Revisiting ensembling for improving the performance of deep learning models

Antonio Bruno[1[0000−0002−9152−8112]], Davide Moroni[1[0000−0002−5175−5126]], and Massimo Martinelli[1[0000−0001−7419−5099]]

Institute of Information Science and Technologies,
National Research Council of Italy, Via Moruzzi, 1 – 56124-Pisa (IT)
{Name.Surname}@isti.cnr.it
www.isti.cnr.it

Abstract. Ensembling is a very well-known strategy consisting in fusing several different models to achieve a new model for classification or regression tasks. Over the years, ensembling has been proven to provide superior performance in various contexts related to pattern recognition and artificial intelligence. Moreover, the basic ideas that are at the basis of ensembling have been a source of inspiration for the design of the most recent deep learning architectures. Indeed, a close analysis of those architectures shows that some connections among layers and groups of layers achieve effects similar to those obtainable by bagging, boosting and stacking, which are the well-known three basic approaches to ensembling. However, we argue that research has not fully leveraged the potential offered by ensembling. Indeed, this paper investigates some possible approaches to the combination of weak learners, or sub-components of weak learners, in the context of bagging. Based on previous results obtained in specific domains, we extend the approach to a reference dataset obtaining encouraging results.

Keywords: Ensembling · bagging · machine learning · deep learning · image classification · convolutional neural networks.

1 Introduction

Representation learning and deep learning have achieved amazing results in the last decades, obtaining unparalleled performance under challenging tasks such as image classification and object detection [13]. After the first impressive leap, many works have been of incremental nature in the previous years, often focused on architecture engineering for achieving minor improvements over a sensible increase in complexity. Indeed, the performance gain versus the computational increment ratio has become less attractive. Therefore, research has moved to find optimal tradeoffs between accuracy and computational load [17]. This is also motivated by the widespread adoption of deep learning paradigms that calls for the sustainable use of artificial intelligence (AI). AI has operational costs than can be directly measurable in energy consumption, having therefore a relevant

environmental footprint [16]. Ensembling is a well-known approach that enables using a collection of different models (e.g., classifiers or regressors) to obtain a new model having other (and generally superior) performance with respect to predefined metrics. Ensembling has a long history that dates back even before the birth of machine learning. Indeed, it is customary to state that the first application of Ensembling is majority voting in statistics as in the claim of the theorem by Marquis de Condorcet: in 1785 he proved that if the probability of each voter being correct is more than one-half and the voters are independent, then the addition of more voters increases the likelihood of the majority vote being correct (see, e.g., [2]). Long after that, Ensembling has been used to turn weak models into superior models showing encouraging results in several domains. It has been reported that ensemble models often become in the first place in public competitions such as those promoted by Kaggle [1]. The relationship between deep learning and Ensembling is at least twofold. From one side, basic constructions of Ensembling known as bagging, boosting and stacking have somehow influenced architectures commonly used in deep learning and the way they are trained. For instance, residual networks behave like ensembles of relatively shallow networks [18]. On the other side, thanks to Ensembling strategies, deep learning models can be used as basic models to build more complex models. This paper focuses on this second aspect by recapping Ensembling and its role in deep learning, exploring several directions. Based on previous results obtained in a specific domain, preliminary results are then reported on a benchmark dataset. This paper extends the short paper [3].

2 Related works

Ensembling generally refers to machine learning approaches in which a set of *weak learners* (or *basic models*) is turned into a *strong learner* (or *ensemble model*). The set of weak learners might consist of homogenous models (i.e., they are all from the same family or architecture) or might be heterogeneous, i.e., the basic models belong to different machine learning paradigms. The basic example is to put together multiple models trained for solving the same classification or regression task and then combine them together in some fashion, e.g., by performing majority voting in the case of classification or averaging in the case of regression. The scope of performing Ensembling is generally related to the desire to reduce the bias or variance that affects a machine learning task [7]. As it is well known, a very simple model might have a great error in achieving good performance on a dataset, even during training. This is generally linked to the low representation capabilities of simple models that cannot capture all the complex patterns in the training datasets. Such error during training is referred to as the bias of the model. By converse, very complex models have many degrees of freedom to adhere to the training dataset completely and obtain excellent performance during training. However, they capture not only the relevant features of the problem but also learn insignificant features of the training dataset. This results in relatively poor performance during test and validation: the model is

