ON THE EFFECTIVENESS OF DEEP ENSEMBLES FOR SMALL DATA TASKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks represent the gold standard for image classification and many other tasks. However, they usually need large amounts of data to reach superior performance. In this work, we focus on classification problems with a few labeled examples per class and improve sample efficiency in the low data regime by using an ensemble of relatively small deep networks. For the first time, our work broadly studies the existing concept of neural ensembling in small data domains. We compare different ensemble configurations by varying the complexity of base members given a total fixed computational budget. We found that the state of the art is improved by keeping low the complexity of single models and increasing the ensemble dimension. Furthermore, we investigate the effectiveness of different losses and show that their choice should be made considering different factors. Finally, according to our results, our proposed ensemble resulted to be the most robust configuration considering variation in dataset type and size. Our findings are empirically validated on popular datasets after an extensive number of experiments.

1 INTRODUCTION

The computer vision field has been revolutionized by the advent of deep learning (DL) [Y. LeCun & Hinton, 2015]. The convolutional neural network (CNN) is the most popular DL model for visual learning tasks thanks to its ability to automatically learn general features via gradient-based optimization algorithms. However, the cost to reach high recognition performances involves the collection and labeling of large quantities of images. This requirement can not always be fulfilled since it may happen that collecting images is extremely expensive or not possible at all. For instance, in the medical field, high-quality annotations by radiology experts are often costly and not manageable at large scales [Litjens et al., 2017]. Different approaches have been proposed by the research community to mitigate the necessity of training data, tackling the problem from different perspectives.

Transfer learning aims at learning representations from one domain and *transfer* the learned knowledge (e.g. pre-trained network) to a target domain [Bengio, 2012], [Tan et al., 2018]. Similarly, few-shot learning uses a *base set* of labelled pairs to generalize from a small *support set* of target classes [Vanschoren, 2018]. Both approaches have obtained interesting results and especially transfer learning is widely used. Still, they suffer from the need of collecting a pool of annotated images and the *source* and *target* domains must be somewhat related. Self-supervised learning is another approach that is trying to reduce the demand for annotations. Usually, a large set of images is used to teach how to solve a *pretext* task to a CNN [Jing & Tian, 2020]. This task does not require explicit human annotations and is conceived to teach general visual features that can be transferred to the *downstream* task. In this manner, costly human annotations are not needed but the challenge of collecting many images remains.

The previously cited research directions, in a way or another, still rely on many samples/annotations. Our grand goal is to develop learning algorithms that are as sample-efficient as the human visual system. In other words, we aim to solve a classification problem with only a limited amount of labeled examples. Due to the great difficulty, this problem is still largely unsolved and hardly experimented.

In this work, we propose the use of neural ensembles composed of smaller networks to tackle the problem of learning from a small sample and show the superiority of such approach. Similarly to what has been done in recent works [Arora et al., 2020], [Barz & Denzler, 2020], we benchmark the

approaches by varying the number of data points in the training sample while keeping it low with respect to the current standards of computer vision datasets. It has been shown that large CNNs can handle overfitting and generalize well even if they are severely over-parametrized [Kawaguchi et al., 2017], [Neyshabur et al., 2018]. A recent study have also empirically shown that such behaviour might also be valid in the case of tiny datasets, making large nets a viabile choice even when the training sample is limited [Bornschein et al., 2020]. A well known technique to reduce model variance is to average predictions from a set of weak learners (e.g. random forests [Breiman, 2001]). An ensemble of low-bias decorrelated learners, combined with randomized inputs and prediction averaging, generally mitigates overfitting. Neural network ensembles have been widely used in the past for both regression and classification problems [Hansen & Salamon, 1990], [Giacinto & Roli, 2001], [Lee et al., 2015], [Li et al., 2018]. However, state-of-the-art classification pipelines rarely make use of ensembling techniques due to the large dimensions of modern CNNs. Such an approach would become prohibitive from a computational point of view. On the other side, for domains in which training data is not abundant, ensembles of relatively small neural networks become highly attractive thanks to the aforementioned factors. Furthermore, ensemble approaches have also been shown to be good estimators of predictive uncertainty, an important tool for any predictive system [Lakshminarayanan et al., 2017].

Motivated by the nice variance-reduction properties of ensembles, we systematically study their effectiveness in small data tasks by a) fixing a computational cost for the ensemble and b) evaluating it varying the complexity of the members. The obtained results confirm our intuition and show that running a large ensemble is advantageous in terms of accuracy and overall training time over smaller groups with higher complexity of the base members.

Furthermore, all networks are trained with two losses: 1) the widely used cross-entropy and 2) the recently proposed cosine loss [Barz & Denzler, 2020]. Despite the latter loss has been specifically proposed to tackle small data problems, we have noticed some cases in which the cross-entropy still gives higher accuracy. The combined factors of model complexity and the amount of available data seems to influence the final performance.

