Using Intrinsic Dimensionality to Improve Dropout Regularization.

Javier Fernandez Cruz

A Thesis
in
The Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Master of Computer Science (Computer Science) at
Concordia University
Montréal, Québec, Canada

July 2021

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Javier Fernandez Cruz**
Entitled: **Using Intrinsic Dimensionality to Improve Dropout Regularization.**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science (Computer Science)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. Adam Krzyżak

_____ Examiner
Dr. Adam Krzyżak

_____ Examiner
Dr. Eugene Belilovsky

_____ Thesis Supervisor
Dr. Thomas Fevens

Approved by _____
Dr. Leila Kosseim, Graduate Program Director

July 8, 2021 _____
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Using Intrinsic Dimensionality to Improve
Dropout Regularization.

Javier Fernandez Cruz

The intrinsic dimensionality (ID) of multi-dimensional data collections is one of their most fundamental characteristics. Estimates of ID provide an important notion of the complexity of the data, which, in turn, is crucial to selecting the right approach and designing effective machine learning models. There is a wide range of applications for ID estimation, from widely used dimensionality reduction to adversarial attacks, outlier detection and search indices to more theoretical fields like similarity search, discriminability, graph construction, and extreme value theory. However, the notions provided by ID estimations of the data stop at the threshold when designing machine or deep learning models, providing little to no insight when selecting model hyperparameter configurations.

In this work, we explored the idea of using a relation between the intrinsic and extrinsic dimensionality of an image manifold to provide an intuition for selecting an appropriate dropout rate to regularize neural networks in the context of image classification problems. We studied the characteristics of several ID estimators and applied them to image datasets and introduced a new formula to compute values for the dropout rate dependent on ID estimations.

We empirically studied the effects of using this new rate by analyzing its effects in the training of several state-of-the-art image classification models on benchmark datasets. We showed that using this technique can consistently improve the performance of several well-established models.

Acknowledgments

There is a long list of people who in some proportion provided me with the opportunity to be where I am today and dream about an even better tomorrow. Mentioning all would require careful thought and even then there might be someone unintentionally left out. I will then refrain from naming you all for the sake of the few that cannot be forgotten. I hope you know I carry you all in my heart.

First and foremost I would like to thank my family for the education they provided me with. Specially my parents, who have always encouraged me to take the next step and who have proved many times that there are no lengths they are not willing to go.

I would like to express my deepest gratitude to my supervisor, Prof. Thomas Fevens for accepting me under his care and bestowing in me a great deal of confidence and trust. Thank you for maintaining a constant stream of support both in terms of invaluable advice and much appreciated resources.

It is my undeniable luck to be able to count with a rather small but priceless group of friends. Among them I would like to single out one who was once one of the unlikeliest people I thought I would create a bond with and who has become over the years, member of a very select group of people who I hold dear. Thank you Manfred, without your help I would not be here today and who knows what would have had become of me.

Last but certainly not least, I would like to thank my son for being the light of my days, my strength and my stay in the distance. Know that I live for the day when we'll be together again. I love you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The growing development of machine learning has pushed its application into numerous fields of human life. Many of its applications involve studying large amounts of data generated by a wide variety of sources. These data are usually complex and may be characterized by thousands or millions of features. Applications of statistics and other disciplines have identified problems when dealing with large amounts of features, not the least amongst these is the well-known curse of dimensionality [4], which warns about the potentially exponential increase in a problem's difficulty as the dimensionality of the data associated with it increases.

Avoiding this curse is one of the main reasons why it is preferable to deal with representations of the data that are less complex. This necessity has prompted the creation of dimensionality reduction algorithms aimed at obtaining lower-dimensional representations of the data. Thus, the question arises as to "*What is the right dimension?*". That is Intrinsic Dimensionality (ID), which represents the minimum number of features some data can be represented by without losing valuable information.

ID estimation is a critical part of dimensionality reduction methods, but there is also a wide range of applications where ID can be applied to. For example, ID is also crucial to the choice of machine learning methodology and its application to a given problem, it can

set fundamental limits to the design and subsequent robustness of models, and it provides an important natural measure of the complexity of a dataset, essential for organizing and processing large amounts of data. In addition, it can help understand the behaviour of machine learning and deep learning models, and it can help explain their results.

However, we find that little use is given to the knowledge that ID provides in terms of configuring model hyperparameters.

1.1 Motivation

Deep learning has proved to be a very powerful tool when analyzing large image datasets for a wide range of applications. These are oftentimes high-resolution images from fields like medicine that have very large dimensions. Unfortunately, many of the models used for these types of applications are usually considerably large, with their size resembling the extrinsic dimension of the data (the total amount of features for every sample) rather than the intrinsic dimension of their manifold.

These large networks usually tend to develop complex co-adaptations between different units as they learn the underlying relationships between pixels of the images, causing overfitting problems that jeopardize the generalization capacity of the networks on the test sets. Dropout [27, 61] is a regularization technique used to prevent overfitting by limiting these co-adaptations during training. In essence, dropout reduces the size of the layers of a neural network by dropping units during every training step with a certain probability p, training a random sample of neurons every time.

This probability p, generally referred to as the dropout rate, is one of the configurable hyperparameters of a deep learning model. There are two generally used methods to initialize this parameter. One is following the empirically backed suggestion of the authors and set values of 0.5 for the dropout applied to any hidden layers of the model, and a value of 0.8 is used on the input layer. Another widely used technique is to perform a

grid/random search operation using a small validation set to find the parameter values that would produce good results during the training process. Usually, more than one parameter is evaluated during a hyperparameter search, and the algorithm determines the best combination from the parameter spaces specified.

Some techniques focus on the model state, which propose the use of special schemes for modifying the dropout rates used under certain conditions; some of these are variational dropout [39] and crossmap dropout [57].

The remarkable success of deep learning in computer vision and works to test the well-known manifold hypothesis[24] has represented for quite a while a common intuition among researchers that images are a very good example to this hypothesis, having low-dimensional structure despite the high dimensionality of conventional pixel representations. Recent works in this area have proved this intuition to be correct by applying intrinsic dimensionality estimation tools to popular datasets, proving that there is a significant difference between these dimensionality values [45, 58].

The need to reduce the number of units a network uses to better fit the data and the fact that lower-dimensional representations can accurately express the data fed to the network have motivated the idea that there might exist some correlation between the intrinsic dimension of the image manifold being learnt and the number of units that actually need to be employed to capture its nature.

As the dropout rate of the network determines the probability with which any given unit (if applied dropout to) will be kept, we hoped to find an intuitive method to infer some measure usable as this hyperparameter that directly relates to the data.

1.2 Thesis Objectives

With the development of this thesis, we attempted to achieve the following objectives:

- To successfully apply an intrinsic dimensionality estimator to the selected benchmark datasets and corroborate the difference between image dimensions and the ID of the image manifold.
- To establish a relation between the ID estimate and the dropout rate by proposing a new measure that can approximate the latter based on the ID estimates obtained.
- To use the new approximation as the dropout hyperparameter for the configuration of state-of-the-art image classification models.
- To assess the impact of the new dropout rate on the generalization capacity of the models by comparing their result with their originally reported accuracy.

1.3 Contributions

In this thesis, we hypothesized about the existence of a relationship between the dimensionality of image datasets and the dropout rate used by deep learning models trained on them and proposed a simple calculation to quantify that relationship. We tested our theory and validated it by conducting experiments where we used this new quantity to configure deep learning models and evaluate their performance.

We proved with these experiments that following this method, we can get meaningful values related to the complexity of the data that can be used to define the rate of dropout layers within a neural network. Furthermore, following our approach we achieved consistent improvement for the accuracy of the models used, matching or outperforming its original reported performance under normal conditions and significantly more so under overfitting conditions.

Indirectly, we empirically corroborated that common natural image datasets have very low intrinsic dimensionality relative to their high dimensional pixel representation, a long-standing notion among machine and learning practitioners.

1.4 Outline

The remainder of this work is organized as follows:

Chapter 2 presents a review of background and works related to the concepts of dimensionality, its role in machine learning, and the regularization of neural networks using dropout. We present in Section 2.1 a brief introduction to concepts related to dimensionality and a brief overview of work related to intrinsic dimensionality estimation and its applications in various areas of research. Likewise, a brief introduction to the problem of overfitting, its causes and measures of regularization of neural networks using dropout is presented in Section 2.2.

Chapter 3 describes the area around which we center our objectives and introduces the proposed framework for our research experiments. In Section 3.1, we present a method to calculate a dimensionality-dependent measure of the rate at which units should be dropped from the network during training.

Chapter 4 contains the description of the experiments performed on several state-of-the-art image classification models using popular datasets. We present the results obtained from these experiments and discuss about the effect of using our dimensionality-dependent rate on the dropout layers.

In Chapter 5 we summarize the entirety of our work and present our conclusions based on the results obtained. Also, we provide some ideas for future work to extend this technique and suggestions of other applications where it might be applicable.

Appendix A reports the general hyper-parameter configurations used on the studied models to achieve the results presented in this thesis.

