INCORPORATING NEURAL NETWORK VERIFICATION INTO THE EVALUATION OF PAC BOUNDS

Thomas Walker
Imperial College London
thomas.walker21@imperial.ac.uk

Alessio Lomuscio
Imperial College London
a.lomuscio@imperial.ac.uk

ABSTRACT

The theory of Probably Approximately Correct (PAC) generalization bounds for neural networks has been around since the work of [1]. Initially, they were a theoretical construct that elucidated the details of the learning process and gave probabilistic guarantees on the performance of machine learning models on unseen data. However, their utility in practice was only first realised by [2] who managed to contextualize the bounds for neural networks in a non-vacuous manner. Since then there have been other successful implementations [3][4][5]. Each of these optimise different components of the bounds to ensure their tightness and they are evaluated using finite training sets of discrete points. Although we cannot train networks on regions of the input space, we can often verify the network's performance in these regions. In this work, we want to understand how knowing the network's performance on a region of the input space can be used to condition PAC bounds.

1 Introduction

Usually, neural networks are implemented to form a representation of a distribution defined on an input space. A priori this distribution is unknown, and we only have a sampled training set. What we want to do is manipulate the parameters of the neural network, using the training data, to generate a reasonable approximation of the underlying distribution. To make progress with this we need to understand whether such a configuration of parameters exists, and what we mean by a reasonable approximation.

The neural network training paradigm involves a loss function that is evaluated for a particular neural network on the training data, to identify the discrepancy between the representation and the underlying distribution at the points defined by the training sample. The parameters are refined through back-propagation to reduce the discrepancy and achieve a lower loss value.

On the one hand, modern neural networks are often able to accommodate many more parameters than training samples. Hence, their expressivity encapsulates representations that are able to interpolate the training data. Consequently, it can be shown empirically that neural networks are able to reach close to zero loss for even a random sample of data that has no structure.

Therefore, quantifying what we mean by a reasonable representation using the loss function is perhaps not ideal. However, it has been shown that neural networks trained in the way described above, arrive at representations that are more general than interpolating the training sample, so perhaps the loss value is still indicative of the concept of a reasonable representation. Intuitively, what we mean by a reasonable representation is a general representation. That is, the error between the network representation and the entire distribution is small, rather than just small at the points of the training sample.

Here we will be concerned with developing probabilistic guarantees on the generalisation of learned representation. Where by generalisation we refer to the extent to which the neural network representation captures high-level features of the training set, to enable meaningful inferences to be made about inputs lying outside of the training set. Although similar to the analysis of neural network performance on out-of-distribution inputs, this will not be the focus of this work; instead, we investigate the overfitting of the neural network representations.

It is empirically well-known that neural networks have a remarkable capacity to learn general representations. Although, theoretically this phenomenon still eludes us. Probably Approximately Correct (PAC) bounds is a theoretical tool developed to provide quantitative probabilistic guarantees on the generalisation property of neural networks. With high probability, they bound the difference in network performance on training data and the underlying distribution. Different types of PAC bounds appeal to different components of the learning process. For example, PAC-Bayes bounds are developed under the Bayesian machine learning framework.

Since the formulation of the PAC learning framework in [1], there has been little success in non-vacuously implementing these bounds for neural networks. However, [2] optimised the evaluation of PAC-Bayes bounds for neural networks to the extent to which they provided non-vacuous results. Then, [6] introduced PAC compression bounds derived from compression algorithms that were designed to reduce the number of parameters needed to represent a given neural network whilst guaranteeing a certain level of performance. Using these algorithms they were able to derive bounds on the generalisation error, however, these particular bounds were not meaningful in practice. It did motivate the subsequent work of [3] that combines this paradigm with the Bayesian framework by utilizing the notion of a compression scheme to develop priors that tightened PAC-Bayes bounds sufficiently for practical implementation. Then, [5] extended this line of reasoning to further improve the tightness of the bounds.

In this work, we intend to introduce a different strategy for evaluating these bounds. The current strategy only considers discrete samples, where here we capitalise on our ability to determine network performance in a region of the input space. The intuition is that operating with a region allows us to infer more information on the true behaviour of the network on the underlying distribution. As we can only train neural networks on discrete finite sets, these updates will have to occur after training which will introduce some added constraints to when our updated bounds will hold.

2 Problem Formalisation

2.1 Notation

Suppose our input space is $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the feature space and $\mathcal{Y}$ is the output space. We suppose that there is an unknown distribution $\mathcal{D}$ defined on this space for which the training process aims to learn a network representation, $h : \mathcal{X} \to \mathcal{Y}$. This will involve employing a learning algorithm on a training set $S = \{z_i\}_{i=1}^m = \{(x_i, y_i)\}_{i=1}^m$ which we assume consists of m independent and identically distributed, i.i.d, samples from $\mathcal{D}$. Our neural network will be parameterised by a weight vector $\mathbf{w} \in \mathcal{W}$ with the corresponding network denoted $h_{\mathbf{w}} \in \mathcal{H}$. The sets $\mathcal{W}$ and $\mathcal{H}$ will be referred to as the parameter space and the hypothesis space respectively. To assess the quality of a particular network representation we use a bounded loss function $l : \mathcal{Y} \times \mathcal{Y} \to [0, C]$. The loss function quantifies the difference between a network's output and the true output according to the distribution $\mathcal{D}$. For $z = (x, y) \in \mathcal{Z}$ we identify the quantity $l_z(\mathbf{w}) := l(h_{\mathbf{w}}(x), y)$ as the error of this particular point of the input space under the representation $h_{\mathbf{w}}$. Using this we can define the risk of our network to be

$$R(\mathbf{w}) = \mathbb{E}_{z\sim\mathcal{D}}(l_z(\mathbf{w})).$$

Note that this is dependent on the unknown distribution $\mathcal{D}$ and hence is also unknown. From the previous discussion, we say that a network representation is general, or reasonable if it has low risk. In practice, we work with the empirical risk of the network

$$\hat{R}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} l_{z_i}$$

which is implicitly dependent on a training set and is such that $\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\mathbf{w})\right)$. As mentioned before, it has been shown empirically that low empirical risk correlates with low risk. However, it is immediately clear as to why this should be the case, and there is a lot of work that is addressed to answer this question. It will be useful to introduce the notation $[k] = 1, \ldots, k$ and $[[k]]_m = k, \ldots, m$ for $k, m \in \mathbb{N}$.

