Deep Learning Through the Lens of Example Difficulty

Anonymous Author(s)
Affiliation
Address
email

Abstract

Abstract: Existing work on understanding deep learning often employs measures that compress all data-dependent information into a few numbers. In this work, we adopt a perspective based on the role of individual examples. We introduce a measure of the computational difficulty of making a prediction for a given input: the *(effective) prediction depth*. Our extensive investigation reveals surprising yet simple relationships between the prediction depth of a given input and the model's uncertainty, confidence, accuracy and speed of learning for that data point. We further categorize difficult examples into three interpretable subgroups and show how these subgroups are processed differently inside deep models. These insights lead to a coherent view of a number of separately reported phenomena in the literature: early layers generalize while later layers memorize; early layers converge faster; and networks learn easy data and simple patterns first.

1 Introduction

Much of the existing work on understanding deep learning "integrates out" the data, viewing the inductive bias of the model, or the properties of the optimizer as central to the success of the approach. Examples of such work include studies of eigenvalues and geometry of the loss landscape [1, 2, 3, 4, 5, 6], studies of margin and effective generalization measures [7, 8, 9, 10, 11] and mean-field studies of stochastic optimization [12, 13, 14]. However, in practice, we are rarely concerned with only the average behvaiour of a model.

The principles governing the passage of data through the hidden layers of deep models are not well understood. A clear formalisation of these principles would, for example, aid the development of models that signal their uncertainty, our understanding of how deep networks respond to distributional shift, the advancement of curriculum learning and machine learning fairness, and enable the construction of more efficient networks which return answers early if they are sufficiently confident. We return to these topics in Sec. 5.

In this paper we relate the hidden embeddings for an input to high level concepts about *example difficulty*: "Does this data point look mislabelled?"; "Would classifying this example still be ambiguous if we knew the ground truth label?" as well as to lower level notions of example difficulty: the accuracy, uncertainty and confidence of a prediction, and the speed of learning for the same data point.

Contributions Our main contributions are as follows:

- We introduce a measure of *computational example difficulty*: the *prediction depth* (PD). The prediction depth, illustrated in Fig. 1, represents the number of hidden layers after which the network's final prediction is already (effectively) determined (Sec. 2).
- We demonstrate that the prediction depth is larger for examples that visually appear to be more difficult, and that prediction depth is consistent between architectures and random seeds (Sec. 2.2).
- We show that predictions are on average more accurate for validation points with small ("early") prediction depths and that the expected prediction depth establishes a simple, linear lower bound on the consistency of a prediction (Sec. 3.1).

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

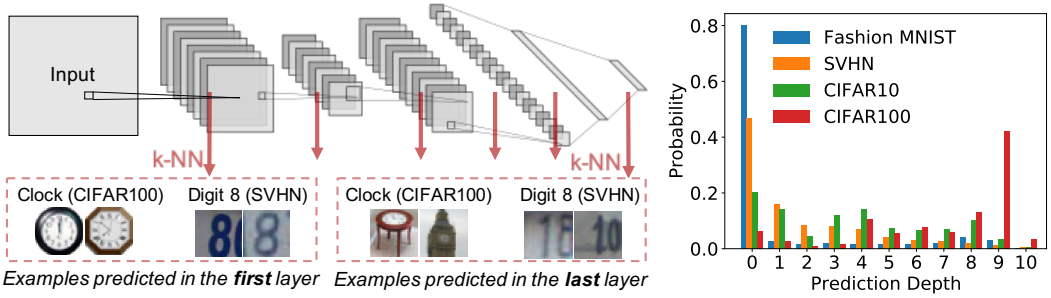

Figure 1: **Deep models use fewer layers to (effectively) determine the prediction for easy examples and more layers for hard examples. Left plot:** A cartoon illustrating the definition of prediction depth (defined in Sec. 2.1). Also shown are training examples from CIFAR100 ("Clock") and SVHN ("Digit 8"). These examples have prediction depths 0 (input) and 10 (SoftMax layer) for ResNet18. Examples with prediction depth zero are visually typical ("easy"), while those with prediction depth 10 are mislabelled and/or visually confusing ("hard" examples). To find the prediction depth, we build k-NN classifiers from the embeddings of the training set in different layers of the model. The prediction depth corresponds to the earliest layer at which the predictions of the k-NN classifiers converge to a fixed value. **Right plot:** Probability of prediction depth in ResNet18 models for four data sets (training split). Ranking the mean prediction depths of these data sets in ascending order, we observe: Fashion MNIST (smallest), SVHN (second), CIFAR10 (third), and CIFAR100 (largest mean prediction depth). This order aligns intuitively with how one might rank the difficulties of these classification tasks.

- We show that final predictions for data points that converge earlier during training are typically determined in earlier layers, and demonstrate a correspondence between the training history of the network and the processing of data in the hidden layers (Sec. 3.2).
- We show that both the adversarial input margin and the output margin are larger for examples with smaller prediction depths. We perform an intervention to reduce the final confidence of a network and show that this leads to predictions being made only in the latest hidden layers (Sec. 3.3).
- We identify three extreme forms of example difficulty by considering the prediction depth in the training and validation splits independently (Sec. 4).
- We use our results to express a coherent viewpoint of deep learning that closely resembles four known deep learning phenomena reported from separate viewpoints: early layers generalize while later layers memorize; networks converge from input layer towards output layer; and networks learn easy data and simple patterns first. (Sec. 5).

Experimental Setup: We trained ResNet18, VGG16 and MLP architectures on CIFAR10 / CIFAR100 [15], Fashion MNIST [16] and SVHN [17] data sets. All models were trained using SGD with momentum. Our MLP comprised 7 linear layers of width 2048 with ReLu activations and a linear head, terminated with a softmax. Details of the data sets, architectures, and hyperparameters used can be found in Appendix A.