Chapter 2

Background and related work

2.1 Intrinsic Dimensionality

The term **dimension** is plainly defined in English as *a measurement of something* [64]; in Mathematics and Physics, the term "dimension" is employed in different areas with different meanings. However, a common yet informal way of defining the dimension of a certain mathematical space could be *the minimum number of coordinates needed to specify any point within it* [52]. Derived from these concepts, the field of Statistics commonly defines the **dimensionality** of a certain dataset as referring to *the number of attributes said dataset has* [67]; namely, the number of features used to describe each data point. Thus, we may have $\mathcal{X} = (x_1, x_2, ..., x_N)$, where $x_{(1,2,...,N)} \in \mathbb{R}^D$, which represents a set of sample points with a D number of features; or rather, with dimensionality $dim(\mathcal{X}) = D$.

Most modern applications of classic AI, data mining, machine learning and the like, frequently deal with huge quantities of data points characterized by a large number of features with diverse objectives: categorizing objects, discovering relations, localizing independent entities, predicting some outcomes, and many others. Unfortunately, working with these high-dimensional datasets can often result in the manifestation of undesired adverse effects

that jeopardize the positive achievements of results. Most prominent among these is Bellman's well-known curse of dimensionality, which warns about the exponential growth in the difficulty of problems in higher dimensions [4].

Aiming to avoid the dangers of high dimensionality, the first step in many practical scenarios is the search for a less complex representation of the original data, here some *dimensionality reduction* technique is applied in order to project the data into a lower-dimensional subspace and produce the dataset $\mathcal{X}' = (x'_1, x'_2, ..., x'_N)$, where $x'_{(1,2,...,N)} \in \mathbb{R}^d$ that shares the most relevant attributes with the original data and where $(dim(\mathcal{X}') = d) \ll D$ [3]. Two main reasons justify the use of dimensionality reduction methods. First, while many of the original features are essential to describe the data, others are (almost) irrelevant and might consequently disturb or at the very least over-complicate subsequent data processing steps. Secondly, the size of the original data itself might effectively render impossible the algorithm's application due to excessive computational cost, be it because of run time or memory consumption issues.

However, the value of d needs to be optimally found. If d is too small, important data features might be lost or "collapsed" during reduction. In contrast, if the value is too large, some irrelevant features might be kept, making the data noisy and the algorithm application unstable. The optimal value of d represents the minimum number of latent features/variables needed to represent the original data with minimum loss of information; this value is known as **intrinsic dimensionality** (ID). The Intrinsic Dimensionality of a dataset provides an important natural measure of its complexity and constitutes an essential aspect to consider for organizing and processing large amounts of data. Furthermore, it may be useful for understanding the properties of models applied to the data it describes and, thereby, useful for selecting the optimal model for that particular data distribution.

To find the ID value that best describes a dataset embedded into a high-dimensional space, one approach is to split the data space into two different subsets–a low-dimensional

one where non-linear methods are effective and a high-dimensional one that allows the effective application of linear methods [25]. In practice, each one of these subsets might have a different ID value, and in turn, those values might not correspond to the ID of the complete dataset [12], demonstrating that ID can be considered a local characteristic of the data space defined in each data neighbourhood. Thus, we call **local intrinsic dimensionality** (LID) to the ID value of each of the neighborhoods [30, 31], and this value can be used in turn to infer the ID of the complete dataset.

Intrinsic Dimensionality estimation has been the subject of much research that have aimed to apply it not just to dimensionality reduction problems or scenarios, but also to a whole range of applications where ID estimation is of great use; some examples are analysis of search indices [5, 33, 37], outlier detection [15], similarity search and similarity graph construction [10, 33, 34, 35], adversarial attacks [1, 47] and some broader fields like overfitting prevention, subspace clustering, feature selection and feature reduction.

As a result of the wide applicability of ID, various characterizations (or models) and accurate estimators have been developed to meet the need of robust and reliable ID estimation.

2.1.1 Intrinsic Dimensionality Estimation

Many theoretical and practical models for measuring ID have been proposed over the course of the last few decades. Some examples are: early theoretical models like the Lebesgue and the classic Hausdorff dimension, the box counting and packing dimensions [18], Multidimensional Scaling (MDS) [42], and other prominent practical models that approach finite data samples.

These ID measures can be regarded as "global," as they consider the dimensionality of the dataset as a whole; however, another important family of models seeks to quantify the ID in the vicinity of a point of interest in the data space. These are considered "local"

intrinsic dimensionality models.

Generally, methods aimed at estimating a measure of global ID can be locally applied to a data neighbourhood by extracting local subsets of the data and then using the original method to each subset separately. And in turn, local methods can be used to infer a global ID estimate using averaging or voting or some more sophisticated technique. Global methods are generally successful on datasets with simple underlying structures but are significantly unsuccessful when used on more complex datasets with non-constant space curvature or highly non-linear features, or dealing with restricted cardinality, point sampling and noise. On these scenarios and in comparison, local ID estimation methods practically make many global methods ineffective [9]. As a result, most recent methods and estimators have completely abandoned the global approach [9], as analyzing a dataset at its biggest scale rarely produces reliable results. Thus, ID estimators dedicated specifically to target local vicinities have been the focus of much research in recent years, as the effectiveness and efficiency of the algorithmic applications of intrinsic dimensionality estimations depend greatly on the quality of the estimator employed. Therefore, we focus on those.

Some well-known and widely used ID estimators are based on Principal Component Analysis (PCA), MDS and on quantifications of the covariance matrix's eigenspectrum using various heuristics [6, 7, 36, 63]. Since their approach is to explicitly find a mapping that projects the data into an appropriate space, these are usually classified as projection methods. Other projection methods include several manifold learning techniques like locally linear embedding, local multidimensional scaling and kernel eigenvalues [59, 65], which can produce an estimate of the ID of the data as a byproduct of their application. However, although these methods have proven to be valuable tools for explanatory data analysis, they usually cannot provide a reliable estimate of ID, as they are too sensitive to noise and parameter settings [45]. Also, many of these methods have been particularly designed for dimension reduction procedures, where the value of ID is a desired known

parameter.

Other ID estimators such as Locally Linear Embedding (LLE) [59], nearest neighbour estimator [46, 56], ISOMAP [62] and Tensor Voting Framework (TVF) [50] exploit the intrinsic geometry of the dataset (often based on the nearest neighbour distances within the data) and are therefore known as geometric methods. Most of these consider hyper-spheres with sufficiently small radius r and centred on the points in the dataset and then estimate some statistics (expressed as functions of the ID) related to the distances of neighbouring points included inside the hyper-sphere.

Finally, fractal-based approaches analyze the space-filling capacity of a local neighbourhood to infer an ID estimation. Several of these have been proposed more recently, such as the Minimum Neighbor Distance (MiND) [46], the expansion dimension [37], the generalized expansion dimension [32], the local continuous intrinsic dimension [29], the correlation dimension (and its variations) [8] and some nearest neighbors approaches. The aforementioned models estimate ID in relation to the rate at which the number of data points encountered increase as the distance from the point of interest expands (the reason why they are also called "expansion models"), which helps to provide a local view of the dimensional structure of the data. As these models use only the inter-point distance values within the neighbourhood with no expensive vector or matrix operations, they also have the great advantage of computational efficiency. Some interesting fractal approaches like the "incising balls" [19] and Maximum Likelihood Estimator (MLE) [45] have demonstrated very solid performance and reliability when dealing with a dataset from different domains [9].

The incising balls method relies on counting the number of nearest neighbours in balls of growing radii and fits a polynomial to those values. The degree of the said polynomial (if complaint with several established constraints) is considered the final ID estimate. The algorithm requires the value selection for certain parameters; nevertheless, it has proved

itself superior to a wide range of other estimators and has shown a competitive performance compared to the well-established MLE, which we will discuss next.

Maximum Likelihood Estimator

The Maximum Likelihood Estimator [45] and several of its refined extensions [13] have become very popular among researchers and developers and have been repeatedly used in many application areas (for miscellaneous applications, and also for selecting the target for dimension reduction techniques) due mainly to their proven consistency, efficiency and reliability [9].

The MLE applies the principle of maximum likelihood to the distances between close neighbors and derives the final point-wise estimator using a Poisson process approximation by randomly sampling within a given radius around each sample point.

This estimator is based in the following assumptions. Let $\mathbf{y}_i$ be points sampled independent and identically distributed from some smooth probability density function f on $\mathbb{R}^m$, where both f and m are unknown. The data $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N \in \mathbb{R}^D$ where $m \ll D$ is then generated using a continuous and sufficiently smooth mapping $g : \mathbb{R}^m \to \mathbb{R}^D$, where $\mathbf{y}_i \mapsto g(\mathbf{y}_i) = \mathbf{x}_i$.