2.2 Problem Statement

Although current methods for generating PAC bounds for neural networks operate under the PAC-Bayes framework. Here we will consider a generic PAC bound, where a prior distribution is not required. This is done for simplicity purposes, and it is indeed possible to extrapolate the discussions made here to the Bayesian setting.

Theorem 2.1 ([7]). *Let $|\mathcal{W}| = M < \infty$, $\delta \in (0, 1)$ and $\mathbf{w} \in \mathcal{W}$ then it follows that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \hat{R}(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{M}{\delta}\right)}{2m}}\right) \geq 1 - \delta.$$

As neural networks often operate in the over-parameterised setting, as we alluded to before, it is possible to obtain zero training error on the training set, that is $\hat{R}(\mathbf{w}) = 0$ for some $\mathbf{w} \in \mathcal{W}$. Assuming this to be the case we see that Theorem 2.1 gives us a bound on the risk of the network that holds with a confidence of $1 - \delta$. More importantly, it tells us how this bound changes as we add points to the training set. The bound gets tighter to the true risk of the network as we increase the number of points on which we train the network. Adding an extra data point and training the network to zero training error reduces the bound of Theorem 2.1 by

$$C\sqrt{\frac{\log\left(\frac{M}{\delta}\right)}{2}}\left(\frac{1}{\sqrt{m}} - \frac{1}{\sqrt{m+1}}\right).$$

Or we can say that the original bound holds with an increased confidence of

$$1 - \delta' = 1 - \frac{M}{\left(\frac{M}{\delta}\right)^{\frac{m+1}{m}}} \geq 1 - \delta.$$

Suppose instead that we have trained to a network $h_{\mathbf{w}}$ for which we can guarantee a zero loss on some region $\Delta \subset \mathcal{Z}$, we will denote this assumption by $l_\Delta(\mathbf{w}) = 0$. If one considers a classification task where the loss function is the 0-1 error, then all this assumption is saying is that our network performs as expected in the region Δ. Implicit here is the assumption that the region Δ is independent of our training set S, which will be important to keep in mind when we begin experimenting.

One of the major implications of our assumption is that it provides information for the distribution $\mathcal{D}$. From the assumption, we can understand the shape of $\mathcal{D}$ in the region Δ, however, we are not able to explicitly calculate

$$p_\Delta = \mathbb{P}_{z\sim\mathcal{D}}(z \in \Delta) = \int_{z\in\Delta} \mathcal{D}(z)dz.$$

For example, in the context of classification tasks and with l being the 0-1 error, our assumption tells us the class of the points in the region Δ. This assumption will only prove useful for improving statistical guarantees if we know p_Δ. One can think of p_Δ as the significance of the region under the distribution $\mathcal{D}$. Therefore, for now, we will suppose that we have access to this quantity and later we will address how we may calculate it in practice. It is important to understand that our assumptions will not affect the quantity $R(\mathbf{w})$ as this value is calculated under the stronger assumption that we know the full distribution $\mathcal{D}$.

3 The Refinement of PAC Bounds

As we saw, there are two ways in which we could improve bounds. We can either make the bound smaller or ensure that the existing bound holds with greater confidence.

3.1 Improving the Tightness of Bounds

Following the steps of the proof of Theorem 2.1 we can determine an updated bound under our assumptions.

Theorem 3.1. *For $\mathbf{w} \in \mathcal{W}$ and $\delta \in (0, 1)$ we have that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \hat{R}(\mathbf{w}) + CB(m, p_\Delta, \delta) \Big| l_\Delta(\mathbf{w}) = 0\right) \geq 1 - \delta$$

for

$$B(m, p_\Delta, \delta) = \sqrt{\frac{\log\left(\frac{(1-p_\Delta)+\sqrt{(1-p_\Delta)^2+4\delta^{\frac{1}{m}}p_\Delta}}{2\delta^{\frac{1}{m}}}\right)}{2}}.$$

Remark 3.2.

- *Note how the result of Theorem 3.1 holds for a fix $\mathbf{w} \in \mathcal{W}$.*
- *Recall that the region Δ must be independent of the sample set S for this result to hold. Moreover, $\mathbf{w}$ is independent of the sample S on which the empirical risk is calculated. These independence conditions will implicitly be present in all the results that follow.*

By letting $p_\Delta = 0$ we just get the statement of the Theorem 2.1 without the factor of M, as we are only focused on a single parameter value. With $p_\Delta = 1$ we see that $B(m, p_\Delta, \delta) > 0$ which is not ideal as in such a scenario we would know that $R(\mathbf{w}) = 0$, and the inequality holds trivially. This issue arises due to a step in the proof of the theorem.

3.2 Improving the Confidence of Bounds

We can also look at improving the confidence with which bounds hold by conditioning on the event that $l_\triangle(\mathbf{w}) = 0$, which is essentially equivalent to improving the tightness of a bound. It is potentially more desirable to improve the tightness of bounds as this improvement manifests more readily in practical applications. However, we proceed with improving the confidence of bounds as often the explicit results are easier to derive. Furthermore, it provides a standardised metric to quantify the improvement imposed by conditioning on the event $l_\triangle(\mathbf{w}) = 0$.

Theorem 3.3. *For $\mathbf{w} \in \mathcal{W}$ and $\delta \in (0,1)$ we have that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \hat{R}(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}}\,\middle|\, l_\triangle(\mathbf{w}) = 0\right) \geq 1 - \left(\sum_{k=1}^{m}\binom{m}{k}\delta_k p_\triangle^{m-k}(1-p_\triangle)^k\right)$$

where

$$\delta_k = \frac{1}{\left(\frac{1}{\delta}\right)^{\frac{m^2}{k^2}}}.$$

Note that $\delta_k \leq \delta$ so that we do get an improvement in the confidence of the bound. Again we see that with $p_\triangle = 0$ we recover the bounds and the confidence of Theorem 2.1. However, what we notice now is that when we let $p_\triangle = 1$ we get full confidence in our bound as we expect, which is not the case in Theorem 3.1.

