Towards Understanding Human-aligned Neural Representation in the Presence of Confounding Variables

by

Sanja Simonovikj

B.S. Computer Science and Engineering
Massachusetts Institute of Technology (2020)

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

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2021

Author . . . .
Department of Electrical Engineering and Computer Science
May 18, 2021

Certified by . . . .
Pulkit Agrawal
Assistant Professor
Thesis Supervisor

Accepted by . . . .
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Towards Understanding Human-aligned Neural Representation in the Presence of Confounding Variables

by

Sanja Simonovikj

Submitted to the Department of Electrical Engineering and Computer Science
on May 18, 2021, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Deep Neural Networks (DNNs) find one out of many possible solutions to a given task such as classification. This solution is more likely to pick up on spurious features and low-level statistical patterns in the train data rather than semantic features and high-level abstractions, resulting in poor Out-of-Distribution (OOD) performance. In this project we aim to broaden the current knowledge surrounding spurious correlations as they relate to DNNs. We do this by measuring their effect on generalization under various settings, determining the existence of subnetworks in a DNN that capture the core features and examine potential mitigation strategies. Finally, we discuss alternative approaches that are reserved for future work.

Thesis Supervisor: Pulkit Agrawal
Title: Assistant Professor

Acknowledgments

I would like to express my deep and sincere gratitude to my research and thesis supervisor, Professor Pulkit Agrawal, for his unwavering support and exceptional guidance in the past year and a half, and for always being generous with his time and dedicated in his advice, ranging from the big picture to tackling small details. I will carry his lessons with me for years to come.

I would also like to extend my sincere thanks to Minyoung Huh, PhD student in MIT, for his continued advice and practical suggestions. During many hours of meetings and conversations, he has helped me in developing research skills and mathematical intuition.

I gratefully acknowledge the assistance of the Improbable AI lab members, including but not limited to Hyojin Bahng, Anurag Ajay, Brian Cheung and Avery Lamp, who provided specialized knowledge and advice that was helpful for realization of this project.

Last but not least, I would like to thank my friends and family for their love and encouragement throughout the years, many of whom are far but feel really close. Their support made all of this possible.

Contents

List of Figures

List of Tables

Introduction

Deep Neural Networks (DNNs) have shown superior generalization ability despite their capacity to fit random data [30]. However, many works have pointed out the brittleness of DNNs in light of adversarial examples [24] and out-of-distribution examples [18] [16]. In practice, DNNs tend to pick up on spurious features and low-level statistical patterns in the train data rather than semantic features and high-level abstractions, resulting in poor Out-of-Distribution (OOD) performance.

As an example, consider the task of classifying images into two classes: digits two and six. Suppose the training set is biased so that background of images of digits two and six are colored in red and blue respectively. There are at least two distinct features that can be employed to classify these images: color or shape. If a DNN is trained on this biased data and tested on a dataset that lacked any correlation between the digit class and the background color, one would notice that DNN is perfect at the training task, but performs at chance on the testing task. This clearly shows that the DNN captures color, but fails to capture shape. Has the DNN learned the "right" feature? In some scenarios, where the data is generated from an underlying causal model, such as physics describing the motion of a ball, it is straightforward to distinguish the right feature (e.g., mass) from a spurious feature (e.g., color).

However, for many problems of interest such as object identification, speech recognition or machine translation the target labels are human defined (i.e., subjective) and there exists no causal model. For these problems, if a DNN learns features that are different from humans, then to a human, the DNN's performance on new data is likely to be counter-intuitive. The DNN would be said to have learned "spurious" features. E.g., in the digit classification example "shape" is the true underlying feature whereas “color“ is the spurious feature.

This study aims to further the current knowledge of deep learning phenomena related to learning reliable and robust features. It also presents a new approach to analyze and compare the types of solutions learned by a DNN and it provides an overview of a preliminary attempt to learn reliable solutions that overcome the undesirable inductive biases in DNNs. Finally, it recommends directions for future research.

This paper is organized as follows. In chapter 1 we begin by giving a background on the problem of poor OOD performance and examining the existing literature on learning core features. The next chapter describes the setup and data used throughout the paper and demonstrates the reliance of DNNs on spurious features. In the third chapter we analyze the effect of NN depth on generalization across biased datasets and degree of bias and we look at the correlation between the simplicity of a solution and distance to initialization. In chapter 4 we give a formal definition of simplicity leveraging the Lottery Ticket Hypothesis (LTH) as well as investigate the existence of subnetworks in a DNN that capture the core features. In chapter 5 we propose a new evolutionary training strategy to mitigate the issues outlined in the earlier chapters. Finally, we provide a summary, address limitations and we propose areas of further research in chapter 6.

Chapter 1

Background and related work on learning core features

1.1 Background

In the past few years the field of deep learning has seen a significant progress. A remarkable success has been achieved in the large-scale image classification on the challenging ImageNet benchmark [19], as well as smaller standard datasets such as CIFAR-10 [13] and MNIST [14]. Beyond achieving high accuracy in these kinds of tasks, a major appeal of deep learning is the ability to learn effective *feature representations* of data (also known as *embeddings*). A major goal in representation learning is to obtain embeddings that encode high-level, more interpretable and more *human-aligned features* (HAF). Indeed, the learned feature representations turn out to be quite versatile - in the field of computer vision for example, they are the driving force behind transfer learning and image-similarity metrics such as VGG [12].

While the utility of learned representations is clear, the inductive biases in model class and training algorithms for Deep Neural Networks (DNNs) exhibit some shortcomings. The presence of *adversarial examples*, where an input perturbed in a way imperceptible to the human eye causes the model to predict a wrong output, suggests that DNNs make predictions based on features that are vastly different from what humans use. This same conclusion can be derived when looking at the failure of DNNs to generalize on out-of-distribution examples, something a typical human would normally have no trouble with. Further empirical studies done in this project and others illustrate that DNNs often rely on features, such as the background, that may be spuriously correlated with the label during training time, resulting in poor accuracy during test-time.

The issue of DNNs and humans using different features to solve the same task is widespread. E.g., [7] find that ImageNet-trained CNNs are strongly biased towards recognising textures rather than shapes, which is in stark contrast to human behavioural evidence and reveals fundamentally different classification strategies; [29] find that despite training on video data, neural networks prefer static cues over temporal ones. While learning of some spurious features can be attributed to biases in the training data, [28] report that balanced datasets by themselves do not mitigate this issue. Learning of such spurious features can be seen as the root cause of the aforementioned challenges that DNNs are faced with: poor true generalization, lack of robustness and interpretability [11]. We therefore believe that these problems can be mitigated by making DNNs learn human-aligned features (HAF).

The main deterrent in learning HAFs is that just given a training set and a powerful learner, there are multiple possible solutions and without priors it is impossible to decide the correct one. Humans possess background knowledge that guides this selection. In machine learning, the role of feature selection has been relegated to regularization. Widely used regularization penalties are based on the *occam's razor*, a heuristic, that encourages the determination of simpler solutions. However, these heuristics have proven insufficient at explaining the success of deep learning. For instance, deeper neural networks with more parameters generalize better than (simpler) DNNs with fewer parameters. Data augmentation/adversarial training [9] are other popular ways of regularizing training. Learning features that can quickly adapt to a new problem [27, 5] or find the causal structure [3] can be seen as yet another form of regularization. However, these mechanisms do-not fundamentally address the problem as they optimize features for transfer to hand-chosen datasets.

