-based mask-level uncertainty extension — excels at picking out tiny and camouflaged anomalies, which are often undetectable to the human eye.

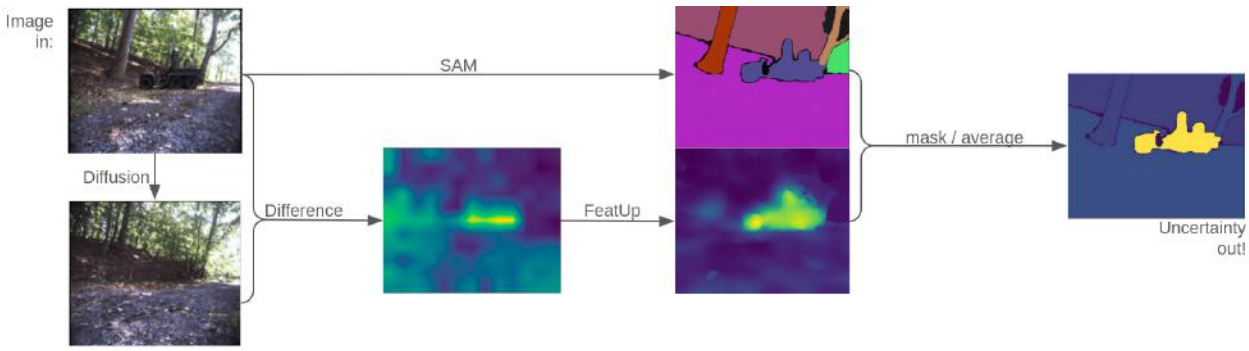

Figure 3.2: DiffUnc estimates pixel- and mask- level uncertainty using the output of a guided diffusion process. By taking the difference between the synthesised output and the original image target, we can get a localised estimate of uncertainty. The challenge is two-fold — first, the diffusion process must enforce the 'right' kind of consistency between output and target; and second, we must take the 'right' kind of difference.

3.1 Introduction

Autonomous robotic navigation has become increasingly pervasive in both data-rich (often urban) environments [117]–[119] as well as data-constrained (often off-road) environments like planetary exploration [120][121], search-and-rescue [122], mines [123] and forests [124]. Navigation systems rely on semantic segmentation of camera images [125]–[127] to detect various semantic types and object classes. Robots operating in such real world environments often face 'out-of-distribution' (OOD) obstacles that are not well-represented in the training data. However, most deep-learning based semantic segmentation models not only make unexpected errors on OOD examples, but they often have no notion of how incorrect their estimates are.

In order to navigate safely and reliably in unfamiliar environments, autonomous robots must *detect anomalies* and *estimate uncertainty* in the visual data they encounter, in order to anticipate potential errors (Figure 3.3) [128]. Uncertainty in general arises from two sources [129]: 'aleatoric' uncertainty is the inherent and irreducible uncertainty due to sensor/label noise, partial observability, environmental dynamics, or similar; whereas 'epistemic' — or model — uncertainty is the hypothetically reducible [129] uncertainty in the modelling algorithm. In neural network-based perception models, epistemic uncertainty is often assumed to be due to unfamiliar inputs not being well-represented in the training dataset [130]. However, this is an oversimplification — epistemic uncertainty can also stem from incorrect assumptions, imprecise problem formulation, bad architectural choices, *etc.*

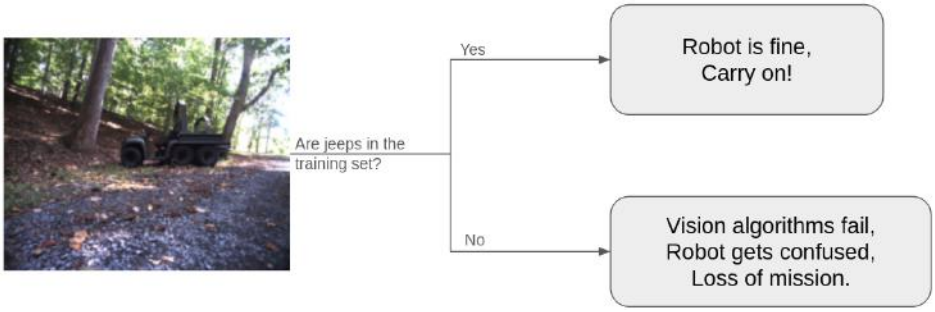

Figure 3.3: When a robot trained on pure forest data (Figure 3.13) and deployed to a wooded region encounters a jeep, one predictor of potential mission failure is the answer to the question 'Were jeeps in the training set?' If the answer is 'No', our robot's vision algorithms are quite likely to fail. If, however, we could accurately estimate the degree to which this jeep image is 'out-of-distribution' (*i.e.*, its epistemic uncertainty), then the robot could *recognise* its own high probability of failure, and could appropriately notify a remote operator or independently make more cautious plans, thus averting loss of mission.

Under this assumption, if one can obtain a measure of how 'out-of-distribution' a sample is relative to the training data, one has obtained an estimate of the model — or 'epistemic' — uncertainty. Measuring the 'OODness' of a sample *also* enables us to flag unusual samples as anomalies for anomaly detection. Our goal in this work is therefore improved estimation of epistemic uncertainty, with the application of anomaly/failure detection firmly in mind.

Uncertainty estimation for deep neural networks is a field of active research and interest in the robotics community [92][128][131]–[136]. Bayesian neural networks [18][137]–[140] provide a principled framework to estimate *both* uncertainties for an input $\mathbf{x}$ by not just predicting a single categorical distribution $\mathbf{p} = (p_1, \ldots, p_C \,|\, \sum_{c=1}^{C} p_c = 1)$ over C classes, but by predicting a hierarchical 'distribution over distributions' $p(\mathbf{p} \mid \mathbf{x})$. However, conventional Bayesian uncertainty estimators like variational inference [131][141][142], ensembles [134][143]–[146], MC-Dropout [128], and test-time augmentation [147][148] require multiple forward passes through a neural network and are prohibitively slow for real-time robotics applications. More recently, 'evidential' uncertainty [132][133][149]–[152] estimators such as the natural posterior network (NatPN) [113][153][154] have emerged as more efficient alternatives that model uncertainty as a Dirichlet distribution [155] and directly predict its parameters.

These Bayesian and evidential approaches have been very successful, but generally produce only image-level uncertainty estimates, sufficient to flag an entire image as anomalous, but often insufficient for a robot's remote supervisor to determine what should be done to resolve the uncertainty (Figure 3.4). To be useful, uncertainty estimates should be localised. Additionally, uncertainty estimates are rarely meaningful and reliable [129], an issue that the community has tried to address via calibration measures and re-calibration methods [156]–[158]. What we really need for anomaly detection, therefore, are *meaningful and localised* estimates of epistemic uncertainty.

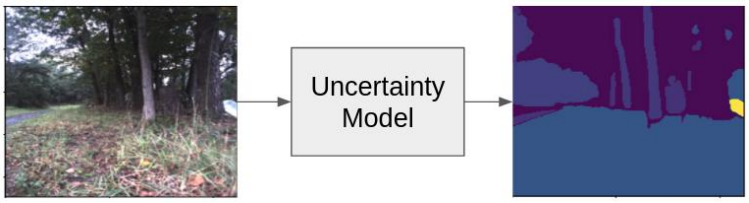

Figure 3.4: Flagging an entire image as anomalous or high uncertainty is often insufficient, because a remote human supervisor cannot always tell without guidance which part of an image is problematic. In this case, for example, the problematic element is the small silver-white patch at the right edge of the input image (left). Without being shown the *localised* uncertainty (right), the supervisor might not find the anomaly.

A variety of approaches for localising uncertainty and detecting anomalies have been attempted (Figure 3.5) [159], including dimensionality reduction approaches like principal component analysis (PCA) [160]–[162] and random projections [163]–[165], novelty functions [166][167], and feature-space comparison of neural embeddings [168]–[171]. More recently, the pixel-by-pixel evidential uncertainty estimation and anomaly detection-by-synthesis approaches have shown particular promise.

The pixel-by-pixel evidential approach is simply to take an evidential method and apply it to each input image pixel x_{ij}, individually, to estimate the likelihood $p(y_{ij} \,|\, x_{ij}, \mathcal{D})$, which is inversely related to the pixel's epistemic uncertainty. Such a method is usually task-specific; for example, the NatPN-based approach of [113] is specifically for semantic segmentation, and y_{ij} indicates the ijth pixel's class. This sort of approach yields results as shown on the left of Figure 3.5, and often fails to detect the entirety of the anomalous object.

Meanwhile, anomaly detection by synthesis has become a popular approach in which an autoencoder-type architecture $G(\cdot)$ [172]–[176] is trained on anomaly-free data $\mathcal{D}_{id}$ [177][178]. Anomaly detection can then be performed by calculating the difference $\text{Diff}(\mathbf{x}, \mathbf{x}')$ between an inference-time input $\mathbf{x}$ and whatever output is reconstructed by $\mathbf{x}' = G(\mathbf{x})$. This approach assumes that the autoencoder will *not* be able to accurately reproduce anomalies because of the priors learned from its anomaly-free training data, so that $G(\mathbf{x})$ is effectively a projection of $\mathbf{x}$ onto the training manifoled $\mathcal{D}_{id}$, and that computing such a difference will yield hotspots in anomalous regions. Recently, generative adversarial network (GAN)- based approaches have shown promise and become increasingly common [179]–[183]. However, these approaches often have high computational requirements, in addition to high false-positive rates (as in, *e.g.*, the rightmost example in Figure 3.5) [159] stemming from the empirical fact that, in practice, GANs often generalise well to anomalies and therefore reconstruct them sufficiently well that detection capability is reduced [177].

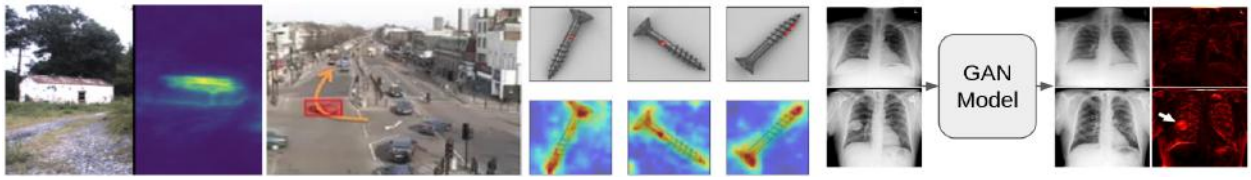

Figure 3.5: Examples of prior attempts to localise uncertainty and detect anomalies, which have included applying evidential methods pixel-by-pixel [113], using neural networks to directly estimate bounding boxes or heatmaps [184][185], and using generative models to project the potentially-anomalous image to its nearest anomaly-free neighbour and computing their difference [179].

DiffUnc is an evidential anomaly-by-synthesis approach, which combines these two promising strategies (Figures 3.2). Specifically, we use a diffusion model [186] — which is essentially a stochastic NatPN [154] — to model our in-distribution (ID) targets $\mathcal{D}_{id}$. By 'guiding' our diffusion process [187], we can enforce consistency between generated images $\mathbf{x}' \equiv G(\mathbf{x})$ and input images $\mathbf{x}$, effectively training the diffusion model to remove anomalies. We can then take the epistemic pixel-level uncertainty to be the difference between input and reconstruction, $\mathbf{u} \equiv \text{Diff}(\mathbf{x}, \mathbf{x}')$. Our premise is that diffusion models are an evidential and justifiable method for modelling distributions and image-level uncertainties [186][188], and that guided diffusion has been empirically shown to be an extremely good generative model [40][187]. Our hypothesis is that this makes guided diffusion a justifiably and empirically good choice of generative model for the anomaly-by-synthesis approach.

Selecting a good $\text{Diff}(\cdot, \cdot)$ operator and guiding the diffusion model is non-trivial. To make DiffUnc a success, we develop and share the following contributions:

1. A new energy function for guiding our diffusion process, 'SoftRect', that is forgiving of small errors but punishes large ones harshly.

2. Empirical evidence and supporting analysis for our choice of the $\text{Diff}(\cdot, \cdot)$ operator to be a cosine-difference between encoded DINO features [27], [189]. We compare this metric with several other difference metrics, including straightforward image-space comparisons and alternative perceptual metrics.

3. FeatUp- and SAM-based extensions to our work, which yield the crisp uncertainty localisation and anomaly detection results shown in Figure 3.2.

We find that including FeatUp [16] in our pipeline greatly improves both TPR and FPR; *i.e.*, our pixel-level uncertainty estimates more thoroughly highlight anomalies while suppressing false positives. Including SAM [3] greatly improves our detection of very small and camouflaged anomalies. We validate our uncertainty estimation approach on the off-road Robot Unstructured Ground Driving (RUGD) , and on the urban Cityscapes dataset [190][191]. We evaluate it on the Fishyscapes and Segment Me If You Can (SMIYC) anomaly detection benchmarks [192][193].

DiffUnc additionally yields an image-level evidential uncertainty score that 'falls out' of the diffusion process [188], and an image-level error estimate computable by averaging pixel-level results, making DiffUnc a good choice for applications where hierarchical uncertainty estimation or anomaly detection is desired.

3.1.1 Problem statement

Given an 8-bit, 3-channel RGB-image sample $\mathbf{x}$ of shape $H \times W \times 3$ that is encountered by a deployed mobile robot, and the dataset $\mathcal{D}_{id}$ of in-distribution images previously encountered by this robot's vision systems during training, we want to estimate an uncertainty map $\mathbf{u}(\mathbf{x}|\mathcal{D}_{id})$ of shape $H \times W \times 1$. We aim with this map to estimate the degree to which each sample pixel x_{ij} is out-of-distribution relative to $\mathcal{D}_{id}$, for all $i \in [0, H)$ and $j \in [0, W)$. We formulate the map as $\mathbf{u}(\mathbf{x}) = \text{Diff}(\mathbf{x}, G(\mathbf{x}))$, where $G(\mathbf{x}) \equiv \mathbf{x}'$ is a projection $G(\cdot)$ of the image $\mathbf{x}$ onto the in-distribution manifold of $\mathcal{D}_{id}$, and $\text{Diff}(\cdot, \cdot)$ is some difference operator. Our goal, therefore, is to identify a projection $G(\cdot)$ and difference $\text{Diff}(\cdot, \cdot)$ such that the resulting uncertainty u_{ij} are minimal for $x_{ij} \sim \mathcal{D}_{id}$, and maximal otherwise.

Additionally, given the image sample $\mathbf{x}$ and the dataset $\mathcal{D}_{id}$, we aim to estimate a set of uncertainty-labelled object masks $\underline{\mathbf{u}}(\mathbf{x}|\mathcal{D}_{id})$.

3.2 Methods

The premise behind DiffUnc is that diffusion models are a justifiable method for modelling distributions and image-level uncertainties [186][188], and that it should therefore be a good choice for our operator $G(\cdot)$ as part of our pipeline for extracting mask- and pixel- level uncertainties. We provide an overview of our diffusion-based uncertainty estimation pipeline in Figure 3.6.

The first step in our pipeline is to use a guided diffusion process as a pseudo-autoencoder, by guiding its output $\mathbf{x}'$ toward consistency with its "input" image $\mathbf{x}$. This input $\mathbf{x}$ is the RGB image collected by the vision model at inference time , and may or may not contain any anomalies. Through a careful choice of the guidance signal (Section 3.2.1), we can effectively train this pseudo-autoencoder to remove any present anomalies from $\mathbf{x}$.

