# Analysing the Robustness of Deep Neural Networks

Divij Khaitan

December 4, 2024

## 1 Introduction

Deep Neural Networks have revolutionised several fields since it was discovered that they could be trained at scale on graphics processing units. Today, deeper and deeper networks trained on unfathomably large datasets are ubiquitous in everyday use, even for someone who may not be technically inclined. The increase in the sizes of neural networks does beg the question, how much is too big. Today, OpenAI's GPT-4 model has over a trillion parameters, even though the number of unique sentences it was trained was several orders of magnitude smaller. In practice it is observed that models do not appear to overfit their training data, a phenomenon that lacks an accepted explanation even today.

The goal of this project is to provide a theoretical and practical analysis of the robustness properties of deep neural networks by analysing the patterns of neuron activations.

### 1.1 Related work

**Generalisation/Implicit Regularisation**: Zhang et. al. showed as early as 2017 that even AlexNet has enough capacity to fit all of imagenet with completely random labels perfectly when given enough time [25]. This suggests that although networks have extremely high 'capacity', they do not use all of it in practice and that contrary to statistical wisdom, they are not overfitting as a result of overparameterisation. Arora et. al. showed that overparameterisation provides a speed up to the optimisation of neural networks and provided bounds for the generalisation error of two layer ReLU networks [1]. Soltanolkotabi and Oymak showed that networks that polynomially overparameterised in the size of the input are resistant to label corrpution, with the final parameters being close to their initialised values [12]. A major hypothesis in this area is that stochastic gradient descent(SGD) as a training method introduces an 'implicit regularisation' into the training of a deep network. Norm minimisation was one explanation for this behaviour, first put forth by [18]. Several notions of complexity have been put forth, including VC dimension and Rademacher complexity, see [17] for a more granular view of these. Examples where all

matrix norms diverge were constructed in [19], indicating that the relationship between weight decay and implicit regularisation requires more inspection. This relationship has also been questioned by [2], which shows that for the case of matrix factorisation, matrix norms are insufficient to describe the action of gradient-based optimisation. This paper also shows that adding depth to matrix factorisation encourages low rank solutions, which improve the accuracy of the reconstruction. [5] formally showed that deep networks learn 'incrementally', and while shallow networks can exhibit similar behaviour they do so for a small set of initial conditions. [3] showed that SGD behaves differently on noise as opposed to actual datasets, and that dropout works better than noise injection for hindering memorisation while promoting generalisation

**Out of Distribution Detection**: When considering open world deployment of networks, it simply isn't possible to train them on exhaustive partition of every input they can expect. Thus, it becomes important to develop mechanisms that allow users to detect when an input is outside the training distribution of a model. The first major work that tackled this problem with some effectiveness was [8], which uses $-\max_i \text{softmax}(f(x)_i)$ as a means of detecting if a particular input is outside the training distribution. This is analogous to treating the magnitude of the softmax output as a confidence measure, which isn't entirely accurate as pointed out by the authors. However, this has proven to be a baseline which is difficult to beat non-trivially. [9] describes a method for fine-tuning trained models to detect OOD inputs using unlabelled data and the KL divergence from a uniform distribution. ODIN [13] has empirically shown excellent performance on OOD detection, but requires tuning hyperparameters beforehand. [11] Uses a similar idea of creating a probability distribution over feature spaces and uses the minimum mahalanobis distance as a classifier, and also use it to show that tuning their classifier on certain adversarial attacks adds robustness to other adversarial attacks. The authors assume a multivariate gaussian distribution on the class-conditional softmax probability vectors, and for a vector output the class for which the probability vector has the smallest mahalanobis distance. [21] Uses the gram matrices of the feature vectors and computes layer-wise deviations of an input from the normal values of the class.

**Activation Patterns of Neural Networks**: There has been some past work on analysing the patterns of activation of neural networks. On the side of interpretability, SUMMIT (citation) and NAP (citation) both attempt to identify 'important' neurons in an attempt to interpret the features associated with different classes. Similar ideas have been used for the detection of rare classes as well as the detection of misclassifications at test-time. Our work is more similar to the latter, however we take a different approach to analysing the patterns.

