Compositional Committees of Tiny Networks

Goh Howe Seng[1], Tomas Maul[2] (⌧),
Manav Nitin Kapadnis[3]

[1] University of Nottingham Malaysia,
University of Nottingham Malaysia, Tomas.Maul@nottingham.edu.my
[2] Indian Institute of Technology Kharagpur

Abstract. Deep neural networks tend to be accurate but computationally expensive, whereas ensembles tend to be fast but do not capitalize on hierarchical representations. This paper proposes an approach that attempts to combine the advantages of both approaches. Hierarchical ensembles represent an effort in this direction, however they are not compositional in a representational sense, since they only combine classifier decisions and/or outputs. We propose to take this effort one step further in the form of compositional ensembles, which exploit the composition of the hidden representations of classifiers, here defined as tiny networks on account of being neural networks of significantly limited scale. As such, our particular instance of compositional ensembles is called Compositional Committee of Tiny Networks (CoCoTiNe). We experimented with different CoCoTiNe variants involving different types of composition, input usage, and ensemble decisions. The best variant demonstrated that CoCoTiNe is more accurate than standard hierarchical committees, and is relatable to the accuracy of vanilla Convolutional Neural Networks, whilst being 25.7 times faster in a standard CPU setup. In conclusion, the paper demonstrates that compositional ensembles, especially in the context of tiny networks, are a viable and efficient approach for combining the advantages of deep networks and ensembles.

Keywords: Neural networks · Ensembles · Compositionality · Hierarchical representations.

1 Introduction

Deep learning or deep neural networks (DNNs) are important engines underlying current artificial intelligence applications. DNNs have many advantages, chief among them being the levels of accuracy attained, frequently surpassing human levels of performance. However, they do also have disadvantages, such as memory and computational cost, and the fact that they can be hard to design and train. On the other hand, we have ensembles, with strengths such as the fact that they can be very simple, predictable, and fast to train, especially when using small classifiers. On the downside, ensembles do not generally achieve the same levels of accuracy as DNNs.

This paper is concerned with the question: can we combine the advantages of both DNNs and ensembles, whilst avoiding their disadvantages? So far, it seems this combination has been done primarily by inserting the key concept of ensembles (i.e. committee of classifiers) into DNNs, i.e. ensembles of DNNs [2]. In spite of improving accuracy further, this does not solve the computational cost problem of DNNs. The opposite idea, i.e. inserting the key concept of DNNs (i.e. hierarchical feature composition) into ensembles, does not seem to have been explored so far in the literature. The literature on hierarchical ensembles is very close to this concept, however, typically only decisions and/or outputs are combined in a hierarchical manner, which is not strictly feature composition [3–5, 7–10, 12–14]. Thus we propose to imbue ensembles with feature compositionality and propose to call this approach Compositional Committee of Tiny Networks (CoCoTiNe). We use the term *tiny networks* (TNs) to denote networks with a small number of narrow layers. In order to simplify the experimental design we restrict ourselves to data belonging to the image domain, although the approach can easily be extended to any type of data.

The main contribution of this research is to provide an alternative solution to DNNs by designing a compositional ensemble that exploits and combines the best attributes of both deep learning and ensemble methods. The proposed approach aims to get realistically close to the accuracy levels of deep learning networks, whilst affording better speed and consistency.

2 Related Works

This section focuses on the research works most closely related to CoCoTiNe, i.e.: hierarchical ensembles. As outlined in [12], hierarchical ensembles can be classified into the following five categories: (1) top-down ensemble methods, (2) Bayesian ensemble methods, (3) Reconciliation methods, (4) true path rule ensembles, and (5) decision tree-based ensembles. We propose that CoCoTiNe is an instantiation of an additional category within this taxonomy, i.e. compositional ensembles, which are essentially hierarchical ensembles that implement some form of feature compositionality across the ensemble's hierarchical layers.