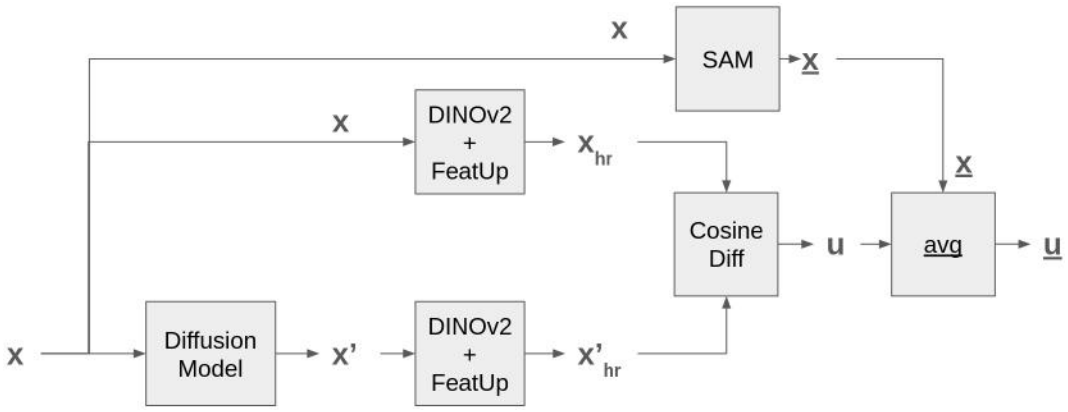

Figure 3.6: DiffUnc is an evidential anomaly-by-synthesis approach to localised uncertainty estimation and anomaly detection. In our full pipeline, the guided diffusion models removes anomalies from the input image, $\mathbf{x}$, to yield $\mathbf{x}'$. We then compute the pixel-wise uncertainty map $\mathbf{u}$ to be the difference $\text{Diff}(\mathbf{x}, \mathbf{x}')$ between the two. This yields a heatmap, which can then be converted to constant-uncertainty masks $\underline{\mathbf{u}}$ by averaging $\mathbf{u}$ within SAM-segmented masks $\underline{\mathbf{x}}$.

Next, we compute the cosine-difference between the upsampled DINOv2 feature embeddings [194] of the input $\mathbf{x}$ and anomaly-free $\mathbf{x}'$. This difference yields a measure of 'perceptual difference', which measures how similar two images are in a way that (a) takes into account pixel interdependence (*e.g.*, due to structure), and (b) coincides with human judgment [195]. This choice of difference metric results from extensive comparisons with a variety of alternative image- and latent/perceptual- space difference metrics (Section 3.2.2). We select FeatUp (Chapter 2) as our upsampling method.

The cosine-difference of the upsamples feature maps yields our pixel-level estimate of epistemic uncertainty, $\mathbf{u} \equiv \text{CosDiff}(F(\mathbf{x}), F(\mathbf{x}'))$, where $F(\cdot) \equiv \text{FeatUp}(\text{DINOv2}(\cdot))$. We use the Segment Anything Model (SAM) [3] to segment the input images $\mathbf{x}$, and get mask-level uncertainties $\underline{\mathbf{u}}$ by averaging the pixel-level uncertainties $\mathbf{u}$ within each segment. Finally, an image-level uncertainty $u = \text{Avg}(\mathbf{u})$ is obtained by taking the mean over pixel-level estimates. We make use of this image-level uncertainty in the proof-of-concept Active Labeller discussed in Section 4.1.

Altogether, DiffUnc produces a set of epistemic uncertainty estimates $\{\mathbf{u}, \underline{\mathbf{u}}, u\}$ for use in anomaly detection and other downstream tasks.

3.2.1 Improving Guided Diffusion for Anomaly Removal

Diffusion models [186][187] are trained and operated in two stages. The 'forward diffusion' process iteratively 'diffuses' noise through a target image $\mathbf{x}$ by progressively adding noise that is sampled from a Gaussian distribution (Figure 3.7). The generative 'reverse diffusion' process, which is often what the community refers to in short-hand as 'diffusion', is a learned denoising process that at each denoising (or time) step $t \in [0, T]$ takes a step $t \to t-1$ via gradient descent on the negative log-probability $-\log p(\mathbf{x}'_t)$, where $\mathbf{x}'_t$ is the image generated by step t. In other words, reverse diffusion aims to maximise the probability $p(\mathbf{x}'_{t-1})$ based on the information it has at the current step t.

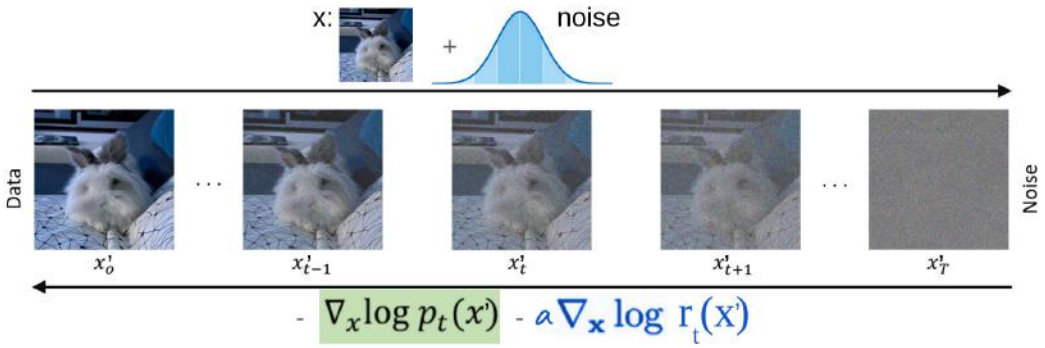

Figure 3.7: Diffusion models have a 'forward' and a 'reverse' diffusion process [186][187]. The forward process progressively adds Gaussian noise to an image. The reverse process learns to iteratively denoise from total noise $\mathbf{x}'_T$ to a realistic, in-distribution image $\mathbf{x}'_0$. If a guidance energy $r(\mathbf{x}'_t)$ is included in the gradient descent, then consistency between the denoised images $\mathbf{x}'_t$ and the original image $\mathbf{x}$ can be encouraged. See Figure 3.22 for the effect of tuning the guidance strength α.

Like other learned generative models, diffusion processes model $p(\cdot)$ based on the distribution of the dataset they are exposed to at training time — *e.g.*, a diffusion model can be overtrained to a single image, in which case the generative process will always produce that same image; or the model can be trained on a dataset of bunny rabbits, in which case the process will very likely produce an image recognisable as a bunny rabbit [186]. For each denoising step t, the negative log-probability $-\log p(\mathbf{x}'_t)$ can be thought of as an uncertainty score [188].

In guided diffusion the reverse-process that steps from $t+1 \to t$ aims to *additionally* minimise the negative log-likelihood $-\log p(\mathbf{x}'|\mathbf{x}'_t)$ of the final output $\mathbf{x}'$ given the current denoised estimate $\mathbf{x}'_t$. In practice, the gradient of this likelihood is unknown or difficult to compute, and so we approximate the likelihood with an energy function $r(\mathbf{x}, \mathbf{x}'_t)$ [187].

A common choice for the energy function $r(\cdot,\cdot)$ is the L2 error, but we instead define a 'SoftRect' function $\mathrm{SR}(\mathbf{x}, \mathbf{x}') \equiv \sigma(\beta(\mathbf{x}-\mathbf{x}')-\epsilon)+\sigma(\epsilon-\beta(\mathbf{x}-\mathbf{x}'))$, where $\sigma(\cdot)$ is the sigmoidal function. SoftRect is effectively a differentiable rectangle function, whose 'boxiness' and width can be tuned by modifying β and ϵ, respectively (Figure 3.8). Our improved diffusion process therefore aims to minimise $-\log p(\mathbf{x}'_t) - \log \mathrm{SR}(\mathbf{x}, \mathbf{x}'_t)$.

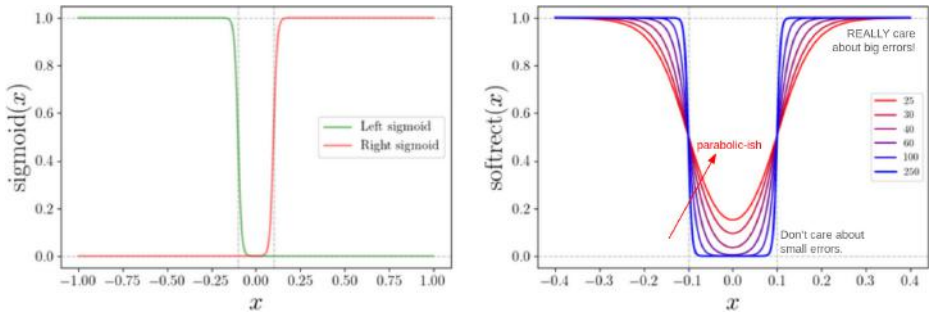

Figure 3.8: The SoftRect energy function is a sum of sigmoids and has a tunable width and boxiness. The boxier it is, the more forgiving it is of errors $\mathbf{x}-\mathbf{x}' < \epsilon$, and the harsher it is toward errors $> \epsilon$.

3.2.2 Choosing the Difference Metric

Selecting the Diff(·) operator is non-trivial. One might imagine from comparing the input and output images of our diffusion model, that taking a simple pixel-wise L2 difference would be adequate to obtain a meaningful uncertainty map with hotspots highlighting any anomalies and coolspots everywhere else. However, actually trying this approach yields a noisy uncertainty map, with low-intensity hotspots distributed across the field of view (Figure 3.9, top).

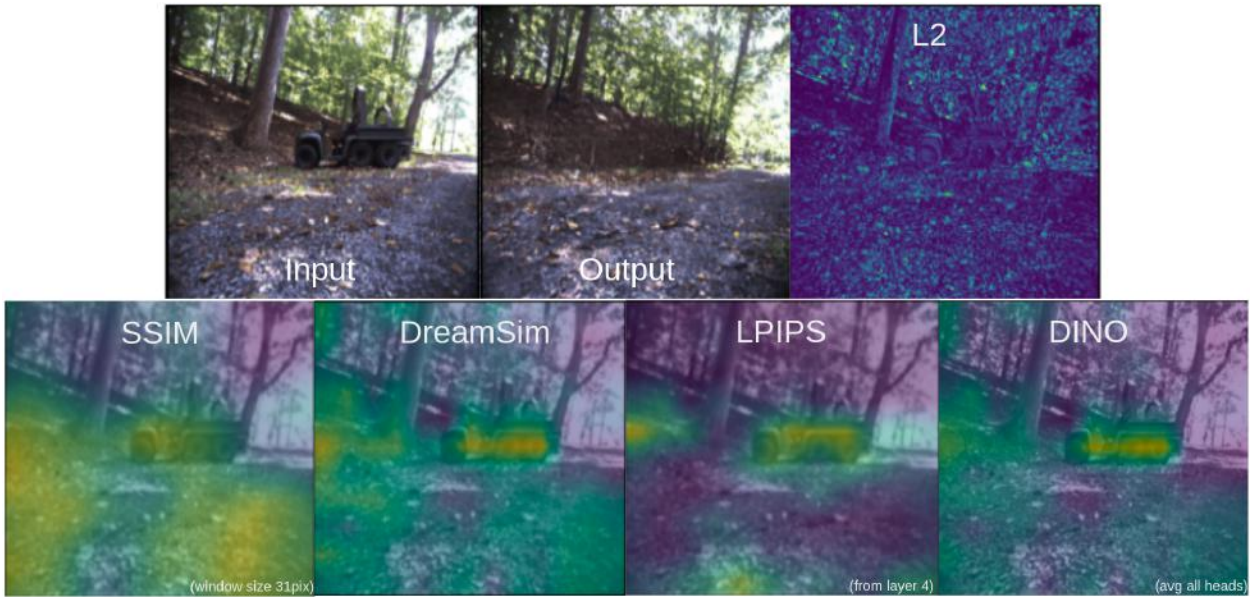

Figure 3.9: A qualitative comparison of the pixel-level uncertainty estimates yielded by different choices of Diff($\mathbf{x}, \mathbf{x}'$) operator. We display these results using bilinear rather than FeatUp upsampling. At top-left we show the anomalous input $\mathbf{x}$ and anomaly-free output $\mathbf{x}'$. DINO does best at highlighting the jeep anomaly with a high peak signal-noise ratio (PSNR), in comparison with other methods which have substantial false-positive hotspots with a PSNR ≤ 1.

The reason that using image-space difference metrics does not work well for comparing a diffusion model's output $\mathbf{x}'$ with its input $\mathbf{x}$, is that even with our improved diffusion guidance, there are many reasonable, small deltas in image intensity that are still considered in-distribution by the diffusion process. Our insight in moving forward is that the difference between $\mathbf{x}$ and $\mathbf{x}'$ that we wish to capture with our uncertainty estimation and anomaly detection, is a 'perceptual' difference — that is, a measure of difference that is computed in the latent feature space of a neural encoder, which is ideally aligned with human estimates of dissimilarity [195].

We therefore qualitatively compare several perceptual difference metrics. Specifically, we compare a DINO-based metric with LPIPS [195] and DreamSim [196]. We also compare with SSIM [197], which is a direct predecessor to perceptual similarity metrics, and share the results in Figure 3.9 (bottom). For this experiment, we compute the DINO-based metric using L2 rather than cosine-difference, for calibrated comparison with the other metrics,

which all (except SSIM) use L2 somewhere in their pipeline. We also use bilinear rather than FeatUp-based upsampling, in order to isolate the impact of the difference metric on the information available for heatmap generation in the absence of FeatUp's guidance. L2 and SSIM are computed in image space, while DreamSim, LPIPS, and DINO are all computed in latent/perceptual space.

We follow this qualitative comparison (Figure 3.9) with a quantitative comparison between DINO16 [27]-, DINOv2 [194]-, ViT [5]-, CLIP [198]-, and MaskCLIP [199]- based perceptual differences. More specifically, we evaluate each difference method via the binary classification task of OOD pixel detection, using masks generated from known held-out data classes (Figure 3.14) as truth. We compare the performance of our anomaly detector using both L2 and cosine distance, and with both FeatUp and bilinear upsampling. We share the ten top-performing combinations in Table 3.1.

Using a perceptual difference computed with DINOv2 features, using cosine-distance [189] and FeatUp upsampling, is the highest performer, and we select this model as our $\text{Diff}(\cdot,\cdot) \equiv \text{CosDiff}(F(\cdot), F(\cdot))$ operator, where $F(\cdot) \equiv \text{FeatUp}(\text{DINOv2}(\cdot))$. We qualitatively show the impact of upgrading to DINOv2 in Figure 3.23.

Embedding	**Distance**	**Upsampling**	**↑ AUCPR**	**↓ FPR95**	**↑ PSNR**
DINOv2	Cos	FeatUp	**0.747**	**0.514**	**5.53**
DINOv2	L2	Bilinear	0.726	0.553	3.69
DINOv2	Cos	Bilinear	0.726	0.553	3.69
MaskCLIP	L2	Bilinear	0.710	0.619	1.94
MaskCLIP	Cos	Bilinear	0.694	0.646	1.73
DINO16	Cos	FeatUp	0.663	0.555	4.4
DINO16	Cos	Bilinear	0.639	0.640	2.67
DINO16	L2	Bilinear	0.614	1.00	2.54
ViT	Cos	FeatUp	0.558	0.598	3.41
MaskCLIP	L2	FeatUp	0.557	0.531	3.81

Table 3.1: Comparison of perceptual difference metrics computed using different feature embeddings, distance metrics, and upsampling methods, sorted in approximately decreasing order of performance. Our DINOv2-based metric using cosine-distance and FeatUp is the top performer on area-under-curve precision-recall (AUCPR), and false-positive rate at 95% true-positive rate (FPR95), and peak signal-noise ratio (PSNR). We additionally share the remaining top-ten performers. Underlined metrics are within a standard deviation of the best value. **Bold** metrics are best performers.