The idea of the estimator is to fix a point x, assume that the density $f(x)$ is approximately constant in a small sphere $S_x(R)$ with fixed radius R around x, and treat the observations contained in $S_x(R)$ as a homogeneous Poisson process. Considering the inhomogeneous process $\{N(t,x), 0 < t < R\}$,

$$N(t,x) = \sum_{i=1}^{n} 1\{\mathbf{x}_i \in S_x(t)\} \tag{1}$$

counts the observations within distance t of x. Approximating this binomial (fixed n) process by a Poisson process and suppressing the dependence on x for now, we can write the

rate $\lambda(t)$ of the process $N(t)$ as:

$$\lambda(t) = f(x)V(m)mt^{m-1} \tag{2}$$

This follows immediately from the Poisson process properties, since $V(m)mt^{m-1} = \frac{d}{dt}[V(m)t^m]$ is the surface area of the sphere $S_x(t)$. Letting $\theta = \log f(x)$, we can write the log-likelihood of the observed process $N(t)$ as:

$$L(m, \theta) = \int_0^R \log \lambda(t) dN(t) - \int_0^R \lambda(t) dt \tag{3}$$

This is an exponential family for which maximum likelihood estimations exist with probability close to one as $n \to \infty$ and are unique. The MLEs must satisfy the likelihood equations

$$\frac{\partial L}{\partial \theta} = \int_0^R dN(t) - \int_0^R \lambda(t) dt = N(R) - e^{\theta} V(m) R^m = 0 \tag{4}$$

and

$$\frac{\partial L}{\partial m} = \left(\frac{1}{m} + \frac{V^{'}(m)}{V(m)} \right) N(R) + \int_0^R \log t dN(t) - e^{\theta} V(m) R^m \left(\log R + \frac{V^{'}(m)}{V(m)} \right) = 0 \tag{5}$$

Substituting (4) into (5) gives the maximum likelihood estimation for m:

$$\hat{m}_R(x) = \left[\frac{1}{N(R, x)} \sum_{j=1}^{N(R,x)} \log \frac{R}{T_j(x)} \right]^{-1} \tag{6}$$

where $T_j(x)$ represents the Euclidean ℓ_2 distance from x to its j^{th} nearest neighbor x_j, and where $x_j \neq x$.

In practice, it may be more convenient to fix the number of neighbors k rather than the

radius of the sphere (R), then the estimate in (6) becomes:

$$\hat{m}_k(x) = \left[\frac{1}{k-1} \sum_{j=1}^{k-1} \log \frac{T_k(x)}{T_j(x)} \right]^{-1} \tag{7}$$

where $T_k(x)$ represents the Euclidean ℓ_2 distance from x to its k^{th} nearest neighbor x_k, which defines the sphere.

It is natural to assume that choosing the right value for k is crucial, since different values of the parameter are likely to result in different dimensionality estimates. A simple and popular solution for this problem is to average multiple ID estimates computed using multiple values of k to produce a final estimate.

Coincidentally enough, the original dimensionality estimator proposes to average the local estimates computed at each point to obtain a global dimensionality estimate defined by:

$$\overline{m}_k = \frac{1}{n} \sum_{i=1}^{n} \hat{m}_k(x_i) \tag{8}$$

where n represents the number of sampled neighborhoods.

However, this approach resulted in some unexpected biased estimation of ID for small values of k, even when there were large quantities of data. To attenuate this problem, a correction was applied by using a better-motivated way of combining the statistics through the application of another likelihood function that resulted in an average of inverses [14]. This approach (also known as harmonic mean) provided the slightly different global ID estimator:

$$\overline{m}_k = \left[\frac{1}{n} \sum_{i=1}^{n} \hat{m}_k^{-1}(x_i) \right]^{-1} = \left[\frac{1}{n(k-1)} \sum_{i=1}^{n} \sum_{j=1}^{k-1} \log \frac{T_k(x_i)}{T_j(x_i)} \right]^{-1} \tag{9}$$

Several extensions of the Maximum Likelihood Estimator exist, which attempt to address certain areas of optimization. Some attempt to reduce the bias for small values of

k by other techniques like regularization, some include mixtures of translated Poisson processes to account for noisy data, some use polynomial regression to account for non-uniformity and non-linearity, and others successfully apply geodesic distances instead of Euclidean distances between data points. A comparison of several of these reveals subtle differences [13]; however, the numerical results for the ID estimation are qualitatively very similar.

The MLE has been shown to produce good results on a range of simulated and real datasets and outperforms several other well-established dimension estimators. Based on many of its advantages, including the quality and robustness of its estimations and its performance in terms of computational cost and run time, we decided to move forward on our experiments using this estimator for ID.

2.2 Training Neural Networks

Artificial neural networks (ANNs) are a set of mathematical algorithms that work in unison to perform operations on numerical data contained in vectors (input) to produce a certain result (output). These constructs are loosely modelled after aspects of the biological brain and mimic some of its aspects.

The advancements in technology and the increase in processing power of computers have allowed ANN to go from shallow representations carefully designed and handcrafted by engineers and experts to deep complex networks built of successive layers (hidden layers) of representation until an output is produced (Figure 1). The most important aspect of these deep neural networks is that human beings do not design these layers; rather, they are learned from data by means of an autonomous learning procedure. This is widely known as *Deep Learning*.

Deep learning's ability to learn complex representations and identify intricate patterns in high-dimensional data has enabled it to become an essential tool for machine learning

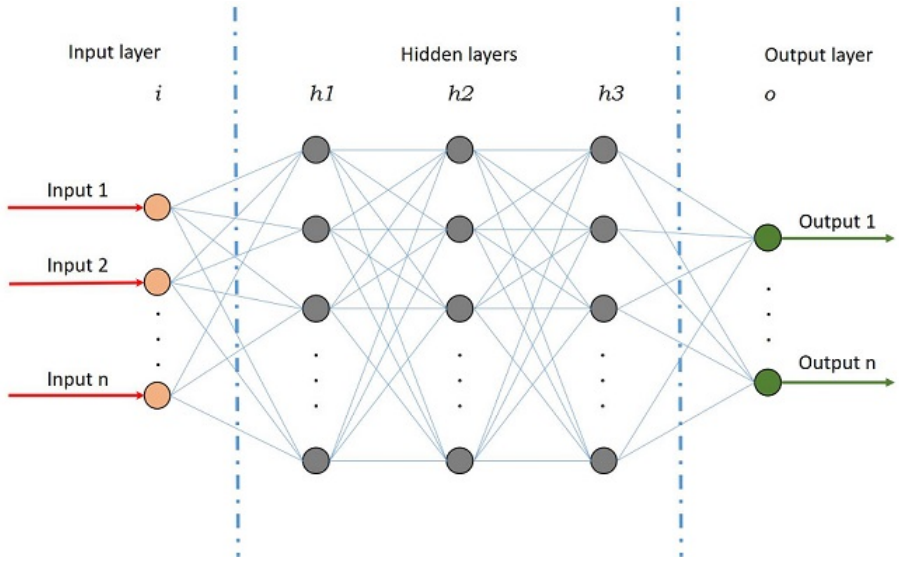


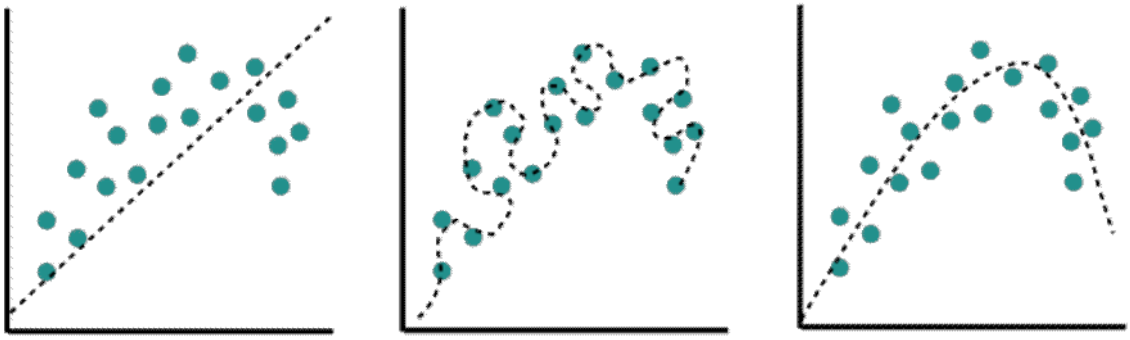
Figure 1: A multi-layer neural network.

practitioners and to make important breakthroughs across a wide variety of fields. For example, its early and outstanding results on image classification problems [41] propelled its adoption by classical machine learning dominated fields like computer vision, signals analysis and natural language processing, making it applicable to a wide domain spectrum that includes science, industry, government and business.

Training a neural network means feeding it with raw data to automatically discover the representations needed to achieve the task it was modelled to perform. For example, for a supervised image classification problem, a network is fed a large dataset of images, each labelled with its category. On each successive training step, the network produces a prediction that is compared to the expected label and an error score is computed. The network then modifies its adjustable internal parameters in order to reduce this error and produce a better prediction.

In a typical deep learning system, there may be hundreds of millions of these adjustable internal parameters (weights) and hundreds of millions of labelled examples with which to train the network [43].

However, the performance of a deep neural network (or any machine learning model for



(a) Underfitting (b) Overfitting (c) Good fit

Figure 2: Underfitting vs. overfitting.[1]

that matter) is measured by its ability to predict accurately (generalize) on new unobserved data as well, not just on training data. A network (or any parametric model for that matter) with limited capacity cannot fit the training data accurately enough (See Figure 2a). An underfitting problem can easily be addressed by adjusting the network capacity, namely, increasing the number of hidden layers or the number of neurons on each layer, and therefore, the number of weights. On the other hand, a network that is too large can excessively fit the training data, memorizing the data (and noise) rather than its underlying characteristics (See Figure 2b). Addressing overfitting requires the use of more sophisticated techniques.

