
Meet You Halfway: Explaining Deep Learning Mysteries

Oriel BenShmuel
 Faculty of Math&CS
 Weizmann Institute of Science
 Israel
 oriel.benshmuel@weizmann.ac.il

Abstract

Deep neural networks perform exceptionally well on various learning tasks with state-of-the-art results. While these models are highly expressive and achieve impressively accurate solutions with excellent generalization abilities, they are susceptible to minor perturbations. Samples that suffer such perturbations are known as “adversarial examples”. Even though deep learning is an extensively researched field, many questions about the nature of deep learning models remain unanswered. In this paper, we introduce a new conceptual framework attached with a formal description that aims to shed light on the network’s behavior and interpret the behind-the-scenes of the learning process. Our framework provides an explanation for inherent questions concerning deep learning. Particularly, we clarify: (1) Why do neural networks acquire generalization abilities? (2) Why do adversarial examples transfer between different models?. We provide a comprehensive set of experiments that support this new framework, as well as its underlying theory.

1 Introduction

Past work had successfully promoted deep neural networks to be one of the top-list models when it comes to learning tasks such as classification, prediction, and generation of new information [35, 26, 11, 27, 9, 34, 6, 1]. Many problems in computer vision and natural language processing have reached state-of-the-art results using deep neural networks [10, 31, 30, 7, 22, 24, 2, 18, 5]. Naturally, these models attracted a lot of attention in the artificial intelligence community. Along with their profound impact, deep neural networks raise many questions regarding their behavioral nature.

Adversarial examples are modified samples based on minor perturbations that are hardly imperceptible to the human visual system [29]. Yet, these perturbations cause even a well-trained network to misclassify the produced image. Furthermore, the nature of adversarial examples is not associated with any specific network. The same adversarial example would manipulate (with high probability) a different network trained with a different architecture and disjoint dataset [29]. The latter phenomenon is known as “transferability”, and currently, its causes remain unclear [21, 33]. Naturally, the “transferability” occurrence might be used to generate black-box attacks, where one might generate an adversarial example without using any information regarding the target model [21].

A series of studies discussed this disadvantage with various efforts for its prevention and for the creation of robust models [23, 32]. Several approaches aimed to explain the nature of adversarial perturbations. Two notable examples are the “centroid” approach and the “robust and non-robust features” theory [13, 15]. The “centroid” approach, while not officially defined, generally refers to the adversarial direction as some weighted direction toward the closest samples from the target class (for example, it was mentioned by Ian Goodfellow in his lecture [8] at 1:11:54). The second approach [13] refers to “robust features” and “non-robust features”. The robust and non-robust features theory,

on the other hand, defines robust features as patterns that are predictive of the true label even when adversarially perturbed. Conversely, non-robust features correspond to patterns that, while predictive, can be “flipped” by an adversary within a predefined perturbation set to indicate a wrong class.

Another series of theoretical studies explore the ability of deep neural networks to generalize [14, 25, 4]. Despite the non-convex nature of the optimization problem, simple optimization methods such as SGD have been found to perform well [20, 3]. The broad set of possible parameters attached to the training process constitutes the possible converged states of the model. This over-parametrized setting may offer many possible solutions for the objective, but not all of them demonstrate a well-generalized solution [36]. Optimization with respect to the training set is partially contradictory to achieving ample performance on the test set, resulting in a bias-variance tradeoff. In other words, the goal is to learn the training set distribution without overfitting. Multiple approaches have been suggested to alleviate this problem [17, 37].

Despite these conceptual ideas and practical suggestions, some of the fundamental questions concerning deep neural networks, such as “Why do networks generalize?” or “Why do adversarial examples transfer?” are still vague. In this paper, we explain the evolution of the decision boundary and its properties during the network’s learning process. In particular, we expose an additional optimization goal of the training. This reveal provides us with a set of straightforward explanations of the intrinsic and counter-intuitive occurrences “generalization” and “transferability”. We begin in sections 2 and 3 by illustrating the properties of the decision boundary and explaining under what constraints it evolves (with emphasis on the new optimization property we expose). Next, in sections 4 and 5, we unfold the two phenomena described earlier through the lens of the elaborated optimization properties. Finally, in section 6, we provide a broad set of experiments to support our claims. For the sake of simplicity, we consider only 2-class classifiers and use ℓ_2 norms.

2 Paradigm shift

Generally, a set of representatives sampled from the data distribution might be separated in multiple independent manners according to the provided labels. However, the separation itself is not the only consideration of a classification network. Deep neural networks are constrained to a set of optimization requirements when learning the data, which lead to the desired goal of correct classification. In this paper, we mention these constraints and describe one novel requirement of the optimization process.

Apart from the optimization requirements, our new framework illustrates the learning process as a difference-based mechanism, where mutual properties of the classes, such as objective background (unrelated to a particular class distribution), will be neglected in the network consideration (with the exception of statistical errors). In fact, we might find many features related to the semantic meaning of the classes, and yet, they will be neglected as they are distributed similarly for both classes. Therefore, these features will not contribute any new insights regarding the differences between the classes. Following this perspective, any feature that the network learns represents some property expressed dichotomously among the two classes.

The optimization process aims to detect dichotomous properties that are not associated only to a specific separator but to the class distributions (as we clarify formally in subsection 2.1). In other words, these dichotomous properties are based on the gap between the two class distributions. In that case, reduction of the entirety of these dichotomous properties would generate indistinguishable distributions of the two classes (subject to the limited accuracy of the settings). However, reduction of only some of these properties would only reduce the distributional difference between the classes. Assume a dataset that appropriately represents the classes’ distributions, with enough samples to indicate some of the mentioned differences. Reduction of these indicative differences would be expressed with a Euclidean distance reduction between the classes, even when using complex datasets and natural images (for more details, see section 6).