In [9] a hierarchical ensemble of convolutional neural networks was proposed for diagnosing Diabetic Macular Edema which is an advanced state of diabetic retinopathy and can lead to permanent vision loss. This approach outperformed the existing state-of-the-art methods on publicly available Diabetic Macular Edema databases, and uses a flexible hierarchical ensemble of CNNs (large networks) rather than tiny networks proposed in this research. Moreover, the hierarchy is built from the outputs of previous layers, rather than hidden representation as proposed in this research.

Similarly, in [3] a hierarchical ensemble was proposed for the task of Indian Language identification. This approach uses 2 layers of ensembles, with the first layer consisting of two ensembles which train on different feature sets. The final classification is computed using a majority vote from the classification outputs of these ensembles. This method differs from our approach as it does not put emphasis on the use of tiny networks and has a fixed hierarchical structure of two layers, with the first layer having two separate ensembles. Moreover, the composition of this approach is also based on the outputs of the networks rather than hidden representations.

In [8] a hierarchical long short term memory (LSTM) ensemble was proposed for short-term forecasting of wind speed. Similar to [3], this approach adopts a fixed 2-layer structure, uses LSTMs as compared to tiny (feedforward) networks, and implements the hierarchy based on previous-layer outputs (rather than hidden representations), all of which are distinct from our approach.

As the examples above illustrate the single most important differentiating factor compared to our work consists of the fact that these works do not implement feature compositions. Apart from lacking feature compositions, most hierarchical ensemble studies, are also based on a fixed/rigid number of layers, and do not intentionally involve tiny networks. Other examples of such studies include [5, 7, 10, 13, 14]. However, some hierarchical approaches do allow for an unconstrained number of hierarchical layers, e.g. [4, 9]. The key conclusion of this brief survey is that, so far, there do not seem to be other systematic studies of feature compositionality in hierarchical ensembles.

3 Methodology

3.1 CoCoTiNe

A CoCoTiNe consists of a multi-layer architecture with an ensemble of tiny networks (TNs) at each layer that generally receive as input the combined hidden representations of a subset of tiny networks from the previous layer. In theory, this connectivity structure should provide good compositionality properties to the model as TNs can build from hidden representations from previous Compositional Committee (CoCo) layers. The use of ensembles also means that the final classification is based on a committee decision which in turn decreases prediction variance. Furthermore, the use of TNs (i.e. fully-connected networks with 1-3 narrow hidden-layers) in each layer also encourages fast and predictable training processes, compared to large networks. Figure 1 provides a simplified representation of our general approach to compositional ensembles.

CoCoTiNe classifies images based on local receptive fields (image sub-regions) that can vary in terms of scale and position. Based on this, one of the assumptions currently adopted in this work is that target objects are reasonably centered and scaled, as in the MNIST handwritten digits dataset.

During the training of CoCoTiNe, each tiny network in the first CoCo layer takes in the pre-processed sub-region of a training image and its class label. Each CoCo layer 1 TN is assigned to one random sub-region, and this association is fixed for the lifetime of the model. The hidden representations generated in TNs in COCO layer 1 and associated target class labels (for inputted images) are then fed into TNs in CoCo layer 2 for further training. Each TN in CoCo layer 2 may take in multiple hidden representations produced by multiple tiny networks in CoCo layer 1. Connections with TNs in the previous CoCo layer are randomly determined, and are again fixed for the lifetime of the model. The tiny

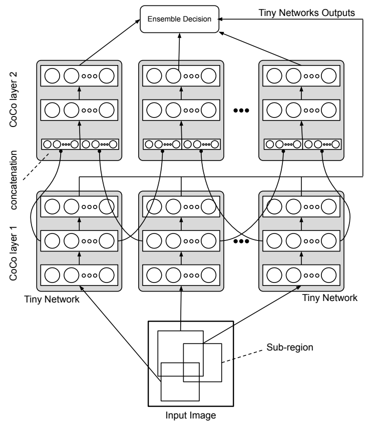

Fig. 1. Simplified diagram of a CoCoTiNe architecture.

networks in subsequent layers are implemented in exactly the same way as CoCo layer 2. Each CoCo layer is trained independently and only after the training of its previous layer. All of our CoCoTiNe experiments were implemented using Python, Tensorflow 2.0, Keras, NumPy, SciPy, Scikit-learn and Pickle, and all experiments were run in the Google Colab environment.

