Understanding Uncertainty in Input Spaces of Deep Classifiers

Sarthak Gupta

AI Shield, B.G.S.W.

Indian Institute of Technology Roorkee

sarthak_g@ma.iitr.ac.in

Abstract

An important metric to analyse the data points on in any machine learning classifier is their position with respect to the classification boundary and the overall data distribution. However, getting these positions is difficult as the input spaces are often high dimensional, and the model classification boundaries can be hard to locate. To address this problem, we propose a metric to estimate the distance of a given point in the input space from the model classification boundary. Alongside this, we test some hypotheses related to model extraction using the uncertainty metric and gain insights into the MNIST and CIFAR-100 datasets. Further, we propose another metric to determine the 'uncertainty' of a given sample with respect to the classification boundary of a particular model. These metrics could have wide-ranging applications in various research areas in machine learning. The methods proposed are very general and could be applied to any deep classifier. We also conducted extensive experiments to check some of the commonly held beliefs regarding vectors in input spaces.

1 Introduction

For several areas of interest in machine learning like model extraction, curriculum learning, continual learning, et cetera, it is often advantageous to get an approximate position of the data point under consideration with respect to the classification boundaries of our model. For instance, this information could be helpful for determining how effective an attack vector is in the case of model extraction [1] [2]. Another use case could be in the memory-based continual learning setup where we might want to sample a diverse set of instances into our limited amount of memory [3]. While there are techniques that apply to neural network classifiers [4][5][6], our method aims to estimate the boundary in a more precise way which is also to applications like data-augmentations without any simplifications.

To this end, we propose a method to estimate this distance in the input spaces of classification models that also applies to deep models. We aim to estimate the distance by first applying varying degrees of augmentation to the given sample until it gets classified into another class. Then we run a binary search to get a vector very close to the classification boundary, and the distance between this final vector and the original vector is approximately the same as the distance between the original vector from the boundary. By increasing the number of steps of binary search that we take, we can a very good approximation of this distance.

Since we only augment the data and observe the outputs of the classification model, we do not require any modification of the model architecture, making it easily applicable and further making our method model-agnostic. We use jacobian-based data-augmentation techniques to move in the direction of the shortest distance from the classification boundary. For our experiments, we have used

Preprint. Under review.

a fully-connected model and a basic Convolutional Neural Network for the MNIST dataset and two fully-connected networks for extraction. In contrast, for experiments on the CIFAR-100 dataset, we used a pre-trained ResNet-20, while for extraction, we used MobileNetV2 and VGG-16 models. We have conducted several experiments using these models, which are described further in the results section.

Further, we also conducted experiments on model extraction which answer questions like whether vectors sampled from near the classification boundary would be more advantageous for extracting a victim model or vice-versa?. Through empirical observations on MNIST and CIFAR-100, we find the samples near the classification boundary alone are not sufficient to learn the boundary properly. In fact, the samples near class distribution centres lead to better extraction results. This might be because flip points do not represent the original distribution well and hence perform worse when tested on data sampled from the original distribution.

2 Related Work

Currently, most methods for estimating the distance of datapoints from classification boundaries use assumptions specific to the use case in question. These are often not transferable to other uses. For instance, the Rainbow-Memory method proposed in [3] uses data augmentations and their corresponding labels as a proxy for the distance to the classification boundaries. Nevertheless, the augmentations used might not apply to other datasets. Some others have made assumptions about the decision boundaries, which would be valid only in particular cases. Like in [7] the authors assumed the decision boundary to be locally linear. However, as revealed in [8], the decision boundaries could be highly non-linear even in local regions.

Other methods to calculate the distance to the class boundaries exist, which use numerical optimisation to find the flip points. In [8] the authors use the fact that the probabilities of two classes for a flip point must be equal and then use numerical optimisation to find it. Another approach for calculating the flip points is the homotopy algorithm proposed in [9]. However, [9] requires us to put lower bounds on the gradients of the model and change some of the model parameters. This could be problematic in cases like model extraction where we do not have complete access to the model.