There are several approaches to tackle an overfitting network. The first and most obvious approach would be to reduce the capacity of the network so that it contains fewer parameters. Another straightforward solution would be to train on more data, using data augmentation and/or cross-validation techniques and other data sampling schemes to feed the network with more unseen combinations of data. Feature engineering or feature selection techniques can be performed so that the network can focus on relevant features [54]. As overfitting is easily diagnosed by monitoring the performance during training and validation stages, an early stopping approach can be applied so that training is stopped the moment the validation error begins to drop. [69]. Regularization techniques like L1 and L2

[1]Image modified from: `https://www.fastaireference.com/overfitting`

Regularization, weight decay and soft weight share the aim of reducing the complexity of the network by adding a penalty term to the loss function, leading to smaller weights and simpler models [54, 55].

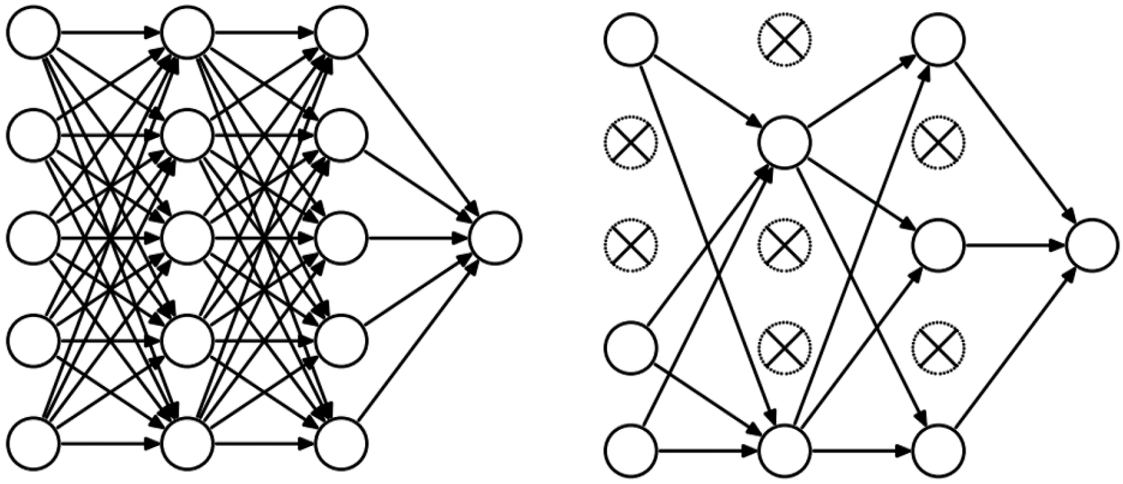
Model combination (or ensemble learning) is a technique by which the (weighted) outputs of multiple models (or all possible parameter settings for one single model) are averaged to produce a final prediction; this method nearly always improves the performance of machine learning methods. However, training large neural networks with different designs or on different data for the purpose of averaging their output is prohibitively expensive [61].

Dropout is an interesting, popular and extensively used regularization technique that exploits the concepts of model combination. *"It prevents overfitting and provides a way of approximately combining exponentially many different neural network architectures efficiently"* [61].

2.2.1 Dropout

The idea behind dropout is to temporarily remove (drop) units from a neural network along with all their incoming and outgoing connections, as shown in Figure 3. On the left (Figure 3a), we can see a standard net with two hidden layers. On the right, an example of a thinned net is produced by applying dropout to the network on the left. Crossed units have been dropped [61].

On every training step, neurons are retained in the network with a fixed probability p independent of other units. A neural network containing n neurons, if trained using dropout, can be considered as the training of an ensemble of 2^n thinned networks with extensive weight sharing. Every time a training pair is provided, a different "thinned" network (with only the units that survived the deactivation) is sampled from the original and trained. Each sampled network gets trained very rarely, if at all [27, 61]. In contrast with an ensemble, at test time, instead of averaging the predictions of all thinned networks



(a) Standard neural net.

(b) After applying dropout.

Figure 3: Dropout on Neural Networks [27, 61].

trained, the predictions are made using a single neural network (the original) where the outgoing weights of the nodes to which the dropout scheme was applied are scaled-down by using the probability p.

Given that applying dropout produces a randomly selected sample of units, with each training step, every hidden unit must learn to work efficaciously with every combination, making it more robust and forcing it to create useful features without the help or corrections from another specific unit. Furthermore, this process prevents complex co-adaptations of neurons that tend to overfit the training data and provides a more capable model that can generalize well on test data [27].

Dropout was originally shown to significantly improve results on a variety of tasks [27, 61]. Its extensive use and effectiveness have solidified its position and have inspired a wide range of techniques for use with deep learning models, allowing them to be trained for longer periods while avoiding overfitting and reaching better test accuracy. Regularization approaches include adaptive dropout [2], dropconnect [66], maxout networks [23], continuous dropout [60] and concrete dropout [22], among many others. Specific applications for regularizing convolutional neural networks include techniques like cutout [17]

and max-pooling dropout [68], while techniques like RnnDrop [49], fraternal dropout [70] and weight-dropped-LSTM [48] specialize in regularizing recurrent neural networks. Also, Bayesian approaches to quantify model uncertainty have been derived from dropout applications [21, 22].

All these applications of dropout share the selection of at least one tunable hyperparameter, the probability p for the Bernoulli distribution or dropout rate. The original dropout proposed in [27] and [61] suggest two ways to find this value. The first approach suggests using a validation set and some hyperparameter search algorithm to find an optimal value. The second approach suggests that for the hidden layers, p "*can simply be set at 0.5, which seems to be close to optimal for a wide range of networks and tasks*", and for the input layers "*the optimal probability of retention is usually closer to 1 than to 0.5*". This last approach seems to be the wide norm for using dropout, as it does not involve exhaustive calculation and empirical results have shown it to be a good fit.

Nonetheless, some works have challenged this cannon and attempted to find values for p that could produce better results for specific architectures or problems. Examples include variational dropout (where each weight of a model has its individual dropout rate) [39], biased and crossmap dropout (where different schemes are used to group certain nodes and apply different rates to these groups) [57], among others.

However, to our knowledge, most (if not all) of these works attempt to propose different values for p based on model state or architecture, but they rarely take into account the characteristics of the data being fed to the networks.

Chapter 3

Proposed Framework

As mentioned in Chapter 2, the default interpretation of the dropout rate hyperparameter for regularizing neural networks is the probability of training a given node in a layer during each specific training step. As discussed, since the proposal of dropout for training neural networks, a common practice has been to use predefined dropout rates for specific types of layers. The foundational works of this technique [27, 61], empirically showed this approach to provide close to optimal results in a suite of standard machine learning problems and a number of useful heuristics.

In accordance with this notion, it is usual to see applications of dropout where the rate applied to input layers is close to 0.8, convolutional layers are assigned rates near 0.7 and fully connected hidden layers are usually given a probability of 50% (0.5) of its units being retained during training.

Several important classic and modern state of the art models follow this path, while other approaches attempt to find more meaningful values for p on their experiments. As discussed in Section 2.2.1, most of this approaches focus on the model architecture or weight matrices' values to experiment with different rates, rather than on the data that is being processed.

However, knowing the structure, quantity, type and other characteristics of the data to

be processed provides valuable insight needed to decide several aspects of a model design. The quantity of features determines how many input units the network will have, the availability of data and its structure offers an idea of how big a network should be to fit the data. Its nature might mandate the type of network that should be used or even the type of layers the network should include and their size. One particular type of data of great significance in current applications of deep learning is images. Widely used for many important applications, images still present outstanding challenges for researchers and are therefore the focus of several active fields of study.

Computer Vision (CV) is one such area of study. CV is concerned with giving computers the ability to extract useful information from images; in a sense, giving computers the ability to *see* and understand the content of digital images. Its applications are huge and span across multiples aspects of life, including fields like agriculture, industrial processing, robotics, military technology, human-computer interaction, medical image analysis and many others. Common CV applications include object classification, identification, segmentation and recognition on images.

Neural networks have proved to be an instrumental tool for computer vision, pushing state-of-the-art results on persistent and challenging CV problems, like for instance, image classification. For the purpose of this research work, we focus our scope on supervised image classification problems using deep learning. We will use several of these models and study the effect of using tuned dropout rates computed by some measure of intrinsic dimension.

3.1 Dropout rates from ID estimates

Image datasets are generally highly complex manifolds that usually contain many symmetries and modes. The sizes of their images can range from small low resolution 28x28 pixel grayscale images like we find in some popular dataset (or even smaller), to high resolution

Figure 4: A Labrador (size: 540x540px).

images with millions of pixels like we could find in many medical image datasets.

The color image shown in Figure 4 have an extrinsic dimension (number of pixel values) of 3x540x540. However, it may be contained in an image manifold with an intrinsic dimensionality value that is significantly lower than that number.

Some studies on learning theory and dimensionality [51, 58] have looked into the relation between ID and its impact on the ability of deep learning models to generalize. They prove empirically that there is a direct relationship between the intrinsic dimension of the image manifold and the number of samples required for a deep learning model to generalize well. Following a somewhat parallel line of thought, we consider that the ID of a dataset can be further used to provide some insight for the configuration of deep learning models used to learn from it.

Thus we set out to assess whether a measure of how many latent variables are needed to represent images on a dataset with respect to their extrinsic dimension (ED) might be a good intuition as to the amount of units needed to fit the data in order to improve its generalization ability.