**Adversarial Robustness**: Szegdy et. al. discovered [24] in 2014 that classifiers on vision models are susceptible to extremely small changes (in some $L^p$ space) that can cause a misclassfification. Since then, a considerable amount of work has gone into attempting to explain their existence ([22], [23]) and the development of attacks (PGD [14], FGSM[6]) and defenses to those attacks,

either by changing the loss function or by modifying the training algorithm in some way.

**Manifold Hypothesis**: This is an idea that pre-dates deep learning, that most real-world datasets in high dimensions have an underlying low dimensional structure. In the context of machine learning, it has primarily been studied from the perspective of dimensionality reduction, reducing a set of points in a high dimensional space to one in a low dimensiona space which is equivalent to it. [4] develops an algorithm to test the hypothesis, and [16] discusses the sample complexity of the problem. See [15] for a survey of the most recent methods in the field.

# 2 Methods

## 2.1 Class-Wise Frequencies of Significant Neuron-Weight Interactions

As an additional approach, we decided to study the interactions in a neural network at a more granular level. Instead of just looking at the final value of the neuron, we decided to look at each individual weight being multiplied into each neuron. In this case, what we did is described below.

The final layer of the neural network operates as follows. Given an activation vector $x \in \mathbb{R}^{n_{in} \times 1}$ and a weight matrix $W \in \mathbb{R}^{n_{out} \times n_{in}}$

$$\begin{bmatrix} w_{1,1} & \dots & w_{1,n_{in}} \\ \vdots & \ddots & \vdots \\ w_{n_{out},1} & \cdots & w_{n_{out},n_{in}} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_{n_{in}} \end{bmatrix} = \begin{bmatrix} w_{1,1}x_1 + w_{1,2}x_2 + \cdots + w_{1,n_{in}}x_{n_{in}} \\ \vdots \\ w_{n_{out},1}x_1 + w_{n_{out},2}x_2 + \cdots + w_{n_{out},n_{in}}x_{n_{in}} \end{bmatrix}$$

This would not allow us to study the individual activation-weight pairs, because they would all be summed up. A vector operation that computes the sum over the rows of a matrix is postmultiplication by the all 1s vector, so by factoring out this vector we can get the required information. Factoring out the all 1s vector from the neurons in the penultimate layer, we get an diagonal matrix with the neurons activations as entries

$$\begin{bmatrix} w_{1,1} & \dots & w_{1,n_{in}} \\ \vdots & \ddots & \vdots \\ w_{n_{out},1} & \cdots & w_{n_{out},n_{in}} \end{bmatrix} \begin{bmatrix} x_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & x_{n_{in}} \end{bmatrix} = \begin{bmatrix} w_{1,1}x_1 & \dots & w_{1,n_{in}}x_1 \\ \vdots & \ddots & \vdots \\ w_{n_{out},1}x_{n_{in}} & \dots & w_{n_{out},n_{in}}x_{n_{in}} \end{bmatrix}$$

To this, we concatenated the bias vector as an additional neuron, giving us a matrix in $\mathbb{R}^{n_{out} \times (n_{in}+1)}$ The definition of significance we decided on was

$$S(x_{ij}) = 1 \text{ if } \frac{x_{ij}}{\sum_j x_{ij}} > \frac{1}{n_{in}}$$

where $n_{in}$ is the number of input features, i.e. number of columns in the matrix. The justification behind this is the idea that if an activation is of an above

average proportion of the final activation, the neuron-weight interaction is significant. As eariler, we hypothesise that the number of significant interactions between neurons and weights are sparse, and that the sparsity increases with depth. To this end, we attempted to bin the proportions of the significance of each neuron-weight interaction. If the activations become more sparse, the number should be larger on the left and become sparse towards the end, which should be exacerbated by depth.