Finally, we show that the most robust ensemble configuration concerning different datasets and sample sizes is the one that we are proposing, that is, a large ensemble with networks of low complexity trained with cross-entropy loss.

In summary, the contributions of our work are the following: *i*) we systematically study the use of neural ensembles in the small sample domain; *ii*) we show that a large neural ensemble of small scale networks outperforms its computationally equivalent competitor with larger models; *iii*) we compare through a large set of experiments state-of-the-art losses showing that their performance depends on diverse factors and we provide a way of choosing the right configuration depending on the situation; *iii*) we show that our proposed ensemble is the most robust configuration considering the variation of dataset size and dimension.

2 Related Work

In this section, we first present a summary of the main techniques proposed in the literature to learn from a small sample. Secondly, we give an overview of neural ensembling techniques.

Learning from a small sample is extremely challenging and, for this reason, largely unsolved. As previously said, few works have tried to tackle the problem of training DL architectures with a small number of samples due to the difficulty of preventing overfitting.
We start by mentioning a series of works that focused on the classification of vector data and mainly used the UCI Machine Learning Repository as a benchmark. In [Fernández-Delgado et al., 2014] the authors have shown the superiority of random forests over a large set of classifiers including feed-forward networks. Later, [Olson et al., 2018] used a linear program to empirically decompose fitted neural networks into ensembles of low-bias sub-networks. They showed that these sub-networks were relatively uncorrelated which lead to an internal regularization process similar to what happens in random forests, obtaining comparable results. More recently, [Arora et al., 2020] proposed the use of neural tangent kernel (NTK) architectures in low data tasks and obtained significant improvements over all previously mentioned classifiers.
All previous works did not test CNNs since inputs were not images. When the input dimensionality

increases, the classification task inevitably becomes more complex. For this reason, a straightforward approach to improve generalization is to implement techniques that try to synthesize new images through different transformations (e.g. data augmentation [Shorten & Khoshgoftaar, 2019]). Some previous knowledge regarding the problem at hand might turn to be useful in some cases [Hu et al., 2017]. However, this makes data augmentation techniques not always generalizable to all possible image classification domains. It has also been proposed to train generative models (e.g. GANs) to increase the dataset size and consequently, performance [Liu et al., 2019]. Generating new images to improve performance is extremely attractive and effective. Yet, training a generative model might be computationally intensive or present severe challenges in the small sample domain. In our work, we use standard data augmentation and do not focus on approaches that improve image synthesis. Indeed, we assume that any state-of-art data augmentation/generation technique could be coupled with our trained networks to further improve testing accuracy.
[Rueda-Plata et al., 2015] suggested to train CNNs with a greedy layer-wise method, analogous to that used in unsupervised deep networks and showed that their method could learn more interpretable and cleaner visual features. [Barz & Denzler, 2020] proposed the use of the cosine loss to prevent overfitting claiming that the L2 normalization involved in the cosine loss is a strong, hyper-parameter-free regularizer when the availability of samples is scarce. They obtained the best results on fine-grained datasets that have between 20 and 80 samples per class. On the other hand, [Arora et al., 2020] performed experiments with convolutional neural tangent kernel (CNTK) networks on small CIFAR-10 showing the superiority of CNTK in comparison to a ResNet-34 model. Finally, [Bornschein et al., 2020] studied the generalization performance of deep networks as the size of the training set varies. They found out that even larger networks can handle overfitting and obtain comparable or better results than smaller nets if properly optimized and calibrated.

Ensembles of neural networks are widely successful for improving both the accuracy and predictive uncertainty of very different predictive tasks, ranging from supervised learning [Hansen & Salamon, 1990], [Lee et al., 2015] to model-based reinforcement learning [Deisenroth & Rasmussen, 2011]. The operational pipeline of neural ensembles consists of training multiple networks individually and then averaging their predictions obtaining, in the end, better results than single estimates [Opitz & Maclin, 1999]. The main reason behind this success is caused by the diversity of the solution found by each independent optimization process. Although ensemble members usually score similar testing performance, given the same input, they may disagree on the prediction, since they converged to different local minima [Garipov et al., 2018]. By combing the predictions, they benefit from diversity and decorrelated errors [Bishop, 1994]. Furthermore, similar to Bayesian methods [Gal & Ghahramani, 2016], they can estimate predictive uncertainty and provide more calibrated prediction probabilities [Lakshminarayanan et al., 2017]. For this reason, they can score the best performance on out-of-distribution uncertainty benchmarks. The main drawback of standard ensembles regards their computational cost that generally grows linearly with the ensemble dimension. Some works have tried to mitigate such problem in the field of continual and supervised learning [Wen et al., 2020], [Wasay et al., 2020].
However, since in this work we are running neural ensembles in low data regimes, the training time is relatively low. Neural ensembling has also been occasionally applied to specific problems and applications with small data [Veni & Rani, 2014], [Shaikhina & Khovanova, 2017], [Phung et al., 2019], [Han & Jeong, 2020].

