Incorporating Regional Verification of Neural Network Performance in to PAC bounds

Thomas Walker

Supervised by Professor Alessio Lomuscio

Summer 2023

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 [7] who managed to contextualize the bounds practically for neural networks in a non-vacuous manner. Since then there have been other successful implementations [9][10][12]. Each of these optimizes 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 sometimes 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

The problem we will be considering here is that of training a neural network to represent a distribution on an input space. The network will only have access to a training set to learn this representation, and it is our goal to develop probabilistic guarantees on the properties of this learned representation. Neural network generalization in this context refers to the ability of a neural network to learn a representation which captures the 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. As modern neural networks typically operate with a far greater number of parameters than points in the training set, classical statistics tells us that our neural network has the capacity to interpolate the training set. Clearly, this is undesirable as it transforms the learning tasks into a memorization task which does not capture the desired high-level features of the training set. Despite this, it is empirically well-known that neural networks have a remarkable capacity to learn general representations. Theoretically, this phenomenon still eludes us and has motivated the application of mathematical ideas such as information theory and algebraic topology to try and give an explanation.

Probably Approximately Correct (PAC) bounds are a theoretical tool that has been developed to provide quantitative probabilistic guarantees on the generalization 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 those developed under the Bayesian machine learning framework and it was these bounds that [7] could implement non-vacuously. Then, [8] introduced 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 generalization error, however, these particular bounds were not meaningful in practice. It did motivate the subsequent work of [9] 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, [12] extended this line of reasoning to further improve the tightness of the bounds. With this work, we intend to introduce a different strategy for evaluating these bounds. The current strategy only considers discrete samples, where here we capitalize 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 Formalization

2.1 Notation

We will only contextualize the PAC generalization bounds for neural networks. To do this, we define our input space to be $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ 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 and so the training process aims to learn a network $h : \mathcal{X} \to \mathcal{Y}$ that produces outputs in accordance with the distribution $\mathcal{D}$. 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 i.i.d samples from $\mathcal{D}$. Our neural network will be parameterized 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 set respectively. To assess the quality of a particular network we use a loss function $l : \mathcal{Y} \times \mathcal{Y} \to [0, C]$ which we will require to be bounded. The loss function quantifies the difference between a network's output and the desired output according to the distribution $\mathcal{D}$. That is 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 example. Using this we can define the risk of our network to be

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

which is dependent on the unknown distribution $\mathcal{D}$ and hence is also unknown. Instead, 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 $\mathbb{E}_{S \sim \mathcal{D}^m}\left(\hat{R}(\mathbf{w})\right)$. 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

We will formulate the problem by considering a relatively simple PAC bound.

Theorem 2.1 ([13]). *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 is the case with most machine learning problems, we can obtain zero training error on the training set, that is $\hat{R}(\mathbf{w}) = 0$ for some $\mathbf{w} \in \mathcal{W}$. Assuming this 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

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

Note that the bound of Theorem 2.1 holds for all $\mathbf{w} \in \mathcal{W}$, which is why the factor of M appears and is hence called a union bound. However, in our setting, we are usually training the network and so are only concerned with the parameter value we obtain at the end of training. However, note that these bounds are derived under the assumption that the parameter value is independent of the training set. Therefore, when evaluating the bound we need to keep this in my, and only evaluate of points which were not used in the training process to determine the parameter.

Suppose that we have trained to a network $h_{\mathbf{w}}$ for which we can guarantee a zero training error 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 really saying is that our network performs as expected in the region Δ. Implicit in this assumption is that the region Δ is independent of our training set S. Theoretically, this allows us to operate more easily with the assumption, however, we have to keep it in mind when we begin to experiment with the bound in practice.

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 fully the distribution $\mathcal{D}$.

3 Improving PAC Bounds

3.1 Improving the Tightness of 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. 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*

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

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. *The region Δ must be independent from the sample set S for this result to hold. This is something that will implicitly be present in all the results that follow.*

Note that by letting $p_{\Delta} = 0$ we just get the statement of the Theorem 2.1 without the factor of M, as we have not included a union bound argument as we are only interested in 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 $R(\mathbf{w}) = 0$. The 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 $l_\Delta(\mathbf{w}) = 0$, 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 $l_\Delta(\mathbf{w}) = 0$.

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

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

where

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

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

We will now work with a PAC bound that was 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. Furthermore, the requirement of a prior is also not restrictive as networks are often randomly initialized and so one can simply take the prior to be the distribution of the initialization on the parameter space.

Theorem 3.4 ([5])**.** *Simultaneously for all $\mathbf{w} \in \mathcal{W}$ and $\delta \in (0,1)$ the following holds,*

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

As our assumption only holds for a specific parameter value, we are only interested in bounds that hold for this parameter value. 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 $l_\Delta(\mathbf{w}) = 0$ 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*

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

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 $p_\Delta = 0$ and $p_\Delta = 1$ we get the same conclusions we made from our investigation of improving the confidence of Theorem 2.1.

4 Impact on PAC-Bayes Bounds

4.1 Bounding Expected Empirical Error

We now want to operate in the Bayesian machine learning paradigm and investigate PAC Bayes bounds. For this, we make a slightly stronger, but reasonable, assumption that there is a subset $\Omega \subset \mathcal{W}$ in which the weights correspond to networks that achieve zero loss on the region $\Delta \subset \mathcal{Z}$. This seems reasonable if one considers linear classifiers for a dataset which has a non-zero margin between the clusters of classes. In this case, there exists a set of parameters that would achieve zero training error which precisely corresponds to the subset $\Omega \subset \mathcal{W}$.

Recall the motivation of our investigation comes from the observation that neural networks are over-parameterized and have the capacity to learn functions that overfit to training data, however, in practice, they do not exhibit this behaviour. The overfitting arises as there are multiple representations of the data that minimize empirical risk, of which some are more desirable. Hence, it is justified to presume that a subset of our parameter space is capable of achieving zero training error on a particular region of the data space. We must make it clear that we are not considering parameters that achieve zero error on some region, all the parameters in Ω must achieve zero error on the same region. This added assumption will allow us to work with expected risk, as the parameters for which $l_\Delta(\mathbf{w}) = 0$ need to have a non-zero probability mass to have any influence on this value. We will work with Theorem 4.1 and see how we can improve the confidence with which it holds using the assumptions.

Theorem 4.1. *[2] For all $\rho \in \mathcal{M}(\mathcal{W})$ and $\delta \in (0,1)$ we have that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) \leq \hat{R}(\rho) + \sqrt{\frac{\mathrm{KL}(\rho,\pi) + \log\left(\frac{1}{\delta}\right) + \frac{5}{2}\log(m) + 8}{2m-1}}\right) \geq 1-\delta.$$

In the following let

$$p_\Omega = \int_\Omega \rho(\mathbf{w})d\mathbf{w}$$

and $l_\Delta(\Omega) = 0$ be the event that for all $\mathbf{w} \in \Omega$ we have $l_\Delta(\mathbf{w}) = 0$.

Theorem 4.2. *For all $\rho \in \mathcal{M}(\mathcal{W})$ and $\delta \in (0,1)$ we have that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) \leq \hat{R}(\rho) + \sqrt{\frac{\mathrm{KL}(\rho,\pi)\log\left(\frac{1}{\delta}\right) + \frac{5}{2}\log(m) + 8}{2m-1}}\middle| l_\Delta(\Omega) = 0\right)$$
$$\geq 1 - \left(\sum_{k=1}^{m}\binom{m}{k}\left(\delta_k p_\Omega + \delta(1-p_\Omega)\right)p_\Delta^{m-k}(1-p_\Delta)^k\right),$$

where δ_k is such that

$$\frac{m}{k}\sqrt{\frac{\mathrm{KL}(\rho,\pi) + \log\left(\frac{1}{\delta}\right) + \frac{5}{2}\log(m) + 8}{2m-1}} = \sqrt{\frac{\mathrm{KL}(\rho,\pi) + \log\left(\frac{1}{\delta_k}\right) + \frac{5}{2}\log(m) + 8}{2m-1}}.$$

Again we observe that is indeed an improvement in confidence as

$$\sqrt{\frac{\mathrm{KL}(\rho,\pi)+\log\left(\frac{1}{\delta}\right)+\frac{5}{2}\log(m)+8}{2m-1}}$$

is function in δ. So that

$$\sqrt{\frac{\mathrm{KL}(\rho,\pi)+\log\left(\frac{1}{\delta}\right)+\frac{5}{2}\log(m)+8}{2m-1}} \leq \sqrt{\frac{\mathrm{KL}(\rho,\pi)+\log\left(\frac{1}{\delta_k}\right)+\frac{5}{2}\log(m)+8}{2m-1}}$$

implies that $\delta_k \leq \delta$.