3.3 Extension to the Bayesian Framework

We will now transfer these ideas to a result that is derived to hold for all parameters in a countable parameter space on which a prior distribution $\pi(\mathbf{w})$ is defined. The requirement that the parameter space is countable is not as restrictive as it may seem, as computers necessarily need to work with floating point numbers and so the values of the parameters will be from a countable set even if we set out the problem over an uncountable parameter space. One often takes the prior to be the random initialisation of the neural network parameters.

Theorem 3.4 ([8])**.** *Simultaneously for all $\mathbf{w} \in \mathcal{W}$, $\delta \in (0,1)$ and a prior distribution π defined on $\mathcal{W}$ we have that,*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \inf_{\lambda>\frac{1}{2}}\frac{1}{1-\frac{1}{2\lambda}}\left(\hat{R}(\mathbf{w}) + \frac{\lambda C}{m}\left(\log\left(\frac{1}{\pi(\mathbf{w})}\right) + \log\left(\frac{1}{\delta}\right)\right)\right)\right) \geq 1-\delta.$$

We can re-derive this bound to hold for a single parameter value, which will allow us to drop the assumption that the parameter space is countable and the prior distribution will no longer influence the bound.

Theorem 3.5. *For $\mathbf{w} \in \mathcal{W}$ and $\delta \in (0,1)$ we have that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \inf_{\lambda>\frac{1}{2}}\frac{1}{1-\frac{1}{2\lambda}}\left(\hat{R}(\mathbf{w}) + \frac{\lambda C}{m}\left(\log\left(\frac{1}{\delta}\right)\right)\right)\right) \geq 1-\delta.$$

In a similar way to before we can condition this probability on the event that $l_\triangle(\mathbf{w}) = 0$ to improve the confidence with which the bound holds.

Theorem 3.6. *For $\mathbf{w} \in \mathcal{W}$ and $\delta \in (0,1)$ we have that*

$$\begin{aligned}\mathbb{P}_{S\sim\mathcal{D}^m}&\left(R(\mathbf{w}) \leq \inf_{\lambda>\frac{1}{2}}\frac{1}{1-\frac{1}{2\lambda}}\left(\hat{R}(\mathbf{w}) + \frac{\lambda C}{m}\left(\log\left(\frac{1}{\delta}\right)\right)\right)\,\middle|\, l_\triangle(\mathbf{w}) = 0\right)\\ &\geq 1 - \left(\sum_{k=1}^{m}\binom{m}{k}\exp\left(-\frac{m^2\epsilon(\mathbf{w})^2}{2kR(\mathbf{w})}\right)p_\triangle^{m-k}(1-p_\triangle)^k\right),\end{aligned}$$

where

$$\epsilon(\mathbf{w}) = \sqrt{\frac{2R(\mathbf{w})\log\left(\frac{1}{\delta}\right)}{m}}.$$

This is indeed an improvement in confidence as

$$\exp\left(-\frac{m^2\epsilon(\mathbf{w})^2}{2kR(\mathbf{w})}\right) \leq \exp\left(-\frac{m\epsilon(\mathbf{w})^2}{2R(\mathbf{w})}\right) = \delta.$$

Again with $p_\triangle = 0$ and $p_\triangle = 1$ we get the same conclusions as stated for Theorem 3.3

4 Experiments

Our experiment here is going to be on evaluating the bound of Theorem 3.1 for a neural network trained on the MNIST dataset. To do so we are going to need to verify a region Δ and then get a value for p_Δ.

4.1 Approximating p_Δ

So far, we have operated under the assumption that we know the value of p_Δ. We used p_Δ as a theoretical device to explore its interaction with the bounds. For our results to be practically justified we need to get an explicit grip on this quantity. Knowing the value of p_Δ exactly would just amount to knowing the shape and the value of $\mathcal{D}$ in the region Δ, which is theoretically possible but realistically unlikely. Instead, we will determine a confidence region for the value of p_Δ. Suppose that we have verified a region Δ of the input space. Using a sample S_A, of size m_A we estimate p_Δ in the following way.

For $(x_i, y_i) \in S_A$ we can define the $\mathrm{Bern}(p_\Delta)$ random variable

$$Z_i = \begin{cases} 1 & z_i \in \Delta \\ 0 & z_i \in \mathcal{Z} \setminus \mathcal{D}. \end{cases}$$

An estimate of p_Δ is then given by

$$\hat{p}_\Delta = \frac{1}{m_A} \sum_{i=1}^{m_A} Z_i,$$

which we can use to derive confidence intervals for p_Δ and update our results accordingly. Due to the often large sample sizes we deal with in the machine learning setting we should be able to infer relatively small confidence intervals. Ideally, we would like to find the shortest confidence interval such that the variability of the bound using this approximation is minimized. A common method for forming these intervals is by using the normal approximation, which asymptotically provides the shortest confidence interval. However, exact intervals provide a concrete comparison to the bound before we condition on the assumption. Hence, we use the $1-\alpha$ Clopper-Pearson confidence interval that is exact and is given by

$$\left[q_B\left(\frac{\alpha}{2}, m_A\hat{p}_\Delta, m_A - m_A\hat{p}_\Delta + 1\right), q_B\left(1 - \frac{\alpha}{2}, m_A\hat{p}_\Delta + 1, m_A - m_A\hat{p}_\Delta\right)\right],$$

where $q_B(\cdot, \beta_1, \beta_2)$ is the quantile function for the beta distribution with shape parameters β_1 and β_2 [9]. We can assume that our bounds are decreasing functions in δ so that we can instead work with the $1-\alpha$ one-sided confidence interval

$$\left[q_B\left(\alpha, m_A\hat{p}_\Delta, m_A - m_A\hat{p}_\Delta + 1\right), 1\right].$$