In deep learning we design models that typically receive as input every pixel value of an image and learn many complex representations to achieve its intended objective. When designing, we avoid drastic changes in size of successive layers so as not to lose information, meaning that many of a network's hidden layers will contain a significant number of units that is more in line with the extrinsic dimension of the images than with the ID of the manifold. If our hypothesis is to hold, this situation should be reversed, and the number of units used should instead have some correlation to the ID estimated for the manifold. However, when reducing the number of units, we risk several pitfalls, like creating a model that is too simple to learn from the data, losing valuable information by simplifying too much or increasing the chances of overfitting.

Our goal would be to use some technique to reduce the size of the network to some measure while avoiding these troublesome situations. It so happens, that we know of a simple regularization technique that is used to great effect during training in order to reduce the number of units used on every training step, dropout (Section 2.2.1).

Considering dropout on our approach, naturally leads us to the appealing idea of computing a certain simple measure p', which we could define based on some proportion of ID and ED, to use as an estimation for the dropout rate. We then hypothesize that using this measure as the dropout rate for deep learning models could improve the generalization capacity of these models, particularly under overfitting conditions.

This approach presents the opportunity to validate our assumptions by analysing the effect that tuning the network dropout rate can have on the generalization performance of the models we consider within our proposed scope.

During our experiments, we attempted several approaches to compute this quantity and used every variation on a sampled version of the datasets to validate their effectiveness. Some variations provided values that were too large or outside the expected range for the target rate (greater than 1), or too small values that restricted the networks excessively and

needed an impractically long training time to achieve competitive classification accuracy.

Empirical results showed that calculating the percentage that the intrinsic dimensionality of the dataset represents against the dimensions of the pixel representation of the images it contains provides a good estimation for this value, yielding a number between zero and one, which could be interpreted as a proportion.

Thus, after several attempts to come up with a good estimation, we proceeded to define p' as follows. Given a set of images $\mathcal{X}$ where each image has c number of channels and dimensions (a x b), the value of p' is given by

$$p' = \frac{\overline{m}_k}{(c \times a \times b)} \times 100 \tag{10}$$

where $\overline{m}_k$ is the intrinsic dimensionality of the dataset given in Equation 9.

By using this quantity, we establish a relation between the intrinsic and extrinsic dimensionalities of the data and incorporate this relation into the model configuration using it as the rate for dropout layers. If our experiments provide positive results, we would have successfully proposed a way to use knowledge from the raw data in assessing values for hyperparameter configurations.

Chapter 4 provides a description of the experimentation done and the results obtained on image classification problems.

Chapter 4

Experimental results

In the previous section, we provide a hypothesis about a relation between the ID of a dataset and the size of the network that should be used to learn from it, and we introduce Equation 10 to compute a numerical estimate of that relation. We also propose the idea of using this estimate when configuring models used for image classification problems by setting the dropout regularization rate to this value and analyze the effects on the network classification accuracy.

To investigate this hypothesis, we selected five popular benchmark datasets used for image classification problems:

- MNIST (Modified National Institute of Standards and Technology) [44],
- CIFAR-10 and CIFAR-100 (Canadian Institute for Advanced Research) [40],
- SVHN (Street View House Numbers) [53], and
- ImageNet [16].

It is important to notice that prior to data augmentation procedures described for each particular experiment on image classification (See Section 4.1), no image format modifications were performed for the MNIST, CIFAR-10, CIFAR-100 and SVHN datasets and the

original image sizes (28x28 for MNIST and 32x32 for the rest) were used. In the case of ImageNet, the images were resized and cropped to 224-pixel resolution following relevant prior works [26].

The first stage of our experimentation was to successfully compute estimates of intrinsic dimensionality for each of the selected datasets. For this purpose, we selected the aforementioned Maximum Likelihood Estimator and applied it to each of the studied datasets in turn. Estimations were performed using all the data available on the original datasets for MNIST, CIFAR-10, CIFAR-100 and SVHN. For ImageNet, three different and independent random samples of 100,000 images were used with the objective of validating that the ID estimation did not depend on the composition of any particular subset sampled from the original dataset.

As we mentioned in Section 2.1.1, the choice of the parameter k for the MLE may affect the dimensionality estimates. As observed by the original [45] and subsequent works [58], estimations of ID increase with k, and choosing large enough values can lead to significant overestimations. Thus, we opted for lower values of k that are in line with benchmark estimations and we were able to reproduce previously reported estimations for the datasets we analyse.

MLE	MNIST		CIFAR-10		CIFAR-100		SVHN		ImageNet	
parameter	$\overline{m}_k$	p'	$\overline{m}_k$	p'	$\overline{m}_k$	p'	$\overline{m}_k$	p'	$\overline{m}_k$	p'
k=3	7	0.30	13	0.42	11	0.36	9	0.29	26	0.02
k=5	11	0.47	21	0.68	18	0.59	14	0.46	38	0.03
k=10	12	0.51	25	0.81	22	0.72	18	0.59	43	0.03
k=20	13	0.55	26	0.85	23	0.75	19	0.62	43	0.03

Table 1: Estimates of ID ($\overline{m}_k$) and p' for different configurations of the MLE parameter k.

Table 1 shows intrinsic dimensionality estimates produced by an implementation of the Maximum Likelihood Estimator for ID using four different configurations for the nearest neighbor parameter (k) for each of the five datasets. We also report in Table 1 the value of p' computed through Equation 10 using the ID estimations and the respective image size

for each dataset. For example, for the ID estimation $\overline{m}_k = 21$ obtained for the CIFAR-10 dataset under the MLE parameter configuration of $k = 5$, by substituting in Equation 10 and using 32-pixel RGB image size, we obtain a p' value of 0.68.

$$p' = \frac{\overline{m}_k}{(c \times a \times b)} \times 100 = \frac{21}{(3 \times 32 \times 32)} \times 100 = 0.68$$

Rather than choosing one particular value of k and its corresponding ID and p' estimates, we report all experiments with multiples values, as they might be useful to discover trends or interesting behaviours during our subsequent work.

The next stage of our work involves conducting a series of experiments on known image classification problems by using the previously computed approximations dependent on the ID estimations of the MLE.

After a careful review of the current state-of-the-art (SOTA) for image classification[1] we selected deep learning approaches that use dropout regularization to improve accuracy. We replicate the models' architecture, hyperparameter configurations, data augmentation techniques and other specifications used on the original works until we match or outperform the reported accuracy for each dataset.

At this point, we assess the benefit of using our proposed measures p' by training each model (from scratch) under two different scenarios.

- For the first scenario we conditioned a highly prone to overfitting environment where we randomly sampled only 10% of the training and test sets and train the models for at least twice as many epochs as proposed by the original works.

- The second one is a more real environment and attempts to recreate the conditions of the reported reference, where we trained the models using all the data available for the originally proposed number of epochs.

[1]Found at this URL: `https://paperswithcode.com/task/image-classification`

We use these two scenarios hoping to get an idea of how beneficial the new rates are not only on well-behaved conditions, but also on highly overfitting conditions where the use of dropout is of particular importance. Also, it is relevant to take into account that a lot of work have been put into designing these models, they are therefore so heavily tuned and include so well chosen regularization schemes that it is difficult to improve their generalization ability. Therefore, having an idea of how our new dropout rates can help them overcome overfitting conditions will be of great use.

Each model was trained several times under each scenario. First each model was trained under the original hyperparameter configuration, four successive trainings were then carried out, each time using one of the values inferred for p' as the dropout rate. The five trainings on each scenario was performed from scratch and using the same data and the same number of epochs.

After training on both scenarios, we then compare the classification accuracy achieved by the models on each using the modified dropout rates proposed against training with the original values.

4.1 Work on image classification

As we mention in the previous section, a formal revision of the state-of-the-art for image classification was conducted and we selected two previous reference works to set a baseline for our experiments. These two works combined contain relevant results for each of the datasets we experiment on and provide ample description of the models used to achieve their results. Table 2 contains the models selected to experiment with for each dataset, where column *Accuracy* represents the reported accuracy value for each of the models and column *SOTA* represents the current state-of-the-art accuracy reported for each of the datasets.

For the MNIST dataset [44], we chose the work of Hirata and Takahashi [28]. An

Dataset	Selected Model	Accuracy	SOTA
MNIST	EnsNet	99.85	99.87
CIFAR-10	WRN-28-10 / PyramidNet	98.50 / 98.39	99.70
CIFAR-100	WRN-28-10 / PyramidNet	87.30 / 88.41	96.08
SVHN	WRN-28-10	99.01	99.01
ImageNet	ResNet-152	81.57	90.45

Table 2: Image Classification State-of-the-Art Survey

Ensemble Learning approach (EnsNet) that uses the output of the last layer of a Convolutional Neural Network (CNN) to feed an ensemble of Fully Connected SubNetworks and produces an output by means of a majority vote scheme.

For the other remaining four datasets, we chose the work reported in the recent paper *Sharpness-Aware Minimization for Efficiently Improving Generalization* (SAM) [20], which proposed the use of a new optimization procedure to train several previous SOTA models on these datasets. SAM seeks out parameter values whose entire neighborhoods have uniformly low training loss value, motivated by the connection between sharpness of the loss landscape and generalization.

What follows is a description of the experiments performed using these models on our experimenting datasets. For each, assume that the data used as base for scenario 1 before any data augmentation is as described on the previous section and that the number of epochs was also modified accordingly.