4.2 Data-Dependent Probability Measure

In machine learning, we form the posterior distribution by the application of a learning algorithm on our training sample. For the case of neural networks we often choose our network parameter by drawing a realisation from the posterior distribution defined by stochastic gradient descent (SGD). As SGD is a random process, it necessarily defines some probabilistic distribution on the parameter space.

Definition 4.3 ([13])**.** *Let $\mathcal{M}(\mathcal{W})$ be a set of probability distributions defined over $\mathcal{W}$. A data-dependent probability measure is a function*

$$\tilde{\rho} : \bigcup_{m=1}^{\infty} (\mathcal{X}\times\mathcal{Y})^m \to \mathcal{M}(\mathcal{W}).$$

Theorem 4.4 ([13])**.** *For all $\lambda > 0$, and data-dependent probability measure $\tilde{\rho}$, we have that*

$$\mathbb{E}_{S\sim\mathcal{D}^m}(R(\tilde{\rho})) \leq \mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\rho}) + \frac{\lambda C^2}{8m} + \frac{\mathrm{KL}(\tilde{\rho},\pi)}{\lambda}\right).$$

Suppose that we find that the network $h_{\tilde{\mathbf{w}}}$ is such that $l_\Delta(\tilde{\mathbf{w}}) = 0$. Then we could simply let our data-dependent probability measure be $\tilde{\rho}(\mathbf{w}) = \mathbb{I}(\mathbf{w})_{\{\mathbf{w}=\tilde{\mathbf{w}}\}}$. So that

$$\mathrm{KL}(\tilde{\rho},\pi) = \log\left(\frac{1}{\pi(\tilde{\mathbf{w}})}\right)$$

and

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

Corollary 4.5. *For all $\lambda > 0$, with the data-dependent probability measure $\tilde{\rho}(\mathbf{w}) = \mathbb{I}(\mathbf{w})_{\{\mathbf{w}=\tilde{\mathbf{w}}\}}$, we have that*

$$\mathbb{E}_{S\sim\mathcal{D}^m}(R(\tilde{\rho})) \leq (1-p_\Delta)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\mathbf{w}})\right) + \frac{\lambda C^2}{8m} + \frac{1}{\lambda}\log\left(\frac{1}{\pi(\tilde{\mathbf{w}})}\right).$$

On the other hand, we could assume that we have optimized to some posterior distribution $\rho \in \mathcal{W}$ which we can augment using the information that $l_\Delta(\tilde{\mathbf{w}}) = 0$ by defining the data-dependent probability measure

$$\tilde{\rho} = \gamma\mathbb{I}(\mathbf{w})_{\{\mathbf{w}=\tilde{\mathbf{w}}\}} + (1-\gamma)\rho(\mathbf{w})\mathbb{I}(\mathbf{w})_{\{\mathbf{w}\neq\tilde{\mathbf{w}}\}},$$

for $\gamma \in (0,1)$.

Corollary 4.6. *For all $\lambda > 0$, with the data dependent probability measure*

$$\tilde{\rho} = \gamma \mathbb{I}(\mathbf{w})_{\{\mathbf{w}=\tilde{\mathbf{w}}\}} + (1-\gamma)\rho(\mathbf{w})\mathbb{I}(\mathbf{w})_{\{\mathbf{w}\neq\tilde{\mathbf{w}}\}},$$

for $\gamma \in (0,1)$ we have that

$$\begin{aligned}\mathbb{E}_{S\sim\mathcal{D}^m}\left(R(\tilde{\rho})\right) \leq & (1-\gamma)\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\rho})\right) + \frac{\mathrm{KL}(\tilde{\rho},\pi)}{\lambda} + \frac{\lambda C^2}{8m}\right) \\ & + \gamma\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\mathbf{w}})\right) + \frac{1}{\lambda}\log\left(\frac{1}{\pi(\tilde{\mathbf{w}})}\right) + \frac{\lambda C^2}{8m}\right) - \gamma p_\Delta \mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\mathbf{w}})\right) + \frac{1-\gamma}{\lambda}\log(1-\gamma).\end{aligned}$$

Reassuringly, we see that for $\gamma = 0$ we recover the original bound of Theorem 4.4 and with $\gamma = 1$ we get the bounded we deduced previously. The general expression is minimized by,

$$\gamma = 1 - \exp\left(\lambda(1-p_\Delta)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\mathbf{w}})\right) - \lambda\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\rho)\right) - \mathrm{KL}(\rho,\pi) + \log\left(\frac{1}{\pi(\tilde{\mathbf{w}})}\right) - 1\right).$$

The reason we do not minimize the bound for $\gamma = 1$ is because we have no control on the behaviour of $h_{\tilde{\mathbf{w}}}$ on $\mathcal{Z}\setminus\Delta$. It may be the case that another parameter exists that achieves zero error on Δ and performs better on $\mathcal{Z}\setminus\Delta$ than $h_{\tilde{\mathbf{w}}}$. Similarly, if p_Δ is small then optimizing for performance on this region will probably not be a good proxy for optimizing performance on the $\mathcal{Z}$ which is what the bound is aiming to do. We can also work with the probabilistic version of Theorem 4.4 which takes the form of Theorem 4.7.

Theorem 4.7 ([4])**.** *For all $\lambda > 0$, for all $\rho \in \mathcal{M}(\mathcal{W})$, and $\delta \in (0,1)$ it follows that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) \leq \hat{R}(\rho) + \frac{\lambda C^2}{8m} + \frac{\mathrm{KL}(\rho,\pi) + \log\left(\frac{1}{\delta}\right)}{\lambda}\right) \geq 1-\delta.$$

For our next results, we resume the assumption that $l_\Delta(\Omega) = 0$ for some $\Omega \subset \mathcal{W}$.

Theorem 4.8. *For all $\lambda > 0$, for all $\rho \in \mathcal{M}(\mathcal{W})$ and $\delta \in (0,1)$ it follows that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) \leq \hat{R}(\rho) + \frac{\log\left(B(\lambda,m,p_\Delta,p_\Omega)\right) + \mathrm{KL}(\rho,\pi) + \log\left(\frac{1}{\delta}\right)}{\lambda}\middle| l_\Delta(\Omega) = 0\right) \geq 1-\delta,$$

where

$$B(\lambda,m,p_\Delta,p_\Omega) = p_\Omega\left(p_\Delta + (1-p_\Delta)\exp\left(\frac{\lambda^2C^2}{8m^2}\right)\right)^m + (1-p_\Omega)\exp\left(\frac{\lambda^2C^2}{8m}\right).$$

Remark 4.9. *When $p_\Delta = 0$ we recover the result of Theorem 4.7.*

Corollary 4.10. *If we let ρ be the point mass at $\tilde{\mathbf{w}} \in \Omega$ then for all $\lambda > 0$ and $\delta \in (0,1)$ we have that*

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\tilde{\mathbf{w}}) \leq \hat{R}(\tilde{\mathbf{w}}) + \frac{m\log\left(p_\Delta + (1-p_\Delta)\exp\left(\frac{\lambda^2C^2}{8m^2}\right)\right) + \log\left(\frac{1}{\pi(\tilde{\mathbf{w}})}\right) + \log\left(\frac{1}{\delta}\right)}{\lambda}\middle| l_\Delta(\tilde{\mathbf{w}}) = 0\right) \geq 1-\delta.$$

5 Experiments

5.1 Approximation 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. We have to proceed with caution here as in most

machine learning settings the Δ we determine is not independent of our training sample. For example, if we use SGD to train to the parameter $\mathbf{w}$ which we find has the property that $l_\Delta(\mathbf{w}) = 0$ then there is an implicit connection between our training set S an Δ. To mitigate this issue we generate a different sample, S_A, of size m_A to estimate p_Δ.