To simplify this concept, we define the *projection* of a sample in the input space onto the decision boundary of a classification network to be the sample’s nearest point on the decision boundary. We refer to *global difference* as a measurement tool that indicates the extent to which the projected training set (with respect to the trained network’s decision boundary) reduced the difference between the classes’ distributions in a particular set of directions in the input space. In other words, after reducing a “global difference” property (by using the projection of the training set), it will be reduced

with respect to any possible (separating) boundary. This intuitive mental image is formally explained in subsection 2.1 under the assumptions in subsection 2.2.

2.1 Formal description

Let $x \in \mathbb{R}^n$ be an input sample, and $l \in \{0, 1\}$ is the corresponding label. For a broader context, denote $\mathcal{I} = \{x_i\}_{i=1}^s$ as a set of samples from the data distribution with the corresponding labels $\mathcal{L} = \{l_i\}_{i=1}^s$. To address the problem more elegantly, we define (or redefine) some expressions to the scope of this paper.

Definition 1 (Classifier). The mapping $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a *classifier* for the set of samples and labels $(\mathcal{I}, \mathcal{L})$, if $f(x_i)f(x_j) < 0$ for any $x_i, x_j \in \mathcal{I}$ such that $l_i \neq l_j$. We denote $\mathcal{C}_{(\mathcal{I}, \mathcal{L})}$ as the set of all the classifiers for $(\mathcal{I}, \mathcal{L})$.

Definition 2 (Decision boundary). The *decision boundary* $\mathcal{B}_f \subseteq \mathbb{R}^n$ of any classifier $f \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}$ is defined as $\mathcal{B}_f := f^{-1}(0)$.

Definition 3 (Projection). The *projection* $\mathcal{P}_{\mathcal{B}_f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ of sample $x \in \mathbb{R}^n$ onto the decision boundary \mathcal{B}_f for $f \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}$ is defined as $\mathcal{P}_{\mathcal{B}_f}(x) := \operatorname{argmin}_{b \in \mathcal{B}_f} \|b - x\|$ and the *projection vector* as

$$\overline{\mathcal{P}_{\mathcal{B}_f}}(x) := \mathcal{P}_{\mathcal{B}_f}(x) - x.$$

Note: In this paper (unless stated differently), we discuss decision boundaries with a well-defined function $\mathcal{P}_{\mathcal{B}_f}(x)$, such that exists only a single possible projection for each input. In section 4, we discuss cases with several possible projection sets for the given dataset \mathcal{I} and decision boundary \mathcal{B}_f .

Definition 3 (Global difference). Assume a classifier $f \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}$, where the number of the samples is $|\mathcal{I}| = s$, and the projected set is $\mathcal{I}' = \{x'_i\}_{i=1}^s = \{\mathcal{P}_{\mathcal{B}_f}(x_i)\}_{i=1}^s$. The *global difference* value of f w.r.t $(\mathcal{I}, \mathcal{L})$ is:

$$\phi(f, \mathcal{I}, \mathcal{L}) := s - \max_{g \in \mathcal{C}_{(\mathcal{I}', \mathcal{L})}} \left\{ \sum_{i=1}^s \alpha_i \mid \forall 1 \leq i \leq s : \overline{\mathcal{P}_{\mathcal{B}_g}}(x'_i) = \alpha_i \overline{\mathcal{P}_{\mathcal{B}_f}}(x_i) \wedge 0 \leq \alpha_i \leq 1 \right\}$$

In Figure 1, we can observe an example of different separators for the same problem in the first row (plots 1, 2, 3) and the corresponding projections sets in the second row (plots 4, 5, 6). In plot 4, we can see the projection of plot 1, and as the green separator indicates, there is a separator that could satisfy $\overline{\mathcal{P}_{\mathcal{B}_g}}(x') = \overline{\mathcal{P}_{\mathcal{B}_f}}(x)$ for all the samples. Plot 5 is the projection of plot 2, and as noted by the green circles, we could not separate the samples while generating similar projection vectors (with the same lengths). Instead, we could use $\overline{\mathcal{P}_{\mathcal{B}_g}}(x') = \alpha \overline{\mathcal{P}_{\mathcal{B}_f}}(x)$ for some $\alpha < 1$ to create a separator with the same directions. In plot 3, we generated a separator that puts the examples as far as possible from the decision boundary. In that case, we could not generate a separating boundary for the projected set (hence $\alpha = 0$), as illustrated in plot 6.

Our proposed approach for explaining the intrinsic properties of neural networks claims that the learning process aims to generate a classifier that satisfies the following constraint for any representative training set $(\mathcal{I}, \mathcal{L})$ (limited to the expressive power of the architecture), a claim which we support by informal evidence and quantitative experiments in section 6:

$$f_\phi(\mathcal{I}, \mathcal{L}) = \operatorname{argmax}_{f \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}} \phi(f, \mathcal{I}, \mathcal{L})$$

Among all the possible solutions for $f_\phi(\mathcal{I}, \mathcal{L})$, the network would prefer a classifier that places each training example as far as possible from its decision boundary to maximize the confidence level in its provided labels subject to the limited expressive power of the given DNN. Therefore, we narrow down the possible solutions to the following:

$$F_\phi(\mathcal{I}, \mathcal{L}) = \operatorname{argmax}_{f \in f_\phi(\mathcal{I}, \mathcal{L})} \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\|$$