Details of model configurations can be found in Appendix A, and further details about augmentation techniques and additional training details can be also found on the reference paper [20].

4.1.1 MNIST

The MNIST dataset is a collection of 70,000 grayscale images of dimension 28x28 representing handwritten digits from 0 to 9. The training and test sets consist of 60,000 and 10,000 images, respectively.

Dropout rate (p')	Scenario 1		Scenario 2	
	training	test	training	test
0.35 (base)	92.07	77.11	99.96	99.85
0.30	94.06	66.29	99.95	99.42
0.47	93.91	81.17	99.97	99.90
0.51	91.54	72.29	99.83	99.01
0.55	90.55	69.37	99.76	98.84

Table 3: Classification accuracy results for EnsNet on MNIST.

Our reference work proposes a batch size of 100 images and 1,300 epochs. We also followed the same data augmentation techniques described on the original paper by rotating, scaling, shifting and stretching images.

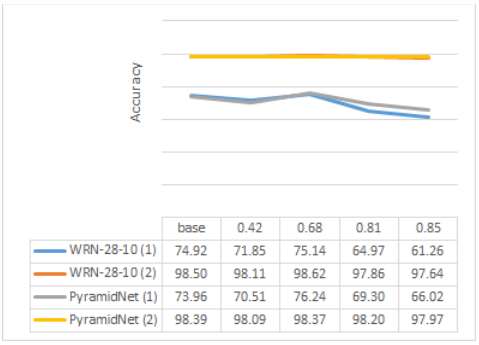
Table 3 shows the accuracy results for the training and test steps of EnsNet on the MNIST dataset using the various configurations for the dropout rate hyper-parameter. Scenario 1 shows evidences of overfitting with high training accuracy but poor generalization on the test set. The model performed best on the test set with a dropout rate of 0.47, surpassing by more than 4% the accuracy obtained by the original configuration (base) with dropout rate of 0.35.

The second scenario shows a similar trend, with the same configuration outperforming the original by a margin of 0.05% on the test set. This performance is, to the best of our knowledge, a new SOTA accuracy value for this dataset.

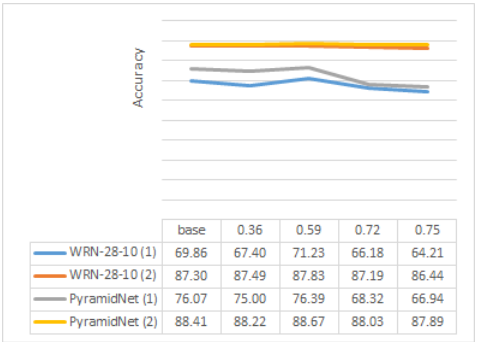
4.1.2 CIFAR-10 and CIFAR-100

CIFAR-10 and CIFAR-100 datasets are both subsets of the Tiny Images dataset and each consist of 60,000 color images of dimension 32x32 (50,000 images on the training set and 10,000 images on the test set), divided into 10 and 100 classes respectively.

From our reference work [20] we selected two models to train on these datasets, a simple WideResNet and a vanilla PyramidNet. All model hyperparameter values are identical to those used on the reference work, and we use auto augmentation [11] and cutout [17] on



(a) CIFAR-10.



(b) CIFAR-100.

Figure 5: Test accuracy of WideResNet and PyramidNet as a function of dropout rate (p') on the CIFAR datasets. (1) and (2) represent each scenario used.

the base data as per prior work. WideResNet was originally trained using a dropout rate of 0.3 during 1800 epochs, while the same parameters are 0.5 and 250 respectively for the PyramidNet.

We report in Figure 5 the performance accuracy of both networks on CIFAR-10 (Figure 5a) and CIFAR-100 (Figure 5b) on the test sets. Consistent with previous SOTA reported for this models [20], they perform relatively similar on each dataset, producing almost overlapping curves for the accuracy results on the complete datasets.

Although accuracy for both models is relatively smooth, on the CIFAR-10 dataset both peaks occur for the model configurations that take 0.68 as the value for the dropout rate instead of their original values, in case of CIFAR-100 the models peak with a dropout rate of 0.59. In both cases the behaviour is consistent for the two scenarios. Which could lead to believe that there is an optimal value for this hyperparameter in the vicinity of those numbers.

It is worth noting that even when all performances for these two datasets are below previously reported SOTA, the tuning of the dropout rate improved the performance of both models, particularly in the case of the first scenario, where overfitting is present and this simple step allowed the simple WideResNet to improve performance by 0.22% on CIFAR-10 and 1.37% on CIFAR-100, and PyramidNet 2.28% and 0.32% respectively. During

(a) Scenario 1.

(b) Scenario 2.

Figure 6: Accuracy of a WideResNet-28-10 as a function of dropout rate (p') on the SVHN dataset.

training with the complete datasets, particularly for CIFAR-100, the models improved by 0.53% and 0.26% which we consider a modest improvement, taking into account the level of regularization and optimization these models already entail.

4.1.3 SVHN

The SVHN dataset is a digit classification benchmark dataset that contains over 600,000 color 32x32 images of printed digits (from 0 to 9) cropped from pictures of house numbers plates where images are centered on the digit of interest.

Following our reference work [20], we selected a WideResNet for this task, and given the complexity of this dataset we used all available data along with auto-augment and cutout to reach the performance of our baseline. All parameter settings follow the configuration used by the original work.

Figure 6 shows the accuracy achieved by a WideResNet-28-10 trained on the SVHN dataset. As expected the results are most promising for configuration settings of rates 0.46 and 0.59 computed from ID values of 14 and 18. Given that the trend of the SOTA for this dataset is very consistent, with over 25 different approaches with an error rate below 2% (and several of those being wide residual networks) we expected a value close to the

original configuration (0.5) to produce similar results.

Figure 6a on the left shows the performance of the network trained under overfitting prone conditions, where the network performs poorly on the test set. We can see that the configuration for $p' = 0.46$ performs better on the test set than the original configuration; furthermore, the performance for $p' = 0.59$ is only a mere 0.01% bellow the reference value.

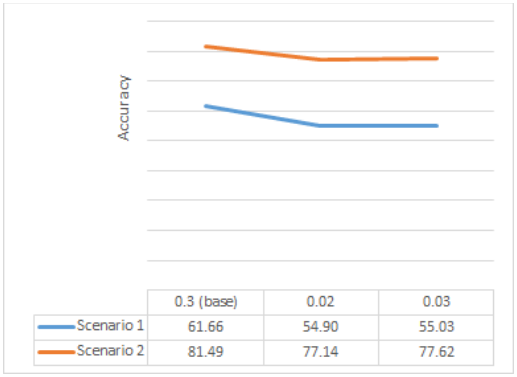
On the right (Figure 6b) we can see accuracy results for the training of the network in scenario 2, where the network was fed all the data and was trained for 160 epochs. Here, it is evident that the trend present on scenario number 1 persists, with the configuration for $p' = 0.46$ matching SOTA performance and configuration for $p' = 0.59$ performing just a little under this value with accuracy 98.93%.

In our opinion, SVHN represents a significantly more difficult problem than previously used datasets do, as images lack any contrast normalization, contain overlapping digits, clutter and distracting and noisy features. It is for this reason, that we consider that matching the state-of-the-art and outperforming the reference model under overfitting conditions is a good result to show that the presented theory might have some validity.

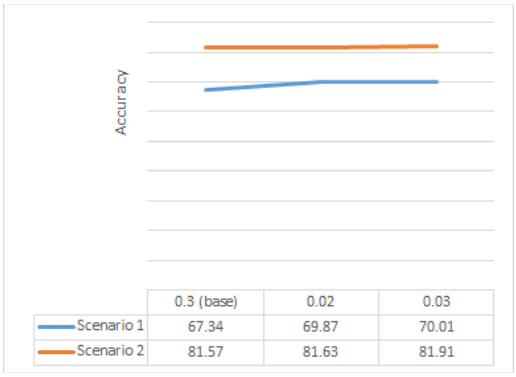
4.1.4 ImageNet

ImageNet is an image dataset organized according to the WordNet hierarchy that contains over 14 million color images, used to test models on larger scales than the ones previously discussed. It is the dataset of choice for important benchmarks in image classification and object detection and it's a very important dataset used widely in computer vision and deep learning research.

For experiments on this dataset we selected a ResNet of depth 152 and followed all configuration settings from the reference model [20]. As per prior work, we resize and crop images to 224-pixel resolution, normalize them. Originally training was done for 200



(a) Training for 400 and 200 epochs.



(b) Training for 800 and 400 epochs.

Figure 7: Accuracy scores for ResNet-152 as a function of dropout rate (p') on ImageNet.

epochs and the dropout rate used was 0.3.

As shown in Table 1, the computed values of p' for ImageNet are significantly smaller than the one estimated for the other datasets, a dropout rate of 0.02 or 0.03 represents very strong regularization and the dropout scheme will sample very thinned networks to be trained on every step. Experimentation with this setting suggested that the high regularization used impeded the network from effectively generalizing within the allotted number of epochs.

Figure 7a shows accuracy results for the network training for 400 epochs during the first scenario and 200 epochs during the second one. Clearly, the performance for the two proposed settings are worse than the results achieved using the original rate, although a small upwards trend can be observed. Hoping to give the network the ability to better adjust its internal parameters and overcome the limits set by the extremely strong regularization, we allowed the network to train from scratch for three times the number of epochs in the discussed scenarios (1200 epochs for scenario 1 and 600 epochs for scenario 2).