Our article contributes to the field of small data and deep ensembles by proposing a structured study with state-of-the-art models on sub-sampled versions of popular datasets. To the best of our knowledge, we are the first ones to perform such an analysis and measure the actual effectiveness of deep ensembles with small data. The proposed method can be exploited for similar classification problems with tiny datasets.

3 Proposed Ensemble Method

We are dealing with a problem in which we assume to have a dataset of samples and corresponding labels $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1) \ldots, (\mathbf{x}_s, \mathbf{y}_s)\}$ where each label $\mathbf{y}$ is a one-hot vector that is encoding K different classes. Furthermore, we are assuming that $\mathcal{D}$ is balanced and that the number of samples

per class is equal to N and relatively small.[1] In general, $\mathbf{x}$ might be a vector of any size. In our work, we are tackling a classification problem where each input sample is an image, therefore, $\mathbf{x} \in \mathbb{R}^{H \times W \times D}$. The objective is to learn the input-output mapping $\mathbf{y} = f_\theta(\mathbf{x})$ with $f(\cdot)$ representing a function approximator and θ its parameters.
More precisely, since we are implementing a weighted average ensemble, we define a set $\mathcal{M} = \{g_{\theta_m}(\cdot) : m = 1, \dots M\}$ of M functions. In our particular implementation, the ensemble is homogeneous, therefore all functions g are equal. Note that during our analysis, we will evaluate different ensembles varying the complexity of the base members g. In other words, we can define $\mathcal{M}$ as a function of g respecting the constraint $\mathcal{M}(g^{(i)}) \sim \mathcal{M}(g^{(j)})$ for any $g^{(i)}$ and $g^{(j)}$ in terms of computational cost. At testing time, given an input sample $\mathbf{x}$, the ensemble prediction is going to be the expected value of the predictions coming from each member in $\mathcal{M}$ scaled by a function ϕ depending on the type of the loss used:

$$\mathbf{y} = f_\theta(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^{M} \phi\left(g_{\theta_m}(\mathbf{x})\right)$$

The weights θ_m are randomly initialized and ensemble members update their parameters receiving the same input batch. Throughout the optimization process, the members are independent, therefore each network will minimize its own loss $\mathcal{L}(\mathbf{y}, g_{\theta_m}(\mathbf{x}))$.
In this work, we implement two loss functions. The first one is the popular categorical cross-entropy which is computed as:

$$\mathcal{L}_{xe} = -\mathbf{y} \cdot \log\left(\frac{\exp\left(g_{\theta_m}(\mathbf{x})\right)}{||\exp\left(g_{\theta_m}(\mathbf{x})\right)||_1}\right)$$

where the exponential and logarithm are applied element-wise. In this case, ϕ corresponds to the soft-max. The second one is the cosine loss, that was lately proposed to improve the generalization of deep networks with small datasets [Barz & Denzler, 2020]. The cosine loss simply computes the dot product among the one-hot encoded categories and the L2 normalized predictions as:

$$\mathcal{L}_{cosine} = \mathbf{y} \cdot \frac{g_{\theta_m}(\mathbf{x})}{||g_{\theta_m}(\mathbf{x})||_2}$$

In this other case, ϕ corresponds to the L2 regularization. The improvements reported in [Barz & Denzler, 2020] were explained by the L2 regularization that prevents the network from focusing on the magnitudes and moves the attention to vector directions. Different studies have shown that directions maintain more information in feature space [Husain & Bober, 2016], [Zhe et al., 2019]. Moreover, differently from the cosine loss, the cross-entropy is steeper, not symmetric, and unbounded. These three characteristics might be indeed problematic when facing very small datasets since noisy gradient updates may cause instability and overfitting.

To sum up, we are going to train multiple deep ensembles varying the complexity of their base members but keeping the overall computational cost fixed. The networks in the groups are homogeneous and trained with the previously cited losses. The objective is two-fold: 1) investigate the variance reduction capabilities of the ensembles and 2) compare the losses in a wide variety of configurations.

4 EXPERIMENTS

In this section, we describe the experiments of our work. First, we depict the used datasets and models. Then, we outline the experimental policy that we have followed. Finally, we show the comparative results.

[1] It is difficult to quantify a small dataset, because it is not an objective measure, but depends on multiple factors. More details are given in Section 4.3.

4.1 DATASETS

We perform our study on two popular benchmarks for image classification, CIFAR-10 and CIFAR-100 [Krizhevsky et al., 2009].
CIFAR-10 is an established computer vision benchmark consisting of color images coming from 10 different classes of objects and animals. The dataset has originally $50,000$ training images and $10,000$ testing images with both sets balanced. The input images are of size 32×32. State-of-the-art results on CIFAR-10 are extremely high. However, we will show that reducing the sample size will make it a much harder task.
CIFAR-100 is the more complex version of CIFAR-10 since is composed of 100 classes and contains the same number of training and testing samples. The training/testing splits and input dimensionalities are equal to the ones of CIFAR-10. Note that since CIFAR-100 has ten times more classes, the number of training images per class is equal to 500 instead of 5000.