Related Work: Hidden layer probes have been used to study deep learning in several papers. [18] compared k-Nearest Neighbors (k-NN) probes to logistic regression and SVM probes and reported that the three gave comparable results. The authors studied average k-NN accuracies in embedding spaces of deep models, and did not study the processing of individual data points or relate the k-NN accuracy to notions of example difficulty. [19] used linear classifier probes in the hidden layers. [20] introduced the Deep k-NN algorithm, which is a deep learning algorithm designed for interpretability and reliability.

Methods like Universal Transformers [21] that exit early during inference can be seen as implementing an estimator for the prediction depth that we study. As such our work can be seen as establishing solid context for such methods, which can be found in both computer vision [22, 23] and in NLU [24, 25, 26]

A number of alternative definitions of example difficulty have previously been proposed [27, 28, 29, 30]. Two are of particular pertinence to this work. Firstly, the probability of predicting the ground truth label for an example when it was not used in training [31] (Sec. 3.1), which represents the *"statistical view"* of example difficulty. Secondly, the difficulty of learning an example, parameterised by the frequency of forgetting an example during training [32] (Sec. 3.2). This measure represents the *"learning view"* of example difficulty. We expand on other notions of example difficulty, and Deep k-NN algorithms in Appendix B.

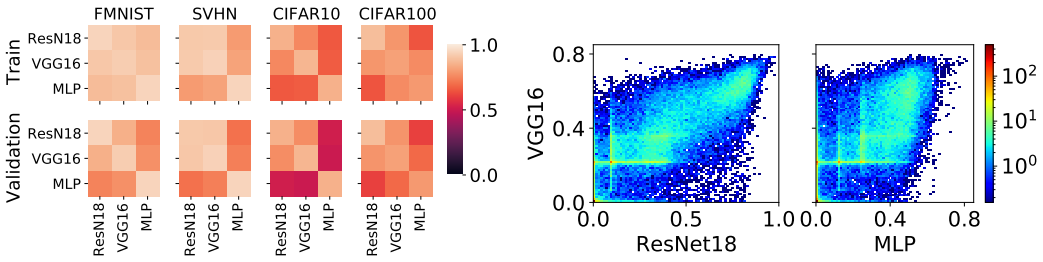

Figure 2: **Consistency of prediction depth between architectures and random seeds.** The **left** panel shows the correlation coefficient between prediction depths in different architectures, for both train and validation splits in four data sets. (Minimum value shown is 0.50.) Diagonal comparisons between an architecture and itself show the correlation for the same architecture trained with different random seeds. The histograms in the **right** panel visually demonstrate these correlations for CIFAR10. This right plot compares mean value of prediction depth obtained for each data point in the training set from an ensemble of 250 trained models. In this plot, for visual simplicity, we rescale prediction depth to the interval $[0, 1]$ for every network. Similar results for all other data sets are presented in Appendix C.2.

2 Prediction Depth: a Computational View of Example Difficulty

In this paper we connect the "statistical" and "learning" views of example difficulty (introduced above) to a third notion: the *"computational"* view of example difficulty. This computational view asserts that predictions are (effectively) already made in the hidden layers of a deep model. For "easy" examples, effective predictions can be made after only a few layers, while more layers are used for "difficult" examples. We parameterize the computational difficulty of an input using the prediction depth, defined in Sec. 2.1. In this paper we present evidence to support this viewpoint, and by connecting the prediction depth to the model's uncertainty, confidence, accuracy and speed of learning for each data point we aim to better understand deep learning.

2.1 Definition of the Prediction Depth

An input's final prediction is typically already assigned in the hidden layers of a network, before the output. We estimate the depth at which a prediction is made for a given input as follows:

1. Construct k-NN classifiers from the embeddings of the training set after each layer of the network, including the input.
2. A prediction is defined to be made at a depth $L = l$, if the k-NN classification after layer $L = l - 1$ is different from the network's final classification, but the classifications of k-NN probes after every layer $L \geq l$ are all equal to the final classification of the network. Data points consistently classified by all k-NN probes are determined to be (effectively) predicted in layer 0 (the input).

When assigning the prediction depth to an input x in the training set, we leave the embeddings of x out of the k-NN data sets. We used $k = 30$ in the k-NN classifiers and established that the k-NN accuracies were insensitive to k over a wide range. See Appendix A.4 for more details.

Note that the prediction depth can be calculated for all data points: both in the training and validation splits. This leads to two notions of computational difficulty:

1. The difficulty of predicting the (given) class for an input (in the training split)
2. The difficulty of making a prediction for an input, unseen in advance (from the validation split)

We examine both notions of computational difficulty in this paper.

k-NN probes are numbered sequentially according to the layer at which they were placed. The placement of k-NN probes is described in Appendix A.5.

In the process of arriving at this definition of the prediction depth we considered several alternatives (see Appendix E), including using the ground truth class in place of the predicted class, and using logistic regression probes in place of k-NN probes.

2.2 Prediction Depth is a Meaningful and Robust Notion of Example Difficulty

In this section we perform sanity checks on the prediction depth and find it to be a consistent and visually intuitive definition of example difficulty.