### 2.2 KL Divergence Between Classes

Building on the same experiment, we wanted to check both the entropies in between different classes over the activation distribution for the same layer. We also measured the same entropies at training and testing time, as well as an in and out of distribution dataset. The in-distribution dataset was the validation set for imagenet as used eariler, and the out of distribution dataset was fixed to be the imagenet-r dataset created by Hendrycks et. al.[7]. This is a collection of stylised versions of the standard classes of imagenet such as caricatures. This dataset has 200 classes compared to the original 1000 from imagenet, and to ensure the comparison was meaningful we sampled 50 classes from the dataset uniformly at random and computed the pairwise KL divergences between pairs of classes from the imagenet validation set, between pairs of classes from the imagenet-r set and between the same classes from imagenet and imagenet-r.

### 2.3 Memorisation and Significance

We hypothesise that a network which has memorised it's training data would have significantly different properties in terms of sparsity and the separation of classes. To this end, we replicate the experiment specified in [25] and train AlexNet on CIFAR10 until it reaches zero error. The details of the training stay the same and will be provided in Appendix A.

### 2.4 Adversarial Attacks

The above gives us a distribution on the activations for a given class. We compared the probabilities conditioned on the distribution of the class of the validation images subject when subjected to a PGD attack [14]. The parameters of the attack are given in Appendix B.

### 2.5 Manifold Entaglement

In the $n-1^{th}$ hidden layer, the manifolds for different classes should be linearly separable. This is because the following transformations are an affine hidden layer computation and a linear softmax computation. We estimate the dimensionality of the manifolds spanned by each class by inspecting the neurons for this layer, combining them into a matrix and finding it's rank. We further estimate the degree of overlap between the manifolds by combining the matrices

for different classes and then applying the principle of inclusion and exclusion. Networks that have not overfir their training data should have manifolds with little overlap, while those which have should have entagled class manifolds.

# 3 Results

We examined the distributions for AlexNet, VGG, ResNet and InceptionV3. The experiments were run using PyTorch, and the weights for the pretrained versions of the networks were also taken from there. The datasets used was the validation set for Imagenet [20], and CIFAR10 [10]

## 3.1 Sparsity of Activations

The following plots have the number of times a neuron activated over the entire validation set divided by the length of the validation set on the x axis, and the y-axis is the number of neurons that activate in that 'bin', divided by the total number of neurons. Inspecting the results from cifar10, the network fitting random labels has a distinct pattern to it's activations in the second-last layer. The random network has most neurons either always activating or never activating. The trained network is somewhat sparse in it's activations, while the overtrained network has more 'stray' neurons which fire infrequently compared to it's trained counterpart.

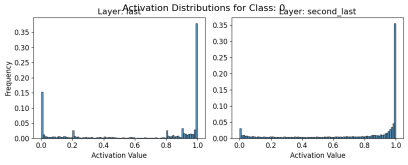


Figure 1: Untrained

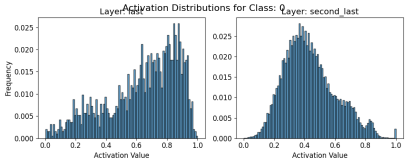


Figure 2: Random Labels

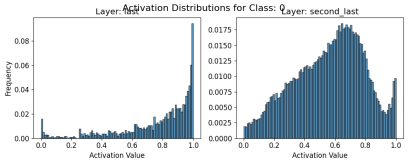


Figure 3: Overtrained

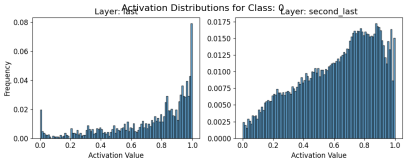


Figure 4: Best

Figure 5: Activation Proportions from the class Airplane on (from left to right) the last and second last layers for the random labels and the last and second last layers clean labels respectively