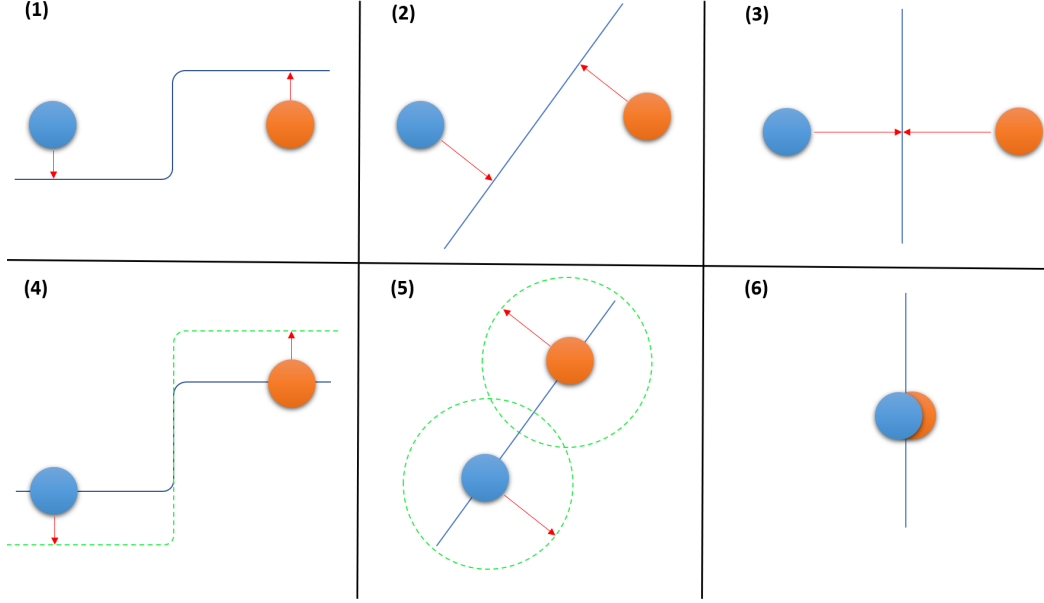


Figure 1: In the first row, there are different classifiers for the same problem. We can see the projections on the first row's boundaries in the second row. Each one of them has a different "global difference" value.

2.2 Assumptions, limitations, and clarifications

Here are some clarifications regarding the new approach:

1. To avoid contradiction with the primary preference of the network (to classify the samples correctly), and due to the limited complexity of the network, a single projection of the training set onto the decision boundary will not reduce (in most natural cases) all of the global differences at once.
2. Our approach describes an optimization problem that aims to reach a *global property*, and it is not necessarily reflected when observing each sample separately.
3. Assume the process of reducing global differences by projecting the training set iteratively (project the samples, find a new separator, project the new samples, etc.). Our definition of "global difference" is sensitive to the iterations' order. Applying the same set of reductions in a different order could not occur according to our maximization requests.
4. We assume that the network is randomly initialized at the beginning of the training process, and the initialized state is not pathologically close to any specific solution (that might not meet the new constraints).
5. The claimed behavior ignores statistical deviations and numerical errors that might occur in some datasets and perform unexpected solutions.
6. Due to the network's degree limitations, the regularization parameters (preventing highly expressive behavior near the samples), and the interest of the decision boundary to stay far from the samples (as described in section 2.1), we assume that the decision boundary does not change its direction frequently and drastically in small regions and specifically near the samples.

3 Euclidean distance and adversarial directions

As shown in section 6, projecting the dataset might be expressed with a Euclidean distance reduction between the classes' representatives in the input space. This puts into question whether our definition

of “global difference” is based on the Euclidean distance between the samples. The concrete notion of “global difference” is not directly related to a Euclidean distance but rather the (possibly) nonlinear properties that differentiate the two classes. However, reducing the gap between the class distributions for a well-representative dataset sampled from the data distribution (particularly for each class) will motivate a Euclidean distance reduction. Generally, this metric reduction will be expressed globally (in average terms) rather than individually for each training set sample.

The “centroid” approach refers to a reduction (namely, the adversarial direction) toward the closest samples from the target class. In contrast, our framework claims for an entirely different occurrence. In our case, samples’ adversarial directions are not influenced by specific samples, but rather all of them are influenced globally. Specifically, they are not influenced due to local regions’ representatives but rather by a global property that might be expressed differently in different local regions. Moreover, we might observe the similarity between the approaches in very simple and low-dimensional cases. In that case, the adversarial directions of close points might be related, not as a result of their individual interaction, but rather due to the similar behavior of the decision boundary in that specific region. While the “centroid” approach might perform well on elementary problems, this is not the case for complex and natural datasets (where the samples are usually very far from each other in the first place). These complex cases are explained clearly with our framework, which we support with experiments (for more details, see section 6).

4 Transferability

In 2013, Szegedy et al. presented the new idea of adversarial examples and some of their essential properties [29]. In the same paper, two additional properties of adversarial examples were introduced:

1. *Cross model generalization*: Adversarial examples will be misclassified, with high probability, by networks trained with different architecture and hyper-parameters.
2. *Cross training-set generalization*: Adversarial examples will be misclassified, with high probability, by networks trained with disjoint datasets (but similar semantic meaning).

These properties are also known as the “transferability” occurrence.

In this section, we aim to explain the “transferability” occurrence. In particular, we claim that our proposed optimization approach for natural datasets generates a single global minimum, and any other minimum is relatively negligible and performs an inferior solution with an improbable convergence process compared to the primary solution. In contrast, datasets with a few meaningful solutions (based on our approach) will have a symmetrical structure, and it is most probable that transferability will not occur among networks that converge to different solutions. We assume that natural datasets are not symmetrically structured due to the possibly unrelated distribution of the background and other natural properties.

Definition 1 (Symmetrical dataset). The dataset \mathcal{I} with the corresponding labels \mathcal{L} is a *symmetrical dataset* if, for any solution $f \in F_\phi(\mathcal{I}, \mathcal{L})$, we could generate a few possible different projection sets $\{\mathcal{P}_{\mathcal{B}_f}(x)\}_{x \in \mathcal{I}}$ for the same classifier f .

In Figure 2, we can observe two different decision boundaries for the same problem. Each decision boundary enables several options for a valid projection.

Assume a pair of possible independent solutions $f, g \in F_\phi(\mathcal{I}, \mathcal{L})$ such that:

$$\forall x \in \mathcal{I} : \overline{\mathcal{P}_{\mathcal{B}_f}}(x) \perp \overline{\mathcal{P}_{\mathcal{B}_g}}(x)$$

In Appendix C, we prove that the described solutions f, g could not co-exist for a non-symmetrical dataset.