3.3 Key Experiments

3.3.1 Performance on RUGD Dataset

We evaluate DiffUnc on the RUGD [190] dataset and share qualitative results in Figure 3.10. The diffusion model is trained on RUGD data without artificial constructs. At test-time, we present DiffUnc with OOD images containing anomalies from held-out classes. See Section 3.4.1 for examples of training and anomalous data.

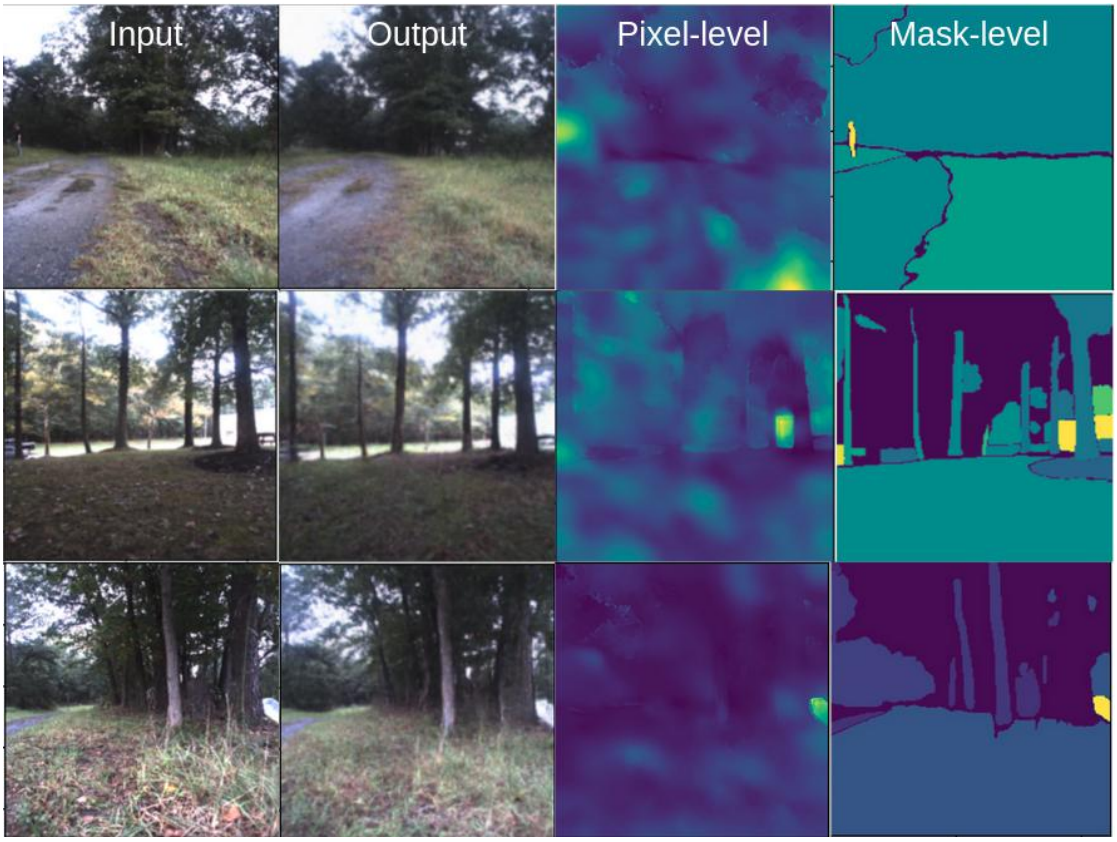

Figure 3.10: Qualitative results on anomalous examples from the RUGD dataset. Our pixel-level uncertainty estimates do generally highlight anomalous regions, but our mask-level estimates is where DiffUnc truly shines. The full DiffUnc pipeline does particularly well at detecting small and camouflaged/human-imperceptible anomalies.

We show the outputs after each stage of our pipeline in Section 3.4.6, and share the impact of each component on model DiffUnc via ablations in Section 3.4.7. We share performance on other datasets in Section 3.4.8.

3.3.2 Validating our Improved Diffusion Guidance

We show that the SoftRect energy function improves our ability to remove anomalies via guided diffusion from the CLEVR dataset [200] in Section 3.4.3 . In Section 3.4.4, we also show what happens as we tune the guidance strength hyperparameter.

3.4 Additional Results

3.4.1 Modelling the RUGD Dataset

The RUGD dataset (Figure 3.11, [190]) is an off-road dataset of video sequences captured from a small, unmanned mobile robot traversing in unstructured environments. It contains over 7,000 frames annotated with pixel-level segmentation over 24 semantic classes. The annotated frames are spaced five frames apart.

The diffusion model spotlighted in Section 3.3 is trained on samples from the RUGD dataset without humans and artificial constructs (Figure 3.13). The remaining image pairs are 'held out' to supply us with examples of out-of-distribution and anomalous images (Figure 3.14). As can be seen from Figure 3.12, our trained diffusion model successfully generates realistic images containing only in-distribution classes such as trees, grass, and the occasional footpath.

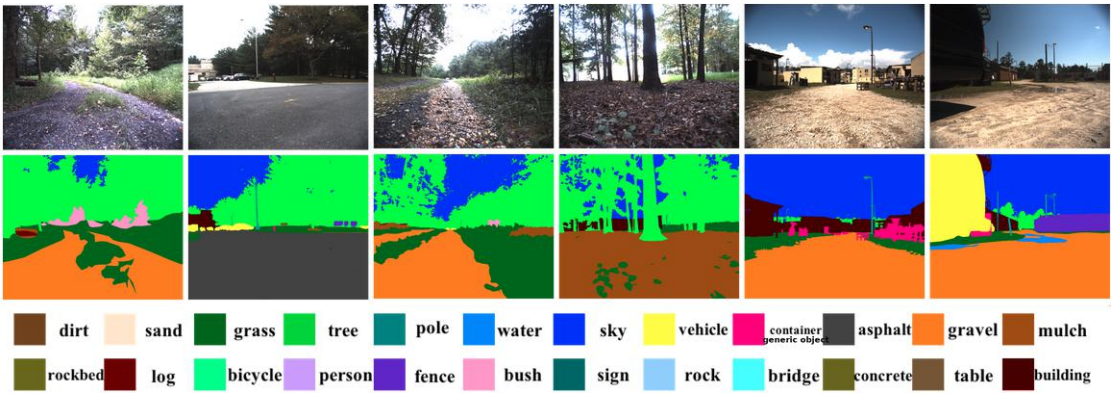

Figure 3.11: Examples of video frames and annotations from the full RUGD dataset [190].

Figure 3.12: Examples of the random-seed in-distribution images generated by the RUGD diffusion model trained on Figure 3.13.

Figure 3.13: Examples of the in-distribution images on which our RUGD diffusion model was trained. In general, these images contain a mixture of forest, meadow, mulch, and paths, without any humans or artificial constructions like buildings or vehicles.

Figure 3.14: Examples of the out-of-distribution RUGD images our forest-deployed robot might encounter, and from which our diffusion model trained on Figure 3.13 must remove anomalies.

3.4.2 Modelling the Urban Datasets and Anomaly Benchmarks

Cityscapes (Figure 3.15, [191]) is a popular urban driving dataset that contains video sequences taken from a car driving in 50 different cities. 5,000 frames are annotated at a 'fine' level with semantic class information, and 20,000 frames are annotated coarsely.

We train a diffusion model on Cityscapes (Figure 3.17) and the generative capability shown in Figure 3.16. Although these generated images display several infidelities to reality, we find that the full DiffUnc is robust to these sorts of textural artifacts. Specifically, we demonstrate this robustness using the Fishyscapes and SegmentMeIFYouCan (SMIYC) anomaly detection datasets (Figures 3.18 and 3.19, [192][193]), and share the results in Section 3.4.8.

Figure 3.15: Examples of the coarse (bottom) and fine (top) semantic annotations of the Cityscapes urban driving dataset [191].

Figure 3.16: Examples of the in-distribution images generated by the Cityscapes diffusion model trained on Figure 3.17. These images display artifacts reminiscent of the earliest Stable Diffusion [40] results, and could likely be improved using the same techniques that have since been applied to Stable Diffusion. However, we find that our DINOv2-FeatUp-SAM pipeline often makes up for these artifacts! (Figures 3.27 and 3.28.)

Figure 3.17: Examples of the in-distribution images on which our Cityscapes diffusion model was trained. In general, these images contain a mixture of forest, meadow, mulch, and paths, without any humans or artificial constructions like buildings or vehicles.

Figure 3.18: Examples of the out-of-distribution Fishyscapes images our urban robot might encounter, and from which our diffusion model trained on Figure 3.17 must remove anomalies.

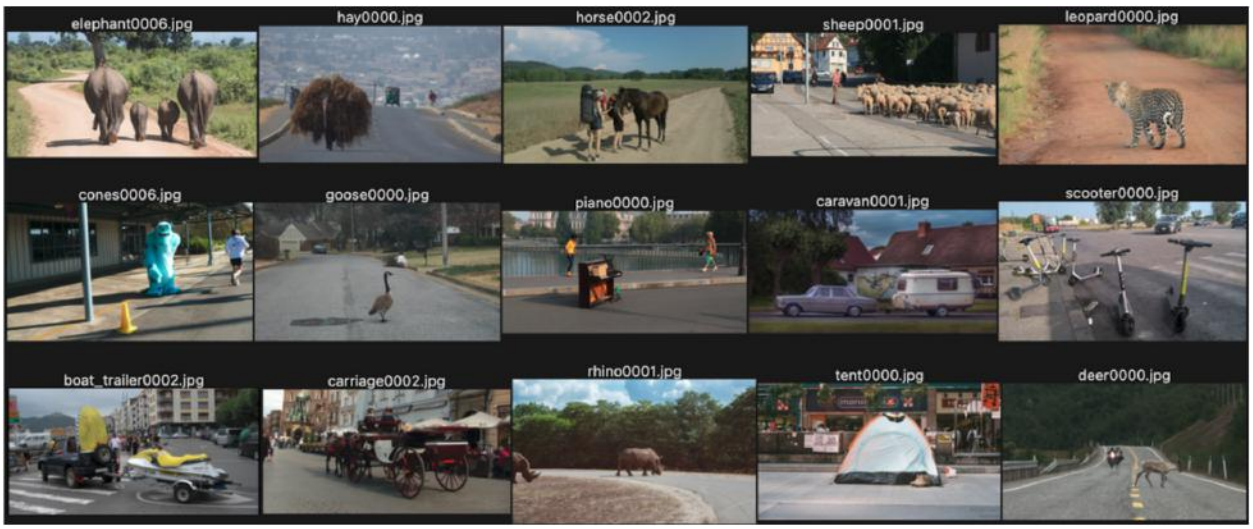

Figure 3.19: Examples of the out-of-distribution SMIYC images a semi-urban robot might encounter, and from which our diffusion model trained on Figure 3.17 must remove non-urban anomalies.

3.4.3 Validating the SoftRect Guidance Function

To validate our choice of SoftRect (Section 3.2.1) over L2 guidance, we train two diffusion models — one for each guidance method — on examples from the CLEVR data [200] containing no reds, yellows, or browns (Figure 3.21). We then present the trained model with anomalous images containing those held out colours, and compare their ability to remove anomalies without modifying non-anomalous parts of the image (Figure 3.20). We find that SoftRect indeed outperforms L2 guidance when it comes to diffusion model-based anomaly removal.

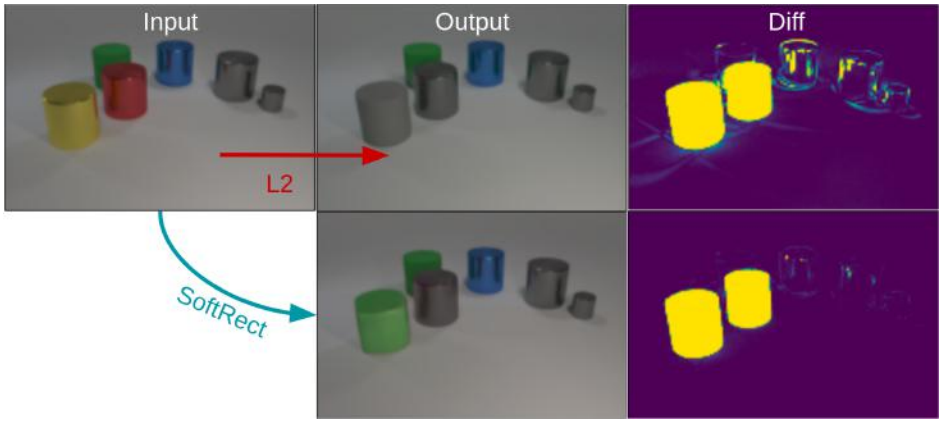

Figure 3.20: Comparing the performance of SoftRect- and L2- guided diffusion models shows that SoftRect successfully guides the diffusion process toward fewer unecessary changes on in-distribution objects. As a result, SoftRect-guided diffusion yields an uncertainty map with few false-positive pixels.

Figure 3.21: Examples of CLEVR images without reds, yellows, or browns, upon which our tabletop diffusion models were trained.

3.4.4 Tuning the Guidance Strength

The strength of the guidance term in our diffusion model can be tuned to enforce a variable level of consistency between the image being generated $\mathbf{x}'$ and the target image $\mathbf{x}$. Figure 3.22 shows the impact of the guidance term on the image generated, for a sample anomalous image $\mathbf{x}$ from the RUGD dataset.

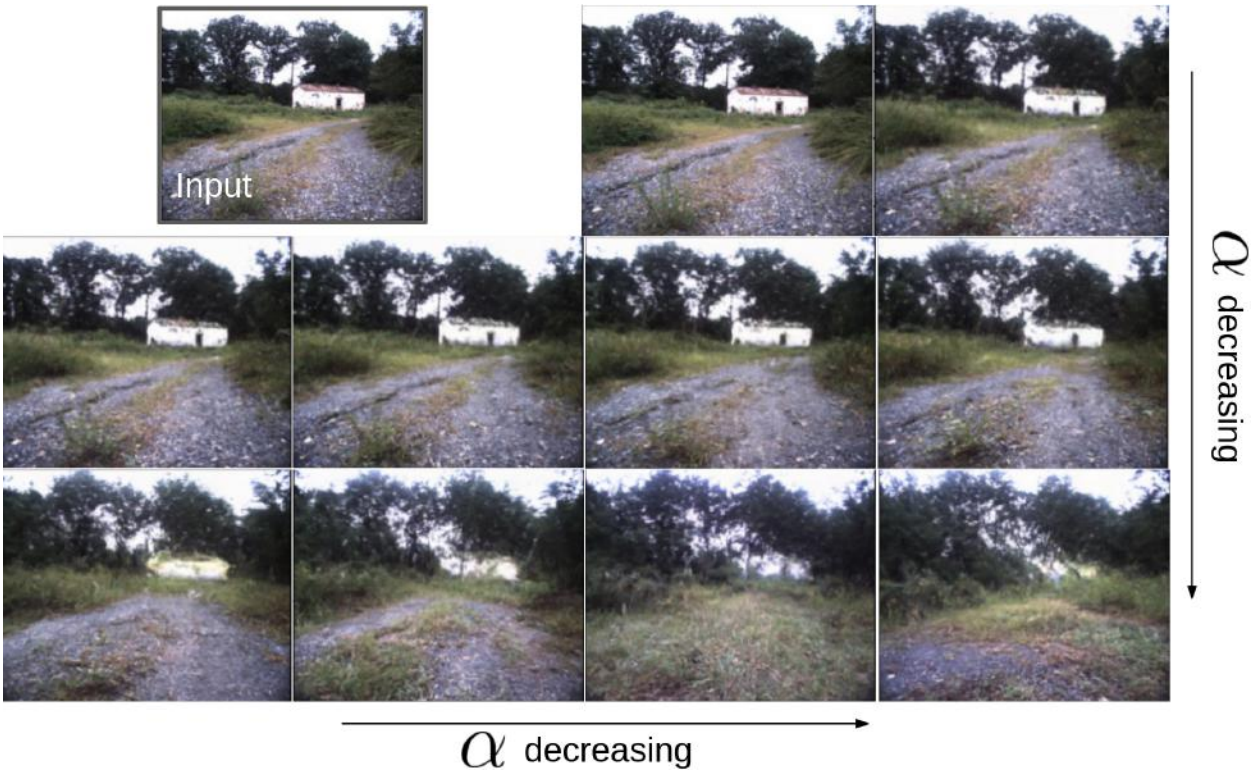

Figure 3.22: The strength of our diffusion model's guidance term can be tuned by a hyperparameter α. As α decreases, the guidance enforcing consistency between the target image and the diffusion process' output weakens, and an increasing number of changes are allowed.