For $(x_i, y_i) \in S_A$ we can define the random variable

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

hence $Z_i \sim \text{Bern}(p_\Delta)$. 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 let us 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 [11]. 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.

5.2 Details

The conducted experiments involve a ReLU neural network classifier trained on the MNIST dataset. 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_Δ. To do this we used the entire dataset of 60000 points to define a discrete underlying distribution. Recall that we have some independence conditions to satisfy and so we sampled from this distribution and partitioned the sample to maintain these.

1. One segment, S_T, was used to train the network.
2. One segment, S_A was used to approximate the value of p_Δ.
3. One segment, S_E was used to obtain an empirical error which was then used to form the bound on the true error.

To partition our points into three segments we introduce two hyper-parameters, η and ζ say. Where η defined the fraction of our entire sample to hold out to approximate the value of p_Δ and ζ defined the fraction of the rest of the sample that was used to obtain the empirical error. Therefore, taking a sample of size m from the underlying distribution we had that

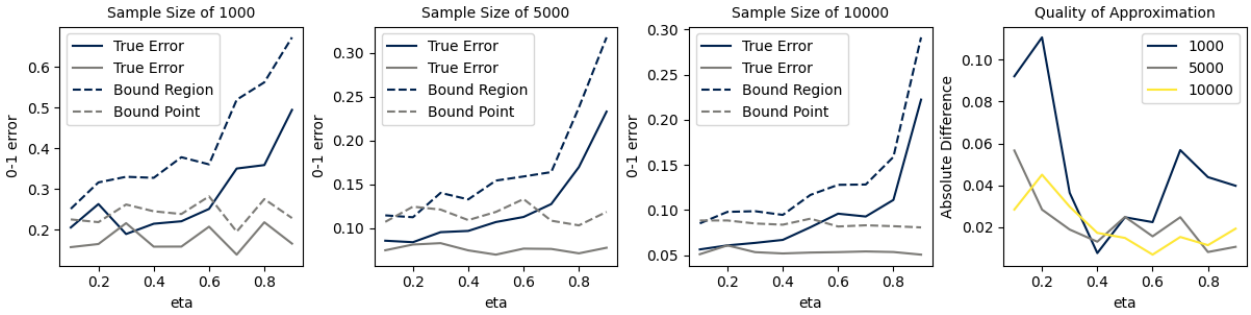
1. $m_A = \eta m$ points were used to approximate p_Δ,
2. $m_E = \zeta(1-\eta)m$ points were used to determine an empirical error, and
3. the remaining $m_T = (1-\zeta)(1-\eta)m$ points were used to train the network.
4. To compare the performance of our bound with its unconditioned version we conducted a similar partitioning, except we did not require a segment to approximate p_Δ. We split the original sample into two segments of size $m_{T'} = (1-\zeta)$ and $m_{E'} = \zeta m$ for training and evaluating the empirical error respectively.

To train the network we used stochastic gradient descent with a learning rate of 0.1 and momentum of 0.9. For training, we used the cross entropy loss function and for evaluating the bound we used the 0-1 loss, as our results require a bounded loss function and the 0-1 is more interpretable. Once our network is trained we then computed explicitly the empirical 0-1 error on the segment S_E, $\hat{R}(\mathbf{w})$, and the true 0-1 error, $R(\mathbf{w})$, using the underlying distribution of discrete points. To define the region Δ we determined the points from the underlying distribution that are correctly classified by the network, $\mathcal{C}$, and then sampled from these points. We could only do this as we gave ourselves access to the underlying distribution which of course in reality is not the case. Consequently, we could control the true value of p_Δ in our experiment. Suppose we wanted to investigate our bound when $p_\Delta = \gamma$, then we could define Δ to be a $\min\left(1, \frac{\gamma}{R(\mathbf{w})}\right)$ portion of $\mathcal{C}$. Obviously, if $R(\mathbf{w}) < \gamma$ then we cannot define a Δ such $p_\Delta = \gamma$. This sampling process from $\mathcal{C}$ is where the dependence of Δ on S arose in our experiments, and is why we required the partitioning of the dataset. Using S_A we approximated the value of p_Δ with a lower bound, and hence could calculate an upper value for our bound.

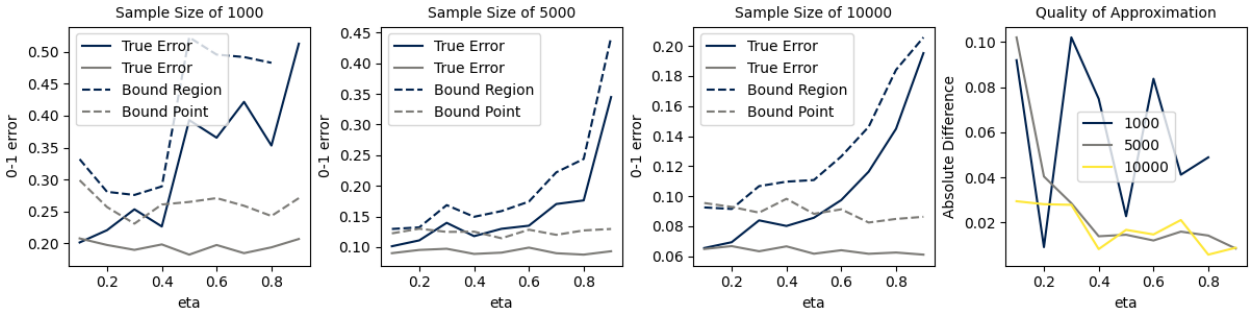
5.3 Bounds on MNIST

With this setup, we investigated the bound derived in Theorem 3.1. We implemented the experiment with $\gamma = 0.5$ and we kept $\delta = \alpha = 0.025$ throughout to give a confidence of ≈ 0.95. We used $\gamma = 0.5$ as we were able to demonstrate that this is attainable in practice. We used different values for the sample size, η and ζ.

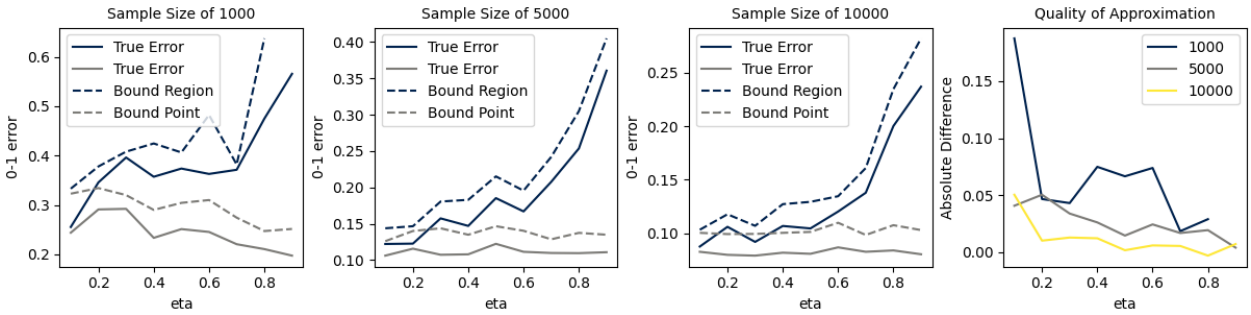
1. $m \in \{1000, 5000, 10000\}$,
2. $\eta \in \{0.1, 0.2, \ldots, 0.9\}$, and
3. $\zeta \in \{0.3, 0.5, 0.7\}$.



(a) $\zeta = 0.3$.



(b) $\zeta = 0.5$.



(c) $\zeta = 0.7$.

Figure 1: The results of our experiment for different values of ζ, the title of each subplot indicates the overall sample size used for the particular experiment.

From Figure 1 we can see that in all cases we get non-vacuous bounds on the true error. When ζ is larger we get bounds that are tighter to the true error, however, when ζ is smaller we get bounds that are smaller in value. This is as expected as with a larger ζ we are using fewer samples to train and so the quality of our neural network classifier is going to be hindered. However, we are using more samples to evaluate our bound which is inversely proportional to the size of the sample. It seems that the improvements gained by decreasing ζ are more significant than any of the negative consequences, and so one would be inclined to choose a lower value for ζ.