Overall, the optimization process of networks that learn natural (and non-symmetrical) datasets has a unique global minimum. Therefore, different networks with different settings that converge to the described global minimum (subject to the architecture’s constraints) will probably provide a similar solution that is expressed with a similar decision boundary. According to [33], decision boundary similarity enables “transferability” as claimed. Additionally, networks that perform well on the dataset with different expressiveness abilities might reach different optimization scores even though it is most probable they will converge to the (same) global minimum. Therefore their decision boundary

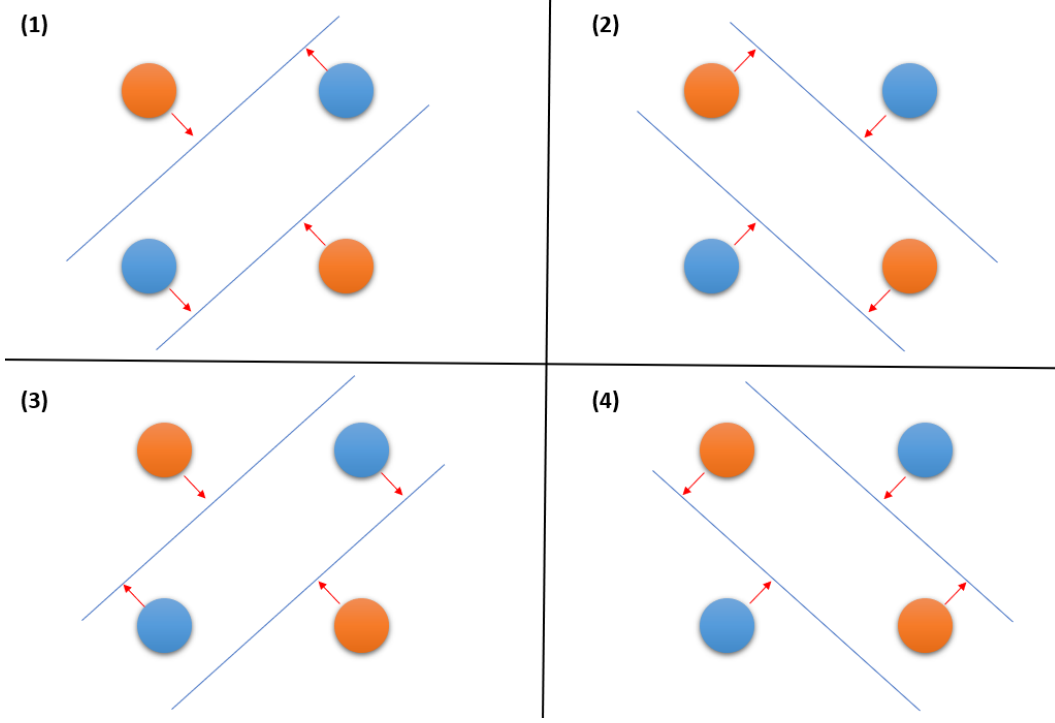


Figure 2: In each column, there are different sets of projections for the same decision boundary.

might look different in some local regions, but the global behavior will stay the same (including transferable adversarial examples).

5 Generalization

A deep neural network trained with a representative dataset will be able (for relevant settings) to generalize its correct classification with high probability, also when presented with samples that did not participate in the training set. This occurrence is known as *Generalization*. In this section, we propose a simple explanation for the described effect through the lens of our framework.

In section 2.1, we defined the term “global difference”. In the described notation, we encouraged the optimization process to find a classifier that maximizes the following expression:

$$\max_{g \in \mathcal{C}(\mathcal{I}, \mathcal{L})} \left\{ \sum_{i=1}^s \alpha_i \left| \forall_{1 \leq i \leq s} : \overline{\mathcal{P}_{\mathcal{B}_g}}(x'_i) = \alpha_i \overline{\mathcal{P}_{\mathcal{B}_f}}(x_i) \wedge 0 \leq \alpha_i \leq 1 \right. \right\}$$

Following that expression, we limit the value of α_i for each example i . Additionally, we assume that the decision boundary is as far as possible from the samples under the described conditions. This approach motivates the optimization process to involve a significant fraction of the training set in order to maximize the expression. For a representative training set, we get that the solutions for $F_\phi(\mathcal{I}, \mathcal{L})$ are directly related to the data distribution. Moreover, as described in section 2 and supported by section 6, the solutions for $F_\phi(\mathcal{I}, \mathcal{L})$ are based on a difference between the classes which is objectively unique through the lens of any classifier (and cannot be reduced twice). Therefore it is not exclusively related to the general distribution of the data but also to the gap between the individual distributions of the classes.

According to our definition of $F_\phi(\mathcal{I}, \mathcal{L})$, the generated classifier is the one that capable of performing the largest reduction of the gap between the class distributions using the most significant distributional differences among them. As explained, it would tend to find global properties expressed with a large fraction of the dataset to satisfy this requirement. Overfitting occurs when these (significant) differences are already considered, and to further increase the reduction, the decision boundary

starts to consider individual properties of specific distributional outliers. Therefore, regularization parameters and appropriate settings would only suppress the latest phase (of overfitting), where an overfitting solution is not inherently different from a generalizing solution but instead contains its global properties (and more).

Overall the detected solution separates the classes based on a distributional distinction. It implies that it would classify correctly, with high probability, other samples from the same distribution (for example, the test set).

6 Experimental results

This section will experimentally analyze the reduction of “global differences” based on Euclidean distance between the two classes.

6.1 Framework

We aim to iteratively project the training set onto the decision boundary as described in Algorithm 1. In each iteration, we train a randomly initialized network using the produced training set. Then we project the training set of the current iteration onto the generated decision boundary of the trained network to replace the old training set with a new one (the projected one). In each iteration, we train the network until the accuracy reaches 90% and all the samples are correctly classified. Additionally, for all iterations, we use an identical architecture and hyper-parameters to train the network, where the number of epochs might change (to reach the required accuracy).

