To Ensemble or Not Ensemble: When does End-To-End Training Fail?

Andrew Webb[1], Charles Reynolds[1], Wenlin Chen[1], Henry Reeve[2], Dan Iliescu[3], Mikel Luján[1], and Gavin Brown[1]

[1] University of Manchester, UK
[2] University of Bristol, UK
[3] University of Cambridge, UK

Abstract. End-to-End training (E2E) is becoming more and more popular to train complex Deep Network architectures. An interesting question is whether this trend will continue—are there any clear failure cases for E2E training? We study this question in depth, for the specific case of E2E training an *ensemble* of networks. Our strategy is to blend the gradient smoothly in between two extremes: from independent training of the networks, up to to full E2E training. We find clear failure cases, where overparameterized models *cannot be trained E2E*. A surprising result is that the optimum can sometimes lie in between the two, neither an ensemble or an E2E system. The work also uncovers links to Dropout, and raises questions around the nature of ensemble diversity and multi-branch networks.

1 Introduction

Ensembles of neural networks are a common sight in the Deep Learning literature, often at the top of Kaggle leaderboards, and a key ingredient of now classic results, e.g. AlexNet (Krizhevsky et al., 2012). In recent literature, an equally common sight is *End-to-End* (E2E) training of a deep learning architecture, using a single loss to train all components simultaneously. An interesting scientific question is whether the E2E paradigm has limits, examined in some detail by Glasmachers (2017), who concluded that E2E can be inefficient, and *"does not make optimal use of the modular design of present neural networks"*.

In line with this question, some have explored training an *ensemble* with E2E, as if it was a modular, multi-branch architecture. Dutt et al. (2020) explore E2E ensemble training and demonstrate it allows one to *"significantly reduce the number of parameters"* while maintaining accuracy. Anastasopoulos and Chiang (2018) use the principle to improve accuracy in multi-source language translation. Furlanello et al. (2018) show good performance across a variety of standard benchmarks. This may seem promising, however, Lee et al. (2015) conclude the opposite, that such End-to-End training of an ensemble is *harmful* to generalization accuracy. The idea raises interesting scientific questions about learning in ensemble/modular architectures. Some authors have noted that some architectures (e.g. with multiple branches, or skip connections) may be viewed as an ensemble of jointly trained sub-networks, e.g. ResNets (Veit et al., 2016; Zhao et al., 2016). It is widely accepted that “diversity” of *independently* trained ensembles is beneficial (Dietterich, 2000; Kuncheva and Whitaker, 2003). and hard evidence lies in the success of the many published variants of Random Forests and Bagging (Breiman, 1996). But if we train a set of networks *End-to-End*, they share a loss function—so their parameters are strongly correlated—seemingly the opposite of diversity. There is clearly a subtle relation here, raising hard questions, e.g., what is the meaning of ‘diversity’ in E2E ensembles?

Our aim is to understand when each scheme—*Independent* ensemble training, or *End-to-End* ensemble training—is appropriate, and to shed light on previously reported results.

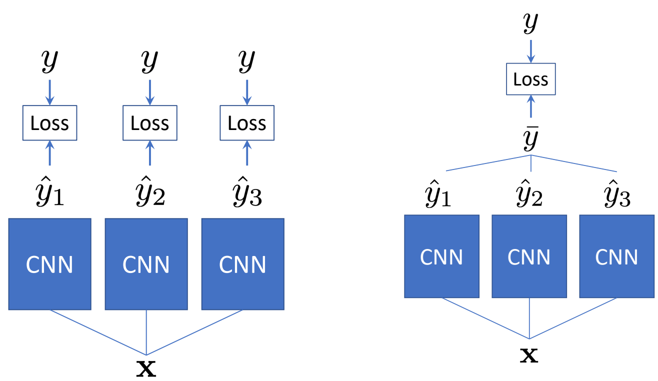


Fig. 1. Computational graphs representing the training of a CNN ensemble: Independent (Left) versus End-to-End training (Right). We study the dynamics of learning when interpolating smoothly in-between these two extremes.

Our strategy is to interpolate the gradient smoothly between the two—a convex combination of Independent and End-to-End training. This uncovers complex dynamics, highlighting a tension between individual model capacity and diversity, as well as generating interesting properties, including ensemble robustness with respect to faults.