3.4.5 Qualitative DINO16 vs. DINOv2 + FeatUp

Early versions of DiffUnc used a perceptual difference metric that was based on DINO16 feature embeddings, which were upsampled back to input image dimensions using bilinear upsampling. When FeatUp (Chapter 2) matured, we were able to upgrade our perceptual difference to a FeatUp-sampled DINOv2-based metric, which led to a considerable leap in performance (Figure 3.23).

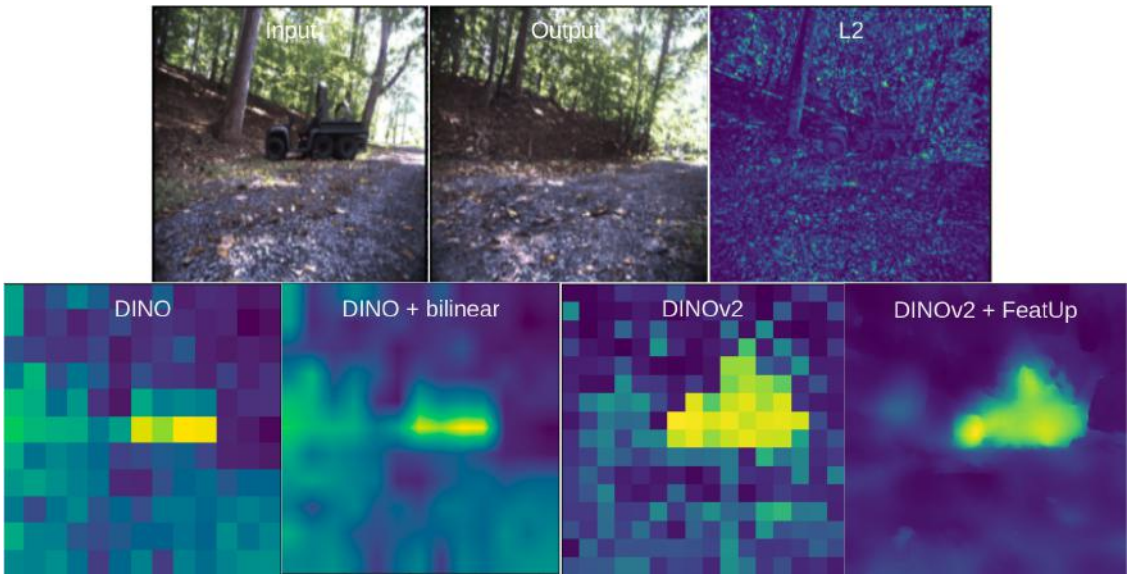

Figure 3.23: Upgrading from DINO16 and bilinear upsampling to DINOv2 and FeatUp upsampling yields a remarkable improvement in qualitative performance! Specifically, the uncertainty map generated via DINOv2 + FeatUp shows far better object-alignment and a much lower false-positive rate. These improvements are verified by our quantitative experiments in Table 3.1.

3.4.6 Qualitative Pipeline Walkthrough

We share several ablations in Section 3.4.7, but the impact of each segment of our pipeline is fairly clear from a glance at the qualitative results (Figure 3.24).

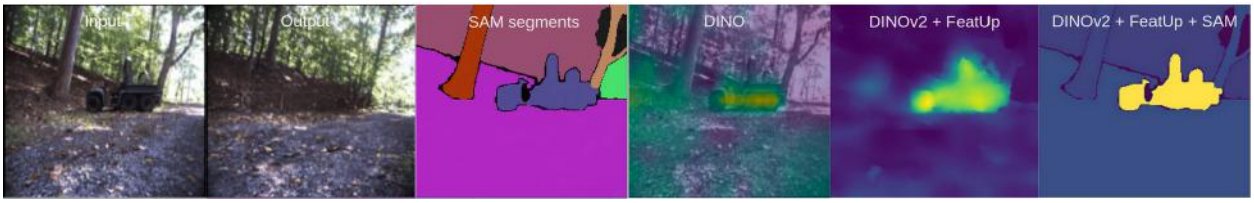

Figure 3.24: Qualitative results from each step of the DiffUnc pipeline (Figure 3.6). The SAM segments are generated by applying SAM to the original input image. Combining DINOv2, FeatUp, and SAM yields crisp, highly-salient anomaly detection.

3.4.7 Ablations

In Figures 3.25 and 3.26 we show the impact of each component of DiffUnc on its performance. The precision-recall curve also shows the impact of selecting cosine-distance over L2 within the perceptual metric. Table 3.2 shows how AUCPR, FPR95, and PSNR trend with the inclusion of SAM and FeatUp .

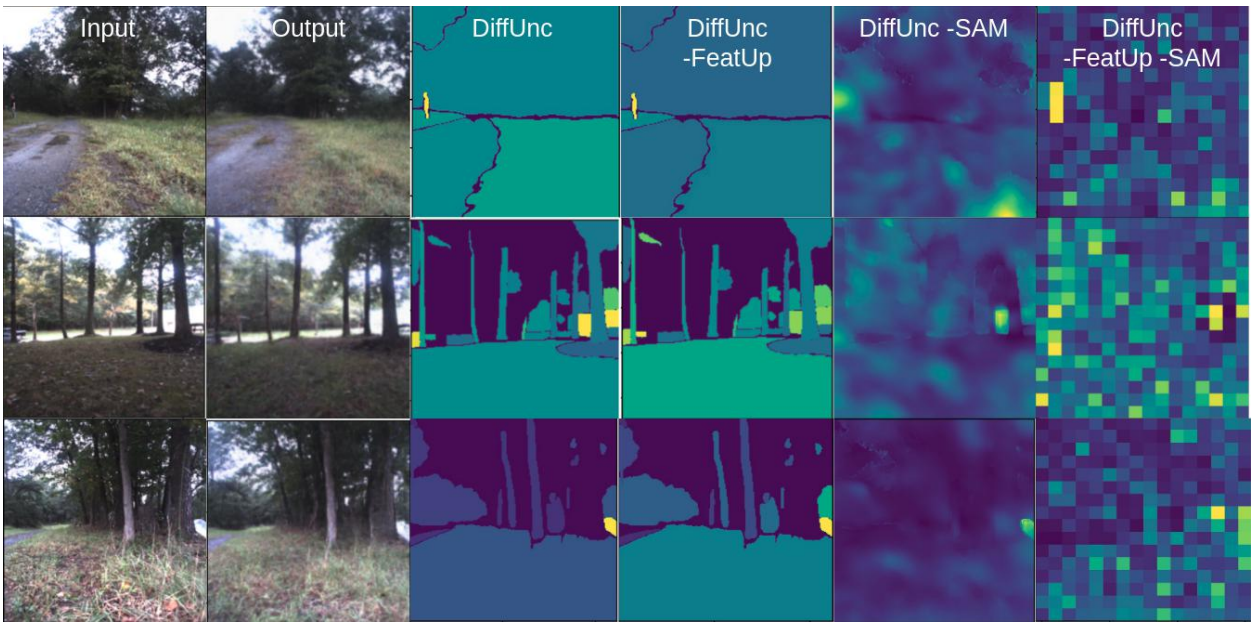

Figure 3.25: Impact of SAM and FeatUp on DiffUnc performance. Removing FeatUp lowers the contrast between in- and out-of- distribution segments. Removing SAM particularly degrades performance on small anomalies.

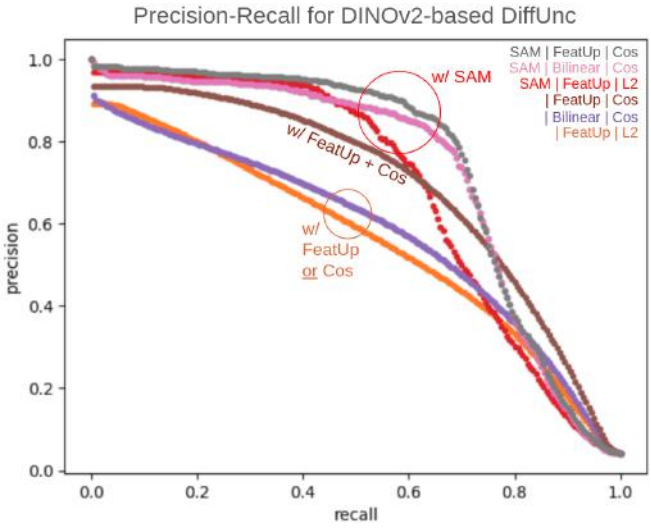

Figure 3.26: Impact of SAM, FeatUp, and cosine-distance (in the Diff$(\cdot,\cdot)$ operator) on DiffUnc precision-recall. Models with SAM maintain a high precision over a larger recall range, while models with FeatUp (and no SAM) have a higher precision at very high recalls.

SAM?	FeatUp?	↑ AUCPR	↓ FPR95	↑ PSNR
Yes	Yes	**0.747**	0.514	5.53
	No	0.726	0.553	3.69
No	Yes	0.693	**0.282**	**6.54**
	No	0.58	0.362	4.68

Table 3.2: Quantitative impact of SAM and FeatUp on DiffUnc performance. SAM generally improves the area-under-curve precision-recall (AUCPR), and degrades the false-positive rate at 95% true-positive rate (FPR95). FeatUp generally improves the peak signal-noise ratio (PSNR). Underlined metrics are within a standard deviation of the best value. **Bold** metrics are best performers.

3.4.8 Performance on Anomaly Detection Benchmarks

We evaluate DiffUnc on the Fishyscapes [192] and SMIYC [193] benchmarks and share qualitative results in Figures 3.27 and 3.28. The diffusion model is trained on urban Cityscapes data (Figure 3.17). At test-time, we present DiffUnc with OOD images from Fishyscapes and SMIYC (Figures 3.18 and 3.19), which not only contain distribution shifts from Cityscapes, but additionally contain entirely new semantic classes.

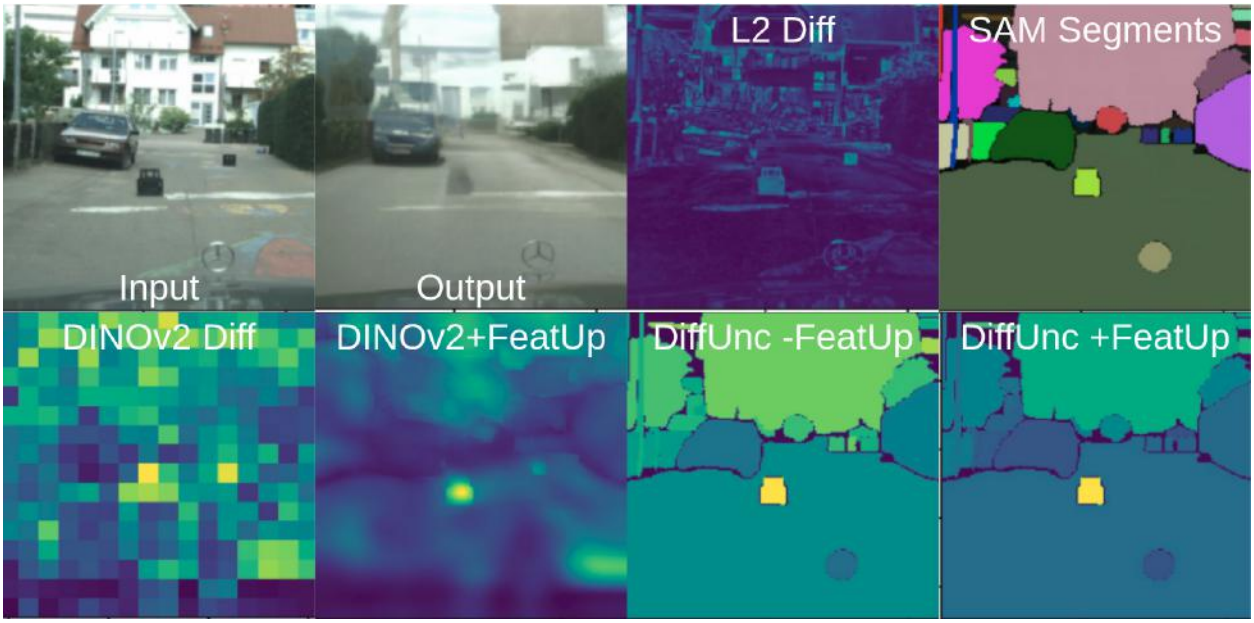

Figure 3.27: Qualitative results on an anomalous example from the Fishyscapes dataset. We are currently expanding these results.

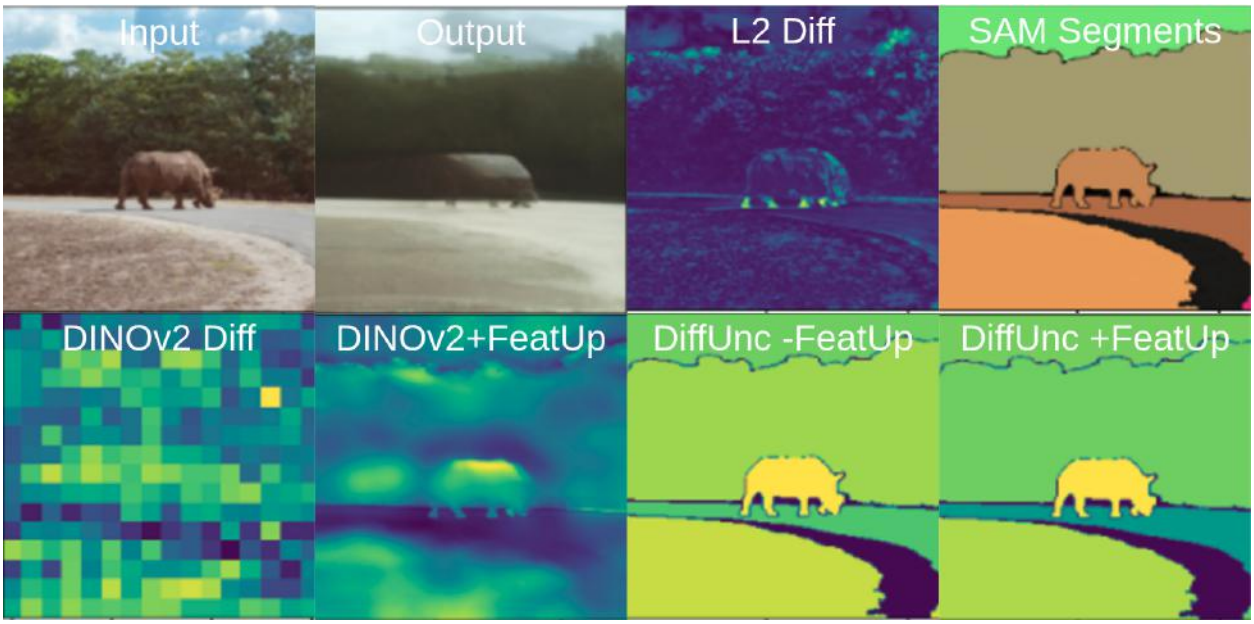

Figure 3.28: Qualitative results on an anomalous example from the SMIYC dataset. We are currently expanding these results.