Algorithm 1 Iterative projection

```

while iterations do
  model ← Net()
  model.train( $\mathcal{I}$ ,  $\mathcal{L}$ )
   $\mathcal{I} \leftarrow \text{Project}(\text{model}, \mathcal{I})$ 
end while

```

For each iteration, we measure the Euclidean distance between each sample of one class to its closest sample from the opposite class. Then, we compute the average of the resulted distances to indicate the distance between the classes for the current iteration. More formally, for a training set $(\{x_i\}_{i=1}^s, \{l_i\}_{i=1}^s)$ where $\forall 1 \leq i \leq s : x_i \in \mathbb{R}^n, l_i \in \{0, 1\}$, we compute for each iteration the following expression:

$$\frac{1}{s} \sum_{i=1}^s \min_{1 \leq j \leq s} \left\{ \|x_i - x_j\| \mid l_i \neq l_j \right\}$$

The analysis includes graphs with the described average distance for each iteration. Due to computational limitations, we randomly sampled a representative subset from the training set in each experiment (with equal classes ratio). We perform several experiments on a few different architectures for three datasets. The architectures for MNIST [19] and CIFAR10 [16] appear in Appendix A, and for ImageNet, [28] we use resnet18 [12] (using two output classes). We trained the networks using the negative log-likelihood loss function. The code was executed using *CUDA 10.1* and *Tesla K80 GPU*.

We show that the Euclidean distance between the classes is consistently reduced, which illustrates a trend rather than a random occurrence. We aim to illustrate the reduced gap between the class distributions. We use a Euclidean distance for this purpose while assuming that the gap between the class distributions is well-represented in the training set. In the experiments, we can see that the Euclidean distance reduced in earlier iterations is greater than the reductions that occur in later iterations. It implies the maximization property described in section 2.1 with “global difference” values and maximal distance of the examples from the decision boundary.

Following that, the “error bar” calculated using different seeds is unnecessary to our purpose in terms of the global trend as we initialized the network randomly in each iteration. Each segment in the presented graphs demonstrates similar behavior, even though we used hundreds of different initializations.

6.2 Results

In Figure 3, we demonstrate the analysis for 400 samples from the MNIST [19] dataset, using the categories “3” and “5”, Adam optimizer, and a learning rate of $1e - 4$.

Figure 4 uses 200 samples from the categories *Airplane* and *Automobile* of the CIFAR10 [16] dataset trained with Adam optimizer and a learning rate of $1e - 4$.

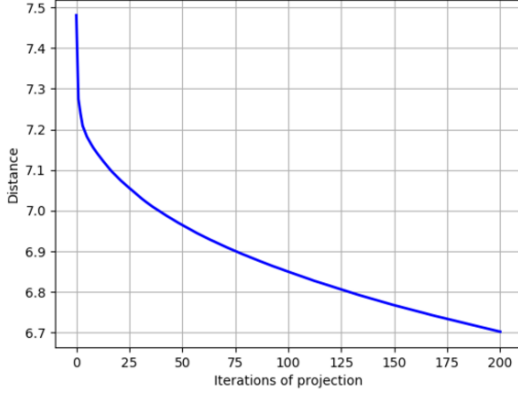


Figure 3: Global differences reduction process (MNIST).

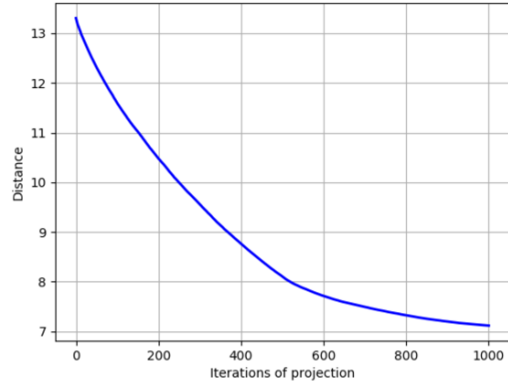


Figure 4: Global differences reduction process (CIFAR10).

Finally, Figure 5 shows the analysis by randomly sampling 100 samples from the categories *Goldfish* and *White shark* of the ImageNet [28] dataset. We used SGD with momentum and a learning rate of $1e - 3$.

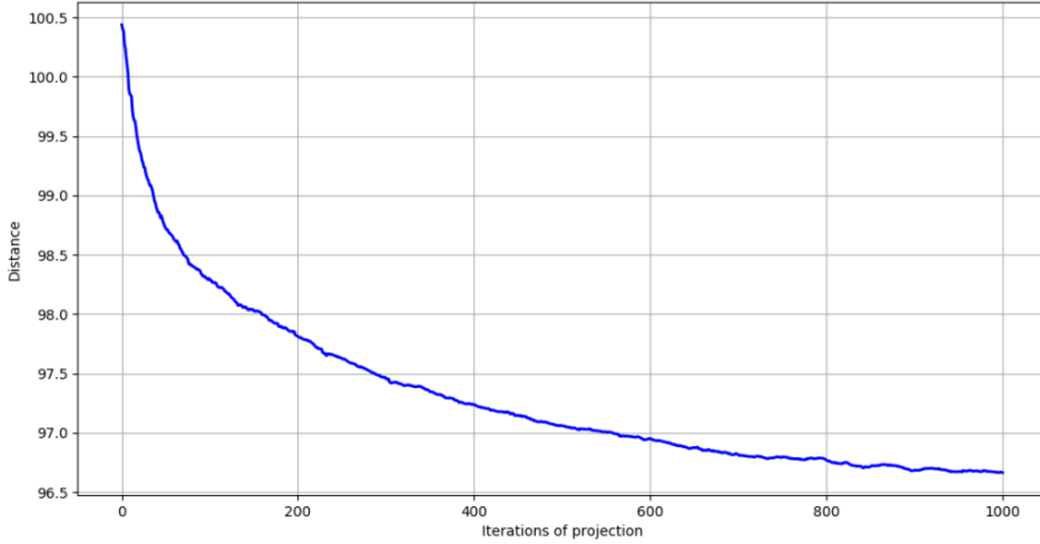


Figure 5: Global differences reduction process (ImageNet).