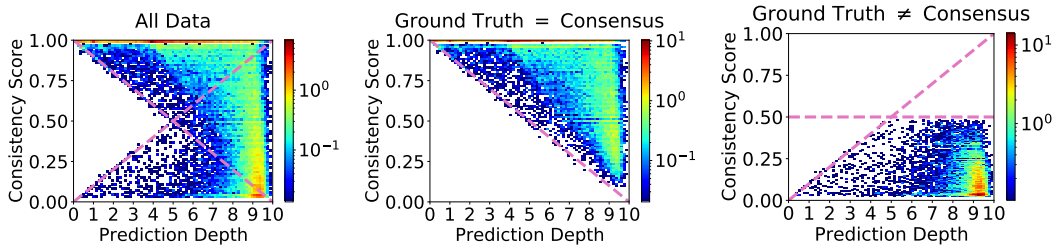

Figure 3: **Consistency score vs. prediction depth in the validation split (left) can be understood as the superposition of two simple patterns (middle and right).** We trained 250 ResNet18 models on CIFAR10, with 90:10% random train:validation splits as described in Appendix A. These histograms compare the frequency of correct predictions to the average prediction depth for a data point when it occurs in the validation split. For points where the ground truth matches the consensus class (middle plot) we see that the prediction depth forms a lower bound on the consistency score. For points where the ground truth and consensus class differ (right plot) at low prediction depth the consistency score is bounded from above by a line that reflects the bound from the middle plot in $\hat{C} = \frac{1}{2}$, suggesting that such points are repeatedly mislabelled with a wrong class label. At high prediction depth, the consistency score is low, which suggests highly inconsistent predictions and low accuracy. This Figure is reproduced for all data sets and architectures in Appendix C.1, illustrating the consistency of this result.

Prediction depth is higher for examples and data sets that appear to be more difficult: If prediction depth is a sensible measure of example difficulty then we would expect to observe two things:

1. Individual data points that are visually confusing or mislabelled should have larger prediction depths as compared to images that are clear examples of their class.
2. Data points from training sets that are intuitively simpler to understand should have lower prediction depths on average.

Fig. 1 shows that the prediction depth passes both of these sanity checks.

Prediction depth is consistent across random seeds and similar architectures: Fig. 2 shows the consistency of prediction depth between different architectures and random seeds, for all architectures and data sets. Perfect agreement is not expected as different deep learning algorithms have different inductive biases, and so can be expected to find different inputs more or less difficult. We observe a higher degree of consistency between prediction depth for ResNet18 and VGG16, than between VGG16 and MLP. This result intuitively makes sense, since ResNet18 and VGG16 are both convolutional networks and we expect their inductive biases to be better matched to one another than to MLP.

3 Deep Learning Phenomena Through the Lens of Prediction Depth

Here we explore how computational difficulty can be used to better understand three important aspects of Deep Learning: accuracy and repeatability of a prediction; the order in which data is learned; and the simplicity of the learned function (as measured by the margin) in the vicinity of a data point.

3.1 Predictions for data points with lower computational difficulty are more likely to be accurate and consistent

Much of the success and interest in deep learning stems from its impressive accuracy. However, this generalization is still not well understood, which is an impediment to improving deep learning. We maintain that the accuracy of deep models has its origin in the processing of data as it passes forward through the hidden layers of a deep network. To better understand the generalization of deep neural networks, we will compare the accuracy and consistency of predictions to the prediction depth for CIFAR100, CIFAR10, SVHN and Fashion MNIST data sets with ResNet18, VGG16 and MLP architectures. We use the consistency score [31] to measure average accuracy. The consistency score represents the *statistical* view of example difficulty.

Consistency score $\hat{C}$: This is the frequency of classifying an example correctly when it is omitted from the training set. An empirical estimator of the consistency score for a validation point (x, y) is given by

$$\hat{C}_{A,\hat{\mathcal{D}},n,r}(x,y) = \hat{\mathbb{E}}^{r}_{D \overset{n}{\sim} \hat{\mathcal{D}} \setminus \{(x,y)\}} \left[\delta_{y_A, y} \right] \tag{1}$$

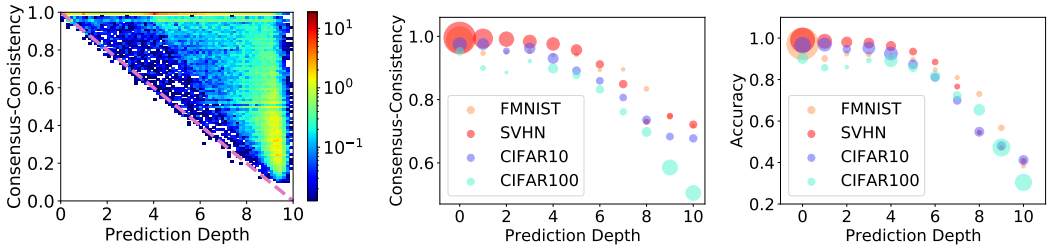

Figure 4: **Left: Average prediction depth in an ensemble bounds the consensus-consistency (left). Middle: Prediction depth in one model predicts the consensus-consistency an ensemble that does not include that model. Right: Prediction depth predicts accuracy. Left:** results for CIFAR100 with ResNet18. We train 250 models (90:10% random train:validation splits) and compare the average prediction depth when a point occurs in the validation set, to the consensus-consistency of the corresponding predictions. Predictions made for points with low mean prediction depths have low uncertainty. Conversely, predictions for points with high mean prediction depths typically have high uncertainty. This left plot shows the result for CIFAR100 with ResNet18. **Middle:** for each data set we train 25 ResNet18 models with the full training set (see Appendix A). The consensus-consistency of each test point is obtained from 24 of the models, while the prediction depth is obtained from the remaining 1 model. We see that prediction depth in one model predicts the consensus-consistency of a separate ensemble. The size of each marker in the middle and right plots shows the fraction of the data set with each prediction depth. **Right:** For each data set we train 250 ResNet18 models (90:10% random train:validation splits). Each time a point appears in the validation split we record the prediction depth and whether the prediction was correct. Predictions effectively made in earlier layers are more likely to be correct. Consistency of these plots is demonstrated for all data sets and architectures in Appendix C.1 where we also show that the prediction depth in one model predicts the *entropy of the predictions* for an ensemble that does not include that model.

where A is a deep learning algorithm (architecture, loss and optimizer), y is the ground truth class for x, D is a random subset of n points sampled from a training data set $\hat{\mathcal{D}}$ excluding (x, y), y_A is the predicted class of x for A trained with data D, δ is the Kronecker delta and $\hat{\mathbb{E}}^r$ denotes empirical averaging with r i.i.d. samples of such subsets D.