The aim of our work is to broaden the current knowledge surrounding spuriously correlated features present in training data. This involves exposing the problem with spurious correlations and DNNs, measuring their effect on generalization under various settings, determining the existence of subnetworks that capture the core features (as opposed to the spuriously correlated ones) and examine potential mitigation strategies. Finally, we discuss alternative approaches that are reserved for future work.

1.2 Related work

1.2.1 Spurious correlations

An increasing number of studies investigate the reliance on spurious features such as background in an image to make a prediction. In [21] the authors find that of all overparameterized models that achieve zero training error, the inductive bias of the model class and training algorithm favors models that use spurious features which generalize only for the majority groups, instead of learning to use core features that also generalize well on the minority groups. This inductive bias favors memorization of as few points as possible. A related work looks at increased regularization as a successful mitigation strategy [20], suggesting that increasing model capacity by reducing regularization and perhaps by increasing overparameterization as well can exacerbate spurious correlations.

1.2.2 Simplicity bias in neural networks

A growing body of literature has studied the simplicity bias (SB) in neural networks - the tendency of standard training procedures such as Stochastic Gradient Descent (SGD) to find simple models [1], [26]. In particular, [23] find that neural networks can exclusively rely on the simplest feature and remain invariant to all predictive complex features. Contrary to conventional wisdom, SB can also hurt generalization on the same data distribution as SB persists even when the simplest feature has less predictive power than the more complex features. The authors also find that the common approaches to improve generalization and robustness — ensembles and adversarial training — can fail in mitigating SB and its pitfalls.

1.2.3 Data augmentation

Data augmentation is a commonly used technique for increasing both the size and the diversity of labeled training sets by leveraging input transformations that preserve corresponding output labels. In computer vision, image augmentations have become a common regularization technique to combat overfitting in deep convolutional neural networks and are ubiquitously used to improve performance on various tasks [17]. Image augmentations have also been shown to improve convergence [15], generalization and robustness on *out-of-distribution* samples [2] [10], and to overall have more advantages compared to other regularization techniques.

1.2.4 Adversarial robustness

Other related work looks at adversarial robustness as a prior towards learning better representations. From a transfer-learning perspective, the work done in [22] indicates that adversarially trained models achieve better transfer accuracy than their standard counterparts. From a feature learning perspective, [4] claim that robust optimization can be used as a regularizer for learning representations by neural networks, resulting e.g. in more semantically meaningful representations, which are approximately invertible, while allowing for direct visualization and manipulation of salient input features. While adversarial robustness seems to be a useful prior for learning more semantically meaningful features, it comes at a cost: [25] discover that there a may exist an inherent tension between the goal of adversarial robustness and that of standard generalization. Specifically, training robust models may not only be more resource-consuming, but also lead to a reduction of standard accuracy, and that this can be attributed to the fact that robust models and standard models might depend on a very different set of features.

Chapter 2

DNN performance in the presence of spurious correlations

In this chapter we will first go over the definition of spurious correlations and provide a general setup and then we will demonstrate the issue of spurious correlations and generalization.

2.1 Spuriously correlated and core features

2.1.1 Definition and example

In statistics, a *spurious relationship* or *spurious correlation* is a mathematical relationship in which two or more events or variables are associated but not causally related, due to either coincidence or the presence of a certain third, unseen factor (referred to as a "common response variable", "confounding factor").

An example which will be relevant for this work is shown in figure 2-1, where we have images of the digits two and six with colored background, labeled 0 and 1 respectively. In this case, the background color (red or blue) is spuriously correlated with the label (0 or 1 respectively) in the task of *digit classification.* A neural network model trained with such data might infer that a red background implies a label 0 and blue background implies a label 1. In reality, there is a third variable that affects both the background and label. In particular, the shape of the digit, two or six, determines whether the label is 0 or 1 respectively, and is also correlated with the background color, either by design or by chance. In this example, the shape of the digit is the confounding variable. We will refer to the shape as a **core feature**, and to the background color as **spurious feature**. Throughout our work we will use the words *spurious feature* and *bias* interchangeably, in accordance to using the term bias to indicate the systematic error arising from incorrect assumptions in the training process.

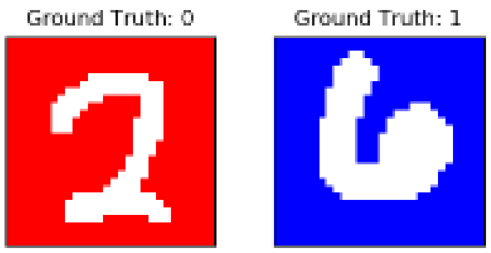


Figure 2-1: Example of spurious relationship: the background color (red or blue) is spuriously correlated with the label (0 or 1 respectively) in the task of *digit classification*. The shape of the digit (2 or 6) is the confounding variable.

2.1.2 Setup

We consider the simplified setup where each datapoint x with label $y \in \mathcal{Y}$ contains some spurious feature or attribute $a \in \mathcal{A}$ in addition to the core features. Each example belongs to a group $g \in \mathcal{G} = \mathcal{Y} \times \mathcal{A}$, where $g = (y, a)$. Importantly, the spurious feature a is correlated with the label y in the training data, but it is not correlated in the test data. In the instance above we have the binary setting where $\mathcal{Y} = \{0, 1\}$ and $\mathcal{A} = \{red, blue\}$.

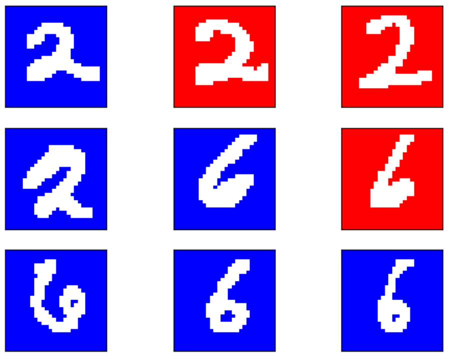
2.2 Datasets with spurious correlations

We curated multiple balanced binary classification datasets where there is a spurious relationship between the variables in the training set but not in the testing set.

Note that the testing set is perfectly balanced, therefore the spurious feature appears equally in each class i.e. $|D^{y_1a_1}| = |D^{y_1a_2}| = |D^{y_2a_1}| = |D^{y_2a_2}|$, where D^{ya} is the dataset of images with spurious feature a and label y. In some experiments we will want to compare performance of a network trained with biased set with that of a network trained with an unbiased set. In that case, both the training and test set are unbiased and perfectly balanced across classes.

2.2.1 Biased Color MNIST dataset

We create the Biased Color MNIST Dataset by taking the digits two and six of the MNIST dataset [14], and coloring the backgrounds with the colors red and blue. Specifically, the train set consists of instances such as $(0, red)$ and $(1, blue)$, while the test set has all four groups, corresponding of the combination of the labels y and spurious feature a. Note that in this case the background color is spuriously correlated with the label. We have 5938 training images, and 1990 testing images.