overfitted to the training dataset and does not reach good general results, having scarce generalization capabilities. We refer to this issue, as saying that the model has high variance. The three basic approaches to performing Ensembling are bagging, boosting, and stacking. In general, bagging reduces the variance among the base classifiers, while boosting-based ensembles lead to bias and variance reduction. Stacking is commonly used as a bias-reducing technique. In more detail, bagging is performed by subdividing the training datasets into different subsets according to some criteria, e.g., balancing class distributions inside each subset or other forms of equalization. Then, each subset of the training set is used to train a weak classifier. Any such classifier ideally has a low bias on the training set but possibly high variance. Using a fusion layer, the outputs of the single classifiers are combined by performing (weighted) voting or performing a weighted average. The model given by the fusion of the weak classifiers is called a strong classifier, and it potentially exhibits lower variance. Notice that the weak classifiers might be trained independently and in parallel. Very often, such classifiers share the same architecture. In boosting instead, weak classifiers are very simple and low complexity but are trained cleverly, for example, using cascading. Adaboost [15] is one of the most popular approaches in which each classifier is trained so as to properly deal with the examples in the training set on which previous weak classifiers have failed. The boosting concept is also known to be the backbone behind well-known architectures like Deep Residual networks [10]. Finally, stacking often considers heterogeneous weak learners. Training is performed in parallel, while a final combination is obtained by training a meta-model to output a prediction based on the different weak models' predictions. Deep convex nets (DCN) [6] are recognized to be a deep learning architecture composed of a variable number of modules stacked together to form the deep architecture. In general, all of these approaches have been used in conjunction with deep learning models. The review [9] presents some recent literature on the subject systematically.

3 Methodology

As seen in the previous section, ensemble and deep learning have a twofold relationship. This section aims to briefly report some experiments on Ensembling that is worthwhile exploring for optimising deep learning models.

1. Varying the number of classifiers. After having fixed a deep learning architecture, such architecture can be regarded as a weak model. Different training runs starting with random weights might result in different classifiers. Majority voting can be applied in this case as shown in Figure 1. The dependence of the performance with respect to the number of classifiers might be studied.
2. Sampling strategies and balancing. Besides performing training of all the weak learners on the full training set as described before, procedures for sampling can be applied. For instance, using disjoint datasets for each weak classifier helps have a set of independent classifiers. In addition, stratification can be applied to keep the same class frequencies in each subset; conversely,

it might be interesting to explore the possibilities given by training each classifier to make it specialised in addressing a special class.
3. Control size and model complexity. The ensembling approaches can be performed by keeping track of the model size and model complexity; this can help understand the heuristics for the optimal choice of the weak learners' dimensions and the ensemble size.
4. Stacking at the deep feature levels. In many cases, the first layers of a deep network perform feature extraction, while the final layers, usually fully connected, perform classification/regression. A possibility in ensembling is given by stacking weak models by removing the final classification layers from each one and training an ad hoc meta classifier as shown in Figure 2.
5. Learning strategies. Given the trained ensemble, it is still possible to fine-tune the model parameters by properly training the model, freezing or not some of the overall network layers.

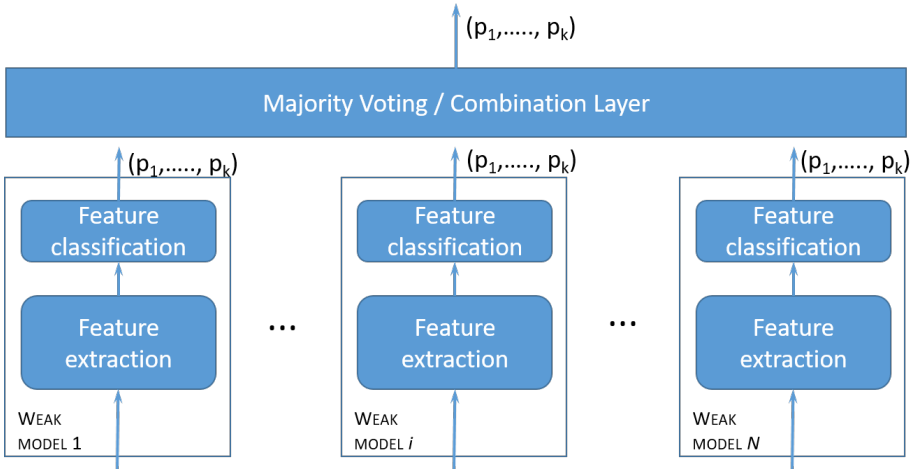

Fig. 1. Majority voting can be applied to fuse the output of multiple weak learners and obtain a strong one. In the figure, we consider a variable number of N weak classifiers (trained for instance using special partitions of the training data), each one producing in output a confidence level vector for k classes. Majority voting or other schemes for combination might be used and the dependence of the performance with respect to the number N of weak classifiers might be studied.