3.4.9 Limitations

DiffUnc's performance appears to degrade on samples with a large number of objects (Figure 3.29). We believe this is due to under-segmentation by SAM, which is fed a sample-invariant hyperparameter that dictates how many segments to make. Future work could aim to find some criterion by which SAM can vary its segmentation resolution, without requiring human intervention.

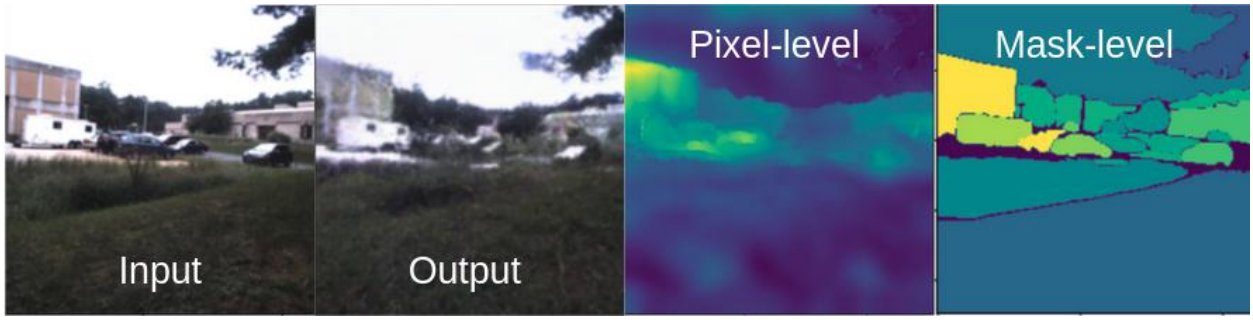

Figure 3.29: In images such as this one, where there are a high number of anomalies — in this case, the vehicles and buildings are all anomalies — the performance of our model appears to degrade. This is likely because SAM is under-segmenting an image with a high number of objects. The number of masks SAM generates is a hyperparameter in our pipeline and so could be adjusted, but the question of how to handle images with a varying number of objects is a non-trivial one.

3.5 Conclusion

DiffUnc is a method for obtaining epistemic uncertainty estimates at the pixel, mask, and image level. It works in any domain for which a guided diffusion model can be trained to remove anomalies, which — with our improvements to the guidance signal — includes at least off-road and urban settings.

A challenge in deploying robotic systems to remote and data-constrained regions like wilderness and underground environments is that there is insufficient data available during training, and it is common to encounter visual data that is out-of-distribution and causes the robot to fail [7]. However, with DiffUnc, such a robotic system can now recognise a source of potentially many failures; and not only that, it can report to its remote supervisor which part of its visual field is causing a problem. This reporting mechanism enables the supervisor to react and address the issue, saving the robot from failure, and in the best case even improving the robot's vision model through active labelling [201] and finetuning [202].

My collaborator, Sid Ancha, has already taken a proof-of-concept step toward this application of DiffUnc by implementing a modification of AnyLabelling, a GUI and tool that makes it incredibly easy to segment and label images data by hand [203]. His modification adds a DiffUnc mode that makes it very easy for a human labeller (*e.g.*, the supervisor) to prioritise labelling images in order of decreasing image-level uncertainty, as estimated by a specified DiffUnc model (Figure 3.30).

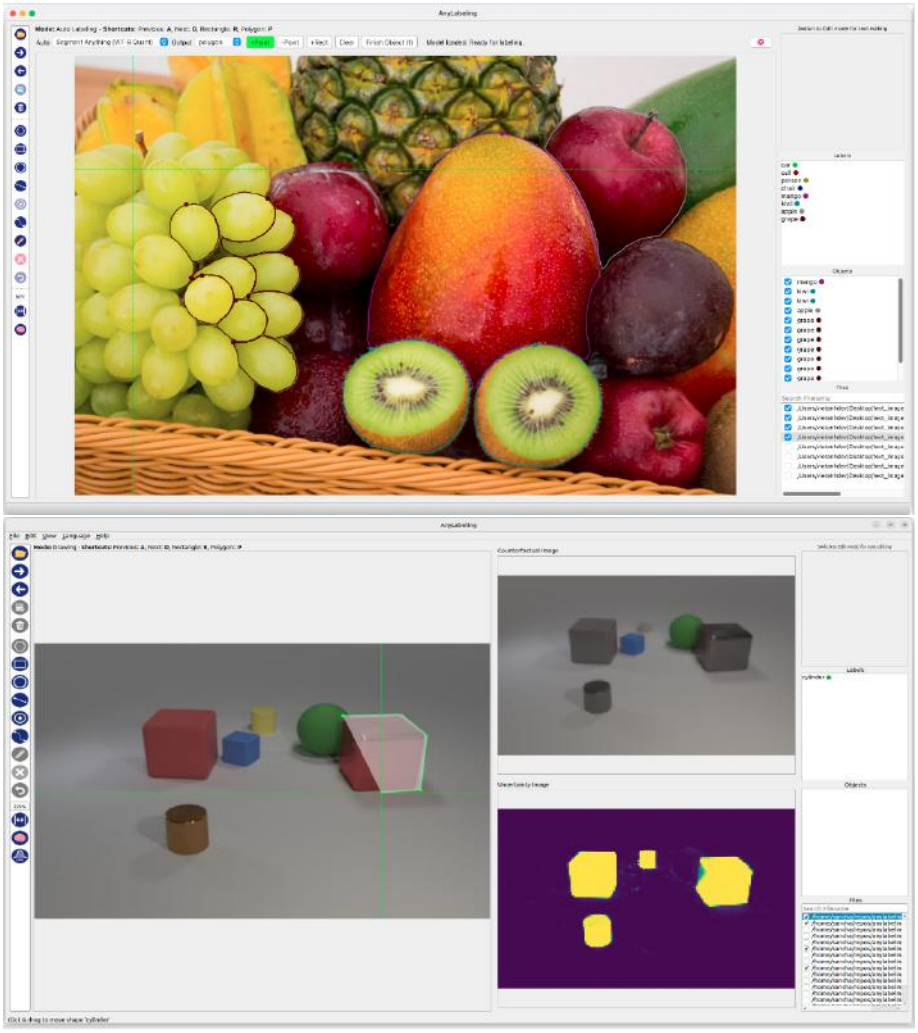

Figure 3.30: AnyLabelling (top) is a GUI and tool for segmenting and labelling image data by hand [203]. Sid Ancha has modified it (bottom) to add a toggleable DiffUnc mode, controlled by the bottom-most button on the toolbar at left. When on, a specified DiffUnc model is loaded, and the images to be labelled are sorted in order of decreasing DiffUnc-estimated uncertainty. He additionally displays the diffusion model's output (upper-right) and the uncertainty masks (lower-right) to help the human labeller interpret and detect anomalies that should be labelled.

I believe that the next step in extending advances in computer vision to data-constrained environments is a more intentional experimental characterisation of the impact of uncertainty-informed active labelling and finetuning on vision model performance, and I discuss this challenge further in Section 4.1.

Chapter 4

Open Questions

I presented the two following questions in Chapter 1:

1. How do we choose the 'right data' in the real world?
2. How can we get good visual estimates out of compute-constrained systems?

In this dissertation I presented two substantive projects that my collaborators and I undertook that each addressed one of these core queries. FeatUp (Chapter 2) is a fast, guided, task-agnostic feature upsampling method that refines low-resolution (and low-compute) results to higher resolution without retraining. It outperforms competitors like CARAFE and Strided Convolution in speed and memory requirements, and those like Bilinear Upsampling and Resize-Convolution in quality. Meanwhile, DiffUnc (Chapter 3) is a diffusion-based method for estimating hierarchical epistemic uncertainty at the pixel- and mask-level. In addition to bringing this hierarchical capability to the table, it performs competitively well on anomaly detection benchmarks like Fishyscapes and SegmentMeIfYouCan (SMIYC).

FeatUp addresses Question 2 and DiffUnc tackles Question 1, but as my time at MIT draws to a close, it becomes increasingly apparent that there is a so-far unasked Question 3 that should be articulated:

3. How should we use the 'right data', and the hierarchical uncertainty estimation we use to find it, for improved vision model performance in data- and compute- constrained environments?

Based on my work so far, there are several research directions that I believe are both promising and exciting. All of which build on or leverage DiffUnc in some manner; some of which could additionally benefit from using FeatUp.

4.1 Uncertainty-informed Active Labelling

As discussed in Section 3.5, active labelling can be used to finetune models that have already been deployed; and I propose that a particularly good criterion for selecting images to label is the image-level epistemic uncertainty of the test-time images, as estimated by DiffUnc. Sid Ancha has already taken a proof-of-concept step by implementing a DiffUnc mode in the AnyLabelling tool (Figure 3.30).

However, simply labelling data is insufficient to demonstrate that uncertainty-informed active labelling is beneficial and effective. One must *use* the labelled data for finetuning, and the details of how to do so are non-trivial. I believe that at least a paper, and quite possibly an entire doctoral thesis, could focus on this topic.

4.1.1 Implementation

The first step would be to implement a *full* pipeline for uncertainty-informed active labelling *and finetuning*. This could look something like Figure 4.1.

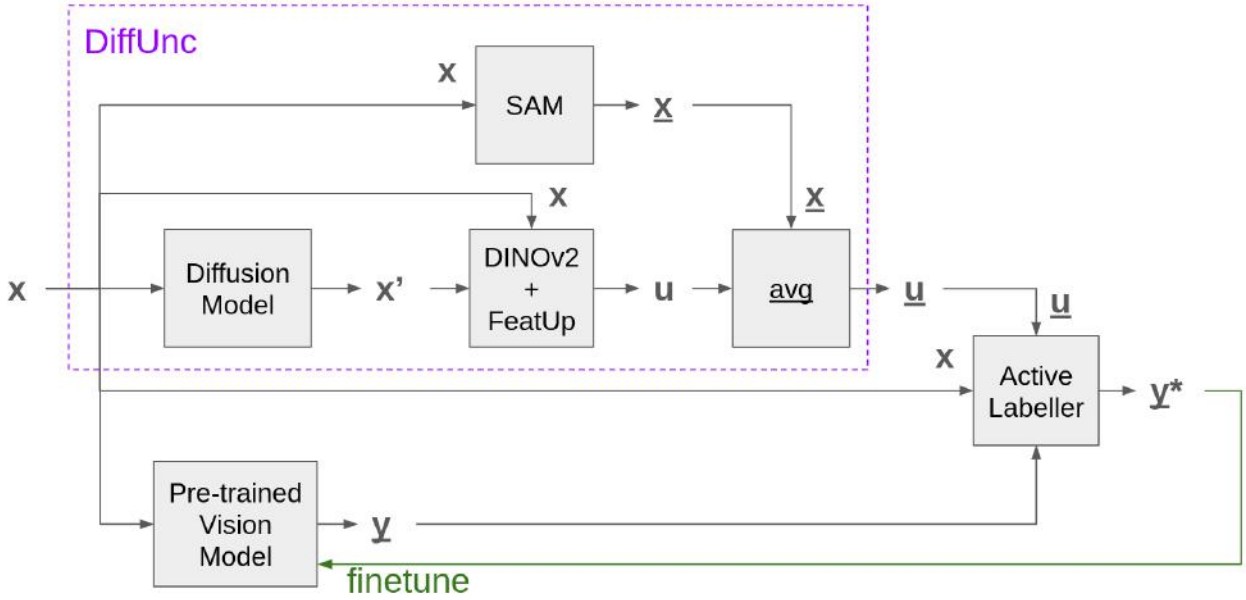

Figure 4.1: A full pipeline for uncertainty-informed active labelling and finetuning. We have already implemented DiffUnc (Chapter 3) and a preliminary version of the Active Labeller (Figure 3.30). Future work could combine these with a semantic segmentation model, and experiment with using data from the Active Labeller to finetune the Pre-trained Vision Model.

4.1.2 Experiments

Prior work from the finetuning literature can be used as a starting point for implementing a finetuning paradigm, but may not hold true for images selected via uncertainty estimate. As a results, several experiments will likely be important to pursue, in order to characterise the vision model's response to the uncertainty-informed finetuning. Examples of parameters that could be changed in the active labelling/finetuning process include:

1. The number of images B per active labelling/finetuning batch.
2. The number of batches N used to finetune the vision model.

3. How frequently DiffUnc is retrained. (It will have to be retrained to account for originally OOD data which is now in-distribution.)

4. The details of the uncertainty-informed sampling method — the straightforward choice is to label the B most uncertain images encountered, but it is conceivable that another sampling method would be better if, for example, those most uncertain images all happened to depict the same anomalous object.

One could imagine generating characteristic plots wherein either N or B is held constant, the other lies along the horizontal, and some performance metric — probably mean-Intersection over Union (mIoU) for semantic segmentation — lies along the vertical. If you have a lot of time, you could generate a full hyperparameter grid for which both N and B are varied at once.

4.2 Perceptually Guided Diffusion

As we were developing DiffUnc (Chapter 3), we found ourselves having to alter the details of the guided diffusion. Specifically, we changed the guidance's penalty function from an L2 to a 'SoftRect'. Another idea we considered was that of switching from an image-space penalty function (*e.g.*, L2, SoftRect) to one computed in the latent — or perceptual — space. Humans clearly learn in a more abstracted manner than direct comparison between image intensities [19], so we wanted to try to do the same in machines. We conducted a preliminary experiment using LPIPS [195] to guide the diffusion process, but it did not go well (Figure 4.2).

Since understanding guided diffusion was not our focus at the time (we were focused on getting good DiffUnc results expeditiously), we moved on without looking into why this is what happened. However, I believe it could be a very interesting project to try to determine whether or not using perceptual similarity as a guidance signal for diffusion is a good or a bad idea.

4.2.1 Implementation

Implementation-wise, such a project could use the DiffUnc codebase as a starting point. All that would need be changed is the guidance method being used by the diffusion model.

4.2.2 Ideas

This experiment (and its failure) came up during my doctoral defense, because my committee actually had the idea independently of me and asked if I had tried it. At the time, Phillip Isola suggested that if one were to pursue this idea further, one might look into replacing LPIPS with E-LPIPS as the guidance; E-LPIPS is a follow-on work that was designed to be more robust in certain ways [204].

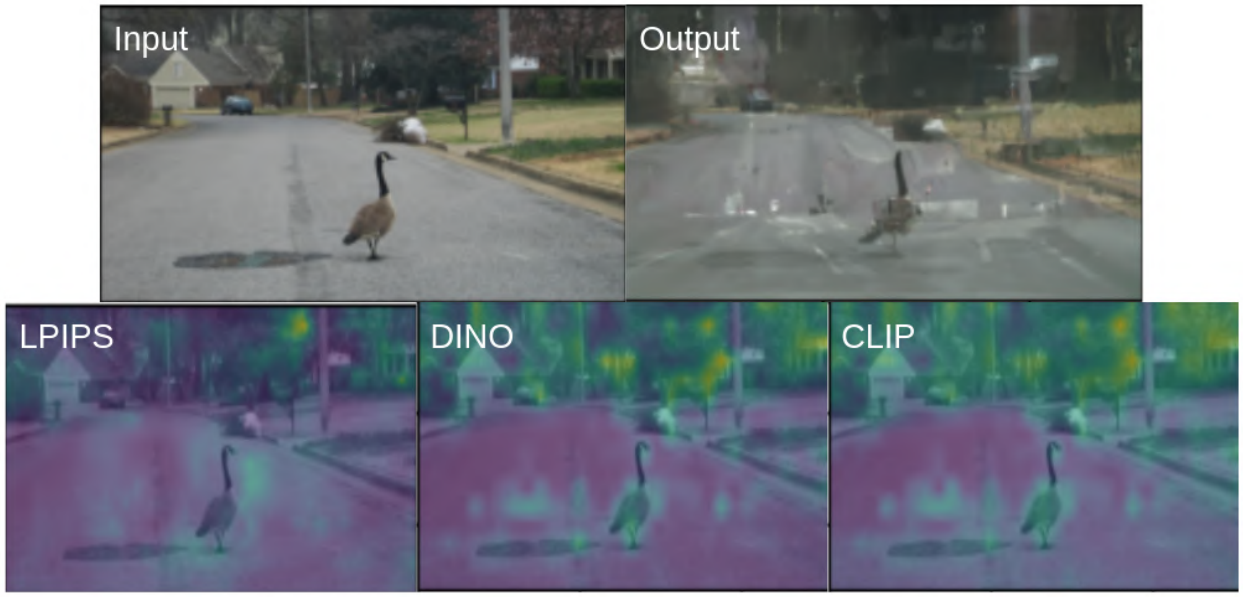

Figure 4.2: We tried guiding the diffusion model using the LPIPS perceptual similarity metric. This yielded the output at top-right, which clearly still had something 'goose-y' in the foreground. We then computed the LPIPS, DINO-based, and CLIP-based differences between input and output, and saw that the majority of perceptual changes had actually occured in the background rather than on the foreground anomaly.