(a) Test set samples

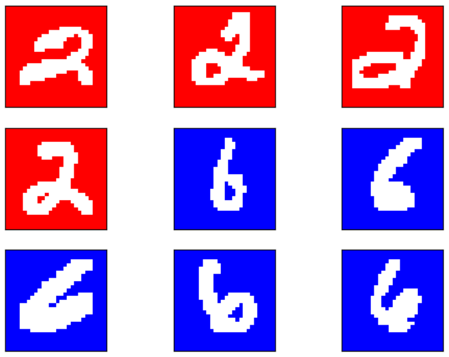
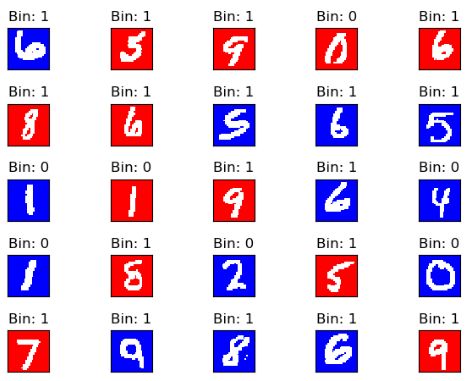

(b) Train set samples

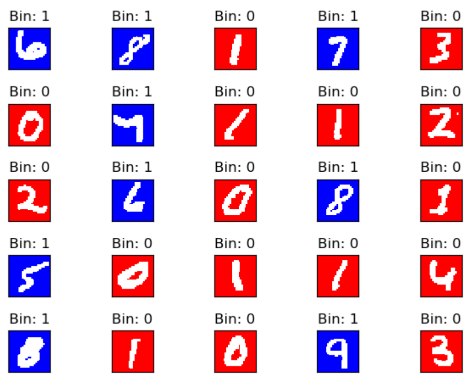
Figure 2-2: Biased Color MNIST dataset; the background color is spuriously correlated with the label in the train set

2.2.2 Biased Two-bin MNIST

The Biased Two-bin MNIST dataset consists of the 10 digits of MNIST where the digits 0-4 get one label (0) and the digits 5-9 get another label (1). We have train set where the backgrounds associated with labels 0 and 1 are red and blue respectively, while in the test set there is no such correlation. Note that this dataset, although similar to the biased color MNIST dataset in 2.2, is more challenging due to the presence of all 10 digits. We have 30000 training images, and 10000 testing images.



(a) Test set samples



(b) Train set samples

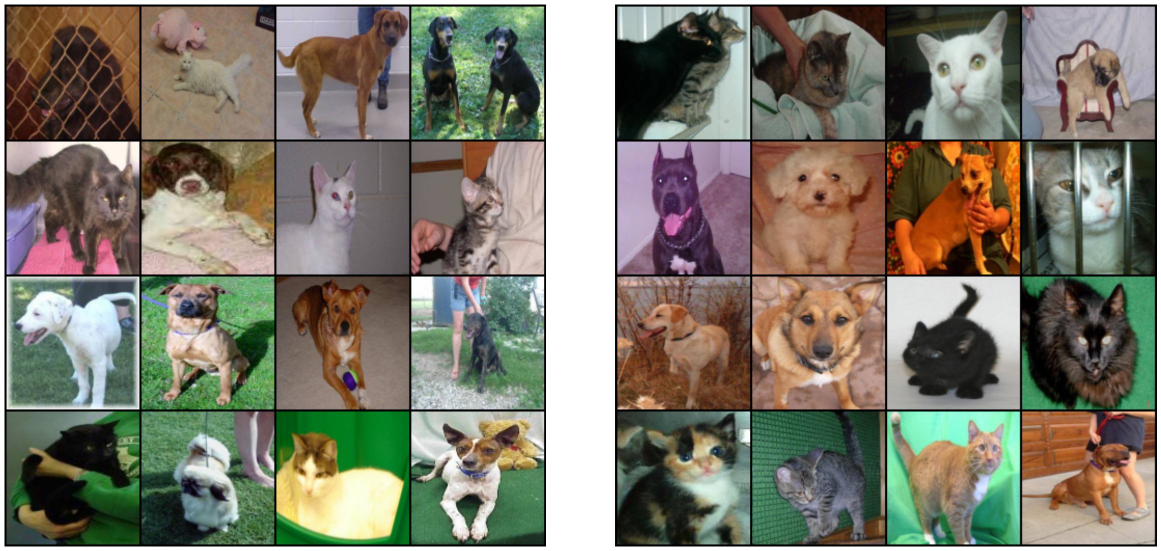
Figure 2-3: Biased Two-bin MNIST dataset consists of all 10 digits and 2 background colors; the bias element is the background color associated with the label

2.2.3 Biased Cats and Dogs

The Biased Cats and Dogs dataset consists of images of cats and dogs with different types of background: outdoors vs indoors (Figure 2-4a). To get images with outdoors background we use an HSV (Hue, Saturation, Value) representation and we get the top "greenest" images based on the percentage of green pixels, where a pixel is considered green if its Hue channel is between 80 and 110. The type of background is spuriously correlated with the labels in the train set (Fig. 2-4b). We note that this dataset contains real images of cats and dogs, which differs from our other datasets which are synthetically constructed. We have 140 training images, and 120 testing images.

2.3 Experiment: Simplicity bias and generalization

Simplicity bias (SB) in neural networks refers to the tendency of standard training procedures such as Stochastic Gradient Descent (SGD) to find simple models. Previous work has found that neural networks can exclusively rely on the simplest feature



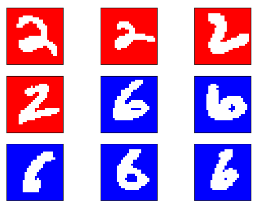
(a) Test set samples

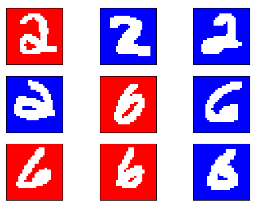
(b) Train set samples

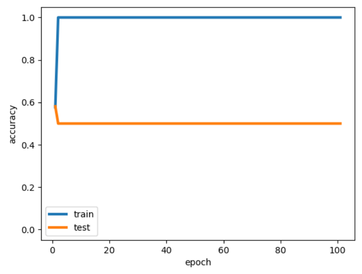
Figure 2-4: Biased Cats and Dogs dataset; the type of background (outdoors vs indoors) is spuriously correlated with labels in the train set

and remain invariant to all predictive complex features [23].

Using our Biased MNIST dataset (2.2.1) as an example and using the notion of simplicity as defined by [23], we can consider the background color of the image to be a simple feature, and the shape of the digit to be a more complex feature. We also provide a more precise definition of simplicity in chapter 4 . If we train a standard Convolutional Neural Network (CNN) with SGD using our biased train set, and evaluate the trained model on an unbiased test set, we notice that while the CNN perfectly fits the training data, it performs at chance in the testing data. This indicates that the NN learns to use color instead of the shape of the digits as the dominant feature for making classification decisions (Fig. 2-5), demonstrating the simplicity bias. These results hold across all of our biased datasets in section 2.2, as well as others not shown in this work that contain spurious correlations in the same fashion.