Additional experiments can be found in Appendix B.

6.3 Limitations

The experiments aim to show the properties we presented in section 2. We address these properties as distributional characteristics. On the other hand, we base our experiments on the consistent reduction

between the classes using Euclidean distance measurements. This detected gap aims to demonstrate the distributional difference between the class distributions.

Moreover, the experimental part states that earlier iterations show larger Euclidean distance reductions than the later ones. The claim is based on a global trend rather than individual observations, where for some specific iterations, we might get a smaller reduction than the one in the next iteration. We explain it by statistical deviations of the dataset, accuracy of only 90%, which might be expressed in a non-optimal decision boundary, and for later iterations, due to numerical errors and the limited expressiveness of the networks.

7 Conclusion

This work illustrates and formally presents an additional goal of the deep neural network’s optimization process. This reveal assists us with the proposed explanations for some of the mysterious phenomena of deep learning. In particular, we explain the occurrences “transferability” and “generalization” through the lens of the presented framework. Finally, we provide support for this hypothesis by conducting novel experiments. The experiments show that the projection of the samples on the decision boundary, reduces the difference between the class distributions.

References

- [1] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.
- [2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [3] Y. Cao and Q. Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in neural information processing systems*, 32, 2019.
- [4] C.-Y. Chuang, Y. Mroueh, K. Greenewald, A. Torralba, and S. Jegelka. Measuring generalization with optimal transport. *Advances in Neural Information Processing Systems*, 34, 2021.
- [5] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.
- [6] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- [8] I. Goodfellow. Stanford university school of engineering - lecture 16 | adversarial examples and adversarial training, 2017.
- [9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.
- [10] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.
- [11] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
- [12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- [13] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features, 2019.

- [14] K. Kawaguchi, L. P. Kaelbling, and Y. Bengio. Generalization in deep learning. *arXiv preprint arXiv:1710.05468*, 2017.
- [15] J. Kim, B.-K. Lee, and Y. M. Ro. Distilling robust and non-robust features in adversarial examples by information bottleneck. *Advances in Neural Information Processing Systems*, 34, 2021.
- [16] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Canadian Institute for Advanced Research, 2009.
- [17] J. Kukačka, V. Golkov, and D. Cremers. Regularization for deep learning: A taxonomy. *arXiv preprint arXiv:1710.10686*, 2017.
- [18] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.
- [19] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]*. Available: <http://yann.lecun.com/exdb/mnist>, 2, 2010.
- [20] Y. Li and Y. Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *Advances in Neural Information Processing Systems*, 31, 2018.
- [21] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks, 2017.
- [22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.
- [23] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.
- [24] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020.
- [25] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. *Advances in neural information processing systems*, 30, 2017.
- [26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- [27] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
- [29] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- [30] M. Tan, R. Pang, and Q. V. Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.
- [31] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34, 2021.
- [32] F. Tramèr, N. Carlini, W. Brendel, and A. Madry. On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems*, 33:1633–1645, 2020.
- [33] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. The space of transferable adversarial examples, 2017.
- [34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- [35] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.

- [36] X. Ying. An overview of overfitting and its solutions. In *Journal of Physics: Conference Series*, volume 1168, page 022022. IOP Publishing, 2019.
- [37] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021.

A Neural network architectures

Table 1: Neural network architectures used in this work.

MNIST	CIFAR10
FC(784,500) + ReLU	Conv(128,3,3) + BN + ReLU
FC(500,256) + ReLU	Conv(128,3,3) + BN + ReLU
FC(256,128) + ReLU	Conv(256,3,3) + BN + ReLU
FC(128,32) + ReLU	MaxPool(2,2)
FC(32,2)	FC(1024,2)

The architectures used in this work appear in Table 1. Conv: convolutional layer, FC: fully-connected layer, BN: batch normalization.

B Additional experiments

We provide additional experiments with the same analysis for different categories. In Figures 6,7,8, we analyze the classification of “0” versus “1” (MNIST), *Cat* versus *Dog* (CIFAR10), and *Hen* versus *Goose* (ImageNet), respectively.