4 Experimental Design

4.1 Datasets

Three datasets were used in this work, namely MNIST [1], Fashion MNIST [15], and CIFAR-10 [6]. All the target objects in all three datasets are reasonably well centered and scaled, complying with one of the key assumptions of the current CoCoTiNe implementation. Due to space constraints, and since these datasets are well-known, we refrain from describing them further here.

4.2 Experimental variants

As the general CoCoTiNe concept allows for multiple variants, we experimented with variations along different dimensions in order to find the most effective implementation, namely: (1) the method for combining hidden representations from previous CoCo layers, (2) how/whether the input information is allowed to influence multiple layers, and (3) the method for making the committee decision. We abbreviate these dimensions to *hidden*, *input*, and *decision*, respectively. Unfortunately, due to space constraints, we can't outline all of the variations here, however we summarize our main findings in the results/discussion section.

4.3 Baseline conditions

In this experiment, a total of seven baseline conditions (BCs) were implemented and tested along with the final implementation of CoCoTiNe. Since to the best of our knowledge there are no other comparable compositional ensemble approaches, we used standard ensemble and hierarchical ensemble techniques for comparison purposes. The baseline conditions are summarized below:

1. **(CNN)** We implemented a vanilla convolutional neural network (CNN) as a deep learning baseline.
2. **(MLP-Mixer)** We chose MLP-Mixer [11] as a recent and strong baseline representative of recent efforts to re-invent multilayer perceptron (MLP) based architectures such that they can be competitive with CNNs.
3. **(tiny-whole)** This condition implements a single TN that takes in the whole image instead of an image sub-region as input.
4. **(tiny-sub)** This condition uses a single TN that takes in a random image sub-region as input. During prediction, it takes a few random sub-regions of the same image and performs classification by taking the max of the summed logits.
5. **(flat-whole)** This condition implements a flat ensemble of TNs that take in the whole image instead of image sub-regions as input, whereby each TN is associated with a random subset of 60% of training instances, as a simple form of bagging.
6. **(flat-sub)** This condition implements a flat ensemble of TNs that take in random image sub-regions as input.
7. **(hierarchical)** This condition is similar to the final implementation of CoCoTiNe, except that the inputs to TNs in CoCo layers 2 and above, are based only on the output logits of TNs in the previous layer (i.e. no hidden representation composition occurs).
   (a) **(hierarch-concat)** The output logits from the previous layer are combined via concatenation.
   (b) **(hierarch-sum)** The output logits from the previous layer are combined by summation.

In order to maximize comparative fairness between BCs and CoCoTiNe, the number of TNs in BCs with flat ensembles was set to 30, which is inline with the number of TNs used in the final implementation of CoCoTiNe. The number of network parameters in the CNN condition was also set to match as closely as possible that of CoCoTiNe. Finally, training was conducted for five epochs per dataset across all conditions.

5 Results and Discussion

5.1 Experimental variants

Our experiments on how to incorporate hidden representations showed that concatenation is a better approach for combining hidden representations, compared for example to summation. Similarly, our experiments suggest that output logits seem to be best combined via concatenation. With regards to input information, our experiments show that adding additional raw input information to TNs in deeper CoCo layers provides no clear advantage to the model, even reducing its accuracy. Finally, with regards to ensemble decisions, our experiments showed that computing decision weights via the number of TNs in each CoCo layer, relative to the CoCo layer with the most TNs (*weight-prop*), and using this to weight votes, yields a better accuracy. The results of these experiments informed our final CoCoTiNe variant which was used in the comparative study below.