4 Experimental results

In this section, we report preliminary experimental results obtained by following the methodology discussed in Section 3. In all the experiments we used as

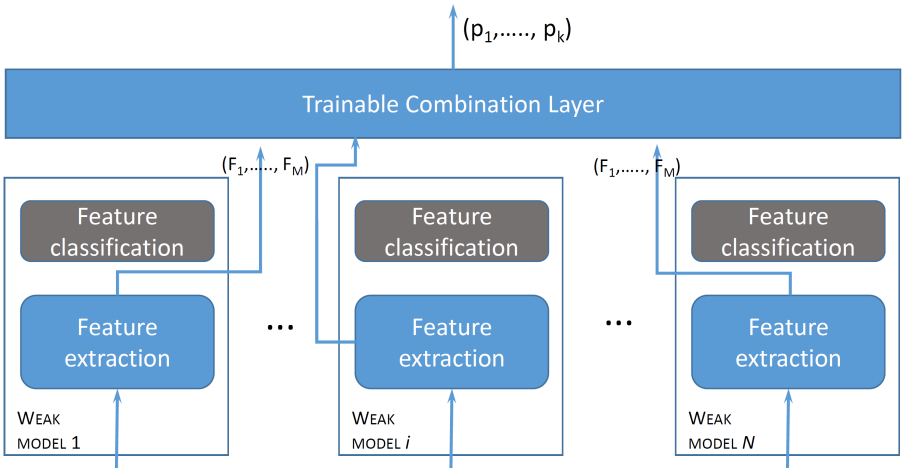

Fig. 2. A different and innovative approach to ensembling of deep weak classifier is reported pictorially in the figure. After having trained each weak classifier independently, only the convolutional layers are kept while the final decision layers of each classifier are disregarded. In summary, we keep a sequence of N *weak feature extractor* modules, each one producing a M-dimensional deep feature vector. These modules can be turned into an ensemble by adding a trainable final decision layer, e.g. made of fully-connected networks, having an input size equal to $N \cdot M$.

a basic weak learner a convolutional neural network belonging to the EfficientNet family [17]. This is a family of eight neural networks named b0, b1,..., b7 featuring different complexity, where b0 is the most simple model ($5,5M$ parameters) while b7 ($66,7M$ parameters) is the most complex. Such networks are obtained through a process of aggregate scaling across depth, width and resolution dimensions starting from an archetype based on the inverted bottleneck MBConv, first introduced in MobileNet [14]. The aggregate scaling is performed in a uniform way so as to optimise according to some heuristics the performance of the network. For this characteristic, EfficientNet is an excellent building block to study ensembling strategies. Notice that in our experiments we use transfer learning. In more detail, all the EfficientNet models we used as weak classifiers share the weights of pre-trained models from ImageNet [5].

To show concretely some possibilities, we report two examples. The first regards an application to a specific dataset for precision agriculture, i.e. the PlantVillage dataset [12], which contains images of plants, either healthy or with a wide range of diseases. The dataset has been analyzed in a recent study [4], which we recap here. A second example is given by a popular benchmark dataset, i.e. the CIFAR-100 dataset [11], which consists of 60000 32×32 colour images divided in 100 classes, with 600 images per class. The 100 classes in the CIFAR-100 are grouped into 20 superclasses. Each image comes with a "fine"

(the class it belongss to) and a "coarse" label (the superclass to which it belongs). In the experiments, the fine-grained version with 100 classes has been used.

For the first example, using particular and ad hoc tricks explained in the paper [4], we were able to obtain the results reported in Table 1 for a single *weak* model trained end-to-end.

Model	Test	Valid	Train
EfficientNet-b0	99.6995	99.8454	99.9960
EfficientNet-b1	99.5793	99.8454	100.000
EfficientNet-b2	99.5192	99.7681	99.9140
EfficientNet-b3	99.6394	99.8712	99.9860
EfficientNet-b4	99.6995	99.8454	99.9980
EfficientNet-b5	99.7596	99.7939	99.9920
EfficientNet-b6	**99.7596**	**99.8712**	**99.9880**
EfficientNet-b7	99.5192	99.8454	99.9960

Table 1. Table with best Weighted F1-score results, for each EfficientNet variant. Best values are in **bold.**

Then ensembling was applied using the methodology depicted in Figure 2. Given the generally good performance of the models reported in Table 1 and aiming at reduced complexity, we restricted to performing ensembling (i) only on b0 models and (ii) using the minimum number of weak learners, i.e. only two learners. As a combination layer for the ensemble, we used the final layer of the weak learners scaled to accommodate the input of two learners. Five runs of training were conducted to train the combination layers keeping frozen the deep feature extractor module of each weak classifier. The results demonstrate a 100% F1-score in training and test for all the five runs, while in validation four runs achieved 100% performance but one, which featured a 99.998% performance.