Organisation of the Paper:

In Section 2 we cover the necessary notation and probabilistic view we adopt for the paper. In Section 3 we ask and expand upon the reasoning behind the primary question of the paper: *is it an ensemble of models, or one big model?*. This leads us to define a hybrid training scheme, which turns out to generalise some previous work in ensemble diversity training schemes. In Sections 4 and 5 we thoroughly investigate the question, uncovering a rich source of unexpectedly complex dynamics. Finally in Sections 6 and 7 we outline how this connects to a wide range of work already published, and opens new doors for research.

2 Background and Design Choices

We assume a standard supervised learning scenario, with training set $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ drawn i.i.d. from $P(\mathcal{X}\mathcal{Y})$. For each observation $\mathbf{x}$, we assume the corresponding y is a class label drawn from a true distribution $p(y|\mathbf{x})$. We approximate this distribution with a model $q(y|\mathbf{x})$, corresponding to some deep network architecture. We will explore a variety of architectures: including simple MLPs, medium/large scale CNNs, and finally DenseNets (Huang et al., 2018).

To have an ensemble, there is a choice to make in how to combine the network outputs. In our work we choose to average the network logits, and re-normalise with softmax. This was used by Hinton et al. (2015), and Dutt et al. (2020)—we replicate one of their experiments later. There are of course alternatives, e.g. majority voting or averaging probability estimates. Majority voting rules out the possibility of E2E ensemble training by gradient methods, since the vote operation is inherently non-differentiable. Averaging probability estimates is common, but we present a result below that encouraged us to consider the averaging logits approach instead, beyond simply replicating Dutt et al. (2020).

We approach the classification problem from a probabilistic viewpoint: minimising the KL-divergence from a model q to the target distribution p. Given a set of such probability models, we could ask, what is the optimal combiner rule that preserves the contribution from each, in the sense of minimising KL divergence? In a formal statement we refer to this as the 'central' model, denoted $\bar{q}$, that lies at the centre of the set of probability estimates:

$$\bar{q} = \arg\min_z \Big[\frac{1}{M}\sum_{i=1}^{M} D(z \mid\mid q_i)\Big] \;=\; \arg\min_z \Big[\frac{1}{M}\sum_{i=1}^{M} \int z(y|\mathbf{x}) \log \frac{z(y|\mathbf{x})}{q_i(y|\mathbf{x})}\, dy\Big] \quad (1)$$

It can easily be proved that the minimizer here is the normalized geometric mean, corresponding to a Product of Experts model, though modeling only the means (i.e. a PoE of Generalized Linear Models):

$$\bar{q}(y \mid \mathbf{x}) \overset{\text{def}}{=} Z^{-1} \prod_{j=1}^{M} q_i(y \mid \mathbf{x})^{1/M}. \quad (2)$$

This can be written in terms of the distribution's canonical link f and its inverse f^{-1}. The inverse link for the Categorical distribution is the softmax $f^{-1}(\boldsymbol{\eta})$, where $\boldsymbol{\eta}$ is a vector of logits. Correspondingly, the logits are given by the link applied to class probabilities $\boldsymbol{\eta} = f(q(y|\mathbf{x}))$.

$$\bar{q}(y|\mathbf{x}) = f^{-1}\Big(\frac{1}{M}\sum_i f(q_i(y|\mathbf{x}))\Big) \quad (3)$$

i.e. a softmax operation on the averaged logits — this provides additional motivation for the ensemble combination rule used by Hinton et al. (2015) and Dutt et al. (2020), since it is the rule that preserves the most information from the individual models, in the sense of KL divergence.

3 Is it an Ensemble, or one Big Model?

Ensemble methods are a well-established part of the ML landscape. A traditional explanation for the success of ensembles is the 'diversity' of their predictions, and particularly their errors. When we reach the limit of what a single model can do, we create an ensemble of such models that exhibit 'diverse' errors, and combine their outputs. This *diversity* can lead to the individual errors being averaged out, and overall lower ensemble error obtained.

A common heuristic is to have the individual models intentionally lower capacity than they might be, and compensate via diversity. Traditional ensemble methods, such as Bagging and Random Forests, generate diversity via randomization heuristics such as feeding different bootstraps of training data to each model (Brown et al., 2005a). *Stacking* (Wolpert, 1992) is similar in spirit to E2E ensemble training, in that it trains the combiner function, although only after individual models are fixed, using them as meta-features.