5.2 Baselines vs. CoCoTiNe

Condition	MNIST	Fashion-MNIST	CIFAR-10
CoCoTiNe	98.71 (0.04)*	86.43 (0.07)*	53.47 (0.08)*
MLP-Mixer	96.89 (0.24)	86.10 (0.57)	40.26 (1.39)
CNN	**98.82 (0.96)**	**90.33 (1.2)**	**70.86 (5.32)**
tiny-whole	97.07 (0.16)	86.29 (0.09)	42.7 (0.62)
tiny-sub	82.48 (0.85)	70.06 (0.13)	31.9 (0.98)
flat-whole	97.08 (0.02)	86.43 (0.07)	44.18 (0.46)
flat-sub	91.06 (0.05)	73.56 (0.08)	34.04 (0.51)
hierarch-concat	97.87 (0.04)	84.13 (0.07)	47.9 (0.17)
hierarch-sum	97.32 (0.19)	82.87 (0.21)	46.28 (0.16)

Table 1. Average test accuracy (across 10 trials) for all main conditions. [3]

Condition	MNIST	Fashion-MNIST	CIFAR-10
CoCoTiNe	**180 (0.04)**	**184 (0.07)**	**182 (0.08)**
MLP-Mixer	300 (1.59)	305 (0.7)	290 (0.9)
CNN	4665 (0.96)	4838 (1.2)	4533 (5.32)

Table 2. Average training times across 10 trials in seconds.[4]

Based on the results obtained in Table 1, CoCoTiNe has a better accuracy compared across all conditions which take image sub-regions as input, either as only a single tiny network or a flat ensemble of tiny networks (*tiny-sub* and *flat-sub*). This demonstrates the effectiveness of enhancing ensembles with the architectural feature of compositionality, obtained via the hierarchical composition of hidden representations. However, the accuracy advantage of CoCoTiNe, while present, is somewhat less significant when compared to conditions which take in the whole image as input (*tiny-whole* and *flat-whole*). This suggests that future work should look into new ways of exploiting and combining local/global image information in the context of compositional ensembles.

[3] Values in bold refer to the best conditions, and values with an asterisk refer to the second-best conditions.

[4] Values in brackets correspond to standard deviations.

When comparing conditions of single tiny networks vs. flat ensembles of tiny networks with their respective input types, i.e. (*tiny-whole* vs. *flat-whole*) and (*tiny-sub* vs. *flat-sub*), conditions that use an ensemble of tiny networks (*flat-whole* and *flat-sub*) significantly improve the accuracy compared to their single tiny network peers, as expected.

When comparing CoCoTiNe against non-compositional hierarchical baselines (similar to CoCoTiNe but without the compositionality of hidden representations), CoCoTiNe consistently achieves better results. This further demonstrates the importance of the use of hidden representations to achieve useful compositional properties and supports the view that compositionality of hidden representations is an important architectural feature for further research in the domain of hierarchical ensembles.

When comparing CoCoTiNe to the CNN baseline using the MNIST dataset, CoCoTiNe exhibits similar average accuracy. This proves that CoCoTiNe's first aim of obtaining comparable accuracy to DNNs is achievable. However, the accuracy gap widens when using the Fashion MNIST and CIFAR-10 datasets, which is likely to be due to the fact that these datasets are more challenging. It is likely that future extensions of CoCoTiNe, as discussed below, can bridge this gap.

The second and third aims of CoCoTiNe, which were affording significantly better speed and consistency compared to DNNs, were also achieved as CoCoTiNe showed significant training time reduction and lower standard deviations across multiple trials when compared to CNN, as depicted in Table 2, in a classic central processing unit (CPU) setup. Overall CoCoTiNe was found to be 25.7 times faster than the corresponding CNN condition. We have omitted a speed comparison in a hardware setup with graphics processing units (GPUs) since this is deemed to be an unfair comparison, given that GPUs and deep learning frameworks have to an extent co-evolved in order to optimize the training and inference time of DNNs, whereas this process has not yet occurred for compositional ensembles. The overall stability and consistency of CoCoTiNe can also be seen in the standard deviations of accuracy measures as depicted in Table 1, which are small for CoCoTiNe, especially when compared to the CNN and MLP-Mixer conditions.