We will present evidence that data points in the validation split are processed as though they belong to their predicted class.

Consensus class $Mo\,(x)$: We call the modal predicted class from an ensemble of r models, for an input x, the *consensus class* of x. Each model is trained on a randomly chosen subset of n data points $D \overset{n}{\sim} \hat{\mathcal{D}} \backslash \{(x, y)\}$. Further implementation details can be found in Appendix A.6.1.

For each data set and architecture we train 250 models with 90:10% train:validation split as described in Appendix A. Each time a data point appears in the the validation split we record the prediction depth and whether the prediction is correct. In Fig. 3 we analyse the relationship between consistency score and the corresponding measured average value of the prediction depth. Later in this section we will see that the prediction depth in a model is sufficient to estimate the accuracy of the prediction for test samples drawn i.i.d. from the same distribution as the training set.

Until this point we have described how the expected prediction depth in an ensemble, for an input, relates to the expected accuracy. Now we show a stronger result: that the specific prediction depth of in a particular model can be used to estimate the uncertainty (and the expected accuracy) of the corresponding prediction.

The distribution of predictions in an ensemble is a popular measure of calibration (E.g. [33, 34, 35]). One measure of uncertainty is the frequency with which an ensemble predicts its own consensus class for an input. We call this measure the consensus-consistency score.

Consensus-consistency score C^*: The fraction of models in an ensemble that predict the ensemble's consensus class $Mo\,(x)$ for an unseen input x.

$$C^*_{A,\hat{\mathcal{D}},n,r}(x) = \hat{\mathbb{E}}^r_{D \overset{n}{\sim} \hat{\mathcal{D}} \backslash \{(x,y)\}} \left[\delta_{y_A, Mo(x)} \right] \tag{2}$$

where symbols have the same meanings as in (1). Further details of our implementation are given in Appendix A.6.2.

In Fig. 4 we show that the mean prediction depth in an ensemble (validation split) bounds the consensus-consistency score (2) (left). This linear bound is surprisingly simple, and the middle and right plots report still stronger results. Left: the prediction depth for a test point in a particular trained model predicts the consensus-consistency of an ensemble that *does not include that model*. Right: the prediction depth for a test point in a particular trained model can be used to estimate the probability

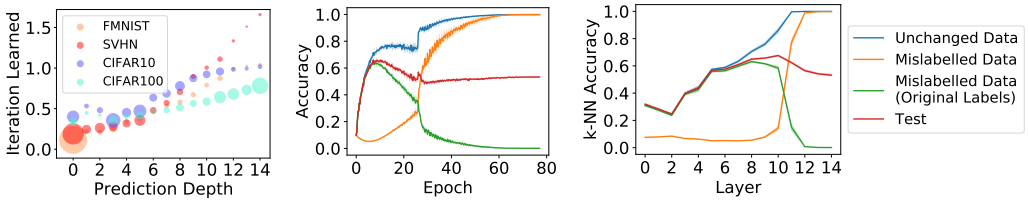

Figure 5: **Left: Data points with small prediction depths are on average learned before data points with higher prediction depths. The training learning curve (middle) shares several important features with the inference learning curve (right).** These plots show results for VGG16. **Left:** We train 250 models for each data set, using a 90:10% random train:validation split as described in Appendix A. Each time an input appears in the validation split we record the prediction depth and the iteration learned in that model. This plot shows the average iteration learned for data points at each prediction depth. Marker size shows the fraction of the data set with each prediction depth. This positive correlation is reproduced for both train and validation splits, in all architectures and data sets in Appendix C.3. In the **middle** and **right** plots, blue, yellow and green curves represent different components of the CIFAR10 training split, in which we have randomized (and fixed) 40% of the labels, and red curves show the test split. The middle and right plots show results from 5 random seeds. The inference learning curve (right) is the sequence of k-NN probe accuracy values for an input. During training (middle), mislabelled data are initially processed as though they are a member of their original class (before they were mislabelled) [36]. After an initial period of learning, the network begins to learn the new (random) labels that have been assigned to those data points, so the orange curve moves upwards, and the green curve downwards. At this point, a maximum is observed in the training accuracy [37]. In the right plot we see that these same phenomena occur in the inference learning curve. The hyperparameters used are given in Appendix A.

the prediction is correct. The observation that prediction depth bounds the consensus-consistency score unifies the behaviours shown in Fig. 3 for points where the consensus and ground truth classes are matching and different: since the prediction depth bounds *consensus-consistency*, in early layers we observe low consistency score for points with different ground truth and consensus classes.

3.2 The computational difficulty of an input is correlated with its learning difficulty

We have reported the relationship between the prediction depth and generalization. Now we turn our attention to the difficulty of learning an example, and its relationship to the prediction depth. We measure the difficulty of learning an example by the speed at which the model's prediction converges for that input during training. Since we are reporting results for classification, the speed of convergence corresponds to the training iteration at which the data point is "learned". We use a definition of iteration learned adapted from [32].

Iteration learned A data point is said to be learned by a classifier at training iteration $t = \tau$ if the predicted class at iteration $t = \tau - 1$ is different from the final prediction of the converged network and the predictions at all iterations $t \geq \tau$ are equal to the final prediction of the converged network. Data points consistently classified after all training steps and at the moment of initialisation, are said to be learned in step $t = 0$.