(a) biased train set


(b) unbiased test set



(c) Train/test accuracy

Figure 2-5: (a) A neural network (NN) is trained on a dataset in which all images of digits two and six have been colored in red and blue respectively. (b) The performance of this network is evaluated on an unbiased dataset where color and digit class have no correlation; (c) Classification accuracy on training and test sets indicates that while the NN perfectly fits the training data, it is at chance performance on the testing set. This result indicates that the NN learns to use color instead of the shape of the digits as the dominant feature for making classification decisions.

Chapter 3

Investigation of the effect of NN depth and distance to initialization on the type of learned solution

3.1 Effect of NN depth on generalization across biased datasets and degree of bias

In section 2.3 we found that a standard high-capacity and deep NN performs at chance on an unbiased test set when trained on a biased set. However, would this still be the case if the network were shallower? In particular, *are shallower networks more likely to capture the core features in the train set that generalize better?*

We want to investigate the effect of the depth of a Deep Neural Network on the generalization performance when there are spurious correlations in the training data. To this end we explore four CNNs (Convolutional Neural Networks) of varying depth, that is CNNs containing 1, 2, 3 and 4 convolutional layers. In addition, we vary the type of bias and the degree of the bias. We curate three simple datasets with three biases: background color, shape and texture (Figure 3-1). We also vary the bias degree/intensity in the training set, denoted by α, where $\alpha = 1$ indicates full bias, and $\alpha = 0$ indicates no bias. To be precise, given bias degree α, bias_mask, and digit, the final image is obtained as: $img = (1 - \alpha) \cdot digit + \alpha \cdot bias\_mask$.

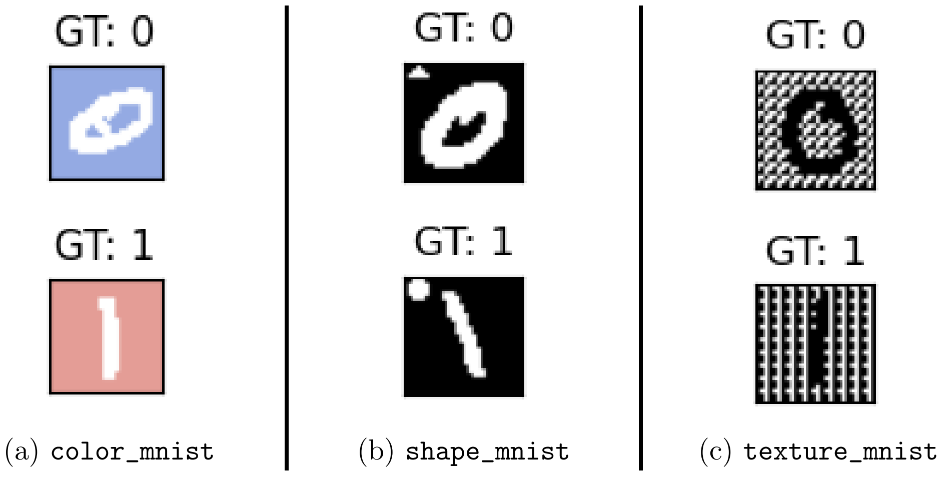


Figure 3-1: Samples from datasets with spurious feature a, where a is the background color in (a), the small symbol (shape) in the top left corner of an image (b) and the background texture (c). The bias degree in this case is 1 (full bias)

We run 10 experiments with different random seeds for each (model, dataset, bias degree) configuration. In each experiment, we train with a biased version of the dataset, and evaluate on the unbiased test set. The test accuracy curves across different models (varying depth) and datasets for a fixed small bias degree of $\alpha = 0.01$ are given in Figure 3-2. We mark in red the average test curve across different trials (one for each random seed). We notice that on average, for a small bias degree, shallower networks are more likely to capture the core digit shape features as opposed to the bias (Figure 3-2).

The complete results are shown in Figure 3-3. As an evaluation metric we use Area Under the Curve (AUC), computed using the trapezoidal rule, of the average test accuracies curve across trials. We use the test accuracy on the unbiased set because that is indicative of whether we have learned the right features. We use AUC as a metric as opposed to only the final test accuracy in order to capture cases where the test accuracy drops over epochs. For each of the four models of varying depth, we explore the performance when the model is trained with biased data with bias $a \in \{color, shape, texture\}$ and bias degree $\alpha \in \{0.01, 0.05, 0.5, 1\}$.

Based on the results in Figure 3-3, if the bias degree is not low/high enough to

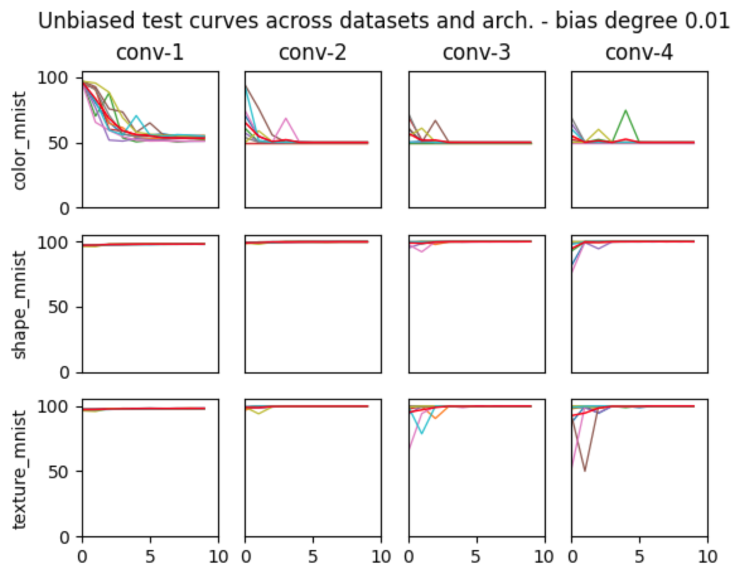
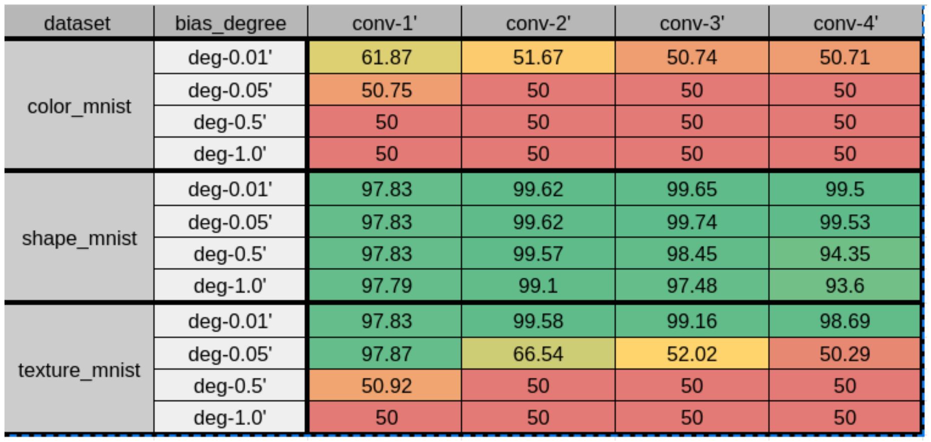