Recall that we have determined a region Δ for which we know the shape of the unknown distribution $\mathcal{D}$. We can say that with probability $1-\alpha$ the probability mass of the region Δ under $\mathcal{D}$ is greater than $p_L := q_B\left(\alpha, m_A\hat{p}_\Delta, m_A - m_A\hat{p}_\Delta + 1\right)$. If we have a bound $B(p_\Delta)$ that is a decreasing function of p_Δ such that

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + B(p_\Delta)\right) \leq \delta.$$

Then $B(p_L) \geq B(p')$ for all $p' \geq p_L$ so that,

$$\begin{aligned} \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + B(p_L)\right) =& \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + B(p_L) \middle| p_\Delta \geq p_L\right) \mathbb{P}_{S\sim\mathcal{D}^m}(p_\Delta \geq p_L) \\ & + \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + B(p_L) \middle| p_\Delta < p_L\right) \mathbb{P}_{S\sim\mathcal{D}^m}(p_\Delta < p_L) \\ \leq& \delta(1-\alpha) + (1)(\alpha) \\ =& \delta + \alpha(1-\delta). \end{aligned}$$

The intention is choose to α sufficiently small so that the added $\alpha(1-\delta)$ term is small. Decreasing α will always decrease p_L and hence increase $B(p_L)$ and so to maintain a tight bound we must supplement the decreasing of α with an increase in m_A, as this will increase p_L and subsequently decrease $B(p_L)$. In the machine learning paradigm, this may involve holding out some data from the training set to approximate p_Δ. We will explore this trade-off in our experiments.

4.2 Setup

We conducted our preliminary experiments on the MNIST dataset, using a ReLU neural network classifier. Specifically, we trained a ReLU network with three hidden layers of size 256 using stochastic gradient descent, a learning rate of 0.01 and a momentum of 0.9. Recall, that we have some independence conditions to satisfy. This requires us to partition the training sample to evaluate the bound.

1. Our first partition, S_1, was used to train the network.
2. Our second partition, S_2, was used to approximate the value of $p_\triangle$ and evaluate the empirical risk of our network.
    - Note how we can use a single partition to both approximate $p_\triangle$ and evaluate $\hat{R}(\mathbf{w})$, as there is no dependence between these steps.

Our dataset, S, consists of 60000 images of digits of resolution 28×28. To generate the partitions we introduced a hyperparameter η, where $|S_1| = (1-\eta)|S|$ and $|S_2| = \eta|S|$. We can now proceed in training our network using S_1, where we perform 25 epochs. It is important to note here that we train the network using a cross-entropy loss. Despite being an unbounded loss this is not an issue, as the empirical risk is evaluated using the 0-1 loss. We train in this way as the cross-entropy loss is a richer metric for the training process, however, our assumption that $l_\triangle(\mathbf{w}) = 0$ only really makes sense when using the bounded 0-1 loss.

To explicitly analyse the performance of our bound we gave ourselves access to some underlying distribution from which to evaluate the true error of our neural network and establish the quality of our approximation of the value $p_\triangle$. To do this we used the entire dataset of 60000 points to define a discrete underlying distribution. and so we sampled from this distribution and partitioned the sample to maintain these.

4.3 Verifying the Region Δ

Once our network is training we can then move on to determining the region Δ, and thus employ our method for approximating $p_\triangle$. To define Δ we took the approach of defining ϵ-balls around the correctly classified training points.

1. For a trained network we identify the correctly classified points for each label, let $\mathcal{C}_i$ denote the correctly classified points for label i.
2. For each label we perform singular value decomposition (SVD) on them along with the correspondingly labelled points in S_2.
    - For the label i the SVD produces the matrices $\mathbf{U}_i$, $\mathbf{S}_i$ and $\mathbf{V}_i$. Where the k^{th} column of $\mathbf{V}_i$ can be thought of as the representation of an image in basis provided by the principal components, which are the rows of $\mathbf{U}_i$.
3. For each column of $\mathbf{V}_i$ corresponding to the representation of an image in $\mathcal{C}_i$, increase the radius of an ϵ-ball around representation until the network no longer correctly classifies points in the ball.
    - A limit on the noise's amplitude was enforced as eventually, the noise masks the structure of the digit. At this point, the network would just be making guesses, which for large training sets would result in a few images still being correctly classified for incredibly large amplitudes of noise.
4. The maximum noise amplitude permissible for each training image to maintain correct classification was noted and used to define our region Δ.
    (a) To claim that noise of a particular amplitude was permissible, the network would have to correctly classify five images perturbed with the noise of that amplitude.

Once we have this region Δ, we can use the columns of the matrix $\mathbf{V}_i$ corresponding to the representation of the i^{th} labelled points in S_2 to find a lower bound for the value of $p_\triangle$. Specifically, we can calculate the distance between the representation of this point and the representations of the point in $\mathcal{C}_i$ and determine whether it lies in one of the ϵ-balls defined previously.

The reason we decide to define the ϵ-balls in the space defined by SVD is that it leads to us being able to define more significant regions in the inputs. Refer to Figure 1 which shows how ϵ-balls of equal radius differ in regular Euclidean space and the space defined by the principal components.

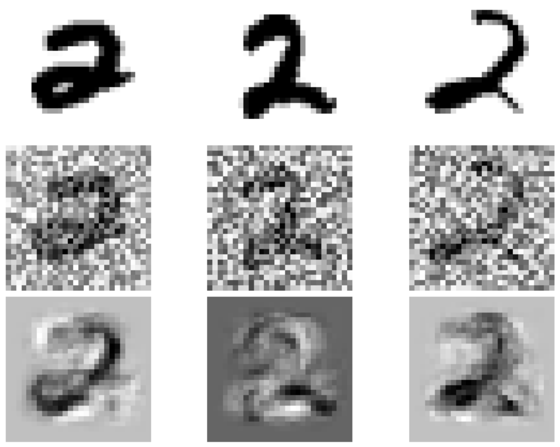

Figure 1: **Top Row:** Images from the MNIST data set, labelled as 2. **Middle Row:** Images of the top row perturbed in Euclidean space with noise of amplitude 0.5. **Bottom Row:** Images of the top row perturbed in the space defined by the principal components of the SVD with noise of amplitude 0.5.