3 Computing Closest Distance to Boundary

In order to compute the closest distance from the boundary, we use the original datapoint and successively add augmentations. We keep increasing the magnitude of augmentation until the datapoint gets classified to a class different from the original sample's class. However, only this will not ensure that we get the closest distance from the classification boundary. To ensure that we move in the direction of the closest distance, we use jacobian-based augmentation methods, whereby we obtain the gradient of the logit of the top-1 class with respect to the input. The negative direction of this gradient will be in the direction of the fastest decrease in the value of the top-1 class logit w.r.t. the input. Now, we add vectors with varying magnitude pointing in the negative direction of this gradient in the input space to the original input as described above. Now, let us define some notations to proceed with the idea in a clear idea. Let x_o be our original datapoint, the classification model F(.) and y = F(x) be the top-1 class logit for the input x. Now we augment the x_o as:

$$x_a = x_o - \nabla F(x_o) * \alpha \tag{1}$$

Where x_a is the augmented vector. We keep increasing $\alpha until F(\mathrm{x}_a) \neq F(\mathrm{x}_o)$. Let the x_a^n be the first such augmented vector, then $F(x_a^k) = F(x_o) \forall k < n$. And let the ordering be such that if $k_1 < k_2$ then $\alpha_1 < \alpha_2$. So, x_a^{n-1} would be the closest vector to the classification boundary with F(x) = F(x_o) and since $F(x_a^n) \neq$F(x_o), we can claim that the classification boundary would lie somewhere between x_a^{n-1} and x_a^n. But this still gives us only a very rough estimate of the decision boundary, so to get a vector much closer to the decision boundary we can conduct a binary search between x_a^{n-1} and x_a^n. The procedure for this search is described in 3. Next, we calculate the distance between the original point and the final flip point obtained from the binary search. This is the approximate measure of the

distance between the original point and the classification boundary.

Algorithm 1 Binary search for flip point

$x_{left} = x_a^{n-1}, x_{right} = x_a^n$
while $i < num\_steps$ **do**
    $x_{middle} = \frac{x_{left} + x_{right}}{2}$
    if $F(x_{middle}) = F(x_{left})$ **then**:
        $x_{left} \leftarrow \mathbf{x}_{middle}$
    else if $F(x_{middle}) = F(x_{right})$ **then**:
        $x_{right} \leftarrow \mathbf{x}_{middle}$
    end if
end while
$x_{flip} = x_{middle}$;
Return x_{flip}

(2)

Algorithm 2 Finding distance to boundary.

$x_a^0 = x_o$
$\alpha = 1$
$i = 0$
while $F(x_o) = F(x_a^i)$ **do**:
    $i = i + 1$
    $x_a^i = x_o - \nabla F(x_o) * \alpha$
    $\alpha = \alpha * 10$
end while
Conduct binary search between x_a^i and x_a^{i-1} using 3 and get x_{flip}.
Return $L_1(x_o, x_{flip})$

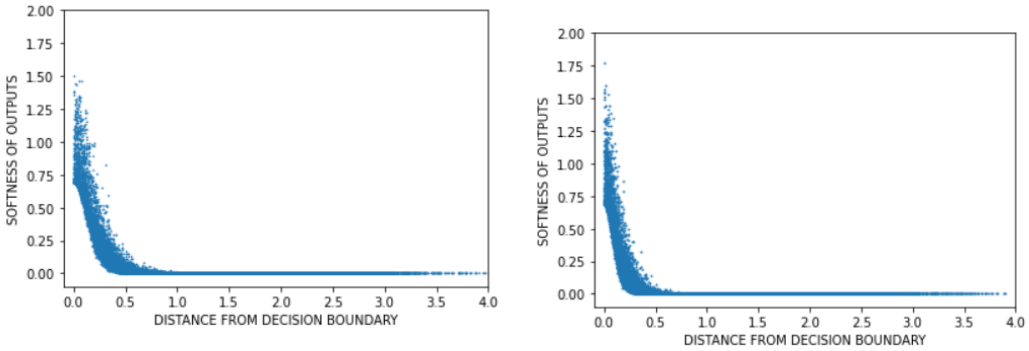
(3)

We use L_1 distance measure to find the distance between x_o. Since we are subtracting only multiples of the gradient of the top-1 class logit, the direction of movement in the input space is fixed; this results in all L_n distance measures giving the same distance. We can find a better approximation of the distance to the boundary by increasing the number of times the binary search. However, increasing the number of binary search steps would also increase the computational overhead. Hence, one must find the appropriate number of steps for the task.