Figure 3-2: Test accuracy curves across 10 trials for models of varying depth and varying biased datasets for a fixed small bias degree of $\alpha = 0.01$. The average curves are marked in red.



dataset	bias_degree	conv-1'	conv-2'	conv-3'	conv-4'
color_mnist	deg-0.01'	61.87	51.67	50.74	50.71
	deg-0.05'	50.75	50	50	50
	deg-0.5'	50	50	50	50
	deg-1.0'	50	50	50	50
shape_mnist	deg-0.01'	97.83	99.62	99.65	99.5
	deg-0.05'	97.83	99.62	99.74	99.53
	deg-0.5'	97.83	99.57	98.45	94.35
	deg-1.0'	97.79	99.1	97.48	93.6
texture_mnist	deg-0.01'	97.83	99.58	99.16	98.69
	deg-0.05'	97.87	66.54	52.02	50.29
	deg-0.5'	50.92	50	50	50
	deg-1.0'	50	50	50	50

Figure 3-3: The Area Under the Curve (AUC) of the average unbiased test accuracies curve across trials for each (model, dataset, bias degree). Shallower networks are more likely to capture the core digit shape feature.

make models of all depths generalize or not generalize well respectively, then it is the shallower networks that are more likely to capture the right feature (the digit shape) initially. We do notice though that in the case of `color_mnist` dataset the test accuracy drops to random the more we train, demonstrating the preference for "simpler" solutions which can fit the train data but do not generalize well.

3.2 Correlation between simplicity of a solution and distance to initialization

Previously we found that DNNs tend to use the spurious features in a biased training dataset and not the desired core features as a predictive signal. In this section we want to investigate if the biased solutions are closer to initialization than the unbiased ones. The distance to initialization is measured as the sum of the L2 distances between the final weights W_{final} and initial weights W_{init} of the NN:

$$d\left(W_{final}, W_{init}\right) = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n_m} \left(W_{final,i,j} - W_{init,i,j}\right)^2}$$

where m is the number of weight matrices in the NN and n_m is the number of weights in matrix m. As we use SGD to train the network, we use this metric as a proxy of the amount of "work" that SGD does to learn the predictive signals.

3.2.1 Biased and unbiased solutions

A biased solution is one obtained by training a network with a biased dataset. In this set of experiments we want to see if biased solutions are closer to initialization than their non-biased counterparts. To this end we train CNNs of varying depth and complexity while ensuring the starting initialization is the same for the biased and unbiased solutions. We use the same three simple datasets with background color, shape and texture biases shown in Figure 3-1. Note that the biased and unbiased datasets are the same size, and they are perfectly balanced across classes and biases. We run five trials of each experiment and we show the means and the average of the L2 distance across trials in Table 3.1. We notice that in most cases the solution obtained by training with biased dataset (one that learns the bias) is closer to initialization than the one obtained by training with unbiased data. However, this is not necessarily the case with very large models such as ResNets [8]. Further studies will need to be undertaken to explain this correlation and whether the distance to initialization is indicative on the type of solution (biased or non-biased) learned.

Dataset	Model	Biased		Unbiased	
		mean	stdev	mean	stdev
Color MNIST	conv-1	**4.26**	0.39	8.31	2.13
	conv-2	**3.92**	0.32	8.79	0.13
	conv-3	**4.04**	0.27	11.31	0.75
	conv-4	**3.94**	0.25	14.88	4.21
	ResNet-18	53.9	21.77	**53.22**	20.17
	ResNet-34	161.94	177.01	**147.76**	99.78
Shape MNIST	conv-1	9.21	0.19	**8.49**	0.19
	conv-2	**8.16**	1.14	8.44	0.05
	conv-3	**7.87**	4.74	9.89	1.65
	conv-4	**5.36**	0.53	10.85	4.05
	ResNet-18	**29.03**	10.91	33.05	4.75
	ResNet-34	78.12	37.85	**75.81**	23.4
Texture MNIST	conv-1	**4.31**	0.07	6.86	0.33
	conv-2	**4.03**	0.19	8.71	0.11
	conv-3	**3.92**	0.07	11.3	0.12
	conv-4	**3.87**	0.1	13.3	2.46
	ResNet-18	102.16	55.28	**40.61**	11.88
	ResNet-34	105.12	106.73	**71.13**	5.89

Table 3.1: L2 distance $d\left(W_{final}, W_{init}\right)$ from final to initial weights for model trained with biased data and unbiased data. In most cases the solution obtained by training with biased dataset (one that learns the bias) is closer to initialization (bolded) than the one obtained by training with unbiased data.

Chapter 4

Existence of Lottery Subnetworks that Capture Core Features

In this chapter we will provide a formal definition of a simplicity of a solution. We will also investigate the hypothesis that larger deep neural networks contain within themselves smaller subnetworks which can extract different high-level features, such as shape and color. To achieve this we will use our curated datasets from section 2.2 and leverage the *Lottery Ticket Hypothesis* (LTH) [6].

4.1 Motivation

The empirical studies in section 3.1 suggest that deeper and higher-capacity neural networks are more likely than their shallower counterparts to use the spurious features as a predictive signal in a biased training dataset. In this section we investigate whether we can find lower-capacity (smaller) subnetworks in a large DNN that are able to use the core features as a predictive signal. If we can empirically prove the existence of such networks, we can look for ways to uncover these subnetworks during training with only biased data. We attempt the former in this section and the latter in section 5.1.

4.2 Background on Lottery Ticket Hypothesis (LTH)

The Lottery Ticket Hypothesis [6] states that dense, randomly-initialized, feed-forward networks contain subnetworks ("winning tickets") that - when trained in isolation - reach test accuracy comparable to the original network in a similar number of iterations. To uncover such subnetworks one can use a standard pruning technique of removing a percentage of the largest weights in the network for the current pruning level and that would naturally uncover subnetworks whose initializations made them capable of training effectively.

The lottery subnetwork can be uncovered by a method called iterative magnitude pruning (IMP): Starting from a dense initialization we train our network until convergence. Afterwards, we prune $p \cdot 100\%$ smallest magnitude weights, where $0 < p < 1$ is the *pruning factor*. We then retrain the sparsified network with its original initialization. After convergence we repeat the pruning process and reset to the initial weights. We iterate this process (s pruning levels in total) until we reach the desired level of sparsity or the test accuracy drops significantly. For more details please refer to [6] .

4.2.1 Definition of Simplicity using LTH

Now that we have the necessary apparatus, we can provide a precise definition of simplicity leveraging the LTH idea of iterative pruning and retraining. This will allow us to characterize solutions learned with biased datasets as simpler than those learned with unbiased datasets.