4.4 Results

The above experiment was performed for $\eta = 0.1, 0.2, 0.3, 0.4$. The results of these experiments can be summarised in Table 1.

η	$\hat{p}_\Delta$	p_L	Empirical Risk	Conditioned Bound	Unconditioned Bound	Bound on True Risk
0.1	0.8345	0.8264	0.0183	0.0117	0.0158	0.0300
0.2	0.7690	0.7626	0.0179	0.0084	0.0112	0.0263
0.3	0.6315	0.6255	0.0198	0.0072	0.0091	0.0270
0.4	0.6553	0.6502	0.0233	0.0061	0.0079	0.0295

Table 1: Results for Experiments Conducted on MNIST.

From Table 1 we see the most significant region is defined when $\eta = 0.1$, however, this does not lead to the tightest bound. Recall, that there is a trade-off in the bound with the sample size used to evaluate the bound, m, and the significance of the region, $\hat{p}_\Delta$. Consequently, we see that the tightest bound is achieved when $\eta = 0.4$, despite the substantial differences in the sizes of $\hat{p}_\Delta$. Similarly, there is a trade-off between the size of the sample used to train the network, which affects the performance of the network, and the size of the sample we use to evaluate the bound. Hence, the lowest bound on the true risk is given when $\eta = 0.2$.
In these experiments, we are not too interested in the tightness of the bound, as the large sample sizes already mean that the bounds are relatively tight. We are interested in our capacity to define significant regions of the input space, and the comparison between between obtained when we do not verify a region Δ.

η	% of Unconditioned Bound	% Increase in Sample Size to Compensate
0.1	74.1%	82.4%
0.2	75.0%	76.9%
0.3	79.1%	60.5%
0.4	77.2%	67.7%

Table 2: Comparison between conditioned and unconditioned bounds.

From Table 2 we see that verifying a region Δ and conditioning our PAC bounds on this information can lead to significant improvements in the tightness of our bounds. Quantitatively, this improvement is equivalent to increasing the size of the sample on which we evaluate these bound by around 70%.

5 Conclusion

We have seen how the knowledge that a network operates as intended on a region of the input space can tighten PAC bounds. When significant regions of the input space can be verified, then optimal bounds can lead to potentially significant tightening of the bounds. Our methodology for tightening PAC bounds relies on the idea that we can verify significant regions of the input space as determined by the underlying probability distribution. Through experiments on the MNIST dataset, we demonstrated that in practice such regions can be defined and utilized in the tightening of the bounds. Subsequent work should look into implementing optimal bounds, as we noted the bound we used for our experiment was not optimal. In the sense that, having $p_\triangle = 1$ did not give us full confidence bound. Moreover, work could look at using variational techniques to get a better grasp on $p_\triangle$ and extend the experiments to more complex data sets.

Acknowledgments

This was funded in part by both the Department of Mathematics and the Department of Computer Science at Imperial College London. Moreover, it was facilitated and inspired by the Verification of Autonomous Systems group at Imperial College London directed by Professor Alessio Lomsucio.

References

[1] David A. McAllester. Some pac-bayesian theorems. *Machine Learning*, 37:355–363, 1998.

[2] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *CoRR*, 2017.

[3] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz. Non-vacuous generalization bounds at the imagenet scale: A pac-bayesian compression approach, 2019.

[4] Gintare Karolina Dziugaite, Kyle Hsu, Waseem Gharbieh, and Daniel M. Roy. On the role of data in pac-bayes bounds. *CoRR*, 2020.

[5] Sanae Lotfi, Marc Finzi, Sanyam Kapoor, Andres Potapczynski, Micah Goldblum, and Andrew Gordon Wilson. Pac-bayes compression bounds so tight that they can explain generalization, 2022.

[6] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. *CoRR*, 2018.

[7] Pierre Alquier. User-friendly introduction to pac-bayes bounds, 2023.

[8] David A. McAllester. A pac-bayesian tutorial with A dropout bound. *CoRR*, 2013.

[9] Nathaniel E. Helwig. Inference for proportions, 2020.

[10] Clayton Scott. Hoeffding's inequality, 2014.

[11] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

6 Appendix

6.1 Proofs

Lemma 6.1 ([10]). *Let $U_1, \dots, U_n$ be independent random variables taking values in an interval $[a, b]$. Then for any $t > 0$ we have that*

$$\mathbb{E}\left(\exp\left(t\sum_{i=1}^{n}(U_i - \mathbb{E}(U_i))\right)\right) \le \exp\left(\frac{nt^2(b-a)^2}{8}\right).$$

Proof. For $s > 0$ the function $x \mapsto e^{sx}$ is convex so that

$$e^{sx} \le \frac{x-a}{b-a}e^{sb} + \frac{b-x}{b-a}e^{sa}.$$

Let $V_i = U_i - \mathbb{E}(U_i)$, then as $\mathbb{E}(V_i) = 0$ it follows that

$$\mathbb{E}\left(\exp(sV_i)\right) \le \frac{b}{b-a}e^{sa} - \frac{a}{b-a}e^{sb}.$$

With $p = \frac{b}{b-a}$ and $u = (b-a)s$ consider

$$\psi(u) = \log\left(pe^{sa} + (1-p)e^{sb}\right) = (p-1)u + \log\left(p + (1-p)e^{u}\right).$$