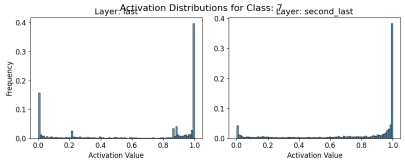


Figure 6: Untrained

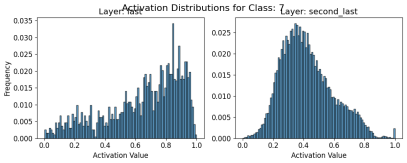


Figure 7: Random Labels

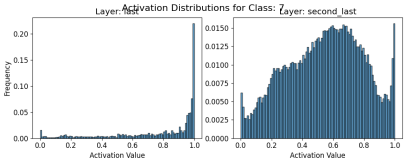


Figure 8: Overtrained

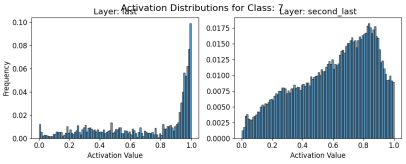


Figure 9: Best

Below are the results for the last 3 layers of alexnet on imagenet and imagenet-r, both trained and untrained. For AlexNet, we can see that the randomly initialised network looks the same in terms of activation proportions for all the layers across both datasets which is to be expected. We see that training encourages sparsity in the network, with the networks having a lower proportion of neurons having an activation proportion of 1 post training. While the second layer shows a smooth decay in both cases, the last and third last layers have much smoother curves for the activations than the OOD networks. The first is monotonically increasing, while the last layer shows a sharp drop off towards the right side of the graph.