4.2.3 Risks

A risk of this project, from the point of view of a conference paper-aspiring graduate student, is that the idea of guiding diffusion models with perceptual similarity might be a bust, and it is unfortunately rather difficult to publish a paper consisting only of negative results in conferences. A journal paper may be a possibility, if the theoretical exploration is sufficiently interesting. Otherwise, this is an example of a project that, if it fails to improve guided diffusion, could probably go into your thesis and self-published on the arXiv, but would not in itself produce a peer-published paper.

In counterpoint, even if this specific idea is a bust, it could lead to another idea which isn't, and negative results *can* generally be published alongside the positive result that followed them.

4.3 Uncertainty-derived Attention / Supervision

The topic I originally wanted to centre the latter half of my doctoral work around was not uncertainty estimation itself, but rather *using* pixel-wise uncertainty to improve vision model training and performance. One of the ideas I spent a fair amount of time considering was that of using pixel-wise uncertainties as a supervisory signal. My thought was that since humans use uncertainty to inform their learning processes, it could make sense to have vision models make direct use of uncertainty, rather than relying only on some *post-hoc* active labelling-based correction.

Alas, I got distracted from this thread by the fact that pixel-level uncertainty estimation was not yet mature enough of a technology. However, my hope is that with DiffUnc (Chapter 3), we have at least partially dealt with that problem, which means that now may be the perfect time to return to this idea.

4.3.1 As attention

The DiffUnc work as I formulated it lends itself more naturally to using the pixel-wise uncertainty as an attentional signal than a supervisory one. This is because DiffUnc operates on the inputs $\mathbf{x}$ rather than on the outputs $\mathbf{y}$ of the vision model, so there is no natural path for a supervisory signal to backpropagate along. However, one could envision using the uncertainty $\mathbf{u}$ as some sort of attentional signal that informs the vision model (Figure 4.3).

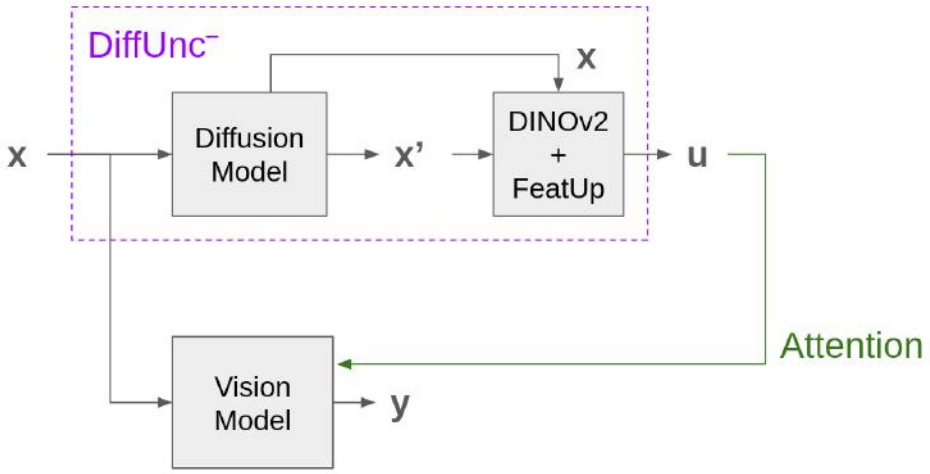

Figure 4.3: The pixel-level uncertainty $\mathbf{u}$ that is produced by DiffUnc$^-$ (without the SAM extension), or some function of it $\alpha(\mathbf{u})$, could be used as an attentional signal that informs the training of the Vision Model.

Implementation

You could simply take DiffUnc and ignore the SAM extension, using the pixel-level rather than mask-level uncertainties as your attentional signal. My unjustified intuition is that this idea would hypothetically work better on a regression problem like depth estimation rather than a classification problem like semantic segmentation; however there is generally more and better semantic segmentation data available than depth data. So maybe give it a try with semantic segmentation first.

4.3.2 As supervision

Using pixel-wise uncertainty as a *supervisory* signal for training vision models is the idea that originally got me interested in pursuing DiffUnc. However, it is a bit tricky to implement, because the best uncertainty estimation methods so far (including DiffUnc) operate on the

inputs $\mathbf{x}$ rather than the outputs $\mathbf{y}$ of the vision model, and therefore do not produce a signal that would pass through the vision model if backpropagated.

I think that in order to pursue this idea further, one would have to try creating a variant of DiffUnc that operated on the vision model output $\mathbf{y}$ (Figure 4.4). One could then backpropagate through the entire pipeline.

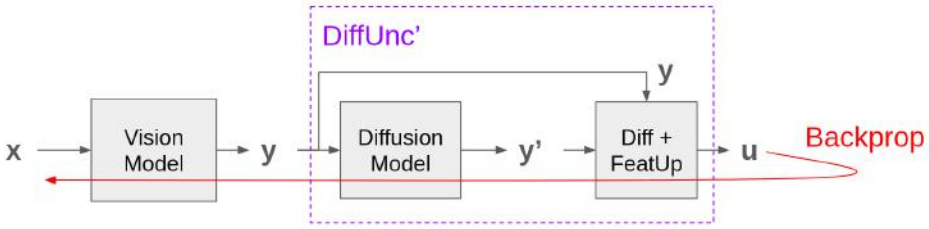

Figure 4.4: If one could train a variant of DiffUnc to operate on Vision Model outputs $\mathbf{y}$ rather than inputs $\mathbf{x}$, then one could backpropagate the uncertainty or an uncertainty-derived loss through the Vision Model as a supervisory signal.

Implementation

You could start with DiffUnc$^-$ (without the SAM extension) and re-train the diffusion model on, *e.g.*, true semantic maps. I think you would need to pad out the semantic maps so that each pixel was a one-hot vector where the one hot corresponded to the class label. That way, you could feed the diffusion model logits vectors at its input.

Alternatively, you could try training DiffUnc on the logit outputs of a pre-trained semantic segmentation model applied to in-distribution data, and then see if an uncertainty-derived loss function could improve finetuning on an out-of-distribution dataset.

Experiments

I do not expect that our empirical results regarding which Diff$(\cdot)$ operator yields the best uncertainty maps would hold, so you would probably want to redo those experiments. In fact, since the diffusion model would be operating on, *e.g.*, semantic logits, my expectation is that something like an L2 or cosine-difference might work better than trying to somehow re-encode a signal that is already encoded.

You could also experiment with where in the vision model you trained DiffUnc' on. For example, perhaps instead of training on the output $\mathbf{y}$, you train on some intermediate layer of the vision models. This might give the diffusion model a stronger signal to work with, at the expense of the supervisory signal not being able to provide feedback to the model's post-DiffUnc layers.

4.4 Uncertainty-based Model Fusion

This is another research direction that I spent a fair amount of time pondering before realising that I needed better uncertainty estimates. It was motivated by the idea that, at times,

humans have several competing methods for accomplishing a visual task. Instead of selecting one method over all the others, humans have a tendency to somehow combine their methods and obtain better visual estimates as a result. For example, humans can perceive depth via both binocular and monocular cues and generally make use of both at once — but when one or the other becomes unavailable, are still capable of depth perception [19].

I wondered if there was a good way to fuse different vision algorithms together in machines, based on uncertainty estimates. The idea of 'mixture of expert' or 'ensembling' methods is not a new one [205], [206], but I was interested in it from a few slightly different angles:

1. I suspected that the method for combining methods should be learned rather than deterministic.

2. In the context of semantic segmentation, I was interested in fusion to reduce domain specificity in vision models, because lightweight semantic segmenters tend to be highly domain-specific. When new data is encountered, one approach is to active label and fine-tune as discussed in Section 4.1, but fine-tuning sometimes leads to 'catastrophic forgetting' [207] of the model's original domain/task. I wanted to, instead of fine-tuning, train a new lightweight segmenter and *fuse* its results with that of the existing segmenter(s).

   One could image combining this with the FeatUp paradigm, and using fusion to fuse the output of several linear probes, which would be very lightweight indeed.

3. In the context of depth estimation, I was interested in fusion to solve the somewhat niche challenge of handling binocular camera occlusion. This is an issue that is not generally a catastrophic failure case on Earth, but actually *is* a significant problem for off-Earth semi-autonomous rovers, which often find one of their stereo cameras occluded by their own bodies when trying to look at objects which are near them and on the ground.

   As a result, I was interested in developing a system that could fuse the outputs of two depth estimation experts — one binocular, one monocular — based on a pixel-wise estimate of the model's uncertainty. My hope was that this would allow an autonomous rover to rely on monocular cues in regions of visual field where one of these stereo cameras was obstructed.

4.4.1 For semantic/panoptic segmentation

Implementation

Although I did implement and conduct preliminary experiments for this idea [208], that was pre-DiffUnc and so follow-on work would probably be best served by starting from scratch. The experimental setup could look something like Figure 4.5. It may look a bit complicated, but implementation could start with DiffUnc (Chapter 3) and our AnyLabelling-based Active Labeller (Section 4.1). What would be left to you to implement would be the choice of expert vision model(s) along with the details of the learned fuser. In the few years since I considered this problem, a few papers have come out that use learned fusers for panoptic segmentation, so you may wish to take those as a starting point [209], [210].

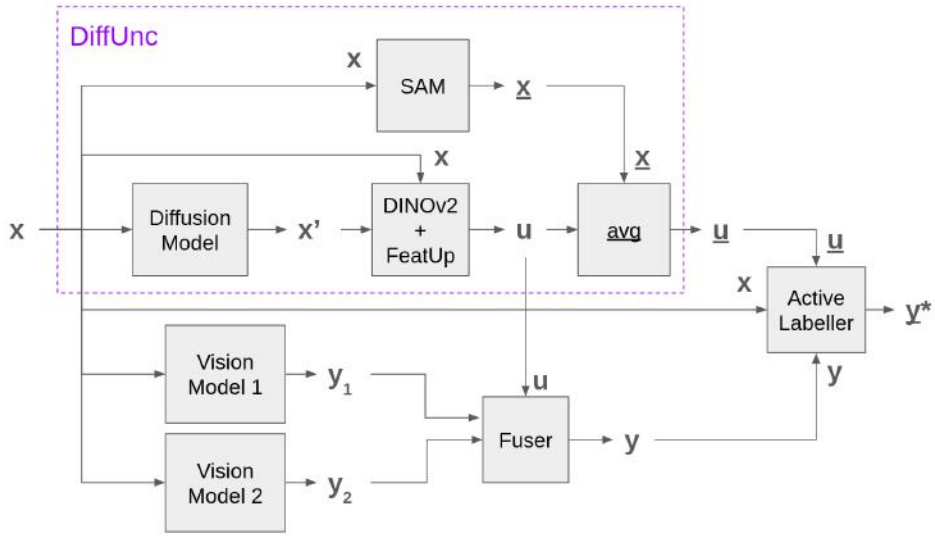

Figure 4.5: Using DiffUnc and an Active Labeller, you can accumulate modular datasets to train new domain-expert vision models (*e.g.*, a Vision Model 3) based on previously unseen data. This would address the common issue in fine-tuning of catastrophic forgetting. One could envision combining this concept with FeatUp (Chapter 2) so that one can work with domain-expert linear probes rather than full models.

Experiments

Rather than using active labelling for fine-tuning as in Section 4.1, in this project you would be using active labelling in order to create training set modules to train new domain expert models with. Once sufficient data was accumulated, you could, *e.g.*,

1. Train the new domain expert.
2. Update the fuser.
3. Update DiffUnc.

How exactly you undertake these three steps would be where the research lies!

4.4.2 For depth estimation

The architecture for monocular/binocular fusion would be the same as for semantic segmentation (Figure 4.5) except that there would be no need for an active labeller or the SAM extension. Vision Models 1 and 2 would be Monocular and Binocular depth estimators. The binocular estimator would take in a second image $\mathbf{x}_2$.

Experiments

I think that the tricky part of this project is finding the data that proves your point. Space industry data is generally a controlled commodity, and the best way to get access to it is to collaborate directly with, *e.g.*, NASA or Blue Origin. If you cannot do that, then you

could try simulating data. But I think that you will struggle to argue the importance of this work to the research community without concrete and real examples, even though folks in the space industry will know what's up.

4.5 Conclusion

In this chapter, I have outlined four different research directions that I view as natural follow-ons to the DiffUnc work (Chapter 3). I believe that each of them could likely produce an entire doctoral thesis, if explored thoroughly.

There are two additional FeatUp-derived research directions about which I have thought far less, and so will merely share briefly here:

1. The idea shared as the motivation for FeatUp (Chapter 2) itself; that is, designing (relatively) light-compute vision systems based on transfer learning via FeatUp + linear probes from a single backbone to several vision tasks.

2. Designing a *new* backbone into which FeatUp — or ideas from FeatUp — is directly integrated, so thereby *obsoleting* the conventional trade-off between spatial resolution and semantic/abstracted information.

This second idea came up in conversation with Phillip Isola, so if it piques your interest, a chat with him might be a good place to start. It also calls to mind the vast number of reverse connections known physiologically to be present in the (macaque monkey) brain, for which the purpose is often unclear [211]. Using something like FeatUp directly (or cyclically) within the backbone might lead to similar connections, thereby (perhaps) providing a hypothesis to vision and neuroscientists for their utility.

In any case, may this chapter aid you in your pursuit of human-inspired methods for extending advances in computer vision to data- and compute-constrained environments!

Chapter 5

Why Human-inspired?

Real-world platforms rarely have access to large amounts of cleanly labelled and high-resolution data, and the computational strength of real robots is often limited by size and power constraints. As a result, roboticists are often limited to LiDAR, stereo vision algorithms, or at best a bounding box detector like YOLOv3 Tiny. The modern vision algorithms and foundational models that have the vision community fearing that 'computer vision is solved' [212] are not even applicable in the majority of robotics, the field of research that originally birthed computer vision in the first place.

In this dissertation, I have sought to address the questions

1. How do we choose the 'right data' in the real, data-constrained world?

2. How can we get good visual estimates out of compute-constrained systems?

by using the human visual system as a model for how data- and compute-restricted systems can still get the important jobs done. This lens has inspired my research throughout my time at MIT, and led to the ideas behind FeatUp (Chapter 2), DiffUnc (Chapter 3), and the approaches to future work outlined in Chapter 4. It is my hope that, with this dissertation, I can provide a launchpad (or perhaps merely a trampoline) to give a boost to any researcher hoping to carry on the torch of exploring human-inspired methods for extending advances in computer vision to data- and compute- constrained environments.