Similarly, one is motivated to choose a smaller value of η. As we are dealing with relatively large samples the confidence interval is sufficiently tight for even small values of η. The changes in our bound over the interval $p_\Delta \in [0, 1]$ are insignificant compared to the improved performance gained by increasing the size of our training sample, and so there is little justification for holding out a larger segment to get an improved approximation on p_Δ.

We support these conclusions with a focused investigation of the bound. We conducted the experiment five times for nine values of η in the interval $\eta \in [0.01, 0.1]$, an overall sample size of 10000, $\zeta = 0.3$ and $\gamma = 0.5$.

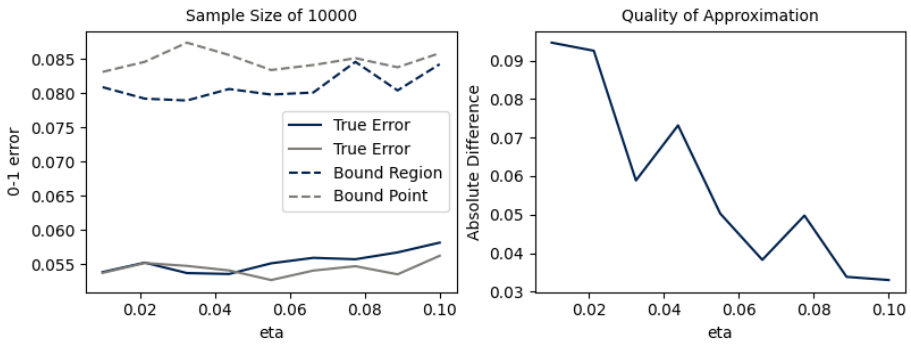


Figure 2: With this more focused investigation, and averaged results, we can more clearly see that conditioned bounds provide an improved tightness.

5.4 Realistic Values of p_Δ

The experiments performed above could control for the value of p_Δ, and it was set to take values around 0.5. This choice was motivated by the following experiment. In neural network verification, the performance of a network is analysed in small regions around a set of test points. The aim of these analyses is to guarantee that despite small perturbations made to the inputs of a neural network, the output will not deviate largely from the expected output. For example, for the MNIST dataset, one would expect that adding some noise to an image shouldn't change the classification given by the network, as the general structure of the written digit remains relatively clear. An initial attempt to approximate p_Δ from this framework proceeded as follows.

1. A network was trained on a training set.
2. Noise of increasing amplitude was added to a correctly classified training image to the point where the trained network misclassified the point.
   (a) 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.
3. The maximum noise amplitude permissible for each training image to maintain correct classification was noted.
   (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.
4. For images in an independent sample, pairwise distances to the training images were calculated. If the distance was below the maximum permissible noise amplitude for the image, the image from the independent sample was deemed to lie in the region Δ. Otherwise, it was deemed to lie in the region $\mathcal{Z} \setminus \Delta$. Using this we could evaluate our estimator $\hat{p}_\Delta$ and determine a confidence interval for p_Δ.

It was discovered that this approach defines an insignificant region Δ under the distribution $\mathcal{D}$. The reason for this is that the added noise was uniform in all dimensions, and the MNIST dataset has 784. One can instead imagine balls on an alternative basis, rather than the standard basis. Choosing the alternative basis judiciously will allow for the determination of more impactful regions. This motivated our next attempt to define a region Δ, which we detail in the case of binary classification for simplicity.

- We took a bi-categorical sample.
- We partitioned it into a 80-20 train-approximate split.
- We trained the network and identified the correctly classified points from the train set.
  1. Suppose m_C points are correctly classified as 0, and assume that $m_C \geq 784$. Form the matrix $X \in \mathbb{R}^{784 \times m_c}$, where each column is the flattened image tensor for the correctly classified point.

2. Compute the singular value decomposition (SVD) $X = USV$, where $U \in \mathbb{R}^{784\times 784}, V \in \mathbb{R}^{m_C \times m_C}$ and
$$S = \begin{pmatrix} s_1 & 0 & \dots & \dots & \dots & 0 \\ 0 & s_2 & \dots & \dots & \dots & 0 \\ \vdots & \vdots & \ddots & & & \vdots \\ 0 & \dots & \dots & s_{784} & \dots & 0 \end{pmatrix}.$$
   - One can see that the rows of U define principal directions in the space of correctly classified 1 labelled points. The columns of V express each of the image vectors as a linear combination of these principal directions weighed by the corresponding singular value.
3. For each point we now analyse the ϵ-balls as defined by the principal directions by perturbing the column of V corresponding to the point in question.
4. We carried out the same operation but now formed X using correctly classified points labelled as 1.
   - The ϵ-balls defined in the respective spaces form our defined region Δ.
5. From here we proceed in the same way as before, by noting that we can transform an image from our approximating set, $\tilde{x}$, from Euclidean space to one of the alternative spaces, $\tilde{v}$, by
$$\tilde{v} = \begin{pmatrix} \frac{1}{s_1} & \dots & \dots & 0 \\ 0 & \frac{1}{s_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{1}{s_{784}} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 0 \end{pmatrix} U^{-1}\tilde{x}.$$
   We can then compute the distance between image vectors in this new space to determine whether they are within the region defined by the ϵ-balls in the new space.

When we carried out this procedure we did not consider singular values below 0.01, that is we discarded the rows of S, and the corresponding columns of U, for which the $s_i < 0.01$. Note that by convention the s_i are such that $s_1 \geq s_2 \geq \cdots \geq s_{784} > 0$. This was done to improve stability in the inverting step.

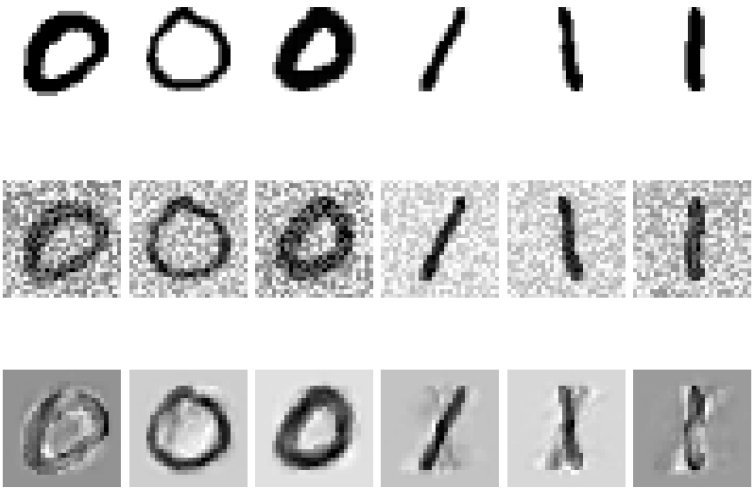


Figure 3: **First Row:** Images from the test set. **Second Row:** Images perturbed in standard space.**Third Row:** Images perturbed in the space defined by the principal components.

In Figure 3 we see that if we make perturbations in the space defined by the principal components, then we get more coherent images than if we make perturbations of the same amplitude in the standard space. Therefore, working in the space defined by the principal components allows us to define more significant regions of the input space.

In our execution of this procedure, we took the MNIST dataset and extracted the digits labelled 0 and 1. Our sample had 10000 points containing equal amounts of points from the two categories. The training set contained 3995 points labelled 0 and 4005 points labelled 1, and all points were correctly classified by the trained network. For the tensor matrix with points labelled 0 we used the first 511 singular values, whereas, for the tensor matrix with points labelled 1 we used the first 465 singular values.

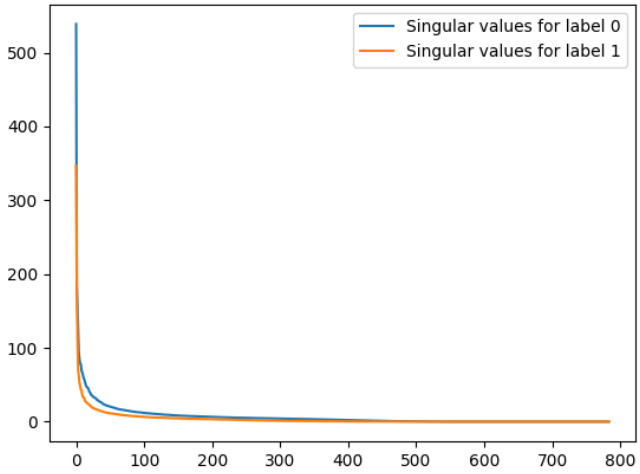


Figure 4: The singular values of the SVD of the tensor matrix corresponding to correctly classified images. The values quickly diminish which reassures us that we can safely disregard the tail end of the singular values without significant compromises in precision.