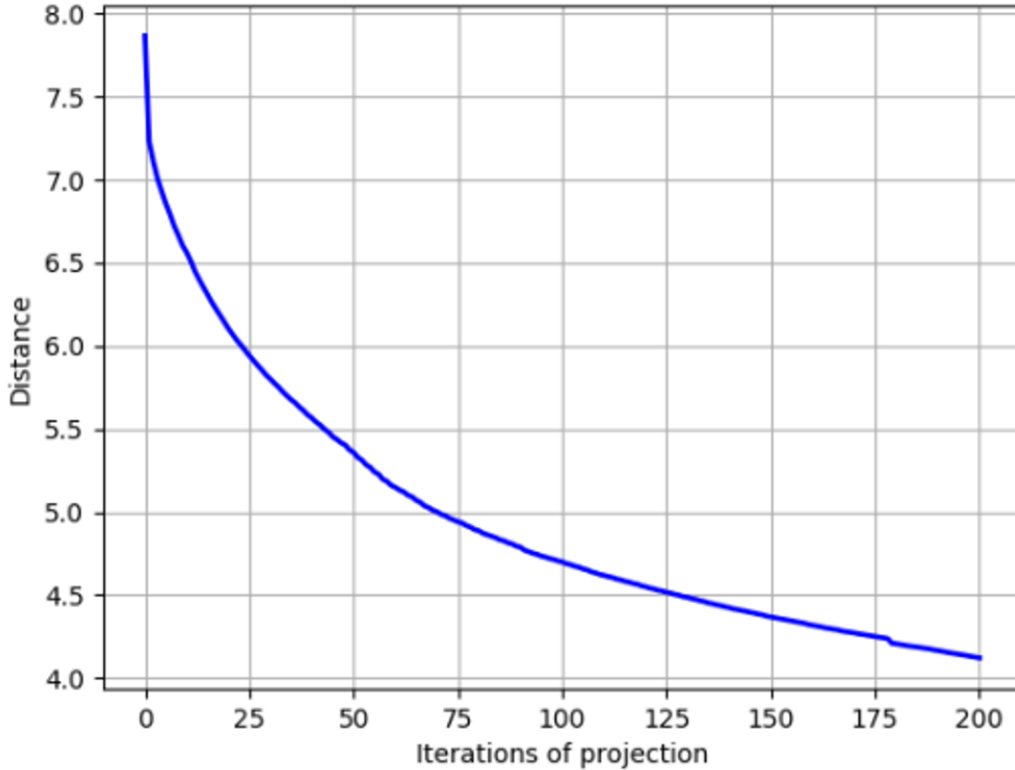


Figure 6: Global differences reduction process (“0” versus “1” of MNIST).

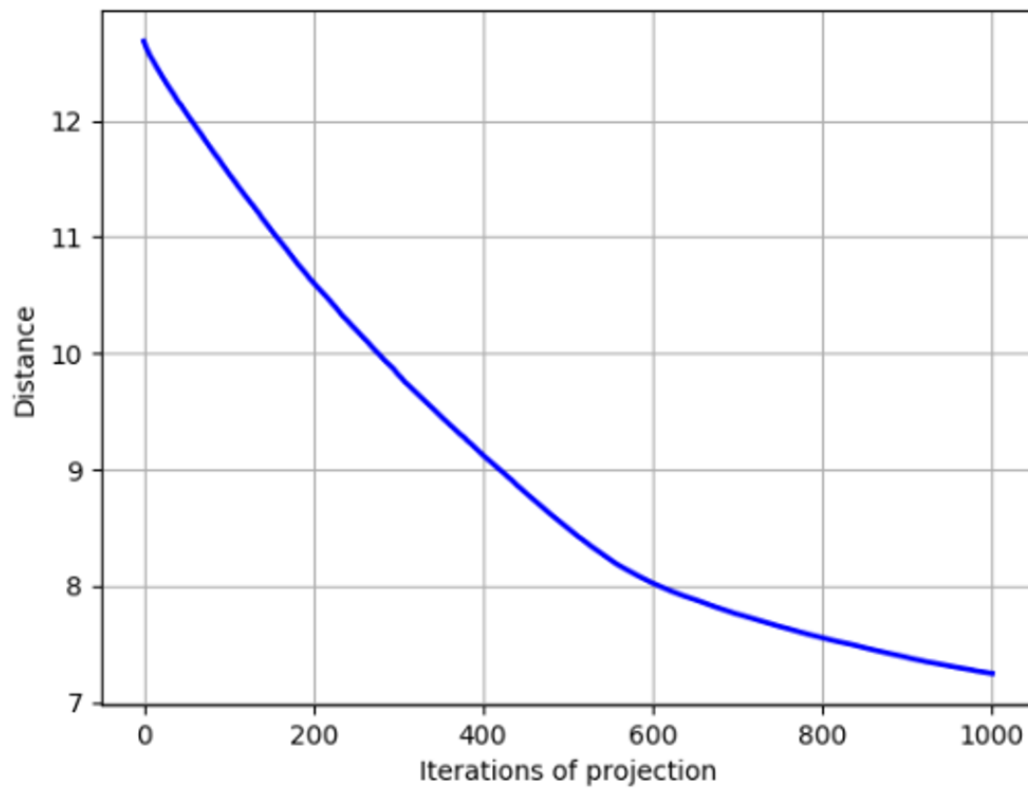


Figure 7: Global differences reduction process (*Cat* versus *Dog* of CIFAR10).

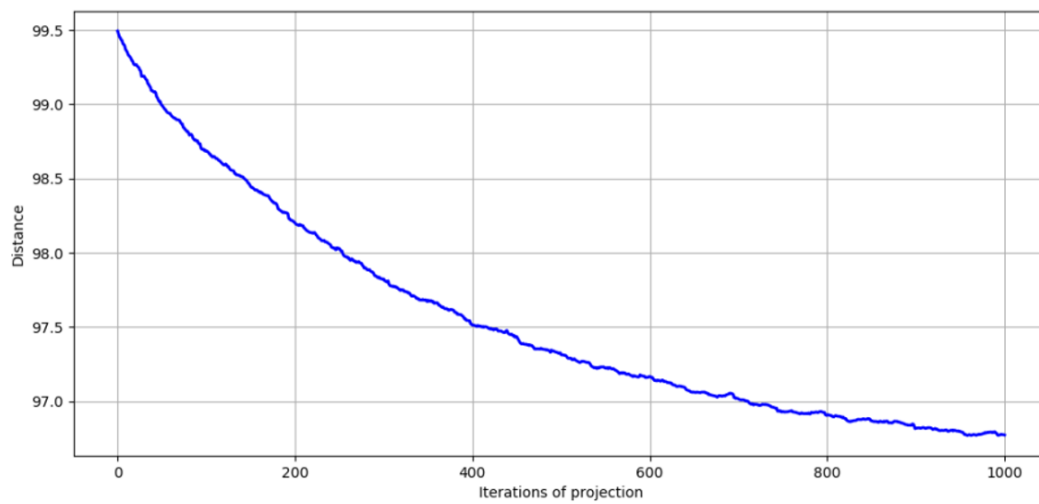


Figure 8: Global differences reduction process (*Hen* versus *Goose* of ImageNet).

C Solution uniqueness for non-symmetrical datasets

Claim 1: For a pair of possible independent solutions $f, g \in F_\phi(\mathcal{I}, \mathcal{L})$ such that:

$$\forall x \in \mathcal{I} : \overline{\mathcal{P}_{\mathcal{B}_f}}(x) \perp \overline{\mathcal{P}_{\mathcal{B}_g}}(x)$$

the dataset \mathcal{I} is symmetrical.