However, if the ensemble combining procedure was fully *differentiable*, we could in theory train all networks End-to-End (E2E), as if they were branches or components of one "big model". In this case, what role is there for *diversity*? Furthermore, with modern deep networks, it is easy to have extremely high capacity models, but regularised to avoid overfitting. With this in mind, is there much benefit to ensembling deep networks? Various empirical successes, e.g. Krizhevsky et al. (2012), suggest a tentative "yes", but understanding the overfitting behaviour of deep network architectures is one of the most complex open challenges in the field today. It is now common to heavily *over-parameterize* and regularize deep networks. These observations raise interesting questions on the benefits of such architectures, and for ensemble diversity.

To study the question of failure cases for E2E ensembles, we could simply train a system E2E and report outcomes. However, to get a more detailed picture, we define a hybrid loss function, *interpolating between* the likelihood for an independent ensemble and the E2E ensemble likelihood. We refer to this as the *Joint Training* loss, since it treats the networks jointly as components of a single (larger) network, or as members of an ensemble. We stress that we are not advocating this as a means to achieve SOTA results, but merely as a forensic tool to understand the behaviour of the E2E paradigm.

Definition 1 (*Joint Training Loss*):

$$L_\lambda \stackrel{\text{def}}{=} \lambda \mathrm{D}(p \,\|\, \bar{q}) + (1-\lambda)\frac{1}{M}\sum_{j=1}^{M} \mathrm{D}(p \,\|\, q_j)\,, \tag{4}$$

where D is the KL-divergence as defined before. The loss is a convex combination of two extremes: $\lambda = 1$, where we train $\bar{q}$ as one system, and $\lambda = 0$, where we train the ensemble independently (with a learning rate scaled by $1/M$). When λ lies between the two, it could be seen as $\bar{q}$ being 'regularized' by the individual networks partially fitting the data themselves. Alternatively, we can view the same architecture as an *ensemble*, but trained *interactively.* This is made clear by rearranging Equation (4) as:

$$L_\lambda = \frac{1}{M}\sum_{j=1}^{M} \mathrm{D}(p \,\|\, q_j) - \frac{\lambda}{M}\sum_{j=1}^{M} \mathrm{D}(\bar{q} \,\|\, q_j) \ . \tag{5}$$

A simple proof is available in supplementary material. This alternative view shows that λ controls a balance between the average of the individual losses, and a term measuring the *diversity between the ensemble members.*

From this perspective, this is a diversity-forcing training scheme similar to previous work Liu and Yao (1999). In fact for the special case of a Gaussian p, q, Eq (5) is *exactly* that presented by Liu and Yao (1999). However, the probabilistic view allows us to generalise, and for the case of classification problems, this can be seen as *managing the diversity in a classification ensemble.* However, seen as a 'hybrid' loss, (4), explicitly varies the gradient in-between the two extremes of an ensemble and a single multi-branch architecture, thus including architectures studied previously (Lee et al., 2015; Dutt et al., 2020) as special cases. In the following section we use this to study the learning dynamics in terms of tensions between model capacity and diversity.

4 Experiments

End-to-End training of ensembles raises several interesting questions: How does End-to-End training behave when varying the capacity/size of networks? Should we have a large number of simple networks? Or the opposite—a small number of complex networks? What effect does varying λ have in this setting, smoothly varying from Independent to End-to-End training? Suppose we have a high capacity, well-tuned *single* network, performing close to SOTA. In this situation, presumably, independent training of an ensemble of such models will still *marginally* improve over a single small model, due to variance reduction. However, it is much less clear in this case whether there will be further gains as $\lambda \to 1$, toward End-to-End training. We investigate this first with simple MLPs, exploring trade-offs while the number of learnt parameters is held constant, and then with higher capacity CNNs, including DenseNets (Huang et al., 2017).

Spending a Fixed Parameter Budget. We first compare a single large network to an ensemble of very small networks: each with the *same* number of parameters. The study of such architectures may have implications for IoT/Edge compute scenarios, with memory/power constraints. We set a budget of memory or number of parameters, and ask how best to "spend" them in different architectures. This type of trade-off is well recognised as highly relevant in the current energy-conscious research climate, e.g. very recently Zhu et al. (2019). We stress again however that we are not chasing state of the art performance, and instead observe the dynamics of Independent vs E2E training in a controlled manner.