Note that this definition can be applied equally to points in the training and validation splits. In order to compare different networks and data sets we rescale the iteration learned in each network so that the 95th percentile occurs at 1.0 and network initialization at 0.

Fig. 5 (left plot) shows the positive correlation between the prediction depth and the iteration learned, for all four data sets, in VGG16. Consistent results are presented for all data sets, in both the validation and training splits, and all architectures, in Appendix C.3. As a result of the reported correlation, we anticipate that many of the data points correctly classified by the k-NN probe in a particular layer would also be correctly classified by the network at a corresponding interval of training steps. Indeed, we see in Fig. 5 (middle and right plots) that many of the important features of the *training learning curve* (which shows how the accuracy of the network changes during training) are also present in the *inference learning curve*, which is the accuracy of the k-NN probes as data passes from input, through the network, towards the output layer. These results reveal a surprising relationship between the processing of data inside a converged network and the network's training history.

3.3 Deep models exhibit larger margins for inputs with lower computational difficulty

It is reported in the literature that deep networks learn functions of increasing complexity during training [38, 39]. We frame this observation in a different manner: that the learned function is also "locally simpler" around the data points with smaller prediction depths, and these points are typically learned earlier in training (Sec. 3.2).

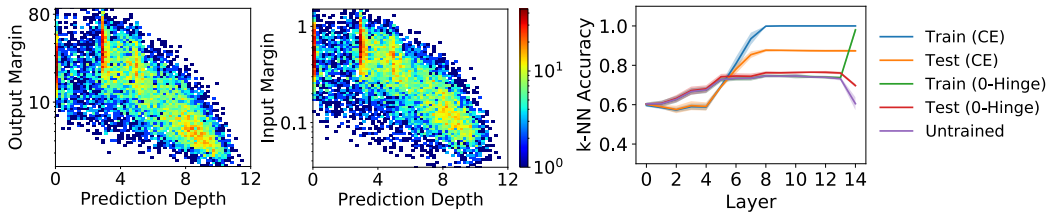

Figure 6: **Left and middle: Test examples with smaller prediction depths, on average, have larger output and input margins. Right: An intervention to minimise the output margin ("0-Hinge" training) results in models where the data clusters by class only in the latest layers. In contrast, standard training ("CE") results in accurate clustering after only 8 layers. Left:** We train 25 models with CIFAR10 on VGG16 (see Appendix A for details). We compare the mean prediction depth of each test point in these 25 runs to its mean output and input margins (log scales). Correlation coefficients are -0.70 (output margin) and -0.69 (input margin). Although the k-NN probes used number up to 14, no data point has an average prediction depth greater than 12. **Right:** Mean and standard deviations for 5 models for each of 0-Hinge (trained to minimize output margin) and CE (standard training) as described in Sec. 3.3. In 0-Hinge the k-NN probe accuracies are very similar to those for an "Untrained" (freshly initialised) network, until the last layers of the network. As described in Appendix A.7, this plot is produced using a subset of CIFAR10, unlike the left and middle plots. Consequently CE achieves accurate clustering after only 8 probes, rather than 12.

Two measures of the simplicity of a learned function are the output margin (the difference between the largest and second largest logits) and the adversarial input margin (the distance between an input x and the closest input by euclidean distance $x + \delta$ with a different predicted class). We estimate the adversarial input margin with a linear approximation [10]: for an input x with predicted class i, $|\delta| \simeq \min_{j \neq i} \frac{|z_i - z_j|}{|\nabla_x (z_i - z_j)|}$ where z_j is the logit returned by the network for class j. Fig. 6 shows that data points with smaller prediction depths have larger margins on average, which is our point.

To illustrate the strength of the relationship between the prediction depth and output margin, we demonstrate that modifying the output margin of the learned function results in a model that clusters the data only in the latest layers: such a solution has a very high average prediction depth. Naturally there are many unknowns that may contribute to this effect. We simply report the intervention and the outcome. The intervention is performed as follows: we construct a loss function that does not promote confidence: a zero-margin hinge loss (3), and optimize the network using *full batch* gradient descent with momentum and *very small learning rate*. For an input x with label i this loss is given by

$$l(x) = \sum_{j \neq i} \max(0, z_i - z_j) \tag{3}$$

where z_j represents the logit for class j. As a control, we additionally train a model in the standard fashion using the cross-entropy loss and SGD with momentum and large initial learning rate. Full-batch gradient descent is computationally expensive, so we train on a subset of CIFAR10 (see Appendix A.7, where we also give the hyperparameters and learning curves.) Fig. 6 (right plot) compares the accuracies of the k-NN probes resulting from these training approaches: the result is striking and just as predicted. Additionally, we observe markedly different output margins for these two training methods: $2.0 \times 10^{-4} \pm 2.0 \times 10^{-4}$ for the zero-margin hinge loss, and $1.6 \times 10^{1} \pm 0.50 \times 10^{1}$ for cross-entropy loss. The hinge loss training also achieves only a marginal improvement in accuracy (red) over an untrained network (purple). The efficacy of this intervention is remarkable.

4 Forms of Example Difficulty

In this section we seek to understand the nature of example difficulty at a higher level: to develop a perspective that is general to different architectures and data sets, and one which motivates future deep learning research.

Fig. 7 shows that the prediction depth can be different when x occurs in the train split vs. the validation split. Thus, there are two axes of example difficulty:

1. Difficulty of making a prediction when an input is in the validation set
2. Difficulty of finding commonalities with other examples of an inputs class during training

The first can be understood as the difficulty of finding commonalities with other images in any class, the second the difficulty of finding commonalities with the ground truth class in particular. **Both axes have a range from "clear" to "ambiguous".** In Sec. 3.1 we showed that predictions made for validation points with later prediction depths are often inconsistent, with low consensus-consistency.