4.2 COMPARED MODELS

To build our ensembles we considered the ResNet-v1 family that was originally proposed in [He et al., 2016]. We chose ResNet architecture for two main reasons: 1) ResNet scores high results on both CIFAR-10 and CIFAR-100 and 2) it is easy to change the computational complexity of the network by varying the original n parameter that controls network depth.
We decided to measure the models' computational complexity as the number of floating-point operations (FLOPs). This is a standard metric that measures algorithmic/model complexity.
The available computational budget was set to be roughly close to the FLOPs of a single ResNet-110 that is obtained by setting $n = 18$ in the original formulation. Among the tested networks on CIFAR-10, ResNet-110 was the one that scored the least testing error [He et al., 2016]. Therefore, our baseline is a single ResNet-110. We compare such architecture with a large ensemble of low-complexity members and a medium-size ensemble with medium-complexity networks. The first one corresponds to 20 ResNet-8 while the second one to 5 ResNet-20. Such networks are obtained by setting, respectively, $n = 1$ and $n = 3$. The amount of FLOPs for all three ensembles on the two datasets is available in Appendix A. Since numbers are in the same order of magnitude, these three ensembles are directly comparable.
We will report the comparison among our ensembles trained with the two losses as well. Moreover, since ensemble optimization implies training multiple single networks independently, we will also analyze them to study the effect of the loss on architectures of different complexity. Therefore, despite not being computationally fair, we will compare the single ResNet-110 with the single ResNet-8 and ResNet-20 trained with the cross-entropy and cosine losses.

4.3 EXPERIMENTAL PROCEDURE

Sub-sampling our training sets is necessary since we are interested in benchmarking the capabilities of our ensembles when the sample size is limited. To this end, we sub-sample the training sets of the chosen datasets to match our specifications. More in detail, the number of samples per class is varied in the set $\{10, 50, 100, 250\}$. A similar procedure was already proposed in [Barz & Denzler, 2020]. Note that the test datasets are not changed, and remain fixed throughout all the evaluations. To ensure consistency of the results, we perform 5 runs for each sub-sampled version of each dataset.
We use the optimization schedule that was used for training ResNets on CIFAR-10 in the original paper [He et al., 2016] with slight modifications. For all the details, see Appendix B.
When we report a single number from a testing run, we have considered the best performance achieved after any epoch. Consequently, means and standard deviations, when computed, consider those results.

4.4 ENSEMBLES COMPARISON

First, we will analyze the effectiveness of the medium size and large size ensemble comparing it to the single more complex network. The results for this analysis are shown in Figure 1 for both CIFAR-10 and CIFAR-100. In each subplot, we report the average testing accuracy profile obtained from different runs.
On CIFAR-10, it is clear that ResNet-8 and ResNet-20 ensembles always outperform the single