We use single layer MLPs in four configurations with ∼815K parameters: a single network with 1024 nodes (1×1024H), 16 networks each with 64 nodes (16×64H), 64 networks with 16 nodes (64×16H), and 256 networks with 4 nodes (256×4H). We evaluate these on the 10-class classification problem, Fashion-MNIST. Full experimental details are in the supplementary material.

Figure 2 summarizes results for the 1×1024H network versus the 256×4H ensemble. Again, these have the same number of configurable parameters, just deployed in a different manner. The 'monolithic' 1024 node network (horizontal dashed line, 10.2% error) significantly outperforms the ensemble trained independently (16.3%), as well as the 'classic' methods of Bagging and Stacking. However, *End-to-End training* of the small network ensemble almost matches performance, and Joint Training ($\lambda = 0.95$) gets closer with 10.3% error.

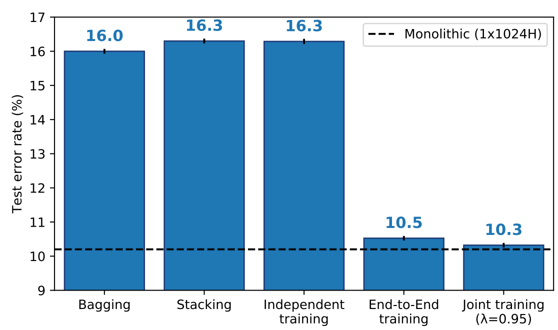


Fig. 2. Large single network vs. small net ensemble. Joint Training the ensemble ($\lambda = 0.95$) comes closest to match the large network accuracy, while classic ensemble methods significantly underperform.

Figure 3 shows detailed results, varying λ on other configurations. The distinction between End-to-End training and independent training is most pronounced for the large ensemble of small networks, 256×4H, where *E2E training is sub-optimal*.

However, as the capacity of the networks increases, the difference between End-to-End and independent training vanishes — illustrated by the almost uniform response to varying λ with larger networks. This is investigated further in the next section with state-of-the-art *DenseNet* networks (Huang et al., 2017) with millions of parameters.

High Capacity Individual Models. In deep learning, a recent effective strategy is to massively overparameterize a model and then rely on the implicit regularizing properties of stochastic gradient descent. We ask, is there anything to be gained from End-to-End training of such large models? What does the concept of diversity even mean when the models have almost zero training error? We address this by examining ensembles of DenseNets (Huang et al., 2017), which achieve close to SOTA at the time of writing. We train a variety of DenseNet ensembles

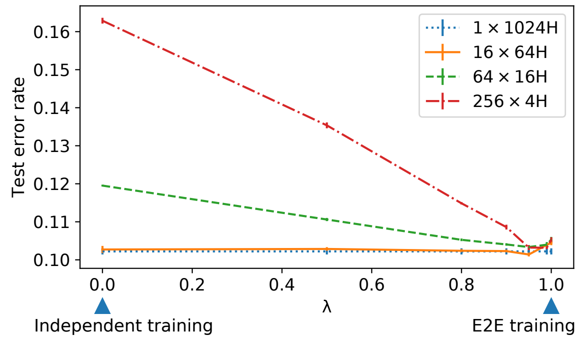


Fig. 3. Error rates for MLPs with 815K parameters, on Fashion-MNIST. Joint Training with higher λ values improves the performance of small network ensembles, matching the performance of a single large network.

on the CIFAR-100 dataset, with the size of the ensemble increasing as the complexity of each ensemble member decreases, such that each configuration occupies approximately 12GB of GPU RAM. These configurations are described in Table 1. Full experimental details can be found in the supplementary material.

Table 1. DenseNet ensembles. A proxy measure of capacity is (d, k), the depth d and growth rate k, which we decrease as the size M of the ensemble increases. Each architecture occupies approximately 12GB RAM.

Name	**Depth**	$\boldsymbol{k}$	$\boldsymbol{M}$	**Memory**	**Params.**
DN-High	100	12	4	∼12GB	3.2M
DN-Mid	82	8	8	∼12GB	2.1M
DN-Low	64	6	16	∼12GB	1.7M

Table 2 shows results for independent, End-to-End, and Joint training, vs. Bagging/Stacking. Each row contains results for a DenseNet ensemble, with minimum error rate in bold. Results for DN-High replicate the results of Dutt et al. (2020, Table 2).