This is a smooth function so that by Taylor's theorem we have that for any $u \in \mathbb{R}$ there exists $\xi = \xi(u) \in \mathbb{R}$ such that

$$\psi(u) = \psi(0) + \psi'(0)u + \frac{1}{2}\psi''(\xi)u^2.$$

As

$$\psi'(u) = (p-1) + 1 - \frac{p}{p + (1-p)e^u}$$

we have that $\psi(0) = 0$ and $\psi'(0) = 0$. Furthermore, as

$$\psi''(u) = \frac{p(1-p)e^u}{(p+(1-p)e^u)^2}, \text{ and } \psi^{(3)}(u) = \frac{p(1-p)e^u(p+(1-p)e^u)(p-(1-p)e^u)}{(p+(1-p)e^u)^2}$$

we see that $\psi''(u)$ has a stationary point at $u^* = \log\left(\frac{p}{p-1}\right)$. For u slightly less than u^* we have $\psi^{(3)}(u) > 0$ and for u slightly larger than u^* we have $\psi^{(3)}(u) < 0$. Therefore, u^* is a maximum point and so

$$\psi''(u) \leq \psi''(u^*) = \frac{1}{4}.$$

Hence, $\psi(u) \leq \frac{u^2}{8}$ which implies that

$$\log\left(\mathbb{E}\left(\exp(sV_i)\right)\right) \leq \frac{u^2}{8} = \frac{s^2(b-a)^2}{8}.$$

Therefore,

$$\begin{aligned}\mathbb{E}\left(\exp\left(t\sum_{i=1}^{n}(U_i - \mathbb{E}(U_i))\right)\right) &= \prod_{i=1}^{n}\mathbb{E}\left(\exp\left(t(U_i - \mathbb{E}(U_i))\right)\right) \\ &\leq \prod_{i=1}^{n}\exp\left(\frac{t^2(b-a)^2}{8}\right) \\ &\leq \exp\left(\frac{nt^2(b-a)^2}{8}\right)\end{aligned}$$

which completes the proof. □

Proof. (Theorem 3.1). Using the law of total expectation and Lemma 6.1 we observe that

$$\begin{aligned}&\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(t\sum_{i=1}^{m}(\mathbb{E}(l_{z_i}(\mathbf{w})) - l_{z_i}(\mathbf{w}))\right)\middle| l_\Delta(\mathbf{w}) = 0\right) \\ &\quad= \sum_{k=0}^{m}\binom{m}{k}\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(t\sum_{i=1}^{m}(\mathbb{E}(l_{z_i}(\mathbf{w})) - l_{z_i}(\mathbf{w}))\right)\middle| z_{[i]} \notin \Delta, z_{[[k]]} \in \Delta, l_\Delta(\mathbf{w}) = 0\right)p_\Delta^{m-k}(1-p_\Delta)^k \\ &\quad\leq \sum_{k=0}^{m}\binom{m}{k}\exp\left(\frac{kt^2C^2}{8}\right)p_\Delta^{m-k}(1-p_\Delta)^k.\end{aligned}$$

Then applying Markov's inequality we get that

$$\begin{aligned}\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + s \middle| l_\Delta(\mathbf{w}) = 0\right) &\leq \frac{1}{\exp(mts)}\sum_{k=0}^{m}\binom{m}{k}\exp\left(\frac{kt^2C^2}{8}\right)p_\Delta^{m-k}(1-p_\Delta)^k \\ &= \frac{1}{\exp(mts)}\left(p_\Delta + (1-p_\Delta)\exp\left(\frac{t^2C^2}{8}\right)\right)^m.\end{aligned}$$

This holds for all $t \geq 0$, hence, we would like to find the t for which this bound is minimized. For $p_\Delta = 0$ this is done by letting $t = \frac{4s}{C^2}$. However, for $p_\Delta \in (0,1)$ we cannot find explicitly the minimizer of this expression and so we will simply use $t = \frac{4s}{C^2}$ as well. If $p_\Delta = 1$ we know that $R(\mathbf{w}) = 0$ and so the result holds trivially. Proceeding for the case where $p_\Delta \in [0, 1)$ we see that

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + s \middle| l_\Delta(\mathbf{w}) = 0\right) \leq \frac{1}{\exp\left(\frac{4ms^2}{C^2}\right)}\left(p_\Delta + (1-p_\Delta)\exp\left(\frac{2s^2}{C^2}\right)\right)^m.$$

Therefore, letting

$$\delta = \frac{1}{\exp\left(\frac{4ms^2}{C^2}\right)}\left(p_\Delta + (1-p_\Delta)\exp\left(\frac{2s^2}{C^2}\right)\right)^m$$

we can rearrange to get that

$$s = \sqrt{\frac{C^2\log\left(\frac{(1-p_\Delta)+\sqrt{(1-p_\Delta)^2+4\delta^{\frac{1}{m}}p_\Delta}}{2\delta^{\frac{1}{m}}}\right)}{2}}$$

which completes the proof. □

Proof. Theorem 3.3. Here we will proceed directly with the law of total probability

$$\begin{aligned}
&\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0\right)\\
&\quad= \sum_{k=0}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right)p_\Delta^{m-k}(1-p_\Delta)^k.
\end{aligned}$$

For each k we have that $l_{z_i} = 0$ for $i \in [[k+1]]_m$, so we can think of each empirical error as $\frac{k}{m}$ of the empirical error of k samples. Therefore,

$$\begin{aligned}
&\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0\right)\\
&\quad= \sum_{k=0}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \frac{k}{m}\hat{R}_k(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right)p_\Delta^{m-k}(1-p_\Delta)^k\\
&\quad= \sum_{k=1}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^k}\left(\frac{m}{k}R(\mathbf{w}) > \hat{R}_k(\mathbf{w}) + \frac{mC}{k}\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right)p_\Delta^{m-k}(1-p_\Delta)^k\\
&\qquad+ \mathbb{I}\left(R(\mathbf{w}) > C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0, z_{[m]} \in \Delta\right)p_\Delta^m\\
&\quad= \sum_{k=1}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^k}\left(\frac{m}{k}R(\mathbf{w}) > \hat{R}_k(\mathbf{w}) + \frac{mC}{k}\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right)p_\Delta^{m-k}(1-p_\Delta)^k
\end{aligned}$$

where in the last inequality we have used the fact that $R(\mathbf{w}) = 0$ to deduce that

$$\mathbb{I}\left(R(\mathbf{w}) > C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}} \middle| l_\Delta(\mathbf{w}) = 0, z_{[m]} \in \Delta\right) = 0.$$