Definition 1 (Simplicity of a Solution)**.** *Given a Deep Neural Network (DNN) and two datasets A and B, we say that the solution the DNN learns while being trained with dataset A is* ***simpler*** *than the solution learned while being trained with dataset B if, for a given pruning factor p, we can iteratively prune the DNN trained with A for at least as large number of levels $s_A \geq s_B$ than the one trained with B before the DNN converges at any threshold difference $\delta\%$ ($0 < \delta < 50$) lower training accuracy than when trained to convergence before any pruning.*

Dataset A	Dataset B	p	δ	s_A	s_B
biased Color MNIST	unbiased Color MNIST	0.5	1%	**7**	5
biased Two-Bin MNIST	unbiased Two-Bin MNIST	0.5	1%	**7**	6
biased Cats and Dogs	unbiased Cats and Dogs	0.5	15%	**3**	3

Table 4.1: Simplicity of DNN solutions obtained through biased and unbiased datasets based on Definition 1 for a given pruning factor p and threshold difference δ. We note that $s_A \leq s_B$ i.e. biased solutions are "simpler" than unbiased ones.

In Table 4.1 we demonstrate the simplicity of the biased solutions in comparison to the unbiased solutions using definition 1 for our three datasets described in section 2.2. We see that for $p = 0.5$ and an arbitrary δ, $s_A \geq s_B$, where s_A is the number of pruning levels for the solutions obtained with a biased dataset, and s_B for the unbiased dataset.

4.3 Experiment setup

We want to know if is there a subnetwork in a larger network that can capture the core features in a biased dataset. To answer this question we design an experiment described below.

Similar to other experiments in this work, we have biased and unbiased set of multiple datasets. In this case we use the Color MNIST Biased dataset (2.2.1) and Two-bin biased MNIST dataset (2.2.2). Our experiments consist of first performing the routine described in Section 4.2 to obtain a lottery subnetwork from a larger network by training on unbiased data. This subnetwork is presumed to be able to capture the core features as it has access to unbiased data. Let us call this subnetwork "shape" subnetwork as it has learned the digit shapes as opposed to spurious features such as the background color. The next part of the experiment is to take the shape subnetwork, reinitialize it with the fortuitous initialization of the starting network and train it only with biased data. We want to see whether we can still reach good performance on an unbiased test data even though we are training only with biased set. A positive answer to this question would mean that the subnetwork we found has

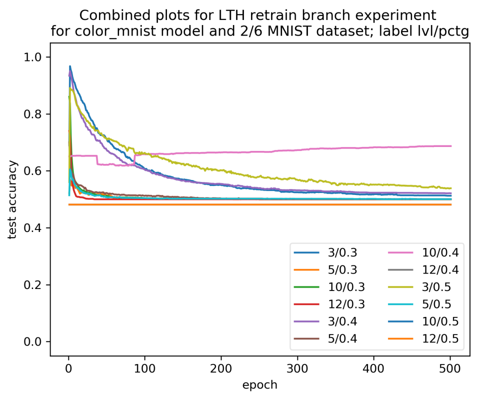
such an initialization and architectural structure that allows for capturing the correct prediction signals. We compare performance by looking at an unbiased test set.
The experiment setup can be summarized as follows:

1. Train on *unbiased* data and perform iterative pruning for s levels with a pruning factor p per level
2. According to LTH there is a subnetwork with similar performance to the starting network. We can say that this subnetwork captures the digit shape features
3. Take the subnetwork from 2), initialize it with the same fortuitous initialization and train it on *biased* data
4. Evaluate the network from 3) on unbiased data
5. If good performance on unbiased data in 4) (even though trained with biased data only), it means the subnetwork had the right initialization/architecture to capture the shape.
6. If 5) is true, it means there is a subnetwork that could capture the shape, and we should find ways to preserve that subnetwork when training the initial network, i.e. do not allow the network that finds the spurious features solution to prevail

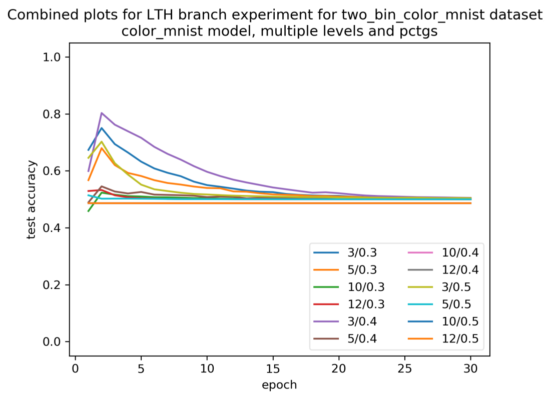
We run multiple versions of this experiment where we vary the datasets and pruning levels and percentage.

4.4 Results

The results are shown in figure 4-1. On the left plot we show the unbiased test learning curve over epochs for the Biased Color MNIST dataset 2.2.1, while on the right we show the same for Biased Two-bin MNIST dataset 2.2.2. Each curve corresponds to a different pruning amount where the label s/p means pruning $p \cdot 100$ % of weights per level for s levels. For both datasets we observe the presence of a peak in the (unbiased) test accuracy learning curve early in the training process (with biased data). While the amount of pruning seems to have an effect on the outcome, in most cases the peak is sharp (short-lasted) and then it flattens around random accuracy. From this we can conclude that the "lottery" subnetwork is learning the core features initially, but then it resorts to learning the spurious features, again demonstrating simplicity bias.



(a) Biased Color MNIST dataset (2.2.1)



(b) Biased Two-bin MNIST dataset (2.2.2)

Figure 4-1: Test accuracy learning curves for two datasets; each line corresponds to the test curve for different pruning amount where s/p means pruning $p \cdot 100$ % of weights per level for s levels. In most cases we notice a peak in the learning curve which then flattens out to random accuracy.

Chapter 5

Explored Mitigation Strategies

In this chapter we look at one potential mitigation strategy with the goal of learning the core features in the presence of spurious correlations in the data. In our discussion we focus on techniques that modify the learning process as opposed ones that modify the training data.

5.1 Diverse Hypotheses through Evolutionary Training Strategies

In chapter 4 we looked at *subnetworks* of a DNN and we experimentally verified that some of them have certain properties (including initialization and architecture) that allow them to capture the correct predictive signals in the early epochs of training with a biased dataset. Each of these subnetworks can be considered to be a *hypothesis*, or a potential solution to the problem. Since there are many subnetworks in a DNN, we can say the DNN contains within itself many different hypotheses. This raises the question of how to uncover the desirable subnetworks (ones that capture the core features in a biased data) if we do not have access to large amounts of unbiased data. We draw inspiration from this open question as well as evolutionary algorithms to formulate one mitigation strategy, which we call *Diverse Hypotheses through Evolutionary Training Strategies.*

5.1.1 Architecture and Setup

For simplicity assume we have a binary classification problem. Let us consider an architecture where instead of having one output classifier we have k such classifiers, or heads of the DNN. Each of these k classifiers is obtained as a matrix product of a matrix W with a matrix C_i, $\forall i$ s.t. $1 \leq i \leq k$, as shown on figure 5-1. For the purposes of this experiment we keep the k heads *fixed*, i.e. we do not update their weights during training. The dataset used is Biased Color MNIST dataset 2.2.1. We train with a biased dataset $\mathcal{D}^b_{train}$ and test on an unbiased set $\mathcal{D}^u_{test}$. Importantly, we also assume we have access to small balanced unbiased validation set $\mathcal{D}^u_{val}$.