The distributions of the size of the determined ϵ-balls for each category of points can be found in Figure 5. Note that we capped the size of the ϵ-balls at 0.25 to maintain viable images.

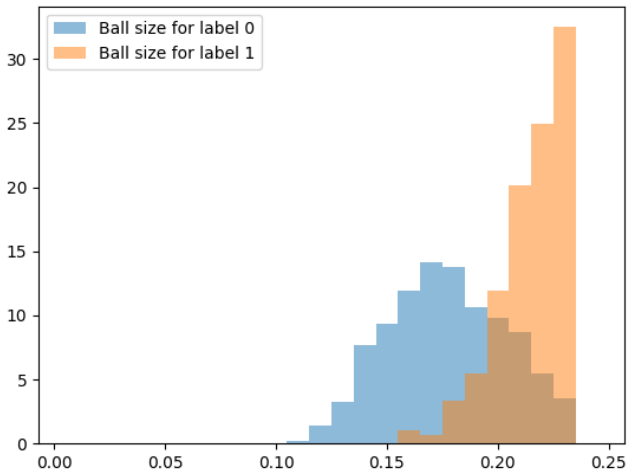
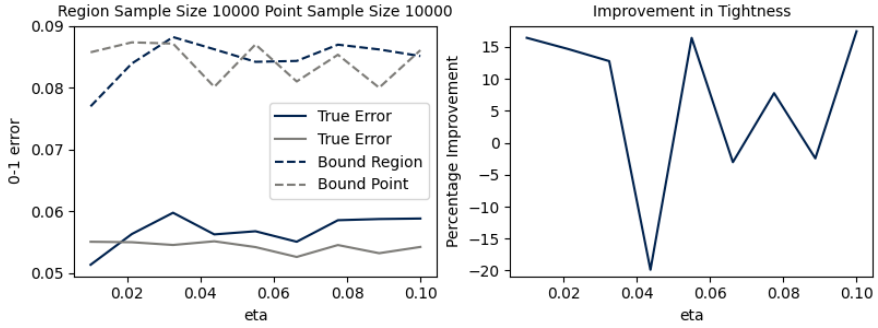


Figure 5: Caption

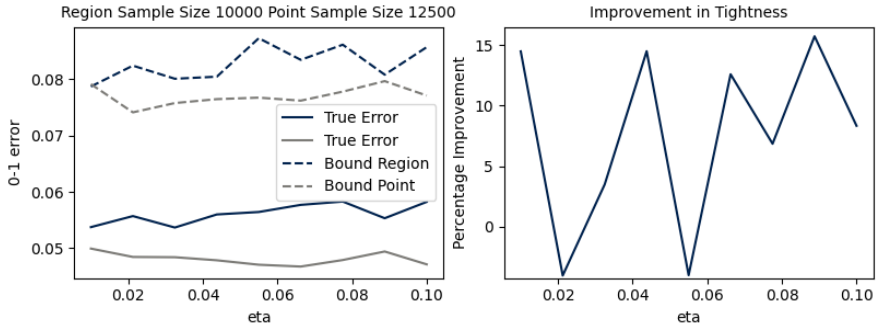
Now that the region Δ is defined by the ϵ-balls we approximate its probability mass using the approximating set. In this approximating set, we had 1005 points labelled 0 and 995 points labelled 1. This gave an estimated value of p_Δ as 0.6285, with a lower bound of 0.607. One can see that we can repeat this procedure with all of the categories in the MNIST dataset, and define a region Δ with a similar probability mass. This motivated our conservative use of $\gamma = 0.5$ throughout the experiments conducted in the work.

5.5 Comparison to Point Bounds

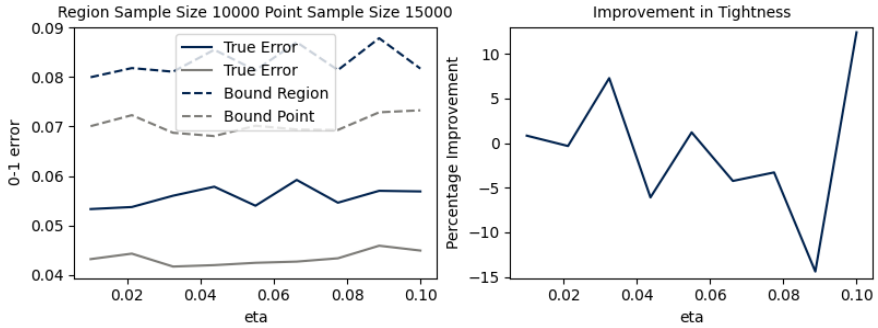
From Figure 2 we can see that our region bound is tighter than the point bound. Now we would like to investigate how much better the conditioned bound is. We will quantify the performance of a bound by its tightness to the true error. We are going to try and answer this question by performing our experiments with a fixed sample size of 10000 to evaluate the region bound and use an increased sample size to evaluate the point bound. We will then compare the tightness of the bound in both cases by calculating the improved tightness of the region bound as a percentage of the tightness of the point bound. By iteratively increasing the sample size used to evaluate the point bound we can equate the improvement provided by the region bound to the increased sample size required to see the same improvement using the point bound.



(a) 10000 samples to evaluate the point bound.



(b) 12500 samples to evaluate the point bound.



(c) 15000 samples to evaluate the point bound.

Figure 6: Experiment to quantify the improvement incurred by the conditioned bound in terms of increasing the sample size used to evaluate unconditioned bounds.

Figure 6 shows that the improved tightness of our conditioned bound equates roughly to a 50% increase in the sample size used to evaluate the unconditioned bound. Recall, that we are only measuring performance in terms of the tightness of the bound to the true error. So although our region bounds are as tight as point bounds evaluated with twice as many points, the increased sample size to evaluate the point bound results in a trained network that performs better.

Despite the absolute performance of the network being degraded by the compromises introduced by implementing the region-conditioned bound, it seems that the bound we do achieve is tighter to the true error of the network. We repeat our original experiment with $\zeta = 0.3$ and $\eta = 0.1$ but with varying values of γ, such that we can understand the performance of the bound for varying values of the probability mass of Δ.

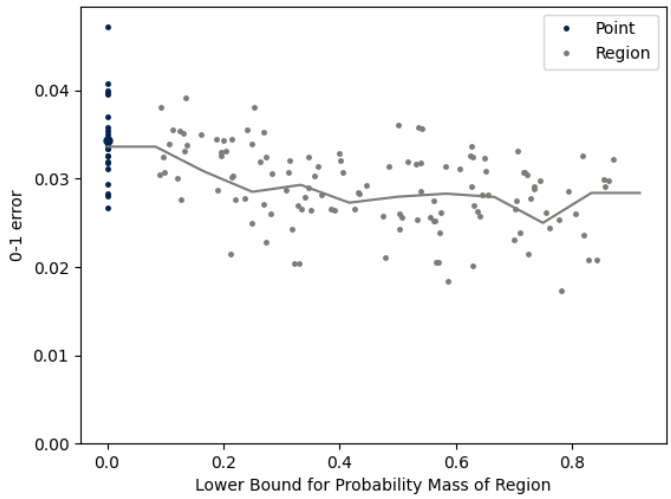


Figure 7: Shows the tightness of the region bounds for different values of the approximated lower bound, p_L, of p_Δ. We also compute the point bound for the same networks and plot their values on the $p_L = 0$ line, with the enlarged dot representing their mean.

In Figure 7 we see that the tightness of the region bound is marginally tighter across the probability masses. This shows that our conditioned bound is not optimal as ideally, we would want our bound to approach 0 as the probability mass of our region increases.