We can see in Figure 7b that simply by allowing the network to train for a higher number of epochs, its performance improved to the point where both proposed configurations for the dropout rate were able to beat the original setting. This underscores the notion that a model with high regularization will take longer to converge.

Just as in the case of the previously discussed CIFAR datasets, even though the accuracy scores achieved by this model are way below the state-of-the-art for this dataset, we see again a positive trend of improvement on highly overfitting situations with more precise selections of dropout rates.

4.2 Summary

During our experimentations, we successfully measured the intrinsic dimensionality of popular image datasets. Our results show that there is indeed a significant difference between this value and the dimension of the traditional pixel representation of the images that conform them. With our empirical findings, we corroborate previous assertions [20, 45] that the estimations of ID generated by the Maximum Likelihood Estimator are dependent on the choice of value for the parameter k and that they tend to grow as larger choices of k are selected.

As an indirect result of our experimentations for estimating ID values for ImageNet using different and independent samples of the dataset, we prove that at least for this data type, this estimation does not appear to be contingent to the specific samples selected, but rather, a measure of the complete dataset regardless of any particular selection of data points.

For the purpose of proving our hypothesis, we conducted experiments on image classification problems by studying the performance of models that use dropout as regularization technique and produce results close to the SOTA reported for each of our datasets. We used the ID estimations produced for each dataset to compute the quantity p' introduced in Chapter 3 and used this value to substitute the rate of the dropout layers used in the models we selected.

We trained these models under two different scenarios using the different dropout configurations obtained and report their accuracy scores for the different runs. After conducting these experiments, we have reached a new SOTA accuracy score for the MNIST dataset improving by 0.05% the previous result, and outperforming by 4.06% the studied model under overfitting conditions. For the remaining four datasets, all models under certain new configurations were able to match the performance of their original settings, and in most cases, they were able to improve those results when favorable conditions for overfitting were present.

In our view, these results confirm our hypothesis that a dropout rate computed from a relation between the ID and the actual size of images processed by a deep learning model could help improve the generalization capacity of the network.

Another indirect result of our experiments came from training the ResNet-152 on the ImageNet dataset. The fact that we needed to triple the amount of epochs needed for the model (under the new dropout rates) to achieve competitive accuracy proves once again that with higher regularization, learning models require more time to capture the trends and relationships on the data they try to fit.

Chapter 5

Conclusions and future work

In this work, we measured the intrinsic dimensionality of popular image datasets and we introduced a method for using this estimation to compute a value that can approximate the dropout rate of a deep learning model. Using this value, we proceeded to train several neural networks within the scope of image classification to verify our hypothesis and assess the impact of using this rate on their generalization capacity.

The experimental results reported in Chapter 4 demonstrate that using the values computed for p' through Equation 10 we were able to match, and in some cases outperform models with high accuracy scores under normal conditions. Under overfitting conditions, however, the accuracy results on validation sets obtained by the models with our parameter modifications are in all cases superior to the results obtained by the same models using their original configurations.

Taking into account these results, we consider our hypothesis valid, as we were indeed able to empirically prove that the new configurations have a favourable impact on the ability of these models to improve their performance, particularly under overfitting conditions.

Our results support the belief that the intrinsic dimensionality of the data constitutes an essential aspect not only for designing models and understanding their behaviour but also for many other applications, including, of course, model configuration.

Our findings with this work may raise a number of subtle directions for future work.

Directly related to our experiments, it would be interesting to investigate the effects of using the new rate for related interpretations and variations of dropout not just for regularization purposes but also in fields like uncertainty estimation.

With regards to ID estimation, future work on computing more stable and reliable estimates specific to image data would empower researchers with more precise tools to study the relationship between the ID of image datasets and other aspects of learning.

Different and more sophisticated interpretations of ID could provide more consequential heuristics for enhancing neural network learning on high-dimensional data and further push the limits of deep learning.

There is potential future research work that could either extend this technique and improve the calculations of p' by providing a more meaningful interpretation or applying it as it is to a different concept to extend the usability of ID estimations.

Due to limitations regarding computational resources and time, we could not complete our experiments using techniques like transfer learning to assess the impact of our method when using pre-trained models. Likewise, regardless of our attempts to acquire relevant non-image datasets in other domains like genetics, we were not able to secure access to assess whether our results could be replicable for other types of data.

Bibliography

[1] L. Amsaleg, J. Bailey, S. Erfani, T. Furon, M. Houle, M. Radovanovic, and N. Vinh. The vulnerability of learning to adversarial perturbation increases with intrinsic dimensionality. *IEEE Workshop on Information Forensics and Security*, pages 1–15, 06 2016.

[2] L. Ba and B. Frey. Adaptive dropout for training deep neural networks. *Advances in Neural Information Processing Systems*, 01 2013.

[3] Y. Bartal, N. Fandina, and O. Neiman. Dimensionality reduction: theoretical perspective on practical measures. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[4] R. Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[5] A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 97–104, New York, NY, USA, 2006. Association for Computing Machinery.

[6] C. M. Bishop. Bayesian pca. In *Proceedings of the 11th International Conference on Neural Information Processing Systems*, NIPS'98, page 382–388, Cambridge, MA, USA, 1998. MIT Press.

[7] C. Bouveyron, G. Celeux, and S. Girard. Intrinsic dimension estimation by maximum likelihood in probabilistic pca. *Pattern Recognition Letters*, 32:1706–1713, 10 2011.

[8] F. Camastra and A. Vinciarelli. Vinciarelli, a.: Estimating the intrinsic dimension of data with a fractal-based method. ieee trans. pami 24, 1404-1407. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24, 05 2002.

[9] P. Campadelli, E. Casiraghi, C. Ceruti, and A. Rozza. Intrinsic dimension estimation: Relevant techniques and a benchmark framework. *Mathematical Problems in Engineering*, 2015:1–21, 10 2015.

[10] G. Casanova, E. Englmeier, M. Houle, P. Kröger, M. Nett, E. Schubert, and A. Zimek. Dimensional testing for reverse k -nearest neighbor search. *Proceedings of the VLDB Endowment*, 10:769–780, 03 2017.

[11] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation policies from data, 2019.

[12] Daniel Rainer Wissel. *Intrinsic Dimension Estimation using Simplex Volumes*. PhD thesis, Rheinische Friedrich-Wilhelms-Universität Bonn, 01 2018.

[13] M. Das Gupta. Regularized maximum likelihood for intrinsic dimension estimation. In *Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 220–227, 01 2010.

[14] Z. G. David J.C. MacKay. Comments on 'maximum likelihood estimation of intrinsic dimension'. In *Gatsby Computational Neuroscience Unit*. Department of Physics, University of Cambridge, 2004.

[15] T. de Vries, S. Chawla, and M. E. Houle. Density-preserving projections for large-scale local anomaly detection. *Knowledge and Information Systems*, 32:25–52, 7 2012.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.-F. Li. Imagenet: a large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 06 2009.

[17] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *Neural Networks*, 2017.

[18] K. Falconer. *Alternative Definitions of Dimension, Techniques for Calculating Dimensions*, chapter 3, 4, pages 39–75. John Wiley and Sons, Ltd, 2003.

[19] M. Fan, H. Qiao, and B. Zhang. Intrinsic dimension estimation of manifolds by incising balls. *Pattern Recognition*, 42:780–787, 05 2009.

[20] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *International Conference on Learning Representations*, 2020.

[21] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *Neural Networks*, 2016.

[22] Y. Gal, J. Hron, and A. Kendall. Concrete dropout. *Neural Networks*, 05 2017.

[23] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *30th International Conference on Machine Learning, ICML 2013*, 1302, 02 2013.

[24] I. J. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[25] A. N. Gorban and I. Y. Tyukin. Blessing of dimensionality: mathematical foundations of the statistical physics of data. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2118):20170237, Mar 2018.

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[27] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint*, arXiv, 07 2012.

[28] D. Hirata and N. Takahashi. Ensemble learning in cnn augmented with fully connected subnetworks, 2020.

[29] M. Houle. Dimensionality, discriminability, density and distance distributions. In *Proceedings - IEEE 13th International Conference on Data Mining Workshops, ICDMW 2013*, pages 468–473, 12 2013.

[30] M. Houle. Local intrinsic dimensionality i: An extreme-value-theoretic foundation for similarity applications. In *Similarity Search and Applications*, pages 64–79, 10 2017.

[31] M. Houle. Local intrinsic dimensionality ii: Multivariate analysis and distributional support. In *Similarity Search and Applications*, pages 80–95, 09 2017.

[32] M. Houle, H. Kashima, and M. Nett. Generalized expansion dimension. In *Proceedings - 12th IEEE International Conference on Data Mining Workshops, ICDMW 2012*, pages 587–594, 12 2012.

[33] M. Houle, X. Ma, and V. Oria. Effective and efficient algorithms for flexible aggregate similarity search in high dimensional spaces. *IEEE Transactions on Knowledge and Data Engineering*, 27:3258–3273, 12 2015.

[34] M. Houle, X. Ma, V. Oria, and J. Sun. Efficient similarity search within user-specified projective subspaces. *Information Systems*, 59, 02 2016.