In the previous section, we saw End-to-End training outperforming independent training for large ensembles of very small models. Here, *we find the opposite is true for small ensembles of large, SOTA models.* In every DenseNet configuration, *End-to-End training (i.e. $\lambda = 1$) is sub-optimal.* In all but DN-Low, which is the configuration with the largest ensemble of smallest models, independent training ($\lambda = 0$) achieves the lowest test error. **These results indicate there is little to no benefit in test error when E2E ensemble training SOTA deep neural networks**. However, the result for DN-Low suggests that there

Table 2. Error rates (%) for DenseNet ensembles. Independent training is optimal (bold) for all but the smallest capacity networks.

	DN-Low	DN-Mid	DN-High
$\lambda = 0.0$ (Independent)	25.0	**19.9**	**17.8**
$\lambda = 0.5$	**23.5**	20.0	18.8
$\lambda = 0.9$	25.9	22.4	20.3
$\lambda = 1.0$ (End-to-End)	29.6	25.7	22.1
Bagging	32.5	29.5	28.4
Stacking	28.3	23.4	21.4

may be a relatively smooth transition, somewhere between E2E ensembles and independent training, as the complexity of the ensemble members increases.

Intermediate capacity. In this section we further explore the relationship between ensemble member complexity and E2E ensemble training. We train ensembles of convolutional networks on CIFAR-100. Each ensemble has 16 members of varying complexity, from $70,000$ parameters up to 1.2 million. Full details can be found in the supplementary material. Note that these experiments are not intended to be competitive with the SOTA, but to illustrate relative benefits of E2E ensembles as the complexity of individual members varies.

Figure 4 shows the test error rate and standard errors for each λ value for each configuration. The 'dip' in each of the four lines in the figure is the optimal λ. We find that the optimal point smoothly decreases: E2E ensembles are the preferred option with simpler networks, but this changes as the networks become more complex. At 1.2M parameters, the optimal point lies clearly *between* the two extremes. This supports the trend observed in the previous sections, that the benefits of E2E training as a scheme depend critically on the complexity of individual networks.

When does End-to-End ensemble training fail? These results suggest a trend: ensembles of low capacity models benefit from E2E training, and high capacity models perform better if trained independently. We suggest a possible explanation for this trend, and a diagnostic for determining during training whether E2E training will perform well. We find that ensembles of high capacity models, when trained E2E, exhibit a 'model dominance' effect, shown in Figure 6. There is a sharp transition in behaviour at $\lambda = 1$ (E2E), whereby a single ensemble member individually performs well—in both training and test error—while all other members perform poorly. This effect is not observed in ensembles of low capacity models.

We suggest that this dominance effect can explain the poor performance of E2E training seen in Table 2. We also note that model dominance occurs in the E2E training experiments of Dutt et al. (2020, Table 2) (called 'coupled training (FC)' by the authors); the authors report the average and standard deviation (within a single trial) of the ensemble member error rates. In the case with 2

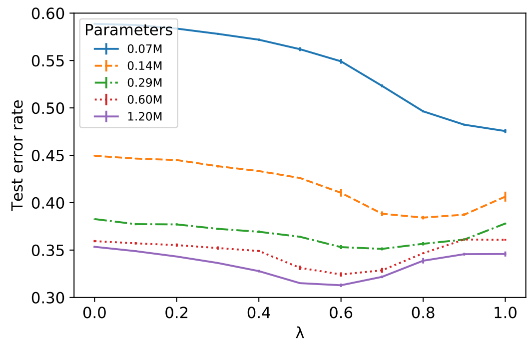


Fig. 4. Ensemble test error against λ for ensembles of convolutional networks of varying complexity on CIFAR-100. The optimal λ value shifts away from E2E ensemble training with greater individual network capacity.

ensemble members, it can be inferred that one ensemble member achieves a much lower error rate than the other.

In our own experiments, the dominance effect manifests early in training (Figure 6). In each trial, the ensemble member with the lowest error rate by the end of epoch 3 dominates by the end of training; model dominance can be used as an early diagnosis during training that the ensemble members are overcapacity for E2E training. Note that model dominance can occur in classification because the ensemble prediction is a normalized geometric mean; a network closely fitting the data can arbitrarily reduce the ensemble error—the well-known 'veto' effect in Products of Experts (Welling, 2007)—and the parameters of other networks can be prevented from moving far from their initial values. This 'stagnation' of other ensemble members can be seen in Figure 5, which shows some of the filters in the first layer of a ConvNet trained independently and E2E, showing a small number of strong filters in the latter case.