5.6 Performance of the PAC-Bayes Bound

Recall, the conditioned PAC-Bayes bound we derived in Theorem 4.8 from Theorem 4.7. To understand the improvement gained by our updated bound we only need to compare the $\frac{\lambda C^2}{8m}$ term of Theorem 4.7 and $\frac{\log(B(\lambda,m,p_\Delta,p_\Omega))}{\lambda}$ term of Theorem 4.8. Note that λ is a hyper-parameter that is optimized non-trivially and so, for the purposes of our discussion, we can ignore it (i.e. let $\lambda = 1$). Hence, we simply need to compare the performance of the quantities $\frac{C^2}{8m}$ and $\log(B(\lambda, m, p_\Delta, p_\Omega))$. Let us assume that we are in the context of Corollary 4.10 so that

$$\log(B(\lambda, m, p_\Delta, p_\Omega)) = m \log\left(p_\Delta + (1 - p_\Delta) \exp\left(\frac{\lambda^2 C^2}{8m^2}\right)\right).$$

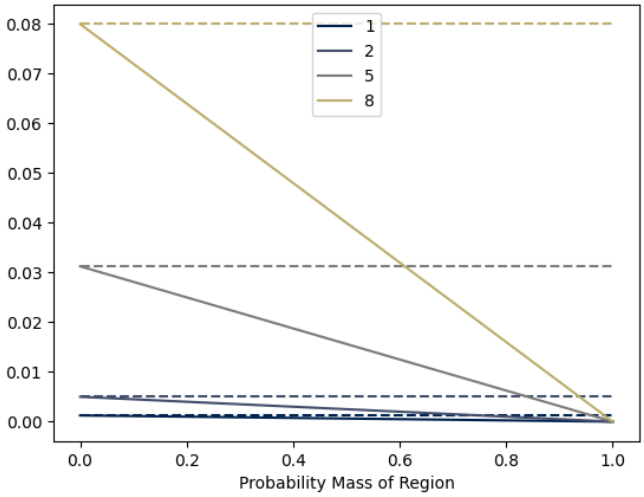


Figure 8: A plot showing the performance of the identified quantities from Theorem 4.7 (dashed) and Theorem 4.8 (solid) for $C \in \{1, 2, 5, 8\}$.

What we see from Figure 8 is a roughly linear improvement in the performance of these quantities as p_Δ increases. However, note that the absolute value of this improvement is small, especially for the smaller values of C. Therefore, if we want to bound the 0-1 error of a network, we will likely see little improvement by using our conditioned bound. The reason for this is that the KL divergence term plays a more significant role in this bound, and the PAC-Bayes framework as a whole. A lot of the work to optimize these bounds for the neural network setting has focused on minimizing this term of the bounds.

6 Conclusion

We have seen how the knowledge that a network operates as intended on a region of the data space can tighten PAC bounds. Through experimentation, we saw that the conditioned bounds' improvements were insignificant. The conditioned bounds were only slightly tighter as any improvements they provided were masked by the added computation required to implement them. However, we only tested one bound in our experiments and we noted that this bound was not optimal. It may be the case that updating other bounds would see a more significant improvement. Furthermore, it may be possible to mitigate some of the added computation we incurred by using our particular method of implementing the conditioned bounds. In the PAC-Bayes setting, we draw similar conclusions. In this framework, the more prominent term of the bound is the KL divergence which is not affected by our method of conditioning on our assumptions. It would be interesting to explore how our assumptions can optimize the prior, similar to much of the work we mentioned in optimizing PAC-Bayes bounds.

References

[1] David A. McAllester. "Some PAC-Bayesian Theorems". In: *Machine Learning* 37 (1998), pp. 355–363.

[2] David A. McAllester. "PAC-Bayesian model averaging". In: *Annual Conference Computational Learning Theory*. 1999.

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

[4] Olivier Catoni. "A PAC-Bayesian approach to adaptive classification". In: (Jan. 2009).

[5] David A. McAllester. “A PAC-Bayesian Tutorial with A Dropout Bound”. In: *CoRR* (2013).

[6] Clayton Scott. *Hoeffding’s Inequality*. 2014.

[7] Gintare Karolina Dziugaite and Daniel M. Roy. “Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data”. In: *CoRR* (2017).

[8] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. “Stronger generalization bounds for deep nets via a compression approach”. In: *CoRR* (2018).

[9] 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.

[10] Gintare Karolina Dziugaite, Kyle Hsu, Waseem Gharbieh, and Daniel M. Roy. “On the role of data in PAC-Bayes bounds”. In: *CoRR* (2020).

[11] Nathaniel E. Helwig. *Inference for Proportions*. 2020.

[12] 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.

[13] Pierre Alquier. *User-friendly introduction to PAC-Bayes bounds*. 2023.

7 Appendix

7.1 Detailing the Proofs

Lemma 7.1 ([6]). *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*

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

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

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

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

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

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,

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

which completes the proof. □

Proof. (Theorem 3.1). Using the law of total expectation and Lemma 7.1 we observe that

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

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

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

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,

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

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 $k = 1, \ldots, m$ we get that

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

which completes the proof of the theorem. □

Theorem 7.2 ([3])**.** *Suppose $X_1, \ldots, X_n$ 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 7.2 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 7.2 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. □

Proof. Theorem 4.2. For ease of notation let

$$B(\rho, \pi, \delta, m) = \sqrt{\frac{\mathrm{KL}(\rho, \pi) + \log\left(\frac{1}{\delta}\right) + \frac{5}{2}\log(m) + 8}{2m-1}}.$$

We first observe that for a sample S if we have $z_{[k]} \notin \Delta$ and $z_{[[k+1]]_m} \in \Delta$, for $k \neq 0$, then

$$\begin{aligned}\hat{R}(\rho|\mathbf{w} \in \Omega) &= \mathbb{E}_{\mathbf{w}\sim\rho}\left(\hat{R}(\mathbf{w})\middle|\mathbf{w} \in \Omega\right) \\ &= \frac{k}{m}\mathbb{E}_{\mathbf{w}\sim\rho}\left(\hat{R}(\mathbf{w})\right) \\ &= \frac{k}{m}\hat{R}(\rho).\end{aligned}$$

So for $k \neq 0$,

$$\begin{aligned}&\mathbb{P}_{S\sim\mathcal{D}^m}\Big(R(\rho) > \hat{R}(\rho) + B(\rho,\pi,\delta,m)\big|l_\Delta(\Omega) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\Big) \\ &\quad= \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) > \hat{R}(\rho|\mathbf{w} \in \Omega) + B(\rho,\pi,\delta,m)\big|l_\Delta(\Omega) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right) p_\Omega \\ &\qquad+ \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) > \hat{R}(\rho|\mathbf{w} \notin \Omega) + B(\rho,\pi,\delta,m)\big|l_\Delta(\Omega) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right)(1-p_\Omega) \\ &\quad= \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) > \frac{k}{m}\hat{R}(\rho) + B(\rho,\pi,\delta,m)\big|l_\Delta(\Omega) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right) p_\Omega \\ &\qquad+ \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) > \hat{R}(\rho) + B(\rho,\pi,\delta,m)\big|l_\Delta(\Omega) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right)(1-p_\Omega) \\ &\quad\leq \mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) > \hat{R}(\rho) + \frac{m}{k}B(\rho,\pi,\delta,m)\right) p_\Omega + \delta(1-p_\Omega) \\ &\quad= \delta_k p_\Omega + \delta(1-p_\Omega)\end{aligned}$$

If $k = 0$ then the inequality clearly doesn't hold so that

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) > \hat{R}(\rho) + B(\rho,\pi,\delta,m)|l_\Delta(\Omega)) = 0, z_{[m]} \in \Delta\right) = 0$$

Therefore,

$$\begin{aligned}&\mathbb{P}_{S\sim\mathcal{D}^m}\Big(R(\rho) > \hat{R}(\rho) + B(\rho,\pi,\delta,m)\Big|l_\Delta(\Omega) = 0\Big) \\ &\quad= \sum_{k=0}^{m}\binom{m}{k}\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho) \leq \hat{R}(\rho) + B(\rho,\pi,\delta,m)\Big|l_\Delta(\Omega) = 0, z_{[k]} \notin \Delta, z_{[[k+1]]_m} \in \Delta\right) p_\Delta^{m-k}(1-p_\Delta)^k \\ &\quad\leq \sum_{k=1}^{m}\binom{m}{k}\left(\delta_k p_\Omega + \delta(1-p_\Omega)\right) p_\Delta^{m-k}(1-p_\Delta)^k.\end{aligned}$$