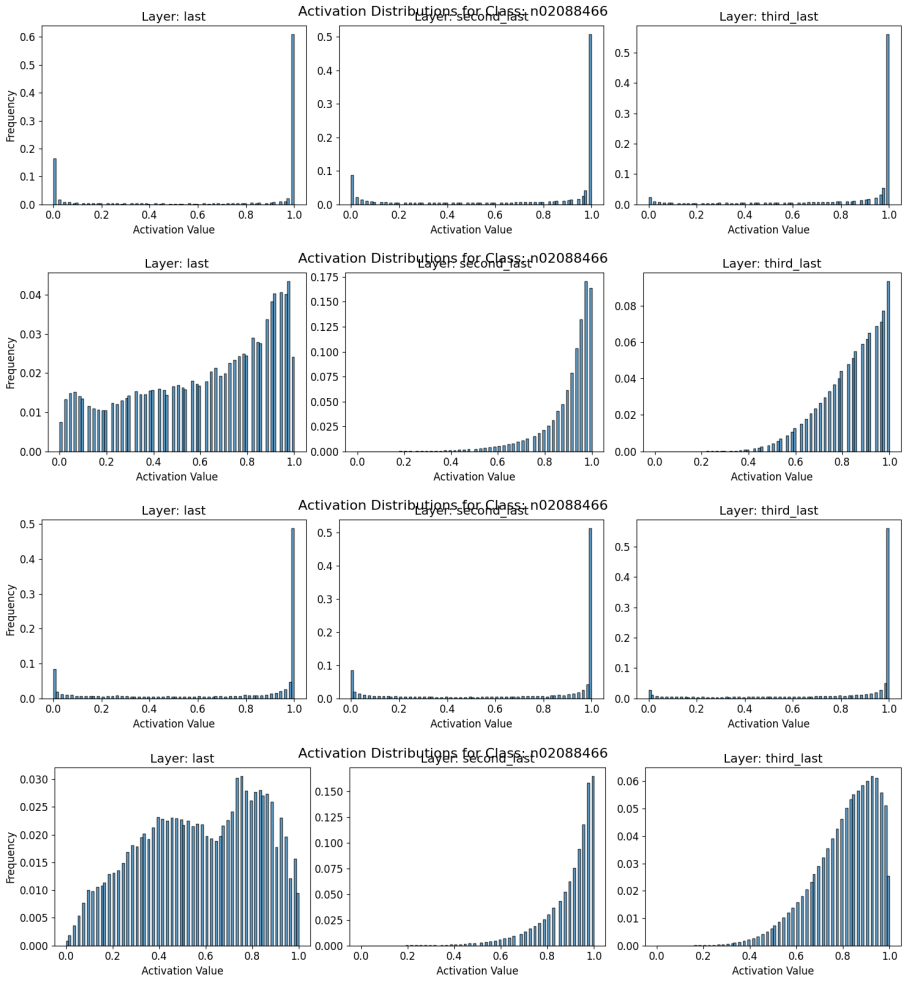


Figure 10: Frequencies of the significant neuron-weight activation proportions for a single class top: untrained on imagenet, second: trained on imagenet, third: untrained on imagenet-r, bottom: trained on imagenet-r

## 3.2 Inter-Class Divergences of Neuron Weight Interactions

Below are the results for the KL-divergences of alexnet normalised corrected for the dimensionality of the space. Between initialisation and the end of training on the in-distribution dataset, we see that the separation of classes increases greatly . The classes becoming more separated is a sign that the network is learning something meaningful about them. However, this also true for the OOD dataset, so this doesn't tell the complete story.

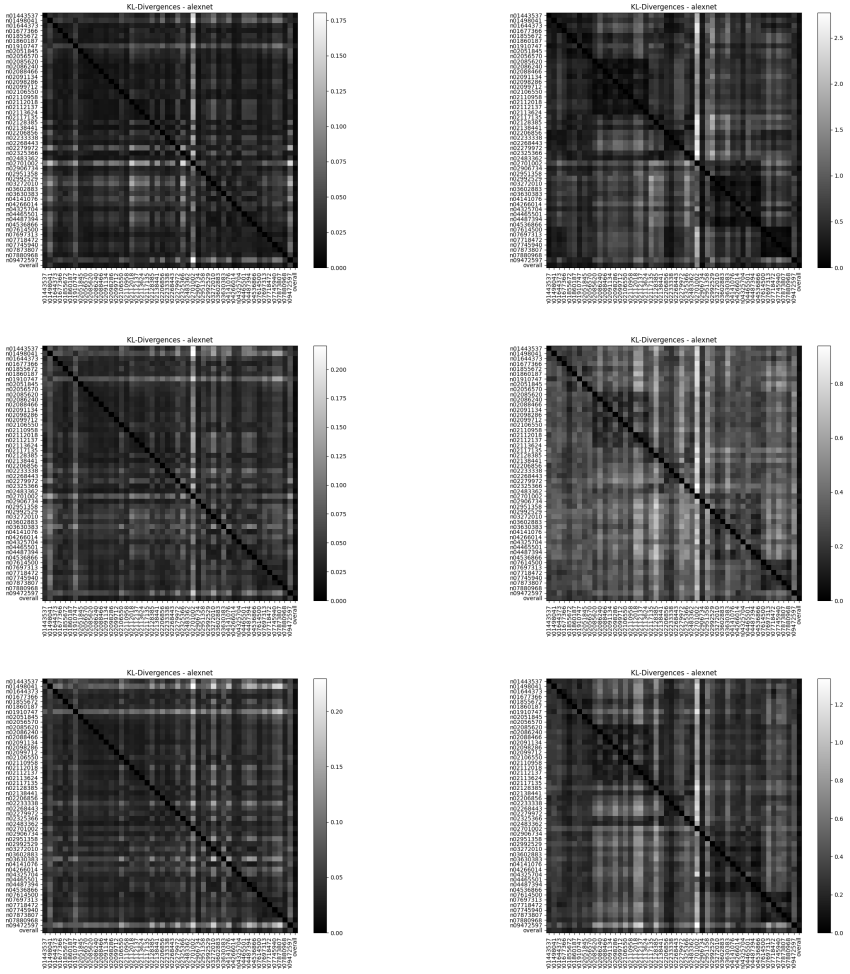


Figure 11: KL divergences between different classes for the untrained(left) and trained(right) classes of alexnet on imagenet. From top to bottom, these are the values for the distributions for the last, second last and third last layers