[35] M. E. Houle, X. Ma, M. Nett, and V. Oria. Dimensional testing for multi-step similarity search. In *2012 IEEE 12th International Conference on Data Mining*, pages 299–308, 2012.

[36] I. Jolliffe. Principal component analysis. 2nd ed. *http://lst-iiep.iiep-unesco.org/cgi-bin/wwwi32.exe/[in=epidoc1.in]/?t2000=017716/(100)*, 98, 10 2005.

[37] D. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. *Conference Proceedings of the Annual ACM Symposium on Theory of Computing*, 09 2002.

[38] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[39] D. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. *Neural Networks*, 06 2015.

[40] A. Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[41] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012.

[42] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29:1–27, 03 1964.

[43] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[44] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998.

[45] E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems*, volume 17, 01 2004.

[46] G. Lombardi, A. Rozza, C. Ceruti, E. Casiraghi, and P. Campadelli. Minimum neighbor distance estimators of intrinsic dimension. In D. Gunopulos, T. Hofmann, D. Malerba, and M. Vazirgiannis, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 374–389, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[47] X. Ma, B. Li, Y. Wang, S. Erfani, S. Wijewickrema, M. Houle, G. Schoenebeck, D. Song, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *35th Int. Conf. on Machine Learning*, 01 2018.

[48] S. Merity, N. S. Keskar, and R. Socher. Regularizing and optimizing lstm language models. *Neural and Evolutionary Computing*, 2017.

[49] T. Moon, H. H. Choi, H. Lee, and I. Song. Rnndrop: a novel dropout for rnns in asr. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 65–70, 12 2015.

[50] P. Mordohai and G. Medioni. Dimensionality estimation, manifold learning and function approximation using tensor voting. *J. Mach. Learn. Res.*, 11:411–450, Mar. 2010.

[51] H. Narayanan and P. Niyogi. On the sample complexity of learning smooth cuts on a manifold. *COLT 2009 - The 22nd Conference on Learning Theory*, 01 2009.

[52] M. Net. Dimensions. In *Math Net*. math.net, 2021.

[53] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Ng. Reading digits in natural images with unsupervised feature learning. *NIPS*, 01 2011.

[54] A. Ng. Feature selection, l 1 vs. l 2 regularization, and rotational invariance. *Proceedings of the Twenty-First International Conference on Machine Learning*, 09 2004.

[55] S. J. Nowlan and G. E. Hinton. Simplifying neural networks by soft weight-sharing. *Neural Computation*, 4(4):473–493, 1992.

[56] K. Pettis, T. A. Bailey, A. K. Jain, and R. Dubes. An intrinsic dimensionality estimator from near-neighbor information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1:25–37, 1979.

[57] A. Poernomo and D.-K. Kang. Biased dropout and crossmap dropout: Learning towards effective dropout regularization in convolutional neural network. *Neural Networks*, 104, 04 2018.

[58] P. Pope, C. Zhu, A. Abdelkader, M. Goldblum, and T. Goldstein. The intrinsic dimension of images and its impact on learning. *Computer Vision and Pattern Recognition*, 04 2021.

[59] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[60] X. Shen, X. Tian, T. Liu, F. Xu, and D. Tao. Continuous dropout. *IEEE Transactions on Neural Networks and Learning Systems*, PP:1–12, 10 2017.

[61] 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*, 15:1929–1958, 06 2014.

[62] J. Tenenbaum, V. Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science (New York, N.Y.)*, 290:2319–23, 01 2001.

[63] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[64] C. University. Dimension. In *Cambridge Dictionary*. Cambridge University Press, 2021.

[65] J. Venna and S. Kaski. Local multidimensional scaling. *Neural Networks*, 19(6):889–899, 2006. Advances in Self Organising Maps - WSOM'05.

[66] L. Wan, M. Zeiler, S. Zhang, Y. Lecun, and R. Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, 01 2013.

[67] L. Wasserman. *All of statistics : a concise course in statistical inference*. Springer, New York, 2010.

[68] H. Wu and X. Gu. Max-pooling dropout for regularization of convolutional neural networks. *Neural Networks*, 2015.

[69] Y. Yao, L. Rosasco, and A. Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26:289–315, 08 2007.

[70] K. Zolna, D. Arpit, D. Suhubdy, and Y. Bengio. Fraternal dropout. *International Conference on Learning Representations*, 10 2017.

Appendix A

Experiments

A.1 EnsNet

The EnsNet model consists of one base CNN and multiple Fully Connected SubNetworks (Figure A.1). The base CNN generates a set of multi-channel feature-maps after each convolutional layer. The set of feature-maps generated by the last convolutional layer is divided along channels into disjoint subsets, and each subset is assigned to one of the FCSNs, which is trained independent of others so that it can predict the class label from the subset of the feature maps assigned to it. The output of the overall model is determined by majority vote of the base CNN and the FCSNs [28].

The complete ensemble is trained by alternating two steps, one is the base CNN training step and the other is the subnetworks training step. The networks are training using an Adam optimizer [38] with parameters $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and a weight decay of 0.

The data augmentation is performed by rotating images by various angles between -10° and 10°, scaling images by various factors between 0.8 and 1.2, shifting images to the width direction or the height direction by a fraction between -0.08 and 0.08 of the total width or the total height, stretching images by the shear transformation with various angles

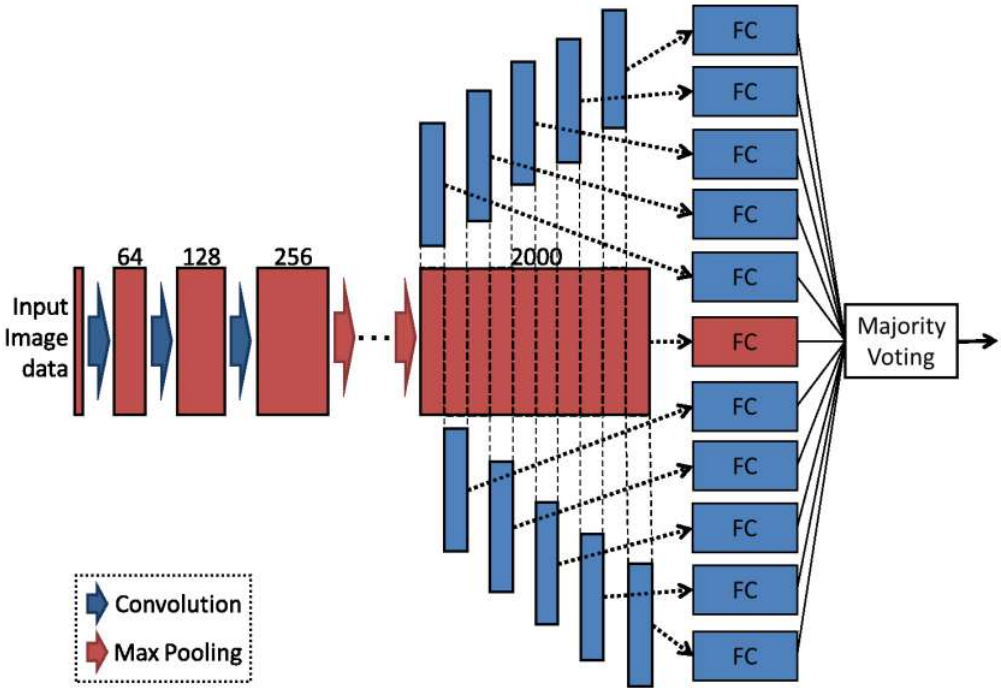


Figure A.1: An example of the architecture of the EnsNet.

between $-0.3°$ and $0.3°$. Batch size and the number of epochs are set to 100 and 1,300 respectively. Dropout rates were set to 0.35 and a dropconnect rate of 0.5 was used on the subnetworks. An additional dropout rate of 0.5 was used on the fully connected layers.

A.2 SAM Experiments

For all experiments using the *Sharpness-Aware Minimization for Efficiently Improving Generalization* (SAM) [20], we refer model configurations to the original paper and replicate them to match the results reported. The SAM hyper-parameter ρ (neighborhood size) is reported for the experiments involving this optimizer.

Please refer to original paper for detailed hyperparameters and additional training details.

A.2.1 WideResNet and PyramidNet

We report in Table A.1 the hyperparameters used for the experiments conducted with the models WideResNet and PyramidNet on the datasets CIFAR-10, CIFAR-100 and SVHN. All other model hyperparameter values are identical to those used in prior work [20] and the batch size used in all cases was 256.

Models \ Parameters	LR	WD	ρ
WRN-28-10 (CIFAR-10)	0.05	0.001	0.05
PyramidNet (CIFAR-10)	0.05	0.0005	0.05
WRN-28-10 (CIFAR-100)	0.05	0.001	0.1
PyramidNet (CIFAR-100)	0.05	0.0005	0.2
WRN-28-10 (SVHN)	0.01	0.0005	0.01

Table A.1: Hyper-parameters used for WideResNet and PyramidNet.

A.2.2 ResNet

For the experiments performed on the ImageNet dataset, we used a ResNet of depth 152 using a batch size of 4096. The initial learning rate is set to 1.0 and is then decayed using a cosine schedule. Weight decay is set to 0.0001 with SGD optimizer and momentum of 0.9 and the value for SAM parameter ρ was set to 0.05.