Letting

$$\delta_k = \frac{1}{\left(\frac{1}{\delta}\right)^{\frac{m^2}{k^2}}} \leq \delta$$

for $k = 1, \dots, m$ we get that

$$
\begin{aligned}
&\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) > \hat{R}(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta}\right)}{2m}}\middle| l_\Delta(\mathbf{w}) = 0\right) \\
&\quad= \sum_{k=1}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^k}\left(R(\mathbf{w}) > \hat{R}_k(\mathbf{w}) + C\sqrt{\frac{\log\left(\frac{1}{\delta_i}\right)}{2k}}\middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right) p_\Delta^{m-k}(1-p_\Delta)^k \\
&\quad\leq \sum_{k=1}^{m}\binom{m}{k}\delta_i p_\Delta^{m-k}(1-p_\Delta)^k
\end{aligned}
$$

which completes the proof of the theorem. □

Theorem 6.2 ([11]). *Suppose $X_1, \dots, X_n$ are independent random variables with range $\{0,1\}$. Let $\mu = \sum_{i=1}^n X_i$. Then for $\delta \in (0,1)$ we have*

$$
\mathbb{P}\left(X \leq (1-\delta)\mu\right) \leq \exp\left(-\frac{\mu\delta^2}{2}\right).
$$

Proof. Theorem 3.5. We deal with the case that $C = 1$ for simplicity as we can just rescale the loss function as required. Let

$$
\epsilon(\mathbf{w}) = \sqrt{\frac{2R(\mathbf{w})\log\left(\frac{1}{\delta}\right)}{m}},
$$

then using Theorem 6.2 we get that

$$
\mathbb{P}_{S\sim\mathcal{D}^m}\left(\hat{R}(\mathbf{w}) \leq R(\mathbf{w}) - \epsilon(\mathbf{w}) - \epsilon(\mathbf{w})\right) \leq \exp\left(-\frac{m\epsilon(\mathbf{w})}{2R(\mathbf{w})}\right) = \delta.
$$

Therefore,

$$
\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \hat{R}(\mathbf{w}) + \sqrt{\frac{2R(\mathbf{w})\log\left(\frac{1}{\delta}\right)}{m}}\right) \geq 1-\delta.
$$

Using $\sqrt{ab} = \inf_{\lambda>0}\left(\frac{a}{2\lambda} + \frac{\lambda b}{2}\right)$ we get that

$$
\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\mathbf{w}) \leq \hat{R}(\mathbf{w}) + \frac{R(\mathbf{w})}{2\lambda} + \frac{\lambda\left(\log\left(\frac{1}{\delta}\right)\right)}{m}\right) \geq 1-\delta
$$

which upon re-arrangement completes the proof of the theorem. □

Proof. Theorem 3.6. We proceed by applying the law of total probability and the reasoning we explained in the proof of Theorem 3.3 to get that

$$
\begin{aligned}
&\mathbb{P}_{S\sim\mathcal{D}^m}\left(\hat{R}(\mathbf{w}) \leq R(\mathbf{w}) - \epsilon(\mathbf{w})\middle| l_\Delta(\mathbf{w}) = 0\right) \\
&\quad= \sum_{k=1}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^m}\left(\frac{k}{m}\hat{R}_k(\mathbf{w}) \leq R(\mathbf{w}) - \epsilon(\mathbf{w})\middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k]]_m} \in \Delta\right) p_\Delta^{m-k}(1-p_\Delta)^k \\
&\quad\quad+ \mathbb{I}\left(\epsilon(\mathbf{w}) \leq R(\mathbf{w}) | l_\Delta(\mathbf{w}) = 0, z_{[m]} \in \Delta\right) p_\Delta^m \\
&\quad= \sum_{k=1}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^k}\left(\hat{R}_k(\mathbf{w}) \leq R(\mathbf{w}) + \frac{m-k}{k}R(\mathbf{w}) - \frac{m}{k}\epsilon(\mathbf{w})\middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k]]_m} \in \Delta\right) p_\Delta^{m-k}(1-p_\Delta)^k \\
&\quad\leq \sum_{k=1}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^k}\left(\hat{R}_k(\mathbf{w}) \leq R(\mathbf{w}) - \frac{m}{k}\epsilon(\mathbf{w})\middle| l_\Delta(\mathbf{w}) = 0, z_{[k]} \notin \Delta, z_{[[k]]_m} \in \Delta\right) p_\Delta^{m-k}(1-p_\Delta)^k.
\end{aligned}
$$

Applying Theorem 6.2 to this we can deduce that

$$\begin{aligned}\mathbb{P}_{S\sim\mathcal{D}^m}\left(\hat{R}(\mathbf{w}) \leq R(\mathbf{w}) - \epsilon(\mathbf{w}) \middle| l_\Delta(\mathbf{w}) = 0\right) &\leq \sum_{k=1}^{m} \binom{m}{k} \exp\left(-\frac{kR(\mathbf{w})\left(\frac{m\epsilon(\mathbf{w})}{kR(\mathbf{w})}\right)^2}{2}\right) p_\Delta^{m-k}(1-p_\Delta)^k \\ &= \sum_{k=1}^{m} \binom{m}{k} \exp\left(-\frac{m^2\epsilon(\mathbf{w})^2}{2kR(\mathbf{w})}\right) p_\Delta^{m-k}(1-p_\Delta)^k.\end{aligned}$$

With this one can proceed in the same way as we do in the proof of Theorem 3.5 to complete the proof of this theorem. □