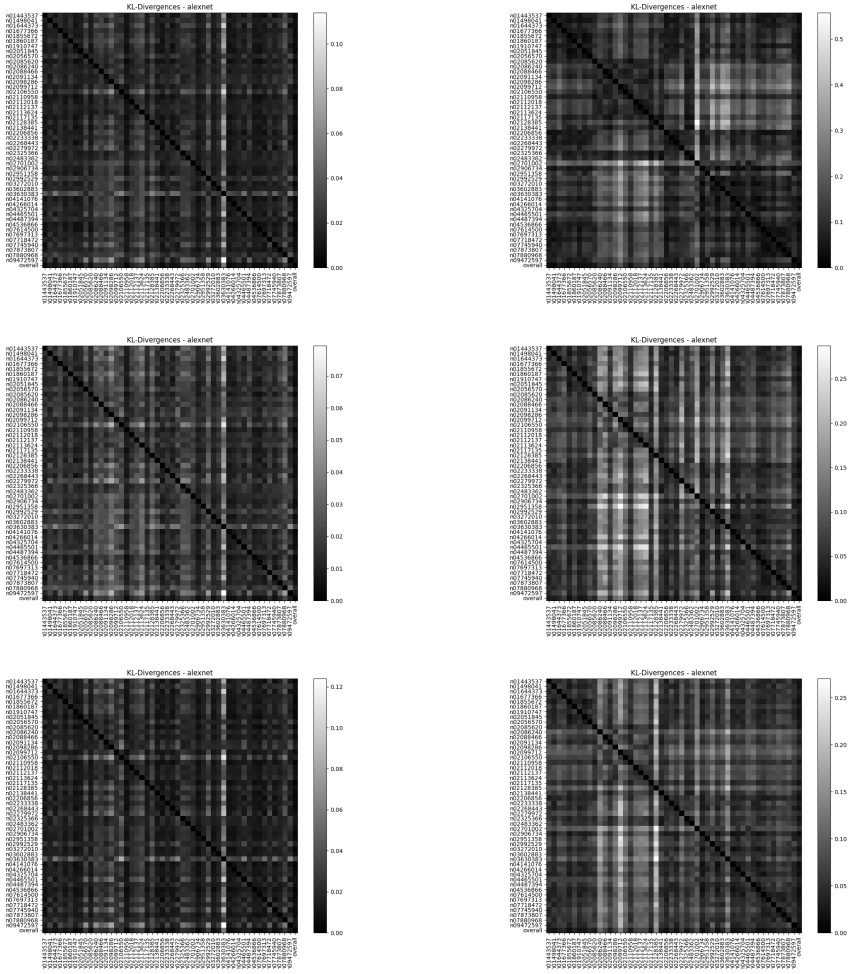


Figure 12: KL divergences between different classes for the untrained(left) and trained(right) classes of alexnet on imagenet-r. From top to bottom, these are the values for the distributions for the last, second last and third last layers

Now, we inspect the effect of training on the separation between a class and its OOD counterpart. This decreases over the course of training, which means that the neuron activation distributions for a class and its OOD counterpart become more similar. Additionally, the OOD distribution has a larger divergence from the ID distribution than the other way around. This would suggest that the OOD distribution is more chaotic than its ID counterpart, which is as expected.


Figure 13: KL divergences from the in-distribution classes to the out of distribution classes untrained(left) and trained(right). From top to bottom, these are for the final layer, the second last layer and the third last layer

For the CIFAR10 networks, we see that the difference between the network at a random initialisation compared to the network fitting random labels are extremely similar. The clean network is similar in the relative difference between classes, but has a much higher degree of separation between classes. For the overtrained network compared to its 'best-fit' counterpart, we see the same similar pattern, but we also see that the divergences are significantly smaller. This suggests that the overtraining is causing the model to 'forget' the relationships between classes which makes them similar.