. . .

There is still plenty of inspiration to be drawn from human vision and perception, far beyond what I have discussed in this dissertation. There are many, many visual tasks that humans do with an ease that robots still cannot mimic! Closing this gap between human and machine not only provides a point of view from which innovation in robot vision may be inspired, but is also in itself both challenging and exciting for those wishing to understand human vision in a more algorithmic manner than our current — primarily behavioural and physiological — understanding.

It is my firm belief that there should be an interplay between human vision science, computer vision science, and robotics. The desire to replicate the human capacity for vision on robotic platforms is where computer vision was born, after all; and, as nearly any practised

roboticist can tell you, there is still quite some ways to go before that desire is attained. Besides, the transfer of knowledge need not be one-way! Our human-inspired models can be used by vision scientists to generate hypotheses about the flow of information within and the likely failings of human visual systems; these hypotheses can then be tested in behavioural and physiological experiments with human subjects; which can in turn inform vision roboticists toward our ambitious goal of designing general, artificial, *embodied* vision systems capable of meeting or exceeding human expectations.

In this dissertation, I presented research that drew inspiration from the knowledge and hypotheses that vision scientists have accumulated and developed regarding human peripheral vision [20] and the impact of human attention and uncertainty on our ability to learn from our mistakes [19]. But there are so, so many more topics in human vision science that are both reasonably mature and highly relevant to tasks that robot vision struggles with, including but not limited to: Low-light adaptation [213], Colour constancy [214], Visual search [215], Scene gist perception [216], [217], and Motion perception [218]. You can explore all of these topics at a high level in the excellent introductory text, *Sensation & Perception*, by Jeremy Wolfe *et al.* [19]; and some pre-deep learning attempts at modelling many of these human visual processes computationally can be found in the classic text, *Vision*, by David Marr [219].

Whatever your background, I hope that this dissertation has piqued your interest in robot vision, and perhaps even — the tiniest bit? — in human vision. If you are yourself a roboticist, I hope that next time your robot does something embarrassing, especially if it has anything to do with perceptual failure, you ask yourself 'Why don't humans do this?' Or if they do do it (consider the toddling years...), 'How do they learn to *stop* doing this?' Perhaps you can make a friend in your institution's brain and cognitive science department (I have several) who don't mind it when you ask them seemingly aggressive questions about what, exactly, a human behaviour translates to *computationally*.

As a final, closing, thought: Interdisciplinary collaboration requires, I think, a certain level of comfort with feeling 'out of your league', 'ignorant', and flat-out 'stupid'. You're not, but that's how it will feel. If nothing else, I hope that my work can provide a bit of motivation as to why enduring that discomfort is worth it.

Just remember that you, too, know more than everyone else about *something*. And pursuing a PhD is about taking that difficult journey from knowing nothing to knowing 'a lot' (whatever *that* is...) about a teeny-tiny niche within your research field of interest. It's okay not to be an expert in the rest.

So go forth and conquer your dreams, of robots who know how to see!

Appendix A

Photo Credits

The images listed below were reproduced or modified from the cited sources, with appreciation for the original authors, and with their permission or under fair use. I hold or share the copyright to the remaining images in this dissertation, unlisted below.

Chapter 1

Figure 1.1: Kirillov *et al.* [3] and Ranftl *et al.* [4].
Figure 1.2: Kirillov *et al.* [3].
Figure 1.3: Biggie *et al.* [7].
Figure 1.4: Intel RealSense website [8].
Figure 1.5: Biggie *et al.* [7].
Figure 1.6: Biggie *et al.* [7] and Erick Fuentes [11].

Chapter 2

Figure 2.1: Ruth Rosenholtz [20].
Figure 2.2: Ruth Rosenholtz [21].

Chapter 3

Figure 3.5: Ancha *et al.* [113], Chalapathy and Chawla [184], Zheng *et al.* [185], and Nakao *et al.* [179].
Figure 3.7: Croitoru *et al.* [186].
Figure 3.11: Wigness *et al.* [190].
Figure 3.15: Cordts *et al.* [191].
Figure 3.30 (top): AI Curious [203].

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional networks," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 1, 2012, pp. 1097–1105.

[2] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, vol. 33, no. 12, pp. 6999–7019, 2021.

[3] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 4015–4026.

[4] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 12 179–12 188.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[6] NVIDIA, "A100 80gb pcie gpu," 2024, [Online; accessed 20-May-2024].

[7] H. Biggie, E. Rush, D. Riley, *et al.*, "Flexible supervised autonomy for exploration in subterranean environments," *Field Robotics*, vol. 3, no. 1, pp. 125–189, 2023.

[8] Intel, "Stereo depth solutions from intel realsense," 2024, [Online; accessed 05-May-2024].

[9] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv*, 2018.

[10] J. Francis, "Yolo v3 trained on open images data," *Wolfram Neural Net Repository*, 2020.

[11] E. Fuentes, "Personal research photograph," 2024.

[12] L. E. Brandt, *Master's thesis: Perceiving shape from surface contours via artificial neural networks*, 2021.

[13] L. E. Brandt and W. T. Freeman, "Toward automatic interpretation of 3d plots," in *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, Springer, 2021, pp. 35–50.

[14] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1125–1134.

[15] W. Cloud, "Pix2pix street-map-to-photo translation," *Wolfram Neural Net Repository*, 2019.

[16] S. Fu, M. Hamilton, L. E. Brandt, A. Feldmann, Z. Zhang, and W. T. Freeman, "Featup: A model-agnostic framework for features at any resolution," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

[17] S. Jiang*, S. Ancha*, L. E. Brandt*, and N. Roy, *in preparation*, 2024 expected, *title and author order TBD.

[18] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, 2016.

[19] J. M. Wolfe, K. R. Kluender, D. M. Levi, L. M. Bartoshuk, R. S. Herz, R. L. Klatzky, S. J. Lederman, and D. M. Merfeld, *Sensation & perception*. Sinauer Sunderland, MA, 2006.

[20] R. Rosenholtz, "Capabilities and limitations of peripheral vision," *Annual Review of Vision Science*, vol. 2, pp. 437–457, 2016.

[21] R. Rosenholtz, "Texture perception," in *The Oxford Handbook of Perceptual Organization*, Oxford University Press, 2015.

[22] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, 2005, pp. 886–893.

[24] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision (IJCV)*, 2004.

[25] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, no. 1, pp. 1–40, 2016.

[26] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9729–9738.

[27] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 9650–9660.

[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv*, 2013.

[29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the ACL Conference of the North*, 2019.

[30] A. Radford and K. Narasimhan, "Improving language understanding by generative pre-training," 2018.

[31] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "Wav2vec: Unsupervised pre-training for speech recognition," *arXiv*, 2019.

[32] W. Hsu, B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *Proceedings of the IEEE Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[33] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, vol. 26, no. 5, pp. 1019–1034, 2014.

[34] J. Ahn, S. Cho, and S. Kwak, "Weakly supervised learning of instance segmentation with inter-pixel relations," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2209–2218.

[35] M. Hamilton, Z. Zhang, B. Hariharan, N. Snavely, and W. T. Freeman, "Unsupervised semantic segmentation by distilling feature correspondences," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[36] Y. Wang, J. Zhang, M. Kan, S. Shan, and X. Chen, "Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 12 275–12 284.

[37] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 33, no. 5, pp. 978–994, 2010.

[38] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 402–419.

[39] S. Kobayashi, E. Matsumoto, and V. Sitzmann, "Decomposing nerf for editing via feature field distillation," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 35, pp. 23 311–23 330, 2022.

[40] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 10 684–10 695.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[42] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele, "Joint bilateral upsampling," *ACM Transactions on Graphics (ToG)*, vol. 26, no. 3, pp. 96–112, 2007.

[43] J. R. Lee, S. Kim, I. Park, T. Eo, and D. Hwang, "Relevance-cam: Your model already knows where to look," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14 944–14 953.

[44] Z. Qin, D. Kim, and T. Gedeon, "Rethinking softmax with cross-entropy: Neural network classifier as mutual information estimator," *arXiv*, 2019.

[45] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 1998, pp. 839–846.

[46] L. Caraffa, J.-P. Tarel, and P. Charbonnier, "The guided bilateral filter: When the joint/cross bilateral filter becomes robust," *IEEE Transactions on Image Processing (TIP)*, vol. 24, no. 4, pp. 1199–1208, 2015.

[47] C. Xiao and J. Gan, "Fast image dehazing using guided joint bilateral filter," *The Visual Computer*, vol. 28, no. 6–8, pp. 713–721, 2012.

[48] D. Mazzini, "Guided upsampling network for real-time semantic segmentation," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2018, p. 117.

[49] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2005, pp. 60–65.

[50] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 597–613.

[51] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7794–7803.

[52] M. Gharbi, J. Chen, J. T. Barron, S. W. Hasinoff, and F. Durand, "Deep bilateral learning for real-time image enhancement," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, p. 118, 2017.

[53] T.-M. Li, M. Gharbi, A. Adams, F. Durand, and J. Ragan-Kelley, "Differentiable programming for image processing and deep learning in Halide," *ACM Transactions on Graphics (ToG)*, vol. 37, no. 4, 139:1–139:13, 2018.

[54] S. Qian, H. Shao, Y. Zhu, M. Li, and J. Jia, "Blending anti-aliasing into vision transformer," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, 2021, pp. 5416–5429.

[55] Z. Lu, J. Li, H. Liu, C. Huang, L. Zhang, and T. Zeng, "Transformer for single image super-resolution," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2022, pp. 457–466.

[56] W. Freeman and A. Torralba, "Shape recipes: Scene representations that refer to the image," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 15, 2002.

[57] H. Su, V. Jampani, D. Sun, O. Gallo, E. G. Learned-Miller, and J. Kautz, "Pixel-adaptive convolutional neural networks," *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[58] N. Araslanov and S. Roth, "Single-stage semantic segmentation from image labels," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[59] T. Prangemeier, C. Reich, and H. Koeppl, "Attention-based transformers for instance segmentation of cells in microstructures," in *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2020, pp. 700–707.

[60] V. Guizilini, R. Hou, J. Li, R. Ambrus, and A. Gaidon, "Semantically-guided representation learning for self-supervised monocular depth," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[61] J. Choi, D. Jung, Y. Lee, D. Kim, D. Manocha, and D. Lee, "Selfdeco: Self-supervised monocular depth completion in challenging indoor environments," in *Proceedings of the IEEE/RAS International Conference on Robotics and Automation (ICRA)*, 2021, pp. 467–474.

[62] H. Choi, H. Lee, S. Kim, S. Kim, S. Kim, K. Sohn, and D. Min, "Adaptive confidence thresholding for monocular depth estimation," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[63] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., 2020, pp. 1–19.

[64] R. Gadde, V. Jampani, M. Kiefel, D. Kappler, and P. V. Gehler, "Superpixel convolutional networks using bilateral inceptions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 597–613.

[65] A. Shocher, N. Cohen, and M. Irani, "Zero-shot super-resolution using deep internal learning," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3118–3126.

[66] Y. Chen, S. Liu, and X. Wang, "Learning continuous image representation with local implicit image function," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 8628–8638.

[67] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," *International Journal of Computer Vision (IJCV)*, vol. 128, no. 7, pp. 1867–1888, 2020.

[68] R. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing (TASSP)*, vol. 29, no. 6, pp. 1153–1160, 1981.

[69] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[70] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel, "Deep vit features as dense visual descriptors," *arXiv*, 2021.

[71] N. Tumanyan, O. Bar-Tal, S. Bagon, and T. Dekel, "Splicing vit features for semantic appearance transfer," pp. 10 738–10 747, 2022.

[72] W. Shi, J. Caballero, L. Theis, F. Huszar, A. Aitken, C. Ledig, and Z. Wang, "Is the deconvolution layer the same as a convolutional layer?" *arXiv*, 2016.

[73] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv*, 2016.

[74] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1520–1528, 2015.

[75] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proceedings of the European Conference on Computer Vision (ECCV)*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., 2016, pp. 694–711.

[76] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *stat*, vol. 1050, p. 11, 2018.

[77] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016.

[78] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," 2015.

[79] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 38, no. 2, pp. 295–307, 2015.

[80] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Proceedings of the Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[81] X. Li, H. Chen, X. Qi, Q. Dou, C.-W. Fu, and P.-A. Heng, "H-denseunet: Hybrid densely connected unet for liver and tumor segmentation from ct volumes," *IEEE Transactions on Medical Imaging (TMI)*, vol. 37, no. 12, pp. 2663–2674, 2018.

[82] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, "Unet 3+: A full-scale connected unet for medical image segmentation," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1055–1059.

[83] J. Fu, J. Liu, Y. Li, Y. Bao, W. Yan, Z. Fang, and H. Lu, "Contextual deconvolution network for semantic segmentation," *Pattern Recognition*, vol. 101, p. 107 152, 2020.

[84] W.-S. Lai, J.-B. Huang, N. Ahuja, and M.-H. Yang, "Deep laplacian pyramid networks for fast and accurate super-resolution," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 624–632.

[85] T. Tong, G. Li, X. Liu, and Q. Gao, "Image super-resolution using dense skip connections," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4799–4807.

[86] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4681–4690.

[87] H. Lu, Y. Dai, C. Shen, and S. Xu, "Index networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 44, no. 1, pp. 242–255, 2022.

[88] Y. Dai, H. Lu, and C. Shen, "Learning affinity-aware upsampling for deep image matting," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 6841–6850.

[89] H. Lu, W. Liu, H. Fu, and Z. Cao, "Fade: Fusing the assets of decoder and encoder for task-agnostic upsampling," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

[90] J. Wang, K. Chen, R. Xu, Z. Liu, C. C. Loy, and D. Lin, "CARAFE: Content-aware ReAssembly of FEatures," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[91] H. Lu, W. Liu, Z. Ye, H. Fu, Y. Liu, and Z. Cao, "Sapa: Similarity-aware point affiliation for feature upsampling," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, 2022.

[92] H. Wu, S. Zheng, J. Zhang, and K. Huang, "Fast end-to-end trainable guided filter," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1838–1847.

[93] H. Hu, Y. Chen, J. Xu, S. Borse, H. Cai, F. Porikli, and X. Wang, "Learning implicit feature alignment function for semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022, pp. 487–505.

[94] V. Sitzmann, J. N. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, 2020.

[95] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5939–5948.

[96] M. Hamilton, E. Shelhamer, and W. T. Freeman, "It is likely that your loss should be a likelihood," *arXiv*, 2020.

[97] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv*, 2016.

[98] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 7537–7547.

[99] P. Krähenbühl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., vol. 24, 2011.

[100] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[101] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research (JMLR)*, vol. 15, no. 1, pp. 1929–1958, 2014.

[102] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *stat*, vol. 1050, p. 21, 2016.

[103] W. B. Johnson, J. Lindenstrauss, and G. Schechtman, "Extensions of lipschitz maps into banach spaces," *Israel Journal of Mathematics*, vol. 54, no. 2, pp. 129–138, 1986.

[104] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[105] A. Torralba, P. Isola, and W. T. Freeman, *Foundations of Computer Vision*. MIT Press, 2024.

[106] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes," *arXiv*, 2016.

[107] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[108] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 34, pp. 12 077–12 090, 2021.

[109] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *International Journal of Computer Vision (IJCV)*, vol. 127, no. 3, pp. 302–321, 2019.

[110] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[111] W. Liu, H. Lu, H. Fu, and Z. Cao, "Learning to upsample by learning to sample," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 6027–6037.

[112] Y. Dai, H. Lu, and C. Shen, "Learning affinity-aware upsampling for deep image matting," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 6841–6850.