One possible downside of this algorithm could be the necessity to calculate the gradient of the outputs with respect to the input. However, if we train our model simultaneously while using its gradient for augmentations, we can use backpropagation to find these gradients efficiently. Such a situation might arise in the case of a continual learning setup, where we would want to select a diverse set of samples based on their distance measure while also training our model to learn the continuous stream of data.

The natural extension to this method might seem to conduct the complete gradient descent algorithm on the top-1 class logit w.r.t. the input values. However, this would be too computationally expensive since it would require us to calculate the gradients at every step. This gradient descent for augmentation might have optimization issues of its own. Furthermore, this would not be able to use the gradients in the way described above for the case of simultaneous augmentation and training. Hence we opt not to go further in the direction of full-fledged gradient descent on the top-1 class logit.

For applications of this metric in areas like model extraction, where we might not have access to the model gradients, we can use gradient estimation methods like zeroth-order gradient estimation. This distance measure could be effective in cases where we only have the top-1 class logit.



(a) The output softness v/s distance plot for MNIST using basic convolutional neural network.

(b) Caption

Figure 1: The output softness v/s distance plot for MNIST using a fully-connected network.

4 Distance Measure and Uncertainty in Outputs

We conducted experiments to establish a relationship between the distance of a datapoint from the classification boundary and the classifier's confidence level while predicting it. For this, we plot the entropy of the softmax outputs of the classifier against the distance of the datapoints. For MNIST, we use a fully connected model and a basic Convolutional Neural Network. For CIFAR-100 we use a pre-trained ResNet-20.

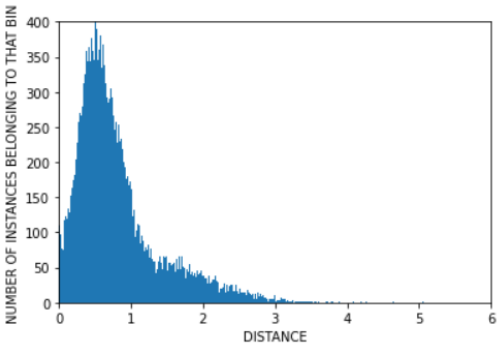
From 1, we can observe a clear correlation between the distance from the classification and the softness of the classifier outputs. That is, as we get closer to the boundary, the classifier becomes less confident about the class of the datapoint. However, after a certain distance, there is only a slight variation in the output softness, and there is a significant difference in confidence in a small distance near the boundary. Thereby we can claim that while the distance from the boundary and the uncertainty/softness of the outputs is inversely proportional in the regions near the classification boundary, there is not much difference in the 'core' regions of classes which are far away from the classification boundaries.

We can analyse this information to identify the regions in the input space where the model cannot give a confident prediction. Then, we can give a higher weightage to samples belonging to that region or try to generate more samples from that region using generative models. The above-described methods could be instrumental in the case of curriculum learning.

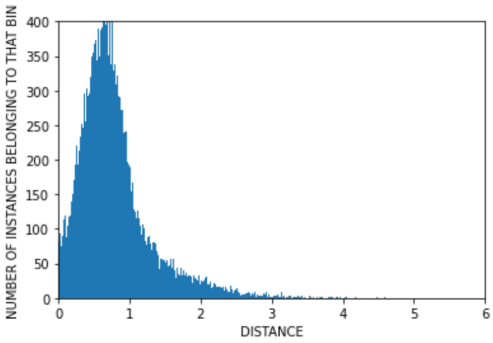
5 Distance Properties of MNIST and CIFAR-100

We use trained models to find the distance of datasets from the boundaries of the models. We then plot histograms for observations and try to make inferences from them. For MNIST, we used the entire dataset along with random uniform and normal sampled vectors from the input space to observe the distance characteristics of these different distributions of datapoints. For CIFAR-100 we only observe the distributions for the natural data and random uniform sampled data in the input space of CIFAR-100. The results from these experiments have been summarized in **??** for MNIST and in **??** for CIFAR-100.