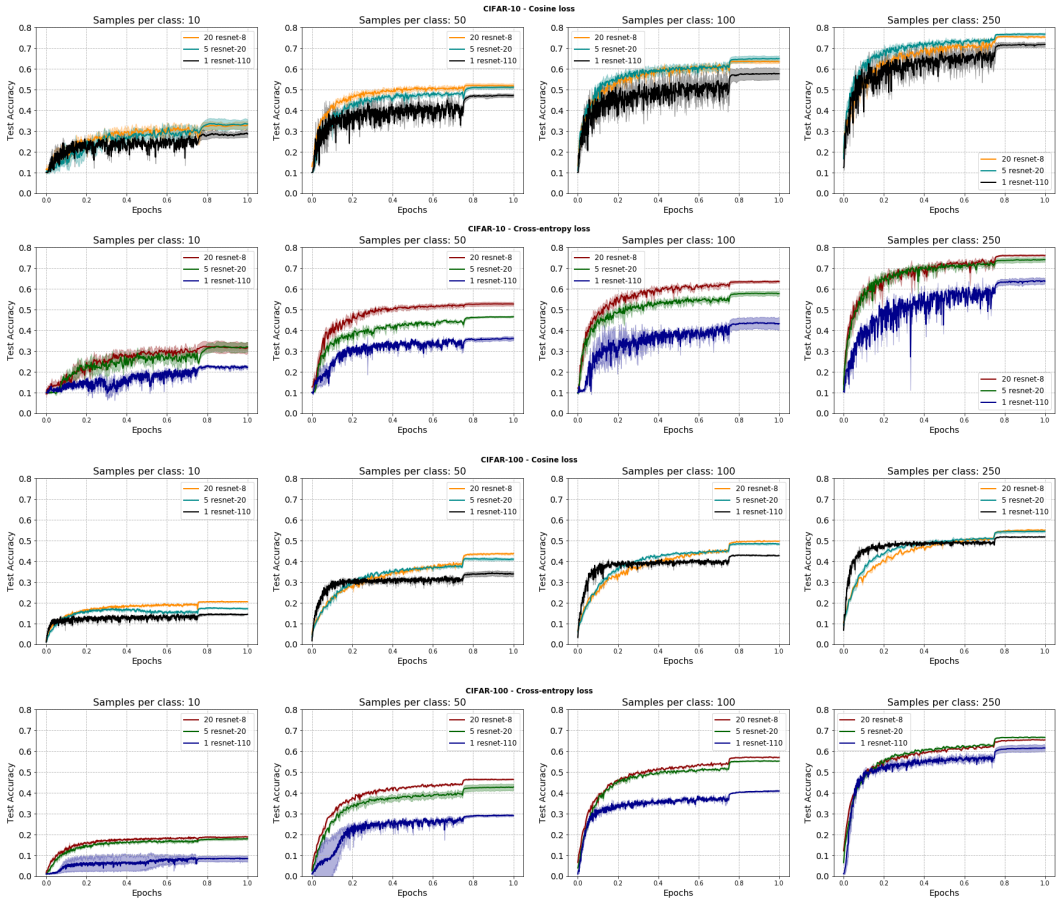

Figure 1: Comparison between 20 ResNet-8, 5 ResNet-20 and a single ResNet-110 divided per dataset and loss. The average testing profile obtained from different runs is reported in each subplot. To improve clarity, profiles are plotted considering the percentage of total epochs. For knowing the respective number of epochs, refer to Appendix B.

ResNet-110. The improvement is higher with the cross-entropy loss but also noticeable with the cosine loss. The ResNet-8 ensemble trained with cross-entropy outperforms ResNet-110 by a maximum of 18.21% and a minimum of 8.98% with, respectively, 10 and 100 samples per class. When using the cosine loss, the maximum and minimum improvements are respectively 7.47% and 4.56% with $N = 100$ and $N = 250$. Moreover, we also note that with the cross-entropy loss, 20 ResNet-8 are better than or comparable to 5 ResNet-20. This is also true if networks are trained with the cosine loss.

On CIFAR-100, results are in line with the ones obtained on CIFAR-10. We note that the higher improvements over the single ResNet-110 are again scored when the ensemble is trained with the cross-entropy loss obtaining a maximum average increase of 17.12% and a minimum of 4.58% with $N = 50$ and $N = 250$. Maximum and minimum improvements with the cosine loss correspond to 9.11% and 3.31% when models are trained with 50 and 250 samples per class. To summarize, we analyzed 16 different configurations derived from the variation of losses and the dimension of the training sample. In all cases 20 ResNet-8 and 5 ResNet-20 outperformed ResNet-110. These results suggest that, given a fixed computational budget, is more convenient to train an ensemble of smaller networks rather than a single one.

4.5 Losses comparison

To provide a complete picture regarding the effectiveness of the two losses, we evaluate them considering both the already presented ensembles and their single members.

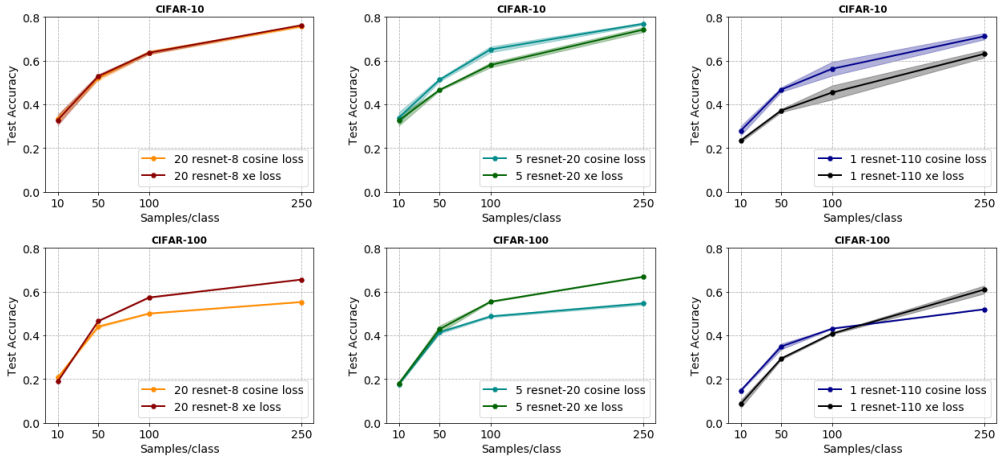

Figure 2: Comparison between the two losses considering the ensembles.

We show the average maximum testing accuracy of each ensemble trained with the two losses in Figure 2 for both CIFAR-10 and CIFAR-100. Moreover, we show the comparison considering single networks in Appendix C.

Looking at both figures, we can note that considerations on the ensemble approaches are directly transferable to the single network case. The relation between the results of the two losses does not change. The only difference regards the highest testing accuracy obtained with the ensemble methods.