Figure 14: KL divergences from the out-of-distribution classes to the in distribution classes untrained(left) and trained(right). From top to bottom, these are for the final layer, the second last layer and the third last layer

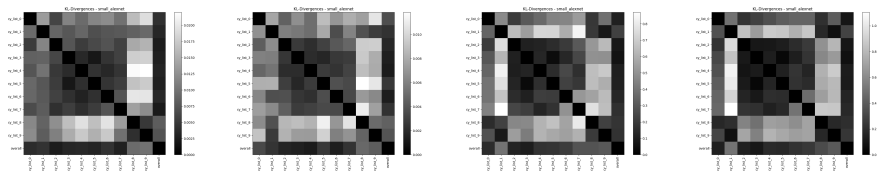

Figure 15: KL divergences between classes for alexnet models on cifar10 at initialisation (left), fitting random labels(centre) and fitting clean labels(right)

# 4 Conclusion

The above results present a new way of looking at and analysing a trained neural network. We analyse the distributions of class activations and the effect of training on their evolution. We also analyse the change in probability under an adversarial attack and the entanglement of class manifolds under different conditions. We believe that as an evaluation framework, this is more robust to overfitting by inspecting the similarities at higher layers as well as lower layers. We find that this framework is able to pick up on evidence of overfitting, forgetting and registers different behaviours for in-distribution data and OOD data. We are also notice significant deviations from the in-distribution counterparts of out-of-distribution classes.

# References


[1] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 244–253. PMLR, 10–15 Jul 2018.

[2] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization, 2019.

[3] 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.

[4] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis, 2013.

[5] Daniel Gissin, Shai Shalev-Shwartz, and Amit Daniely. The implicit bias of depth: How incremental learning drives generalization, 2019.

[6] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015.

[7] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization, 2021.

[8] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks, 2018.

[9] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure, 2019.

[10] Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009.

[11] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks, 2018.

[12] Mingchen Li, Mahdi Soltanolkotabi, and Samet Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 4313–4324. PMLR, 26–28 Aug 2020.

[13] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks, 2020.

[14] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks, 2019.

[15] Marina Meilă and Hanyu Zhang. Manifold learning: what, how, and why, 2023.

[16] Hariharan Narayanan and Sanjoy Mitter. Sample complexity of testing the manifold hypothesis. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010.

[17] Behnam Neyshabur. Implicit regularization in deep learning, 2017.

[18] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning, 2015.

[19] Noam Razin and Nadav Cohen. Implicit regularization in deep learning may not be explainable by norms, 2020.

[20] 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.

[21] Chandramouli Shama Sastry and Sageev Oore. Detecting out-of-distribution examples with in-distribution examples and gram matrices, 2020.

[22] Ali Shafahi, W. Ronny Huang, Christoph Studer, Soheil Feizi, and Tom Goldstein. Are adversarial examples inevitable?, 2020.

[23] Adi Shamir, Odelia Melamed, and Oriel BenShmuel. The dimpled manifold model of adversarial examples in machine learning, 2022.

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

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

# A Details for Modified Alexnet

The modified version of Alexnet is adapted to the CIFAR10 dataset, which is much smaller than the Imagenet dataset that Alexnet was built for. The convolutional block here is a $5 \times 5$ convolution, followed by a $3 \times 3$ max pool and a local response normalisation layer with size 5, $\alpha = 0.0001$, $\beta = 0.75$ and $k = 2$. Two convolutional blocks are followed by a two fully-connected hidden layers with 384 and 192 neurons, with the output having 10 neurons. The output is softmax activated, while the other layers are ReLU activated.

The optimizer used is SGD, with the initial learning rate set to 0.01, decaying by 0.95 after every epoch. No explicit regularisation such as dropout or weight decay are used. Each $32 \times 32$ image is cropped to a $28 \times 28$ resolution, with each image *independently* standardised to have zero mean and unit standard deviation.