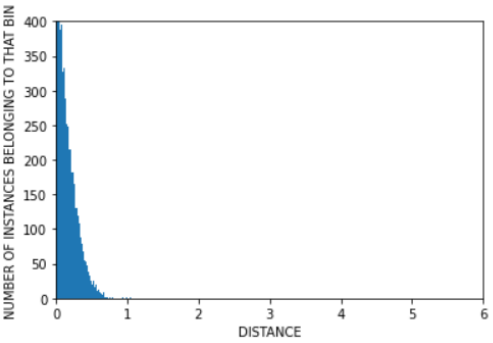
From the plots, we can infer that for MNIST, the mode of the natural samples occurs far away from the classification boundary. The mode occurs at the decision boundary for the uniformly distributed samples. For any closed figure, there is more space closer to the surface than otherwise, and uniform sampling selects samples equally from all parts of the space. Hence, the total amount of samples closer to the boundary would naturally be much greater than far away from the boundary. Similarly, for the random normal sampled dataset with a mean of 0.13 and standard deviation of 0.308, we observe the mode to occur close to the classification boundary.



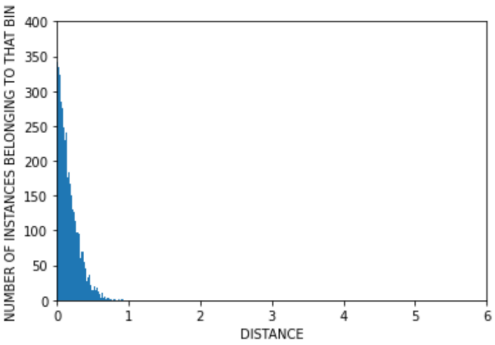
(a) Distribution of natural MNIST samples according to CNN classifier boundaries.



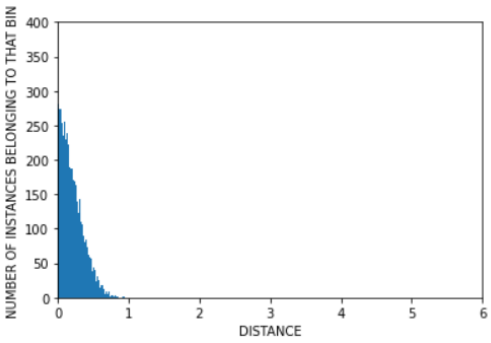
(b) Distribution of natural MNIST samples according fully connected classifier model boundaries.



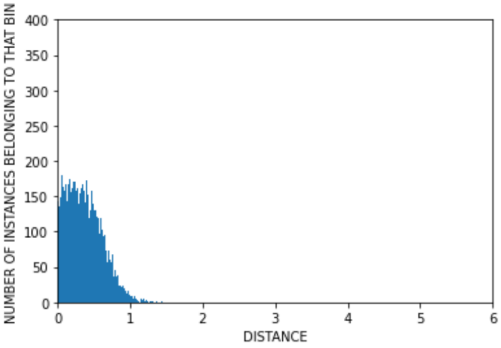
(c) Distribution of randomly selected samples from a normal distribution of pixel values with mean 0.13 and standard deviation 0.308 according to the CNN classifier boundaries.



(d) Distribution of randomly selected samples from a normal distribution of pixel values with mean 0.13 and standard deviation 0.308 according fully connected classifier model boundaries.

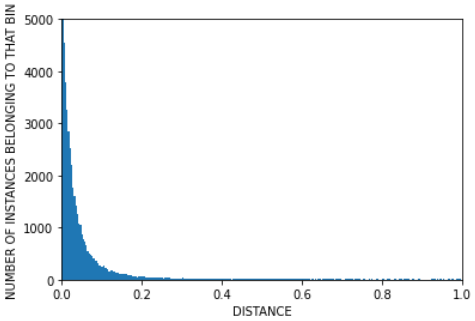


(e) Distribution of randomly selected samples from a uniform distribution of pixel values between 0 and 1 according CNN classifier boundaries.

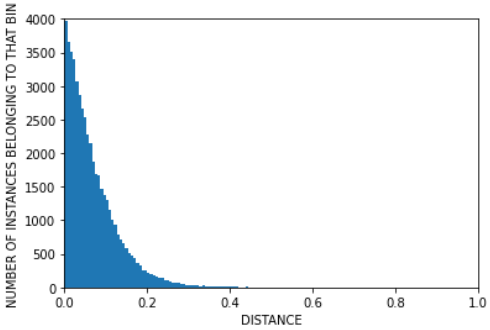


(f) Distribution of randomly selected samples from a uniform distribution of pixel values between 0 and 1 according fully connected classifier model boundaries.

Figure 2: Results from experiments on the input space of MNIST.