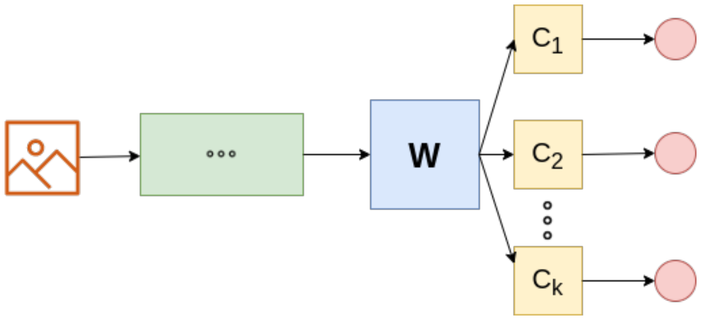


Figure 5-1: Architecture of a k-classifier neural network, where the matrix W is transformed k times to obtain k predictions.

The training approach is based on a classifier rejuvenation strategy, where after every training epoch we remove the worst performant classifiers and sample the remaining ones with replacement based on their performance on the unbiased validation set $\mathcal{D}^u_{val}$. Note that after every epoch we still have k classifiers. The procedure is as follows:

1. Train the DNN for one epoch using mean Binary Cross-Entropy (BCE) loss across the k classifiers

2. Compute the accuracy on $\mathcal{D}^u_{val}$ for each of the k classifiers

3. Take Softmax of the k accuracies from (2) to get a probability distribution $\mathcal{P}$

4. Remove the least probable $p\%$ percent of values in $\mathcal{P}$

5. Softmax the remaining probabilities so that again they sum to 1

6. Sample new k classifiers with replacement based on the new probabilities in $\mathcal{P}$. The value at the i-th element of$\mathcal{P}$, that is $\mathcal{P}_i$ for $1 \leq i \leq k$, can be interpreted as the probability that the i-th classifier is sampled

7. Add small Gaussian noise ϵ to the weights of each sampled classifier, where $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$

8. Repeat process from step 1 until a desired number of epochs

After the model is trained we can perform inference by gathering the predictions from all k classifiers and taking the majority predicted class. An alternative strategy is to take the average of all predicted probabilities per class and use that to predict the final class. Note that for the purposes of analyzing our results in the next section, we do not need to choose an aggregation strategy as we consider the performance of each classifier separately.

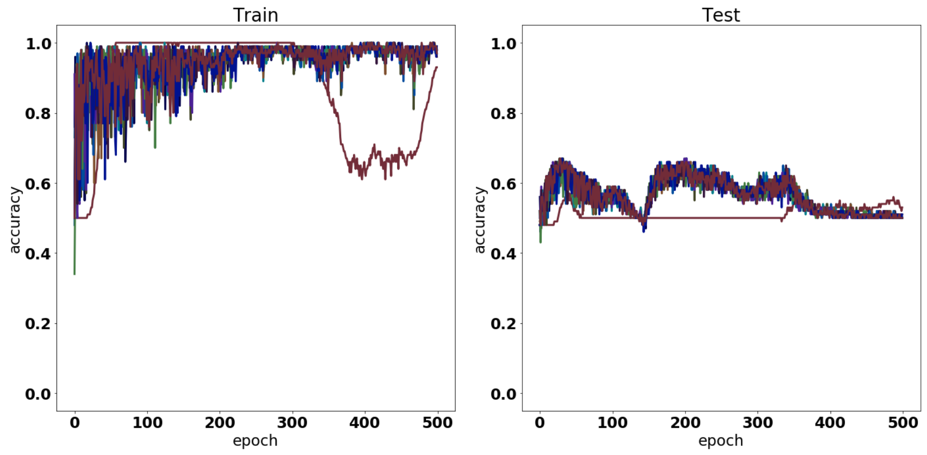
To obtain the results below, we trained a CNN with three convolutional layers, for 500 epochs, using Adam optimizer with a starting learning rate of 1e-6. We vary the value of k. At each rejuvenation step we remove the lowest performant $p = 80\%$ of classifiers. We assume the noise ϵ is Gaussian with mean $\mu = 0$ and variance $\sigma^2 = 0.2$ i.e. $\epsilon \sim \mathcal{N}(0, 0.2)$. Furthermore, the sizes of the train, validation and test sets are as follows: $|\mathcal{D}^b_{train}| = 5938$, $|\mathcal{D}^u_{test}| = 1791$, $|\mathcal{D}^u_{val}| = 199$.

5.1.2 Results

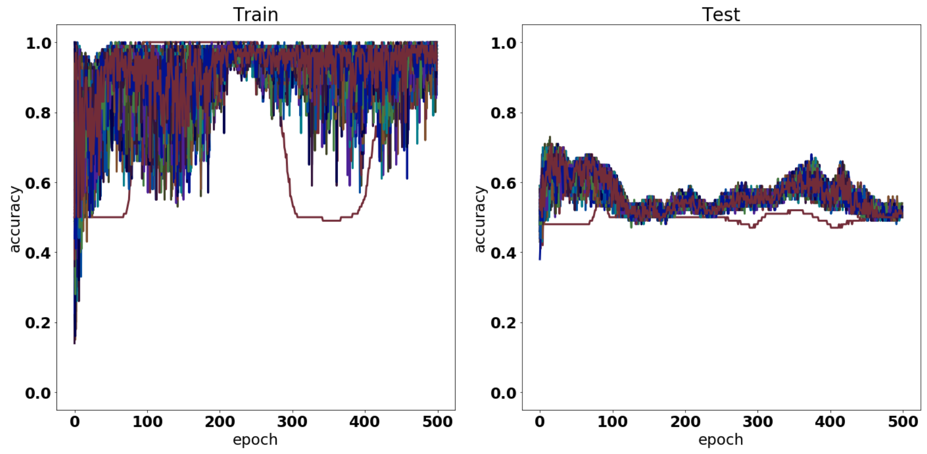
The results for $k = 10$ and $k = 500$ are given in figure 5-2. The x-axis denotes the epoch and ranges from 0 to 500, and the y axis is the accuracy (train or test). On the left side of the plots we show the k train curves for each classifier, whereas on the right plots we show the test curves in the same fashion. We notice that there is not much diversity in the hypotheses, which is expected in the current setup, but the peak test performance reaches around 75% which is significantly higher than the baseline of 50% using standard training techniques when we are training with only biased data. Different hyperparameter settings would yield slightly different results, but we found the given setting to be representative.