6.2 Singular Value Decomposition

For our MNIST experiment suppose that n_i points are correctly classified with label i. That is, $|\mathcal{C}_i| = n_i$. Moreover, suppose that in S_2 there are m_i points with label i. Then we define $\mathbf{X}$ to be the $\mathbb{R}^{784\times(n_i+m_i)}$ matrix, whose columns are the 784 dimensional tensors corresponding to each data point. Performing SVD on $\mathbf{X}$ fives the decomposition

$$\mathbf{X} = \mathbf{USV}.$$

The $U \in \mathbb{R}^{784\times784}$ matrix will have rows that we will refer to as the principal components, these principal components define a basis for $\mathbb{R}^{784}$. The $\mathbf{S} = (s_{ij})$ is a diagonal $784 \times (n_i + m_i)$ matrix, whose diagonal entries we will refer to as the singular values. Moreover, we will assume that $s_{11} \geq s_{22} \geq \cdots \geq s_{784,784} > 0$. The $\mathbf{V} \in \mathbb{R}^{(n_i+m_i)\times(n_i\times m_i)}$ matrix has columns where the first 784 define the representation of the corresponding column in $\mathbf{X}$ in the basis defined by the principal components.

6.3 Results for FashionMNIST

We can conduct similar experiments on the FashionMNIST dataset, as we did for the MNIST dataset.

η	$\hat{p}_\Delta$	p_L	Empirical Risk	Conditioned Bound	Unconditioned Bound	Bound on True Risk
0.1	0.5124	0.5017	0.1093	0.0129	0.0158	0.1222
0.2	0.4980	0.4905	0.1019	0.0092	0.0112	0.1111
0.3	0.5424	0.5363	0.1093	0.0074	0.0091	0.1166
0.4	0.3925	0.3873	0.1115	0.0067	0.0079	0.1182

Table 3: Results for Experiments Conducted on FashionMNIST.

As noted with the results of Table 1, the most significant region is verified when $\eta = 0.1$, the tightest bound is achieved with $\eta = 0.4$, and the lower bound on the true risk is given when $\eta = 0.2$. This highlights the intricate role that η plays in the evaluation of conditioned bounds.
Once again, these experiments highlight our capacity to define significant regions of the input space.

η	% of Unconditioned Bound	% Increase in Sample Size to Compensate
0.1	81.6%	50.0%
0.2	82.1%	47.5%
0.3	81.3%	52.0%
0.4	84.8%	39.0%

Table 4: Comparison between conditioned and unconditioned bounds.

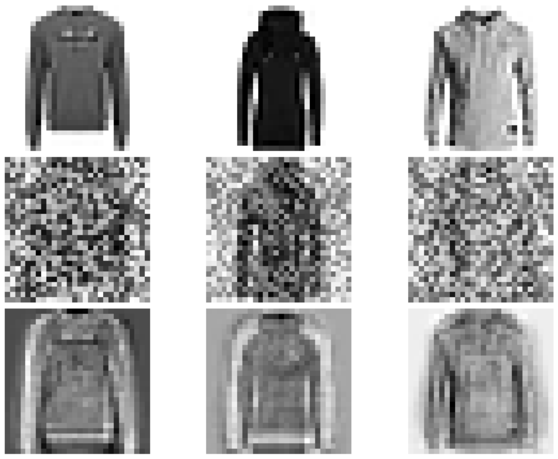

Figure 2: **Top Row:** Images from the FashionMNIST data set, labelled as a pullover. **Middle Row:** Images of the top row perturbed in Euclidean space with noise of amplitude 0.5. **Bottom Row:** Images of the top row perturbed in the space defined by the principal components of the SVD with noise of amplitude 0.5.

6.4 Results for CIFAR10

We conduct similar experiments for the CIFAR10 dataset, with 50000 data points, this time implementing a convolutional neural network rather than a feed-forward ReLU network. For training the CNN we again use stochastic gradient descent, but with a learning rate of 0.001 and a batch size of 4.

η	$\hat{p}_\Delta$	p_L	Empirical Risk	Conditioned Bound	Unconditioned Bound	Bound on True Risk
0.1	0.0014	0.0005	0.4120	0.0173	0.0173	0.4293
0.2	0.0020	0.0013	0.4135	0.0122	0.0122	0.4257
0.3	0.0034	0.0027	0.4333	0.0100	0.0100	0.4432
0.4	0.0015	0.0010	0.4484	0.0086	0.0087	0.4570

Table 5: Results for Experiments Conducted on CIFAR10.

Here are results are not as significant as in the previous case. Firstly, this could be due to the larger dimensionality of the dataset, or the reduced dataset size. Secondly, it may be that the classes in the dataset do not have strongly correlated features. Reducing the effectiveness of SVD in capturing principal directions, hindering the ability to define the region Δ. This indicates that using variational approaches, such as normalising flows, may be necessary for higher-dimensional datasets.

η	% of Unconditioned Bound	% Increase in Sample Size to Compensate
0.1	100%	0%
0.2	100%	0%
0.3	100%	0%
0.4	98.9%	1.3%

Table 6: Comparison between conditioned and unconditioned bounds.

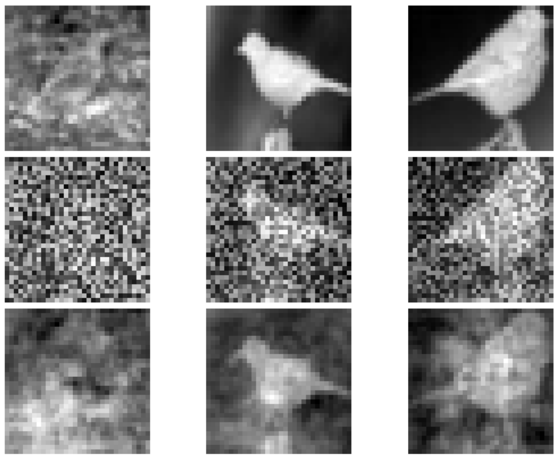

Figure 3: **Top Row:** Images from the CIFAR10 data set, labelled as a pullover. **Middle Row:** Images of the top row perturbed in Euclidean space with noise of amplitude 0.5. **Bottom Row:** Images of the top row perturbed in the space defined by the principal components of the SVD with noise of amplitude 0.5.