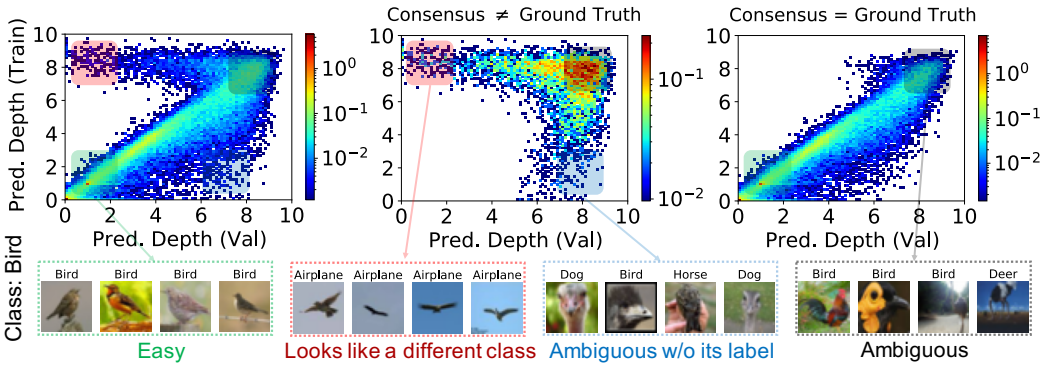

Figure 7: **The prediction depth can be the same, or very different for the same input when it occurs in the train and validation splits. Corners of this plot correspond to different forms of example difficulty.** (See Sec. 4 for discussion.) We train 250 ResNet18 models on CIFAR10 with random 90:10% train:validation splits as described in Appendix A. These histograms compare average prediction depth for each data point when it occurs in the validation split vs the training split. In the full data set (**left**) we see that the prediction depth can be very different in the train and validation splits: the two prediction depths are typically similar for points where the consensus class is equal to the ground truth (**right**), but can be very different when the consensus class is different from the ground truth (**middle**). This behaviour is consistently reproduced for all data sets and architectures in Appendix C.3. Below we show extreme (not hand chosen) images of "Birds" that appear closest to the corners of this plot. The consensus class is given above each image (tie breaks favour the class "Bird".) ResNet18 finds small garden birds easiest, while birds in flight against a blue background "look like aeroplanes", ostriches are "ambiguous without their label" and the "ambiguous" examples are either unclear photographs or examples of rare sub-groups that don't appear frequently in the data. The consensus-consistency of inputs that are "Ambiguous" or "Ambiguous without its label" are significantly lower than those of examples that are "Easy" or "Look like a different class".

Conversely, a low prediction depth typically indicates an input with high consensus-consistency. For Axis 1 we will identify validation points with low prediction depths as "clear" and those with high prediction depths as "ambiguous". We will additionally identify a low or high prediction depth in the training split with examples that are respectively "clear" and "ambiguous" on Axis 2. By making combinations of low/high values of $(\mathrm{LL}_{\mathrm{Val.}}, \mathrm{LL}_{\mathrm{Train}})$ we obtain four extremes of example difficulty:

Easy examples: (Low $\mathrm{LL}_{\mathrm{Val.}}$, Low $\mathrm{LL}_{\mathrm{Train}}$). Such examples are often visually typical members of their class and the predicted label nearly always matches the ground truth.

Looks like a different class: (Low $\mathrm{LL}_{\mathrm{Val.}}$, High $\mathrm{LL}_{\mathrm{Train}}$). In the validation set, there is a clear (and nearly always incorrect) classification for such an input, but it is difficult to connect such inputs to other examples of their ground truth class during training. Mislabelled examples are of this kind, as are visually confusing images which at first appear to show something else.

Ambiguous unless the label is given: (High $\mathrm{LL}_{\mathrm{Val.}}$, Low $\mathrm{LL}_{\mathrm{Train}}$). These examples are difficult to connect to their predicted class in the validation split but easy to connect to their ground truth class during training. These points may, for example, show instances of a class that visually resemble both their own class and another class. They are likely to be misclassified.

Ambiguous: (High $\mathrm{LL}_{\mathrm{Val.}}$, High $\mathrm{LL}_{\mathrm{Train}}$). These examples may be corrupted or show an example of a rare sub-class. Predictions for these inputs can depend strongly on a model's initialization.

In Fig. 7 we visualize CIFAR10 "Bird" images with the extreme forms of example difficulty for ResNet18, as identified using the prediction depth in the training and validation splits.

In order to better understand how networks process examples with different, extreme forms of example difficulty Fig 8 examines how the k-NN confidence (fraction of votes) and accuracy of the ground truth class and of the consensus class progress, as validation points pass through the network.

5 Discussion

Pertinence of example difficulty to topics in machine learning: Our results are relevant to at least three important areas of deep learning research: distribution shift and robustness, algorithmic fairness, and curriculum learning. These problems are all related to the interplay between deep learning algorithms and data quality, "label noise", or bias, which can be understood in terms of example difficulty. We expand upon the relevance of our work to these three topics in Appendix D.