From the plots, we observe that model complexity and problem dimension seem to play a crucial role in the loss choice. Indeed, for the least complex ResNet-8, the cross-entropy is slightly better than the cosine loss on CIFAR-10 and substantially better on CIFAR-100. If we increase model complexity and analyze the ResNet-20 results, we note that the cosine loss outperforms the cross-entropy on CIFAR-10 but the opposite result is obtained on CIFAR-100. Finally, the cosine loss is clearly more convenient with ResNet-110 on CIFAR-10 and CIFAR-100 up to 100 samples per class. On the other hand, when training on CIFAR-100 with $N = 250$, the cross-entropy increases its effectiveness.
The performance of ResNet-110 on CIFAR-100 obtained here follows the same behavior that was shown in Figure 2 of [Barz & Denzler, 2020]. Although the results are not directly comparable since they used a different set-up, their cross-entropy network outperformed the cosine one after 200 samples per class.[2] These findings are in line with the argument regarding the regularization capabilities of the L2 normalization in the cosine loss. Reasonably, the latter seems to increase its effectiveness when the training sample is tiny and the model more complex. However, even if we consider a not computationally fair comparison, we note from Figure 4 that less complex networks trained with cross-entropy may be more effective than larger networks trained with the cosine loss (e.g. CIFAR-100 with 100 samples per class).
Summarizing, according to the results of our study, cosine loss is more convenient if the total amount of data is scarce and the complexity of the network is medium-high. On the other hand, the cross-entropy is better with larger training sets and with medium-low network complexities.[3]

4.6 Evaluation of the best ensemble

In this section, after having evaluated the comparisons between ensembles and losses, we provide details regarding the best choice that we could make in terms of both models and losses. For this reason, we show in Figure 3 the testing accuracy profiles of all three ensembles with both losses in each subplot.

[2]They trained ResNet-110 with twice as many channels per-layer, used SGD with Warm Restarts, and fine-tuned the learning rates.

[3]Fixing the number of samples per class causes the CIFAR-100 training dataset to be ten times larger than the one of CIFAR-10.

Confirming the analysis described in Section 4.2, the single ResNet-110 is always outperformed by the ensembles. Moving to the ensembles, we note that when the sample size is extremely limited (i.e. $N = 10$), they obtain comparable testing accuracy in CIFAR-10 while on CIFAR-100, the best ensemble is 20 ResNet-8 with cosine loss, followed by cross-entropy with an average testing accuracy of 20.73% and 18.92%. With 50 samples per class, 20 ResNet-8 with cross-entropy score the maximum accuracy on both datasets, 52.88% and 46.56%. The ResNet-8 ensemble with cross-entropy confirms to be the best model also on CIFAR-100 and 100 samples per class beating 5 ResNet-20 with cross-entropy by $\sim 2\%$. On the other hand, on CIFAR-10 it ranks second after 5 ResNet-20 and cosine loss by 1.4%. Finally, with $N = 250$, we note a similar trend to the case of 100 samples per class with the improvement of 5 ResNet-20 over 20 ResNet-8, both trained with cross-entropy ($+1.3\%$).
Summing up, we observe from this analysis that 20 ResNet-8 trained with cross-entropy is the most robust method considering both sample-size variation (10 to 250 samples per class) and dataset change (CIFAR-10 and CIFAR-100). We believe that such robustness mainly derives from two factors. The first one is the right balance between model complexity and dataset dimension. In Section 4.5, we have seen it to be particularly important for the cross-entropy loss. Secondly, a large ensemble brings the well-known variance reduction properties that are even more important with limited data.

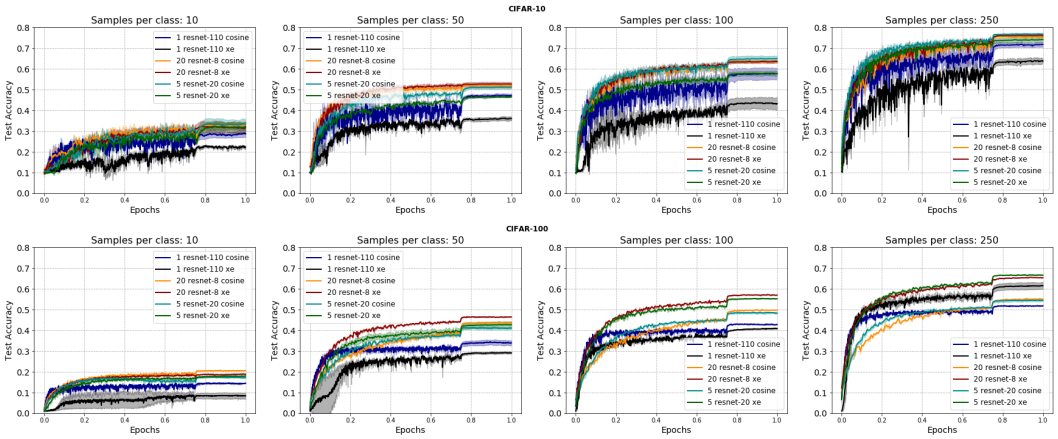

Figure 3: Plots of all our ensembles considering the two loss functions for both datasets. To improve clarity, profiles are plotted considering the percentage of total epochs. For knowing the respective number of epochs, refer to Appendix B.