Taking the complement completes the proof of the theorem. □

Proof. Corollary 4.6. We first observe that

$$\begin{aligned}\mathrm{KL}\left(\tilde{\rho},\pi\right) &= \int_{\mathcal{W}}\tilde{\rho}(\mathbf{w})\log\left(\frac{\tilde{\rho}(\mathbf{w})}{\pi(\mathbf{w})}\right)d\mathbf{w} \\ &= \int_{\mathcal{W}\setminus\{\tilde{\mathbf{w}}\}}(1-\gamma)\rho(\mathbf{w})\log\left(\frac{(1-\gamma)\rho(\mathbf{w})}{\pi(\mathbf{w})}\right)d\mathbf{w} + \gamma\log\left(\frac{\gamma}{\pi(\tilde{\mathbf{w}})}\right) \\ &= (1-\gamma)(\mathrm{KL}(\rho,\pi) + \log(1-\gamma)) + \gamma\log\left(\frac{\gamma}{\pi(\tilde{\mathbf{w}})}\right).\end{aligned}$$

Next we see that

$$\begin{aligned}\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\rho})\middle|l_\Delta(\tilde{\mathbf{w}}) = 0\right) &= \gamma\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\rho})\middle|l_\Delta(\tilde{\mathbf{w}}) = 0, \mathbf{w} = \tilde{\mathbf{w}}\right) + (1-\gamma)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\rho})\middle|l_\Delta(\tilde{\mathbf{w}}) = 0, \mathbf{w} \neq \tilde{\mathbf{w}}\right) \\ &= \gamma(1-p_\Delta)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\mathbf{w}})\right) + (1-\gamma)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\rho)\right).\end{aligned}$$

Now using Theorem 4.4 we get that

$$\begin{aligned}\mathbb{E}_{S\sim\mathcal{D}^m}\left(R(\tilde{\rho})\right) \leq & \gamma(1-p_\Delta)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\tilde{\mathbf{w}})\right) + (1-\gamma)\mathbb{E}_{S\sim\mathcal{D}^m}\left(\hat{R}(\rho)\right) \\ & + \frac{1-\gamma}{\lambda}\left(\mathrm{KL}(\rho,\pi) + \log(1-\gamma)\right) + \frac{\gamma}{\lambda}\log\left(\frac{\gamma}{\pi(\tilde{\mathbf{w}})}\right) + \frac{\lambda C^2}{8m},\end{aligned}$$

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

Lemma 7.3. *For any measurable, bounded function $f : \mathcal{W} \to \mathbb{R}$ we have,*

$$\log\left(\mathbb{E}_{\mathbf{w}\sim\pi}\left(e^{f(\mathbf{w})}\right)\right) = \sup_{\rho\in\mathcal{M}(\mathcal{W})}\left(\mathbb{E}_{\mathbf{w}\sim\rho}\left(f(\mathbf{w})\right) - \mathrm{KL}(\rho,\pi)\right).$$

Proof. Theorem 4.8. Using the law of total expectation we have that

$$\begin{aligned}&\mathbb{E}_{\mathbf{w}\sim\pi}\Big(\mathbb{E}_{S\sim\mathcal{D}^m}\Big(\exp\Big(tm\Big(R(\mathbf{w}) - \hat{R}(\mathbf{w})\Big)\Big)\Big)\Big|l_\Delta(\Omega) = 0\Big) \\ &\quad = \mathbb{E}_{\mathbf{w}\sim\pi}\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(tm\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\Big|l_\Delta(\tilde{\mathbf{w}}) = 0, \mathbf{w}\in\Omega\right)\right)p_\Omega \\ &\qquad + \mathbb{E}_{\mathbf{w}\sim\pi}\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(tm\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\Big|l_\Delta(\Omega) = 0, \mathbf{w}\in\Omega\right)\right)(1-p_\Omega).\end{aligned}$$

Using Lemma 7.1 we have that

$$\begin{aligned}&\mathbb{E}_{\mathbf{w}\sim\pi}\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(tm\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\Big|l_\Delta(\tilde{\mathbf{w}}) = 0, \mathbf{w}\in\Omega\right)\right) \\ &\qquad \leq \sum_{k=0}^{m}\binom{m}{k}\exp\left(\frac{kt^2C^2}{8}\right)p_\Delta^{m-k}(1-p_\Delta)^k \\ &\qquad = \left(p_\Delta + (1-p_\Delta)\exp\left(\frac{t^2C^2}{8}\right)\right)^m\end{aligned}$$

and

$$\mathbb{E}_{\mathbf{w}\sim\pi}\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(tm\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\Big|l_\Delta(\Omega) = 0, \mathbf{w}\in\Omega\right)\right) \leq \exp\left(\frac{mt^2C^2}{8}\right).$$

We get the top inequality as we can think of the sum as only involving k terms rather than m as only k are potentially non-zero. Letting $\lambda = mt$ we get that

$$\begin{aligned}&\mathbb{E}_{\mathbf{w}\sim\pi}\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(tm\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\Big|l_\Delta(\tilde{\mathbf{w}}) = 0, \mathbf{w}\in\Omega\right)\right) \\ &\qquad \leq \left(p_\Delta + (1-p_\Delta)\exp\left(\frac{\lambda^2C^2}{m^2}\right)\right)^m\end{aligned}$$

and

$$\mathbb{E}_{\mathbf{w}\sim\pi}\left(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(tm\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\Big|l_\Delta(\Omega) = 0, \mathbf{w}\in\Omega\right)\right) \leq \exp\left(\frac{\lambda^2C^2}{8m}\right).$$

Therefore,

$$\begin{aligned}&\mathbb{E}_{\mathbf{w}\sim\pi}\Big(\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(\lambda\left(R(\mathbf{w}) - \hat{R}(\mathbf{w})\right)\right)\right)\Big|l_\Delta(\Omega) = 0\Big) \\ &\qquad \leq p_\Omega\left(p_\Delta + (1-p_\Delta)\exp\left(\frac{\lambda^2C^2}{m^2}\right)\right)^m \\ &\qquad\quad + (1-p_\Omega)\exp\left(\frac{\lambda^2C^2}{8m}\right).\end{aligned}$$

Applying Fubini's theorem we can exchange the order of taking expectations to deduce that

$$\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(\lambda\left(R(\pi)-\hat{R}(\pi)\right)\right)\middle|l_\Delta(\Omega)=0\right)\leq p_\Omega\left(p_\Delta+(1-p_\Delta)\exp\left(\frac{\lambda^2C^2}{m^2}\right)\right)^m + (1-p_\Omega)\exp\left(\frac{\lambda^2C^2}{8m}\right)=:B(\lambda,m,p_\Delta,p_\Omega).$$

We can now apply Lemma 7.3 to deduce that

$$\mathbb{E}_{S\sim\mathcal{D}^m}\left(\exp\left(\sup_{\rho\in\mathcal{M}(\mathcal{W})}\left(\lambda\left(R(\rho)-\hat{R}(\rho)\right)\right)-\mathrm{KL}(\rho,\pi)-\log\left(B(\lambda,m,p_\Delta,p_\Omega)\right)\right)\middle|l_\Delta(\Omega)=0\right)\leq 1$$

to which we can apply Markov's inequality to get that

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(\sup_{\rho\in\mathcal{M}(\mathcal{W})}\left(\lambda\left(R(\rho)-\hat{R}(\rho)\right)\right)-\mathrm{KL}(\rho,\pi)-\log\left(B(\lambda,m,p_\Delta,p_\Omega)\right)>s\middle|l_\Delta(\Omega)=0\right)\leq e^{-s}$$

for fixed $s>0$. Letting $s=\log\left(\frac{1}{\delta}\right)$ and taking the complement we get that