Proof: According to prior definitions $f, g \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}$. Therefore, for any $x_i, x_j \in \mathcal{I}$ with the corresponding labels $l_i, l_j \in \mathcal{L}$ such that $l_i \neq l_j$, the following must be applied:

$$\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| \leq \|x_i - x_j\|$$

$$\|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\| \leq \|x_i - x_j\|$$

Assume that the dataset is not symmetrical. Equality for f could appear only when:

$$\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i) + \overline{\mathcal{P}_{\mathcal{B}_f}}(x_j) = x_i - x_j$$

and the same for g . Following the given assumption that $\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i) \perp \overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)$ and $\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j) \perp \overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)$, at least one of the above expressions is a strict inequality. Therefore we get:

$$\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\| < \|x_i - x_j\| + \|x_i - x_j\|$$

$$\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\| < 2\|x_i - x_j\|$$

$$\frac{1}{2}(\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|) + \frac{1}{2}(\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|) < \|x_i - x_j\|$$

According to the triangle inequality:

$$\|\frac{1}{2}(\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i) + \overline{\mathcal{P}_{\mathcal{B}_g}}(x_i))\| + \|\frac{1}{2}(\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j) + \overline{\mathcal{P}_{\mathcal{B}_g}}(x_j))\| < \|x_i - x_j\|$$

Following that, and the fact that $f, g \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}$, we could generate a new classifier $h \in \mathcal{C}_{(\mathcal{I}, \mathcal{L})}$ such that for any $x \in \mathcal{I}$ we define $\mathcal{P}_{\mathcal{B}_h}(x) = x + \frac{1}{2}(\overline{\mathcal{P}_{\mathcal{B}_f}}(x) + \overline{\mathcal{P}_{\mathcal{B}_g}}(x))$.

In addition, Claim 2 provides a proof for the following:

$$\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_h}}(x_i)\| > \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| = \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|$$

We also assume that the “global difference” value of f and g is minimized. The classifier h has the following projection $\mathcal{P}_{\mathcal{B}_h}(x) = x + \frac{1}{2}(\overline{\mathcal{P}_{\mathcal{B}_f}}(x) + \overline{\mathcal{P}_{\mathcal{B}_g}}(x))$, with a direction that follows the directions of $\mathcal{P}_{\mathcal{B}_f}$ and $\mathcal{P}_{\mathcal{B}_g}$ for the specific point. Therefore, increasing the norm of the projection vectors in the same direction (using a larger scalar factor than $\frac{1}{2}$) will have to reach a minimized value of “global difference” as requested.

Overall, unless the dataset \mathcal{I} is symmetrical, we could generate $h \in F_\phi(\mathcal{I}, \mathcal{L})$ with $\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_h}}(x_i)\| >$

$\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| = \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|$ in contradiction. ■

Claim 2: $\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_h}}(x_i)\| > \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| = \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|$

Proof: Based on the maximization property stated in section 2.1:

$$\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| = \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|$$

According to our definition: $\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_h}}(x_i)\| = \prod_{i=1}^s \|\frac{1}{2}(\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i) + \overline{\mathcal{P}_{\mathcal{B}_g}}(x_i))\|$

After multiplying the elements, each operand in the summation has the following structure:

$$\frac{1}{2^s} \prod_{i=1}^s \|p_i\|$$

where $p_i \in \{\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i), \overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\}$. For each operand like this, we also have the complement operand with $p_i^c \in \{\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i), \overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\}$ such that $p_i \neq p_i^c$ for any i .

Assume that for a specific operand, T is the collection of indices where $p_i = \overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)$. In that case, we could express the summation of the complement operands in the following manner:

$$\begin{aligned} & \frac{1}{2^s} \prod_{i=1}^s \|p_i\| + \frac{1}{2^s} \prod_{i=1}^s \|p_i^c\| = \\ & \frac{1}{2^s} \left(\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| \cdot \frac{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|}{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\|} \right) + \frac{1}{2^s} \left(\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\| \cdot \frac{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\|}{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|} \right) = \\ & \frac{1}{2^s} \left(\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| \cdot \frac{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|}{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\|} \right) + \frac{1}{2^s} \left(\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| \cdot \frac{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\|}{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|} \right) = \\ & \frac{1}{2^s} \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| \left(\frac{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|}{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\|} + \frac{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\|}{\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|} \right) \end{aligned}$$

For $a, b \in \mathbb{R}$ where $a, b > 0$:

$$\frac{a}{b} + \frac{b}{a} \geq 2$$

and we get equality only when $a = b$. Therefore, if $\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| \neq \prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|$, we necessarily will get that:

$$\frac{1}{2^s} \prod_{i=1}^s \|p_i\| + \frac{1}{2^s} \prod_{i=1}^s \|p_i^c\| > \frac{2}{2^s} \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\|$$

Applying that for all the operands will generate the following inequality:

$$\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_h}}(x_i)\| > \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\|$$

The same process could be applied to g .

In the case where $\prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| = \prod_{j \in T} \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|$ and in particular for the cases where $|T| = 1$, we get the following:

$$\forall_{1 \leq i \leq s} : \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| = \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|$$

Combined with the earlier statement:

$$\frac{1}{2}(\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|) + \frac{1}{2}(\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| + \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_j)\|) < \|x_i - x_j\|$$

we get that for any i and j :

$$\|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| + \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_j)\| < \|x_i - x_j\|$$

In that case, we could generate $\alpha \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x)\|$ where $\alpha > 1$ for at least one example while preserving the other properties and projection vectors. This is a contradiction to the maximization property of f . The same could have been claimed for g . Therefore, we got that $\prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_h}}(x_i)\| > \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_f}}(x_i)\| = \prod_{i=1}^s \|\overline{\mathcal{P}_{\mathcal{B}_g}}(x_i)\|$. ■