[113] S. Ancha, P. R. Osteen, and N. Roy, "Deep evidential uncertainty estimation for semantic segmentation under out-of-distribution obstacles," 2024.

[114] T. Chen, L. Zhu, C. Deng, R. Cao, Y. Wang, S. Zhang, Z. Li, L. Sun, Y. Zang, and P. Mao, "Sam-adapter: Adapting segment anything in underperformed scenes," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 3367–3375.

[115] G.-P. Ji, D.-P. Fan, P. Xu, B. Zhou, M.-M. Cheng, and L. Van Gool, "Sam struggles in concealed scenes—empirical study on "segment anything"," *Science China Information Sciences (SCIS)*, vol. 66, no. 12, p. 226 101, 2023.

[116] M. J. Farah, *Visual agnosia*. MIT press, 2004.

[117] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[118] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE Access*, vol. 8, pp. 58 443–58 469, 2020.

[119] H. Fujiyoshi, T. Hirakawa, and T. Yamashita, "Deep learning-based image recognition for autonomous driving," *International Association of Traffic and Safety Sciences (IATSS) Research*, vol. 43, no. 4, pp. 244–252, 2019.

[120] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker, "Ambler: An autonomous rover for planetary exploration," *Computer*, vol. 22, no. 6, pp. 18–26, 1989.

[121] M. Massari, G. Giardini, F. Bernelli-Zazzera, *et al.*, "Autonomous navigation system for planetary exploration rover based on artificial potential fields," in *Proceedings of the conference on Dynamics and Control of Systems and Structures in Space (DCSSS)*, 2004, pp. 153–162.

[122] G. Kantor, S. Singh, R. Peterson, D. Rus, A. Das, V. Kumar, G. Pereira, and J. Spletzer, "Distributed search and rescue with robot and sensor teams," in *Field and Service Robotics: Recent advances in research and applications*, 2006, pp. 529–538.

[123] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A.-a. Agha-mohammadi, "Step: Stochastic traversability evaluation and planning for risk-aware off-road navigation," in *Proceedings of Robotics: Science and Systems (RSS)*, 2021.

[124] J. Frey, D. Hoeller, S. Khattak, and M. Hutter, "Locomotion policy guided traversability learning using volumetric representations of complex environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 5722–5729.

[125] A. Valada, G. L. Oliveira, T. Brox, and W. Burgard, "Deep multispectral semantic scene understanding of forested environments using multimodal fusion," in *Proceedings of the International Symposium on Experimental Robotics (ISER)*, 2017, pp. 465–477.

[126] A. Valada, J. Vertens, A. Dhall, and W. Burgard, "Adapnet: Adaptive semantic segmentation in adverse environmental conditions," in *Proceedings of the IEEE/RAS International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4644–4651.

[127] T. Guan, D. Kothandaraman, R. Chandra, A. J. Sathyamoorthy, K. Weerakoon, and D. Manocha, "Ga-nav: Efficient terrain segmentation for robot navigation in unstructured outdoor environments," *IEEE/RAS Robotics and Automation Letters (RAL)*, vol. 7, no. 3, pp. 8138–8145, 2022.

[128] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the International Conference on Machine Learning (ICML)*, PMLR, 2016, pp. 1050–1059.

[129] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. Kruspe, R. Triebel, P. Jung, R. Roscher, *et al.*, "A survey of uncertainty in deep neural networks," *Artificial Intelligence Review*, vol. 56, no. Suppl 1, pp. 1513–1589, 2023.

[130] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 30, 2017.

[131] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2015, pp. 1613–1622.

[132] A. Malinin and M. Gales, "Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 32, 2019.

[133] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 31, 2018.

[134] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 30, 2017.

[135] J. Van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal, "Uncertainty estimation using a single deep deterministic neural network," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2020, pp. 9690–9700.

[136] T. Ramalho and M. Miranda, "Density estimation in representation space to predict model uncertainty," in *Proceedings of the international workshop on Engineering Dependable and Secure Machine Learning Systems (EDSMLS)*, 2020, pp. 84–96.

[137] K. P. Murphy, *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023.

[138] A. Mobiny, P. Yuan, S. K. Moulik, N. Garg, C. C. Wu, and H. Van Nguyen, "Dropconnect is effective in modeling uncertainty of bayesian deep networks," *Scientific Reports*, vol. 11, no. 1, p. 5458, 2021.

[139] A. Amini, A. Soleimany, S. Karaman, and D. Rus, "Spatial uncertainty sampling for end-to-end control," *arXiv*, 2018.

[140] D. Krueger, C.-W. Huang, R. Islam, R. Turner, A. Lacoste, and A. Courville, "Bayesian hypernetworks," *arXiv*, 2017.

[141] J. M. Hernández-Lobato and R. Adams, "Probabilistic backpropagation for scalable learning of bayesian neural networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2015, pp. 1861–1869.

[142] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, "A simple baseline for bayesian uncertainty in deep learning," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 32, 2019.

[143] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2011, pp. 681–688.

[144] M. Valdenegro-Toro, "Deep sub-ensembles for fast uncertainty estimation in image classification," *arXiv*, 2019.

[145] Y. Wen, D. Tran, and J. Ba, "Batchensemble: An alternative approach to efficient ensemble and lifelong learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

[146] V. Peretroukhin, B. Wagstaff, and J. Kelly, "Deep probabilistic regression of elements of so (3) using quaternion averaging and uncertainty injection.," in *Proceedings of the IEEE/CVF International Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019, pp. 83–86.

[147] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[148] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu, "Time series data augmentation for deep learning: A survey," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2021.

[149] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 31, 2018.

[150] M. Raghu, K. Blumer, R. Sayres, Z. Obermeyer, B. Kleinberg, S. Mullainathan, and J. Kleinberg, "Direct uncertainty prediction for medical second opinions," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2019, pp. 5281–5290.

[151] A. Amini, W. Schwarting, A. Soleimany, and D. Rus, "Deep evidential regression," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 33, pp. 14 927–14 937, 2020.

[152] D. T. Ulmer, C. Hardmeier, and J. Frellsen, "Prior and posterior networks: A survey on evidential deep learning methods for uncertainty estimation," *IEEE Transactions on Machine Learning Research (TMLR)*, 2023.

[153] B. Charpentier, D. Zügner, and S. Günnemann, "Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 33, pp. 1356–1367, 2020.

[154] B. Charpentier, O. Borchert, D. Zügner, S. Geisler, and S. Günnemann, "Natural posterior network: Deep bayesian predictive uncertainty for exponential family distributions," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[155] Wikipedia contributors, "Dirichlet distribution: Wikipedia, the free encyclopedia," 2023, [Online; accessed 27-July-2023].

[156] H. Guo, H. Liu, R. Li, C. Wu, Y. Guo, and M. Xu, "Margin & diversity based ordering ensemble pruning," *Neurocomputing*, vol. 275, pp. 237–246, 2018.

[157] J. Wenger, H. Kjellström, and R. Triebel, "Non-parametric calibration for classification," in *Proceedings of the International Conference on Artificial Intelligence and Statistics (ICAIS)*, 2020, pp. 178–190.

[158] Z. Zhang, A. V. Dalca, and M. R. Sabuncu, "Confidence calibration for convolutional neural networks using structured dropout," *arXiv*, 2019.

[159] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection: A review," *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–38, 2021.

[160] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 1–37, 2011.

[161] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 1997, pp. 583–588.

[162] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proceedings of the ACM Special-Interest Group on Knowledge Discovery and Data Mining (SIGKDD) International conference*, 2017, pp. 665–674.

[163] P. Li, T. J. Hastie, and K. W. Church, "Very sparse random projections," in *Proceedings of the ACM Special-Interest Group on Knowledge Discovery and Data Mining (SIGKDD) International conference*, 2006, pp. 287–296.

[164] G. Pang, L. Cao, L. Chen, and H. Liu, "Learning representations of ultrahigh-dimensional data for random distance-based outlier detection," in *Proceedings of the ACM Special-Interest Group on Knowledge Discovery and Data Mining (SIGKDD) International conference*, 2018, pp. 2041–2050.

[165] T. Pevnỳ, "Loda: Lightweight on-line detector of anomalies," *Machine Learning*, vol. 102, pp. 275–304, 2016.

[166] M. A. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[167] C. A. Richter, "Autonomous navigation in unknown environments using machine learning," Ph.D. dissertation, Massachusetts Institute of Technology, 2017.

[168] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016.

[169] R. T. Ionescu, F. S. Khan, M.-I. Georgescu, and L. Shao, "Object-centric auto-encoders and dummy anomalies for abnormal event detection in video," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 7842–7851.

[170] D. Xu, J. Song, Y. Yan, E. Ricci, N. Sebe, *et al.*, "Learning deep representations of appearance and motion for anomalous event detection," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2015, pp. 1–12.

[171] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, "Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks," in *Proceedings of the ACM Special-Interest Group on Knowledge Discovery and Data Mining (SIGKDD) International conference*, 2018, pp. 2672–2681.

[172] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *AAAS Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[173] X. Jiang, J. Gao, X. Hong, and Z. Cai, "Gaussian processes autoencoder for dimensionality reduction," in *Proceedings of the Pacific-Asia conference on Knowledge Discovery and Data Mining (PAKDD)*, 2014, pp. 62–73.

[174] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[175] C. Doersch, "Tutorial on variational autoencoders," *arXiv*, 2016.

[176] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.," *Journal of Machine Learning Research (JMLR)*, vol. 11, no. 12, 2010.

[177] V. Zavrtanik, M. Kristan, and D. Skočaj, "Reconstruction by inpainting for visual anomaly detection," *Pattern Recognition*, vol. 112, p. 107 706, 2021.

[178] X. Dan, E. Ricci, Y. Yan, J. Song, N. Sebe, *et al.*, "Learning deep representations of appearance and motion for anomalous event detection," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2015, pp. 8–1.

[179] T. Nakao, S. Hanaoka, Y. Nomura, M. Murata, T. Takenaga, S. Miki, T. Watadani, T. Yoshikawa, N. Hayashi, and O. Abe, "Unsupervised deep anomaly detection in chest radiographs," *Journal of Digital Imaging*, vol. 34, pp. 418–427, 2021.

[180] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Proceedings of the International Conference on Information Processing in Medical Imaging (IPMI)*, 2017, pp. 146–157.

[181] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "F-anogan: Fast unsupervised anomaly detection with generative adversarial networks," *Medical Image Analysis*, vol. 54, pp. 30–44, 2019.

[182] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2019, pp. 622–637.

[183] S. Akçay, A. Atapour-Abarghouei, and T. P. Breckon, "Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection," in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8.

[184] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv*, 2019.

[185] Y. Zheng, X. Wang, R. Deng, T. Bao, R. Zhao, and L. Wu, "Focus your distribution: Coarse-to-fine non-contrastive learning for anomaly detection and localization," in *2022 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, 2022, pp. 1–6.

[186] F.-A. Croitoru, V. Hondru, R. T. Ionescu, and M. Shah, "Diffusion models in vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2023.

[187] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2015, pp. 2256–2265.

[188] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[189] Wikipedia contributors, "Cosine Similarity: Wikipedia, the free encyclopedia," 2024, [Online; accessed 14-May-2024].

[190] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, "A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 5000–5007.

[191] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3213–3223.

[192] H. Blum, P.-E. Sarlin, J. Nieto, R. Siegwart, and C. Cadena, "Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, 2019.

[193] R. Chan, K. Lis, S. Uhlemeyer, H. Blum, S. Honari, R. Siegwart, P. Fua, M. Salzmann, and M. Rottmann, "Segmentmeifyoucan: A benchmark for anomaly segmentation," in *Proceedings of the international conference on Neural Information Processing Systems (NIPS) Workshops*, J. Vanschoren and S. Yeung, Eds., vol. 1, 2021.

[194] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. HAZIZA, F. Massa, A. El-Nouby, *et al.*, "Dinov2: Learning robust visual features without supervision," *IEEE Transactions on Machine Learning Research (TMLR)*, 2023.

[195] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 586–595.

[196] S. Fu, N. Tamir, S. Sundaram, L. Chai, R. Zhang, T. Dekel, and P. Isola, "Dreamsim: Learning new dimensions of human visual similarity using synthetic data," *Proceedings of the international conference on Neural Information Processing Systems (NIPS)*, vol. 36, 2024.

[197] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing (TIP)*, vol. 13, no. 4, pp. 600–612, 2004.

[198] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2021, pp. 8748–8763.

[199] X. Dong, J. Bao, Y. Zheng, T. Zhang, D. Chen, H. Yang, M. Zeng, W. Zhang, L. Yuan, D. Chen, *et al.*, "Maskclip: Masked self-distillation advances contrastive language-image pretraining," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 10 995–11 005.

[200] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2901–2910.

[201] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang, "A survey of deep active learning," *ACM Computing Surveys (CSUR)*, vol. 54, no. 9, pp. 1–40, 2021.

[202] K. W. Church, Z. Chen, and Y. Ma, "Emerging trends: A gentle introduction to fine-tuning," *Natural Language Engineering*, vol. 27, no. 6, pp. 763–778, 2021.

[203] AI Curious, "Anylabeling - effortless data labeling," 2023, [Online; accessed 14-May-2024].

[204] M. Kettunen, E. Härkönen, and J. Lehtinen, "E-lpips: Robust perceptual image similarity via random transformation ensembles," *arXiv*, 2019.

[205] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: A literature survey," *Artificial Intelligence Review*, vol. 42, pp. 275–293, 2014.

[206] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data mining and knowledge discovery*, vol. 8, no. 4, 2018.

[207] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[208] L. Brandt and N. Roy, "Fusion for reducing domain specificity in computer vision models," *Bulletin of the American Physical Society*, 2023.

[209] K. Sirohi, S. Marvi, D. Büscher, and W. Burgard, "Uncertainty-aware panoptic segmentation," *IEEE/RAS Robotics and Automation Letters (RAL)*, vol. 8, no. 5, pp. 2629–2636, 2023.

[210] J. Deery, C. W. Lee, and S. L. Waslander, "Propandl: A modular architecture for uncertainty-aware panoptic segmentation," in *Proceedings of the IEEE Conference on Robots and Vision (CRV)*, 2023, pp. 137–144.

[211] D. J. Felleman and D. C. Van Essen, "Distributed hierarchical processing in the primate cerebral cortex," *Cerebral Cortex*, vol. 1, no. 1, pp. 1–47, 1991.

[212] M. Awais, M. Naseer, S. Khan, R. M. Anwer, H. Cholakkal, M. Shah, M.-H. Yang, and F. S. Khan, "Foundational models defining a new era in vision: A survey and outlook," *arXiv*, 2023.

[213] M. A. Webster, "Light adaptation, contrast adaptation, and human colour vision," *Colour Perception: Mind and the physical world*, pp. 67–110, 2003.

[214] R. W. Pridmore, "Complementary colors theory of color vision: Physiology, color mixture, color constancy and color perception," *Color Research & Application*, vol. 36, no. 6, pp. 394–412, 2011.

[215] J. M. Wolfe, "Visual search: How do we find what we are looking for?" *Annual Review of Vision Science*, vol. 6, pp. 539–562, 2020.

[216] M. S. Castelhano and J. M. Henderson, "The influence of color on the perception of scene gist," *Journal of Experimental Psychology: Human perception and performance*, vol. 34, no. 3, p. 660, 2008.

[217] A. Oliva, "Gist of the scene," in *Neurobiology of Attention*, 2005, pp. 251–256.

[218] Z.-L. Lu and G. Sperling, "The functional architecture of human visual motion perception," *Vision Research*, vol. 35, no. 19, pp. 2697–2722, 1995.

[219] D. Marr, *Vision: A computational investigation into the human representation and processing of visual information*. MIT press, 2010.