Similar tests were conducted on the more challenging CIFAR-100 dataset, where the ensemble model was able to reach a 96.808% with an improvement of 0.728% over the previous state-of-the-art [8]. We notice that the proposed ensemble has $\approx 11M$ parameters with respect to the previous state-of-the-art model which had $\approx 480M$ parameters. For what regards floating point operations, the proposed ensemble features $\approx 0.9G$ FLOPS by contrast with the $\approx 299G$ FLOPS required by the previous model. This shows that the proposed methodology is effective in producing accurate models with lower complexity in the CIFAR-100 case.

5 Conclusions

This short paper has explored several directions for introducing Ensembling in the deep learning context. The approaches and the involved ideas are well-grounded in previous knowledge and guarantees connected to Ensembling in

machine learning, yet there are many possible pathways and combinations to explore. In some preliminary experiments, we studied an adaptive ensembling based on bagging, making it possible to achieve 100% accuracy on a known dataset in agricultural applications [12]. Other experiments on a well know benchmark dataset have shown a significant boost in performance over the state-of-the-art coupled with a reduction of complexity. Further experiments are underway to show the applicability range of the proposed method and the results will be reported in the next future.

References

1. Bojer, C.S., Meldgaard, J.P.: Kaggle forecasting competitions: An overlooked learning opportunity. International Journal of Forecasting **37**(2), 587–603 (2021)
2. Boland, P.J.: Majority systems and the condorcet jury theorem. Journal of the Royal Statistical Society: Series D (The Statistician) **38**(3), 181–189 (1989)
3. Bruno, A., Martinelli, M., Moroni, D.: Exploring ensembling in deep learning. Pattern recognition and image analysis **32**(3), 519–521 (2022). https://doi.org/10.1134/S1054661822030087
4. Bruno, A., Moroni, D., Dainelli, R., Rocchi, L., Toscano, P., Ferrari, E., Martinelli, M.: Improving plant disease classification by adaptive minimal ensembling. Frontiers in artificial intelligence (2022). https://doi.org/10.3389/frai.2022.868926
5. Deng, J., Dong, W., Socher, R., Li, L., Kai Li, Li Fei-Fei: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
6. Deng, L., Yu, D.: Deep convex net: A scalable architecture for speech pattern classification. In: Twelfth annual conference of the international speech communication association (2011)
7. Dong, X., Yu, Z., Cao, W., Shi, Y., Ma, Q.: A survey on ensemble learning. Frontiers of Computer Science **14**(2), 241–258 (2020)
8. Foret, P., Kleiner, A., Mobahi, H., Neyshabur, B.: Sharpness-aware minimization for efficiently improving generalization. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021 (2021)
9. Ganaie, M.A., Hu, M., et al.: Ensemble deep learning: A review. arXiv preprint arXiv:2104.02395 (2021)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
11. Krizhevsky, A., Nair, V., Hinton, G.: Cifar-10 (canadian institute for advanced research) http://www.cs.toronto.edu/ kriz/cifar.html
12. Mohanty, S.P., Hughes, D.P., Salathé, M.: Using deep learning for image-based plant disease detection. Frontiers in plant science **7**, 1419 (2016)
13. Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M.P., Shyu, M.L., Chen, S.C., Iyengar, S.S.: A survey on deep learning: Algorithms, techniques, and applications. ACM Computing Surveys (CSUR) **51**(5), 1–36 (2018)
14. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018)

15. Schapire, R.E.: Explaining adaboost. In: Empirical inference, pp. 37–52. Springer (2013)
16. Schwartz, R., Dodge, J., Smith, N.A., Etzioni, O.: Green AI. Communications of the ACM **63**(12), 54–63 (2020)
17. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
18. Veit, A., Wilber, M.J., Belongie, S.: Residual networks behave like ensembles of relatively shallow networks. Advances in neural information processing systems **29** (2016)