6 Conclusion

The key finding of this paper is that compositional ensembles are a viable machine learning approach. A particular instance of a compositional ensemble was proposed (i.e. CoCoTiNe) and multiple variations were experimented with in order to find the best variant. The accuracy of this compositional ensemble was found to be better than standard hierarchical ensembles (that do not exploit the composition of hidden representations) and was found to be relatively close to vanilla DNNs. Moreover, the speed properties were found to be significantly better than those of DNNs in the context of standard CPU setups. Future work is recommended in terms of: (1) exploring other strategies for representational composition and architectural tuning (e.g. architecture of TNs and CoCo layers), (2) experimenting with different ways to exploit local/global image information, including learnt or adaptive approaches for patch extraction, (3) exploiting hardware parallelization, (4) exploring compositional ensembles in the context of continual learning. We hope that in the future other authors will explore the idea of compositional ensembles further, in order to maximize the variance reduction and speed advantages of ensembles, whilst capitalizing on the representational advantages of DNNs.

References

1. Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine **29**(6), 141–142 (2012)
2. Ganaie, M., Hu, M., et al.: Ensemble deep learning: A review. arXiv preprint arXiv:2104.02395 (2021)
3. Jain, R., Duppada, V., Hiray, S.: Seernet@ INLI-FIRE-2017: Hierarchical ensemble for indian native language identification. In: FIRE (Working Notes). pp. 127–129 (2017)
4. Kim, B.K., Roh, J., Dong, S.Y., Lee, S.Y.: Hierarchical committee of deep convolutional neural networks for robust facial expression recognition. Journal on Multimodal User Interfaces **10**(2), 173–189 (2016)
5. Kim, K., Lin, H., Choi, J.Y., Choi, K.: A design framework for hierarchical ensemble of multiple feature extractors and multiple classifiers. Pattern Recognition **52**, 1–16 (2016)
6. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
7. Liang, X., Ma, Y., Xu, M.: Thu-hcsi at semeval-2019 task 3: Hierarchical ensemble classification of contextual emotion in conversation. In: Proceedings of the 13th International Workshop on Semantic Evaluation. pp. 345–349 (2019)
8. Marndi, A., Patra, G., Gouda, K.: Short-term forecasting of wind speed using time division ensemble of hierarchical deep neural networks. Bulletin of Atmospheric Science and Technology **1**(1), 91–108 (2020)
9. Singh, R.K., Gorantla, R.: Dmenet: diabetic macular edema diagnosis using hierarchical ensemble of cnns. Plos one **15**(2), e0220677 (2020)
10. Sudderth, E., Freeman, W.: Hierarchical ensemble of global and local classifiers for face recognition. IEEE Signal Processing Magazine **25**(2), 114–141 (2008)
11. Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Keysers, D., Uszkoreit, J., Lucic, M., et al.: Mlp-mixer: An all-mlp architecture for vision. arXiv preprint arXiv:2105.01601 (2021)
12. Valentini, G.: Hierarchical ensemble methods for protein function prediction. International Scholarly Research Notices **2014** (2014)
13. Wang, H., Li, J., He, K.: Hierarchical ensemble reduction and learning for resource-constrained computing. ACM Transactions on Design Automation of Electronic Systems (TODAES) **25**(1), 1–21 (2019)
14. Wang, R., Li, H., Lan, R., Luo, S., Luo, X.: Hierarchical ensemble learning for alzheimer's disease classification. In: 2018 7th International Conference on Digital Home (ICDH). pp. 224–229. IEEE (2018)
15. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)