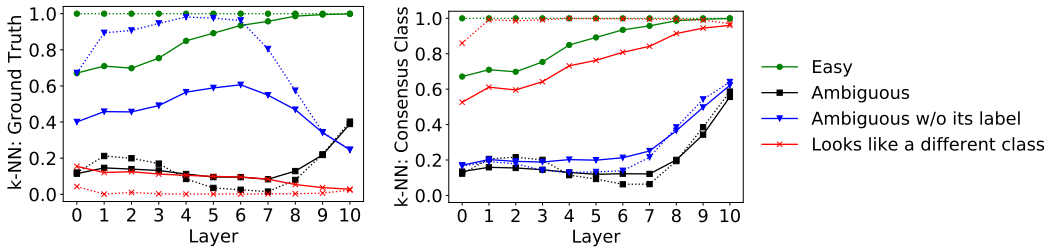

Figure 8: **Average k-NN confidence (frequency of votes) (solid lines) and accuracy (dotted lines) for the ground truth class label (left) and consensus class label (right), in the validation split vs layer for examples exhibiting extremes forms of difficulty.** Mean values for 100 examples with each form of difficulty, identified as the 100 examples closest to the corners in Fig. 7. For "easy" examples the ground truth matches the consensus class. "Easy" examples are classified as their consensus class in all k-NN probes and the confidence in the consensus steadily increases as data points proceed through the hidden layers. For examples that "look like another class" the consensus class is different from the ground truth. These validation points are processed as members of their consensus class, similarly to "easy" examples. Examples that are "ambiguous without their labels" are initially accurately processed as members of their ground truth classes with intermediate confidence, but in later layers become mistaken for their consensus class. "Ambiguous" examples are processed with low confidence and accuracy in all layers, for both ground truth and consensus classes, until the latest layers. Initially, confidence in the ground truth label grows, then they are processed as a member of a third class that is neither the ground truth nor the consensus class, finally they are clustered with their consensus class, which agrees with the ground truth for a fraction of "ambiguous" examples. This result is for CIFAR10 with ResNet18: similar plots for all data sets and architectures are shown in Appendix C.6.

Connecting known phenomena: In the literature, the following phenomena are separately reported from different experimental paradigms:

1. Early layers generalize while later layers memorize [40].
2. Networks layers converge from input layer towards output layer [41, 42].
3. Networks learn easy data [31, 32] and simple functions first [38, 39].

Following this paper, a coherent and closely related picture emerges:

1. Predictions made in early layers are more likely to be consistent than those made in later layers. Predictions which are consistent are likely to be correct and the expected accuracy of inconsistent predictions is naturally low.
2. Data points learned early in training typically have smaller prediction depths than those learned later during training.
3. On average, deep neural networks exhibit wider margins (a common measure of "local simplicity") in the vicinity of data with smaller prediction depths.

Limitations: We believe that the results we report stem from a deep model's representation, which is hierarchical by construction. It is our expectation that the same results will therefore apply in larger models, larger data sets, and tasks other than image classification, but this remains further work. We do not claim that all that is required for good accuracy is to reduce the prediction depth: freezing later layers of the network would not be expected to result in good generalization. We propose practitioners use approaches like using different learning rates in each layer, adjusting the initialisation in different layers, curriculum learning and using new loss functions to encourage learning in early layers.

6 Conclusion

We have shown the prediction depth to be a consistent measure of example difficulty, between architectures and random seeds, and that it qualitatively agrees with our own intuition for image difficulty. We have shown that the expected prediction depth in the validation set forms a surprisingly simple, linear lower bound on the consensus-consistency of the prediction and that validation points with small prediction depths are more likely to be correctly classified. We have reported that data points with smaller prediction depths are on average learned earlier in training, and used this observation to rationalize a visual correspondence between the training learning curve and the inference learning curve. We had demonstrated that both the output and input margins are on average higher for examples with smaller prediction depths and that an intervention to minimize the output margin dramatically increases the layer at which the training split is accurately clustered. Finally we were able to express a novel and coherent viewpoint of deep learning that captures important aspects of four diverse deep learning phenomena, reported from different experimental setups.

References

[1] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, pages 2232–2241. PMLR, 2019.

[2] Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael Mahoney. Pyhessian: Neural networks through the lens of the hessian. *arXiv preprint arXiv:1912.07145*, 2019.

[3] Levent Sagun, Leon Bottou, and Yann LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2016.

[4] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *NeurIPS*, 2018.

[5] Jeffrey Pennington and Yasaman Bahri. Geometry of neural network loss surfaces via random matrix theory. In *International Conference on Machine Learning*, pages 2798–2806. PMLR, 2017.

[6] Levent Sagun, Utku Evci, V Ugur Guney, Yann Dauphin, and Leon Bottou. Empirical analysis of the hessian of over-parametrized neural networks. *arXiv preprint arXiv:1706.04454*, 2017.

[7] Philip M Long and Hanie Sedghi. Generalization bounds for deep convolutional neural networks. In *International Conference on Learning Representations*, 2019.

[8] Thomas Unterthiner, Daniel Keysers, Sylvain Gelly, Olivier Bousquet, and Ilya Tolstikhin. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.

[9] Yiding Jiang, Behnam Neyshabur, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

[10] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. In *International Conference on Learning Representations*, 2018.

[11] Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. Generalization in deep learning. *arXiv preprint arXiv:1710.05468*, 2017.

[12] Samuel L Smith, Benoit Dherin, David GT Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. *arXiv preprint arXiv:2101.12176*, 2021.

[13] M Stephan, Matthew D Hoffman, David M Blei, et al. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 18(134):1–35, 2017.

[14] Samuel L Smith and Quoc V Le. A bayesian perspective on generalization and stochastic gradient descent. In *International Conference on Learning Representations*, 2018.

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[16] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[17] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[18] Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Dnn or k-nn: That is the generalize vs. memorize question. *arXiv preprint arXiv:1805.06822*, 2018.

[19] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

[20] Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *arXiv preprint arXiv:1803.04765*, 2018.

[21] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *International Conference on Learning Representations*, 2018.

[22] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2464–2469. IEEE, 2016.

[23] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations*, 2018.

[24] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and QI JU. Fastbert: a self-distilling bert with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, 2020.