5 Conclusions

In this paper, we have investigated the capability of deep ensembles on classification problems with limited data. Through comparing ensembles given a fixed computational budget, our results show that: a) an ensemble of twenty ResNet-8 is better than or comparable to five ResNet-20; b) both ensembles outperforms a single ResNet-110 for all the experimented configurations. The competitive performance achieved in our study makes deep ensembles a state-of-the-art method to be considered when dealing with small datasets.
Moreover, we performed a broad analysis considering multiple architectures and datasets splits to determine whether the cosine loss has a clear advantage over the cross-entropy. From our results, we partially confirm the nice regularization capabilities of the cosine loss but we also broaden the picture showing that the availability of data and the complexity of the model are two important factors to consider before making a choice.
Finally, according to our study, we have also shown that our proposed solution composed by a large ensemble of lower complexity networks trained with cross-entropy loss, is the most robust ensemble configuration considering variations of dataset size and type.
We believe that the community can benefit from the presented comprehensive comparative experiments by applying and extending this methodology in diverse directions, including different datasets, loss functions, and architectures.

REFERENCES

Sanjeev Arora, Simon S. Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu. Harnessing the power of infinitely wide deep nets on small-data tasks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rkl8sJBYvH.

Bjorn Barz and Joachim Denzler. Deep learning on small datasets without pre-training using cosine loss. In *The IEEE Winter Conference on Applications of Computer Vision*, pp. 1371–1380, 2020.

Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pp. 17–36, 2012.

Christopher M Bishop. Mixture density networks. 1994.

Jörg Bornschein, Francesco Visin, and Simon Osindero. Small data, big decisions: Model selection in the small-data regime. *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, 2011.

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, 15(1):3133–3181, 2014.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059, 2016.

Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson. Loss surfaces, mode connectivity, and fast ensembling of dnns. In *Advances in Neural Information Processing Systems*, pp. 8789–8798, 2018.

Giorgio Giacinto and Fabio Roli. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing*, 19(9-10):699–707, 2001.

Seungmin Han and Jongpil Jeong. An weighted cnn ensemble model with small amount of data for bearing fault diagnosis. *Procedia Computer Science*, 175:88–95, 2020.

Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

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*, pp. 770–778, 2016.

Guosheng Hu, Xiaojiang Peng, Yongxin Yang, Timothy M Hospedales, and Jakob Verbeek. Frankenstein: Learning deep face representations using small data. *IEEE Transactions on Image Processing*, 27(1):293–303, 2017.

Syed Sameed Husain and Miroslaw Bober. Improving large-scale image retrieval through robust aggregation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1783–1796, 2016.

Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. Generalization in deep learning. *CoRR*, abs/1710.05468, 2017. URL http://arxiv.org/abs/1710.05468.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

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

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pp. 6402–6413, 2017.

Stefan Lee, Senthil Purushwalkam, Michael Cogswell, David J. Crandall, and Dhruv Batra. Why M heads are better than one: Training a diverse ensemble of deep networks. *CoRR*, abs/1511.06314, 2015. URL http://arxiv.org/abs/1511.06314.

Hui Li, Xuesong Wang, and Shifei Ding. Research and development of neural network ensembles: a survey. *Artificial Intelligence Review*, 49(4):455–479, 2018.

Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

Lanlan Liu, Michael Muelly, Jia Deng, Tomas Pfister, and Li-Jia Li. Generative modeling for small-data object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6073–6081, 2019.

Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *CoRR*, abs/1805.12076, 2018. URL http://arxiv.org/abs/1805.12076.

Matthew Olson, Abraham Wyner, and Richard Berk. Modern neural networks generalize on small data sets. In *Advances in Neural Information Processing Systems*, pp. 3619–3628, 2018.

David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198, 1999.

Van Hiep Phung, Eun Joo Rhee, et al. A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets. *Applied Sciences*, 9 (21):4500, 2019.

Diego Rueda-Plata, Raúl Ramos-Pollán, and Fabio A. González. Supervised greedy layer-wise training for deep convolutional networks with small datasets. In Manuel Núñez, Ngoc Thanh Nguyen, David Camacho, and Bogdan Trawinski (eds.), *Computational Collective Intelligence - 7th International Conference, ICCCI 2015, Madrid, Spain, September 21-23, 2015. Proceedings, Part I*, volume 9329 of *Lecture Notes in Computer Science*, pp. 275–284. Springer, 2015. doi: 10.1007/978-3-319-24069-5_26. URL https://doi.org/10.1007/978-3-319-24069-5_26.