(a) Distribution of natural CIFAR-100 samples according to pre-trained ResNet-20 boundaries.



(b) Distribution of randomly selected samples from a uniform distribution of pixel values between 0 and 1 according pre-trained ResNet-20 boundaries.

Figure 3: Results from experiments on the input space of CIFAR-100.

Victim Model	Extracted Model	Natural Samples	Uniform Distance Sampled	High Distance Samples
Model 1	Model 3	96.73%	93.94%	86.28%
Model 1	Model 4	93.22%	91.49%	78.29%
Model 2	Model 3	96.90%	93.76%	94.97%
Model 2	Model 4	93.27%	91.56%	87.07%

Table 1: Model extraction accuracy numbers using Natural Samples, Uniform Distance Sampled, High Distance Samples for MNIST.

Victim Model	Extracted Model	Uniform Random Sampled	Flip points	Low Distance Samples
Model 1	Model 3	10.10%	10.32%	76.31%
Model 1	Model 4	10.10%	15.76%	70.55%
Model 2	Model 3	22.81%	22.41%	79.08%
Model 2	Model 4	20.71%	23.16%	75.73%

Table 2: Model extraction accuracy numbers using Uniform Random Sampled, Flip points, Low Distance Samples for MNIST.

6 Use of Distance Measure in Model Extraction Attacks

For MNIST we used two victim models were used with the following architecture:

1. Model 1, Fully connected classifier:
   Flattening Layer -> Dense Layer 1 -> Dense Layer 2 -> Softmax
2. Model 2, Convolutional neural net classifier:
   2D Convolution Layer -> Flattening Layer -> Dense Layer -> Softmax

For the extracted models we use the following architectures:

1. Model 3, Fully connected classifier:
   Flattening Layer -> Dense Layer 1 -> Dense Layer 2 -> Softmax
2. Model 4, Fully connected classifier:
   Flattening Layer -> Dense Layer -> Softmax

During the study, we conducted experiments in which we made model extraction attacks while using data with differing classification distances. We used the original dataset for extraction to serve as a benchmark and uniformly sample 10% of the dataset based on the distance to the boundary. We also sample 10% dataset with the least distance to the boundaries and 10% with the highest distance to the classification boundaries.The results are summarized in 1 and 2.

From the above data, we can infer that the datapoints closer to the class distribution centres are better suited for extracting the model than those near the classification boundaries. One possible reason for this could be a class imbalance in the sampled data due to the geometry of some class boundaries causing the samples to be closer to the boundaries or because of the fact that it is inherently more advantageous to train on more representative samples of classes rather than the

7 Discussion

References

[1] X. Gong, Y. Chen, W. Yang, G. Mei, and Q. Wang, "Inversenet: Augmenting model extraction attacks with training data inversion," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21* (Z.-H. Zhou, ed.), pp. 2439–2447, International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track.

[2] A. Fawzi, S.-M. Moosavi-Dezfooli, and P. Frossard, "The robustness of deep networks: A geometrical perspective," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 50–62, 2017.

[3] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, and J. Choi, "Rainbow memory: Continual learning with a memory of diverse samples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8218–8227, June 2021.

[4] R. Yousefzadeh and D. P. O'Leary, "Investigating decision boundaries of trained neural networks," 2019.

[5] G. Elsayed, D. Krishnan, H. Mobahi, K. Regan, and S. Bengio, "Large margin deep networks for classification," in *Advances in Neural Information Processing Systems* (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, Curran Associates, Inc., 2018.

[6] S. Jetley, N. A. Lord, and P. H. S. Torr, "With friends like these, who needs adversaries?," *CoRR*, vol. abs/1807.04200, 2018.

[7] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, (New York, NY, USA), p. 1135–1144, Association for Computing Machinery, 2016.

[8] R. Yousefzadeh and D. P. O'Leary, "Investigating decision boundaries of trained neural networks," *CoRR*, vol. abs/1908.02802, 2019.

[9] R. Yousefzadeh and D. P. O'Leary, "Deep learning interpretation: Flip points and homotopy methods," in *Proceedings of The First Mathematical and Scientific Machine Learning Conference* (J. Lu and R. Ward, eds.), vol. 107 of *Proceedings of Machine Learning Research*, pp. 1–26, PMLR, 20–24 Jul 2020.