[25] Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A Smith. The right tool for the job: Matching model and instance complexities. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6640–6651, 2020.

[26] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. Deebert: Dynamic early exiting for accelerating bert inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2246–2251, 2020.

[27] Nicholas Carlini, Ulfar Erlingsson, and Nicolas Papernot. Distribution density, tails, and outliers in machine learning: Metrics and applications. *arXiv preprint arXiv:1910.13427*, 2019.

[28] Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019.

[29] John P Lalor, Hao Wu, Tsendsuren Munkhdalai, and Hong Yu. Understanding deep learning performance through an examination of test set difficulty: A psychometric case study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing. Conference on Empirical Methods in Natural Language Processing*, volume 2018, page 4711. NIH Public Access, 2018.

[30] Chirag Agarwal and Sara Hooker. Estimating example difficulty using variance of gradients. *arXiv preprint arXiv:2008.11600*, 2020.

[31] Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. Characterizing structural regularities of labeled data in overparameterized models. *arXiv e-prints*, pages arXiv–2002, 2020.

[32] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

[33] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6405–6416, 2017.

[34] Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *arXiv preprint arXiv:2006.13570*, 2020.

[35] Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2019.

[36] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, 33, 2020.

[37] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR, 2017.

[38] Wei Hu, Lechao Xiao, Ben Adlam, and Jeffrey Pennington. The surprising simplicity of the early-time learning dynamics of neural networks. *arXiv preprint arXiv:2006.14599*, 2020.

[39] Preetum Nakkiran, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L Edelman, Fred Zhang, and Boaz Barak. Sgd on neural networks learns functions of increasing complexity. *arXiv preprint arXiv:1905.11604*, 2019.

[40] Cory Stephenson, suchismita padhy, Abhinav Ganesh, Yue Hui, Hanlin Tang, and SueYeon Chung. On the geometry of generalization and memorization in deep neural networks. In *International Conference on Learning Representations*, 2021.

[41] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6078–6087, 2017.

[42] Ari S Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5732–5741, 2018.

[43] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[44] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. *arXiv preprint arXiv:1912.11370*, 6(2):8, 2019.

[45] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[46] Satrajit Chatterjee. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. In *International Conference on Learning Representations*, 2019.

[47] Piotr Zielinski, Shankar Krishnan, and Satrajit Chatterjee. Weak and strong gradient directions: Explaining memorization, generalization, and hardness of examples at scale. *arXiv e-prints*, pages arXiv–2003, 2020.

[48] Guy Hacohen, Leshem Choshen, and Daphna Weinshall. Let's agree to agree: Neural networks share classification order on real datasets. In *International Conference on Machine Learning*, pages 3950–3960, 2020.

[49] Dara Bahri, Heinrich Jiang, and Maya Gupta. Deep k-nn for noisy labels. In *International Conference on Machine Learning*, pages 540–550. PMLR, 2020.

[50] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

[51] Vaishnavh Nagarajan, Anders Andreassen, and Behnam Neyshabur. Understanding the failure modes of out-of-distribution generalization. In *International Conference on Learning Representations*, 2021.

[52] Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

[53] Terence D Sanger. Neural network learning control of robot manipulators using gradually increasing task difficulty. *IEEE transactions on Robotics and Automation*, 10(3):323–333, 1994.

[54] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of International Conference on Machine Learning*, 2009.

[55] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. *ICML*, 2019.

[56] Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? *International Conference on Learning Representations*, 2021.

[57] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *arXiv preprint arXiv:2008.03703*, 2020.

[58] Sara Hooker, Nyalleng Moorosi, Gregory Clark, Samy Bengio, and Emily Denton. Characterising bias in compressed models. *arXiv preprint arXiv:2010.03058*, 2020.

[59] Karttikeya Mangalam and Vinay Prabhu. Do deep neural networks learn shallow learnable examples first? In *Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   [Yes] To the best of our knowledge the contributions section is accurate and in our view corresponds to both the abstract and the paper.
   (b) Did you describe the limitations of your work?
   [Yes] See Sec. 5
   (c) Did you discuss any potential negative societal impacts of your work?
   [No] The aim of this paper is fundamental insight into the workings of Neural Networks. Although in the long run this may lead to further insights that translate into advances in practical algorithms (see D for some possible directions), even they are quite general and there would be little basis now to speculate about their potential negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results?
   [N/A] We do not use theoretical results in a mathematical proof. We only use a published method of approximating the adversarial input margin in section 3.3 as a motivation for our definition of "margin".
   (b) Did you include complete proofs of all theoretical results?
   [N/A] No new theoretical results are claimed.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We use standard data sets and network architectures, and do not propose new algorithms.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)?
   [Yes] See Appendix A.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)?
   [No] Most plots (Figures 2,3,4,6,7) contain this information in another form, since they plot the point cloud corresponding to all measurements. Figure 5 is already an average over 250 models, any error bar of the mean (standard deviation / $\sqrt{250}$) would be too small to be noticed.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)?
   [No] We used CPUs and cloud compute; most of the energy consumption came from renewable sources, the rest was offset by the cloud provider. We did not track the total amount of compute used for this paper, but it should be modest since we demonstrate fundamental deep learning phenomena on data sets that are "small" by today's standards ("only" CIFAR10, CIAR100, SVHN, FMNIST, and not ImageNet and larger.)

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators?
   [Yes] See Appendix A.1 for the data sets we use.
   (b) Did you mention the license of the assets?
   [Yes] See Appendix A.1 for the data sets we use.
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating?
   [N/A] Only the above common data sets were used.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content?
   [N/A] Only the above common data sets were used.
5. If you used crowdsourcing or conducted research with human subjects...
   [N/A] No crowdsourcing or research with human subjects was used.