Torgyn Shaikhina and Natalia A Khovanova. Handling limited datasets with neural networks in medical applications: A small-data approach. *Artificial intelligence in medicine*, 75:51–63, 2017.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):60, 2019.

Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pp. 270–279. Springer, 2018.

Joaquin Vanschoren. Meta-learning: A survey. *CoRR*, abs/1810.03548, 2018. URL http://arxiv.org/abs/1810.03548.

CV Krishna Veni and T Sobha Rani. Ensemble based classification using small training sets: A novel approach. In *2014 IEEE Symposium on Computational Intelligence in Ensemble Learning (CIEL)*, pp. 1–8. IEEE, 2014.

Abdul Wasay, Brian Hentschel, Yuze Liao, Sanyuan Chen, and Stratos Idreos. Mothernets: Rapid deep ensemble learning. In Inderjit S. Dhillon, Dimitris S. Papailiopoulos, and Vivienne Sze (eds.), *Proceedings of Machine Learning and Systems 2020, MLSys 2020, Austin, TX, USA, March 2-4, 2020*. mlsys.org, 2020. URL https://proceedings.mlsys.org/book/301.pdf.

Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=Sklf1yrYDr.

Y. Bengio Y. LeCun and G. E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015. doi: 10.1038/nature14539. URL https://doi.org/10.1038/nature14539.

Xuefei Zhe, Shifeng Chen, and Hong Yan. Directional statistics-based deep metric learning for image classification and retrieval. *Pattern Recognition*, 93:113–123, 2019.

A ENSEMBLES COMPUTATIONAL COMPLEXITY

In Table 1 we report the computational complexity of the three ensembles on both datasets. To obtain the computational complexity of a single ResNet-8 or ResNet-20, divide the FLOPs by the ensemble dimension.

Dataset	20 ResNet-8	5 ResNet-20	ResNet-110
CIFAR-10	9.3	8.1	10.4
CIFAR-100	9.5	8.2	10.4

Table 1: Computational complexity of the three ensembles in terms of MFLOPs.

B ADDITIONAL EXPERIMENTAL DETAILS

In this section, we provide more details regarding the optimization set-up used to train our ensembles.

Input images are normalized subtracting the per-channel mean computed over the sub-sampled training set. Before being fed to the network, each image is padded 4 pixels on each side and a 32×32 crop is randomly sampled from the padded image or its horizontal flip.

For training the networks, we use stochastic gradient descent (SGD) with weight decay and Nesterov momentum respectively set to 10^{-4} and 0.9. We start with a learning rate of 0.1 and decrease it after 75% of the total number of iterations by an order of magnitude. We increased the number of iterations with the initial learning rate to be sure of decreasing it after having reached the training loss plateau. Furthermore, we have noticed that decreasing a second time the learning rate did not improve further the testing performance.

The networks were fed with a mini-batch of 32 images. We preferred a smaller batch with respect to the original 128 since led to better performance. Despite this effect is well-known in DL [Keskar et al., 2017], we empirically noticed that this is even more influential in our settings since, probably, more noisy gradient updates prevent the network from overfitting the tiny training sets.

Due to the different size of ResNet-8, ResNet-20 and ResNet-110 we train the architectures for a different number of epochs. Assuming that the first element of the set corresponds to datasets with 10 samples per class and the last one to 250, architectures are trained, respectively for $\{400, 300, 250, 200\}$, $\{400, 300, 300, 350\}$ and $\{2000, 1000, 800, 600\}$ epochs.

C COMPARING LOSSES ON SINGLE NETWORKS

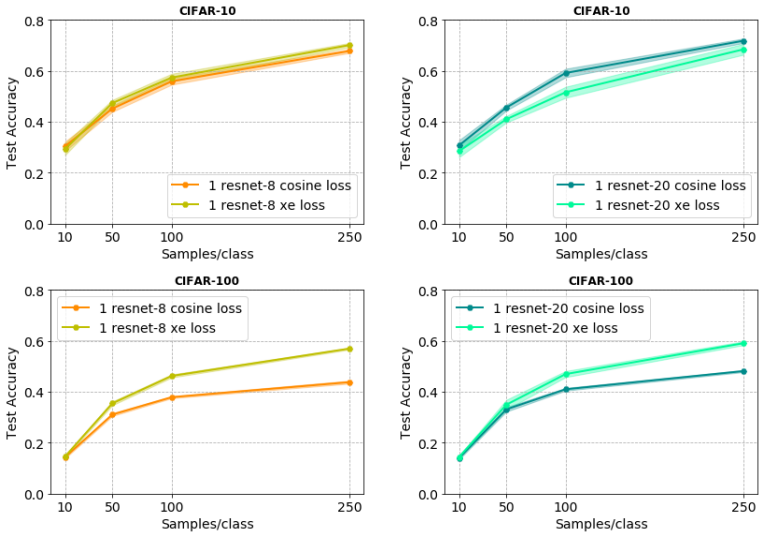

Figure 4: Comparison between the two losses considering single ResNet-8 and ResNet-20.