5.1.3 Analysis and Discussion

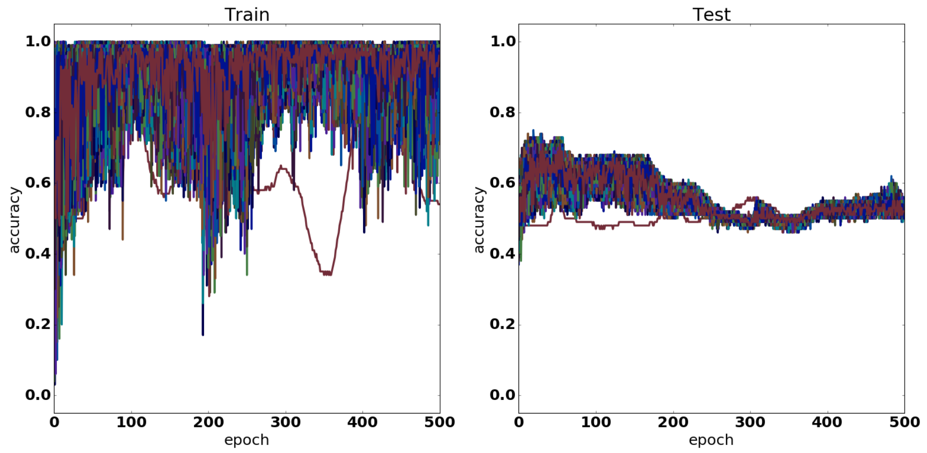
In this section we looked at one possible approach towards finding a hypothesis i.e. a solution learned by a neural network, that captures the predictive signals that generalize well on our validation and test data. This approach can be useful when we cannot curate a large unbiased balanced training set, but we can curate and use a smaller validation set that can directly guide our training process. Note that the validation set in our approach plays a more powerful role than merely choosing the best intermediate checkpoint during in the training phase: it helps us eliminate the unpromising solutions early in the process. A further investigation can probe the benefits of this method in comparison to one where we train the model with the combination of the validation and training set.



(a) k=10 classifiers



(b) k=100 classifiers



(c) k=500 classifiers

Figure 5-2: Evolutionary training approach - Train (left subplots) and test (right subplots) learning curves for each of the k classifiers

Chapter 6

Conclusion

6.1 Summary

The aim of our work was to broaden the current knowledge of the behavior of neural networks (NNs) under spurious correlations in the data using synthetic and image-based datasets that (a) incorporate well-defined spurious correlations, (b) are amenable to experimental analysis, and (c) capture the non-robustness of NNs observed in practice.

We first demonstrated the reliance of NNs on spurious features as a predictive signal in a biased data, which results in at-chance out-of-distribution performance. Then, we analyzed the effect of the NN depth and the strength of the spurious correlation to empirically demonstrate that deeper over-parametrized models are more likely to miss out the core features and rely exclusively on the spurious features. Further empirical analysis showed that there is a correlation between the L2 distance from initialization and the type of solution. Then we introduced a simplicity metric to compare NN solutions and demonstrated that the solutions learned with biased data are simpler under this metric. We proceeded to show that NNs contain within themselves subnetworks whose initialization and architectural structure make them more likely to capture the core features in the early stages of training. Finally, we proposed an evolutionary training strategy that shows promising results towards learning the core features from biased data by having access to a smaller unbiased validation set.

6.2 Limitations

While a lot of our findings are as anticipated and appear to be well substantiated by previous work, others are based on a limited set of experiments and therefore should be treated with considerable caution. It is plausible that a number of limitations could have influenced the results obtained. First, the experiments are performed with well-curated synthetic datasets where the biases are well-defined and amenable to empirical analysis. This differs significantly from the nature of real-world data where quantifying the biases is hard to impossible. Therefore it is not inconceivable that the effectiveness of our proposed techniques or the strength of certain observations and conclusions would have diminished if the data had come from a real source. Second, the NNs, training techniques and hyperparameters used in the experiments, unless otherwise specified, have meant to be representative and "typical" in the task of image-based digit classification, but it might have been the case that different results were observed had these been different. Additionally, for certain experiments, such as the ones in section 3, further analysis would be needed to determine exactly how the depth affects the solution learned or whether there is any causal relationship between the distance to initialization and the type of solution learned.

6.3 Future Work

There are many avenues for future work, as building reliable deep learning systems is an active area of research. These avenues can be divided into three main categories depending on whether we modify the training data, learning algorithm or the architecture. Modifying the training data can be expensive, not scalable and would provide only a short-term solution which would be dataset-specific. A direction which we partially explored in section 5.1 was a combination of modifying the learning process and the architecture, by training in an evolutionary way and introducing multi-head classifiers. A different direction of study can take our findings in section 4 and look into how can we discover the core-feature lottery subnetworks if we do not have access to (a lot of) unbiased data, and maybe more importantly, how can we encourage these subnetworks to avoid reliance on the spurious features later in the training. A related line of work can investigate whether the architecture can have a significant impact on the types of predictive signals learned. A positive answer to this question can motivate more work that relates Neural Architecture Search (NAS) with robust and reliable deep learning. On a wider level, research is also needed to apply the existing findings for real-world datasets and under the computational constraints often faced in practical applications.

6.4 Ending Notes

This work is foundational in nature and seeks to improve our understanding of neural networks, in particular in relation to uncurated datasets which are likely to contain biases of various types. We believe that, in the long term, a concrete understanding of deep learning phenomena is essential to develop reliable deep learning systems for practical applications that have societal impact and we hope that our research will be beneficial in solving the difficulties with these phenomena.

Bibliography

[1] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks, 2017.

[2] Yoshua Bengio, Frédéric Bastien, Arnaud Bergeron, Nicolas Boulanger–Lewandowski, Thomas Breuel, Youssouf Chherawala, Moustapha Cisse, Myriam Côté, Dumitru Erhan, Jeremy Eustache, Xavier Glorot, Xavier Muller, Sylvain Pannetier Lebeuf, Razvan Pascanu, Salah Rifai, François Savard, and Guillaume Sicard. Deep learners benefit more from out-of-distribution examples. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 164–172, Fort Lauderdale, FL, USA, 11–13 Apr 2011. JMLR Workshop and Conference Proceedings.

[3] Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Rosemary Ke, Sébastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. A meta-transfer objective for learning to disentangle causal mechanisms. *arXiv preprint arXiv:1901.10912*, 2019.

[4] Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Brandon Tran, and Aleksander Madry. Adversarial robustness as a prior for learned representations, 2019.

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[6] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Training pruned neural networks. *CoRR*, abs/1803.03635, 2018.

[7] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[9] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

[10] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty, 2020.

[11] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

[12] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution, 2016.

[13] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[14] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[15] Shengchao Liu, Dimitris Papailiopoulos, and Dimitris Achlioptas. Bad global minima exist and sgd can reach them, 2019.

[16] Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics.

[17] A. Mikołajczyk and M. Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, 2018.

[18] Luke Oakden-Rayner, Jared Dunnmon, Gustavo Carneiro, and Christopher Ré. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging, 2019.

[19] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

[20] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization, 2020.

[21] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations, 2020.

[22] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *arXiv preprint arXiv:2007.08489*, 2020.

[23] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks, 2020.

[24] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014.

[25] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy, 2019.

[26] Guillermo Valle-Pérez, Chico Q. Camargo, and Ard A. Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions, 2019.

[27] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, and Daan Wierstra. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[28] Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5310–5319, 2019.

[29] Philippe Weinzaepfel and Grégory Rogez. Mimetics: Towards understanding human actions out of context. *arXiv preprint arXiv:1912.07249*, 2019.

[30] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.