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R(\rho)\leq\hat{R}(\rho)+\frac{\log\left(B(\lambda,m,p_\Delta,p_\Omega)\right)+\mathrm{KL}(\rho,\pi)+\log\left(\frac{1}{\delta}\right)}{\lambda}\middle|l_\Delta(\Omega)=0\right)\geq 1-\delta,$$

which completes the proof of the theorem. □

7.2 Uniform Bounds

The power of our assumptions is that they tell us the shape of the underlying distribution on the region $\Delta\subset\mathcal{Z}$. Now suppose that we have a reasonably accurate approximate of p_Δ, so that in the following we can say 'calculate' instead of 'approximate'. With this we can calculate

$$p_\Delta=\int_\Delta\mathcal{D}(z)dz$$

and if we let $\Delta'=\mathcal{Z}\setminus\Delta$ we can define the distributions

$$\mathcal{D}_\Delta(z)=\begin{cases}\frac{\mathcal{D}(z)}{p_\Delta} & z\in\Delta\\ 0 & \text{otherwise,}\end{cases}\quad\mathcal{D}_{\Delta'}(z)=\begin{cases}\frac{\mathcal{D}(z)}{1-p_\Delta} & z\in\Delta'\\ 0 & \text{otherwise.}\end{cases}$$

Consequently, we can also define

$$R_\Delta(\mathbf{w})=\mathbb{E}_{z\sim\mathcal{D}_\Delta}(l_z(\mathbf{w})),\text{ and }R_{\Delta'}(\mathbf{w})=\mathbb{E}_{z\sim\mathcal{D}_{\Delta'}}(l_z(\mathbf{w})).$$

Despite $l_\Delta(\mathbf{w})=0$ only holding for a potentially small region of the parameter space, we can explicitly calculate $R_\Delta(\mathbf{w})$ for any parameter value as we know the distribution $\mathcal{D}_\Delta$. Throughout we have had to derive results that only applied to a specific parameter value as we were utilizing the property that the empirical error on Δ is zero. Here we will instead see how we could potentially derive results that hold all parameter values. To do this we will not be able to use the fact that the training error is zero to tighten our results. Note that we can decompose the true error as

$$R(\mathbf{w})=p_\Delta R_\Delta(\mathbf{w})+(1-p_\Delta)R_{\Delta'}(\mathbf{w}).\tag{1}$$

For any $\mathbf{w}\in\mathcal{W}$ we can calculate explicitly each term on the right-hand side except for $R_{\Delta'}(\mathbf{w})$. We can incorporate this decomposition into PAC bounds to deduce results of the form,

$$\mathbb{P}_{S\sim\mathcal{D}^m}\left(R_{\Delta'}(\mathbf{w})\leq\frac{\hat{R}(\mathbf{w})+B(\delta,m)-p_\Delta R_\Delta(\mathbf{w})}{1-p_\Delta}\right)\geq 1-\delta,$$

that hold for all $\mathbf{w}\in\mathcal{W}$ and $\delta\in(0,1)$. It may seem as though verifying performance for regions increases our bound and therefore undesirable. However, as p_Δ the probability density for $z\in\Delta'$ under $\mathcal{D}_{\Delta'}(z)$ increases and so $R_{\Delta'}(\mathbf{w})$ increases accordingly.

7.2.1 Experiments

We can investigate the uniform approach in a similar way as we did in the main report.

1. Obtain a sample of size m from our data space according to a discrete underlying distribution defined by 30000 (we use this reduced sample size to reduce computation time).
2. Partition the data set according to some parameter ξ.
   (a) Use ξm data points to determine the region Δ.
      - As before $\eta\xi m$ points will be used to approximate p_Δ, and
      - $(1-\eta)\xi m$ points will be used to train a network to determine the region Δ.
   (b) Use $(1-\xi)m$ points to evaluate our bound.
      - Where $(1-\zeta)(1-\xi)m$ will be used to train the model, and
      - $\zeta(1-\xi)m$ will be used to evaluate the empirical errors for the bound.

From our investigations in the report, we conduct this experiment with $m = 10000$, $\zeta = 0.3$, $\gamma = 0.8$, $\alpha = \delta = 0.025$,

- η taking five equally spaced values in the interval $[0.01, 0.1]$, and
- $\xi \in \{0.3, 0.5, 0.7\}$.

For each set of parameter values, we perform the experiment three times to get an average. The bound we use is that of Theorem 3.1, and from the experiments, we observe the value of

$$\hat{R}(\mathbf{w}) + B(\delta, \zeta(1-\xi)m) - p_L R_\Delta(\mathbf{w}),$$

as from Equation 1 it is clear that this our absolute uncertainty in the value of $R(\mathbf{w})$ that holds with confidence of $1 - \delta - \alpha(1-\delta)$.

1. $\hat{R}(\mathbf{w})$ is the empirical error of the model trained by the partition of size $(1-\xi)(1-\xi)m$ on the partition of size $\zeta(1-\xi)m$.
2. p_Δ is referring to the lower bound on p_Δ calculated using the partition of size $\eta\xi m$ using the region identified by the training a model on the partition of size $(1-\eta)\xi m$ and verifying its performance.
3. We can calculate $R_\Delta(\mathbf{w})$ as we know the shape of Δ.
4. $B(\delta, \zeta(1-\xi)m)$ is the bound of Theorem 3.1.

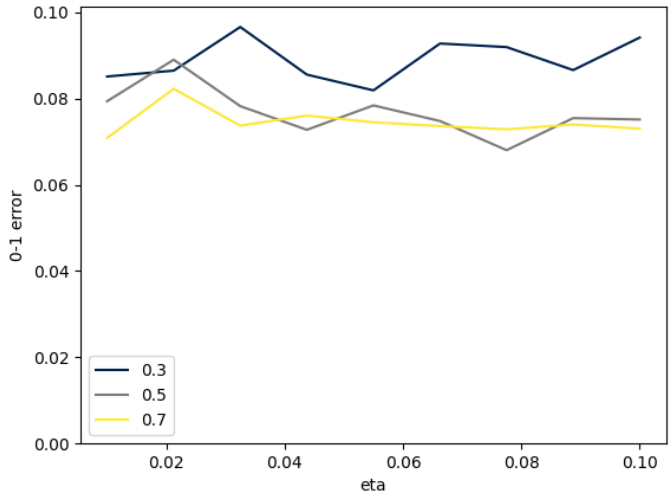


Figure 9: The bound on our absolute uncertainty of the true error. Each line corresponds to the value of ζ used in the experiment.

From Figure 9 we see that for certain values of ξ we get comparable bounds to those calculated via the method in the main report.

7.3 CIFAR10 Experiments

We can repeat the experiments we conducted on MNIST with the CIFAR10 dataset. We define the underlying distribution using only 50000 due to the smaller sample size of CIFAR10 in PyTorch. We perform a focused experiment with $\gamma = 0.8$, $\delta = \alpha = 0.025$, $m = 10000$ $\zeta = 0.3$ and η taking nine values in the range $[0.01, 0.1]$.

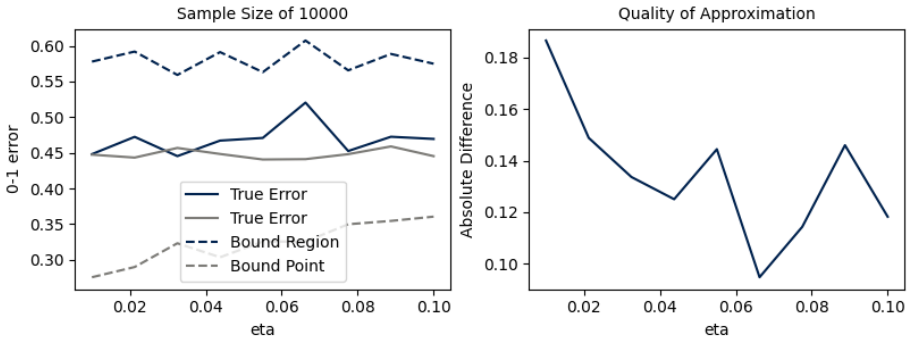


Figure 10: The result of a focused experiment using the CIFAR10 dataset.

For this experiment, we trained a convolutional neural network, which is a more data-dependent task than the MNIST experiment. As a consequence, we observe a few things in the results of Figure 10.

- The 0-1 error is larger.
- The errors when evaluating points bounds are smaller as we have more samples to train our network, and so the representations are more general.

Despite the higher 0-1 error, the region bounds are generally tighter than the point bounds.