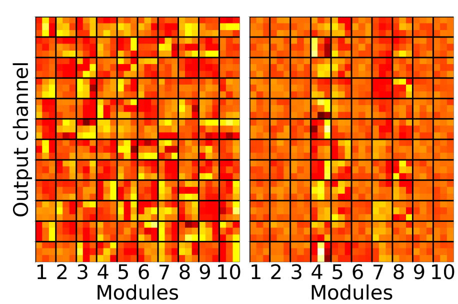
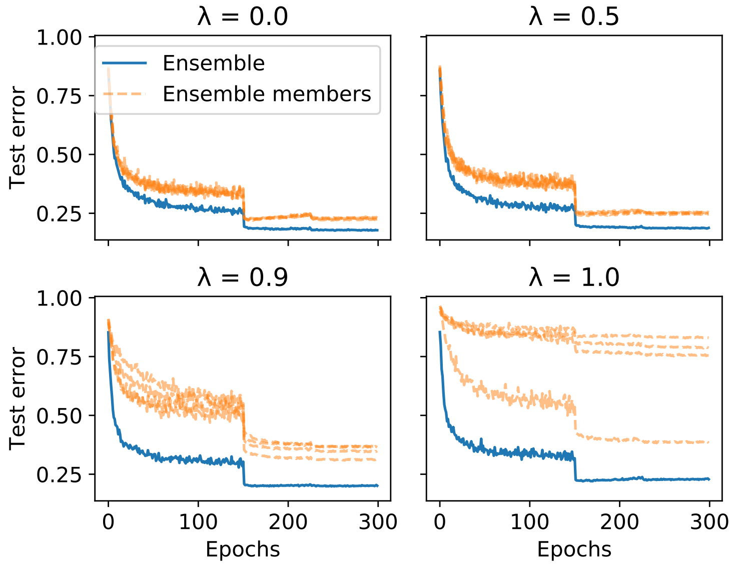


Fig. 5. Model dominance in filters in the first layer of 10 CNN modules. Trained independently (left) and E2E (right). In E2E training, only network 4 has strong filters.



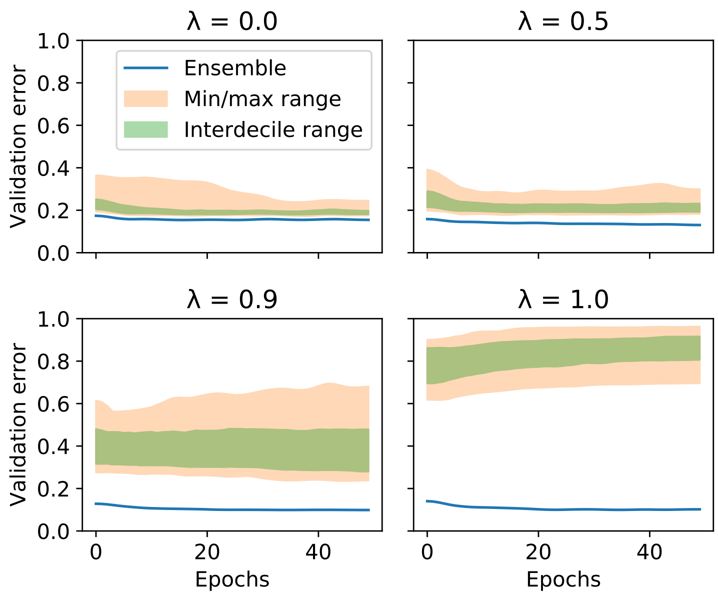


Fig. 6. Ensemble vs Member error: Over-capacity models (DenseNets, top sub-figure) experience a 'model dominance' effect when trained E2E, whereas there is no such effect with 256 × 4H undercapacity models (bottom sub-figure).

5 What happens *between* E2E and independent training?

The previous section has demonstrated clearly that E2E training of an ensemble can be sub-optimal. More interestingly perhaps, the complex behaviour seems to lie between independent and E2E. This section goes beyond just ensemble test error rate, and looks into properties of the networks *inside* the ensemble.

Robustness. We find that Joint Training with λ very slightly less than 1 can greatly increase the *robustness* of an ensemble, in the sense that dropping the response from a subset of the networks at test time does not significantly harm the accuracy. This *partial evaluation*, performing inference for only a subset of the networks, may be useful in resource-limited scenarios, e.g. limited power budgets in Edge/IoT devices.

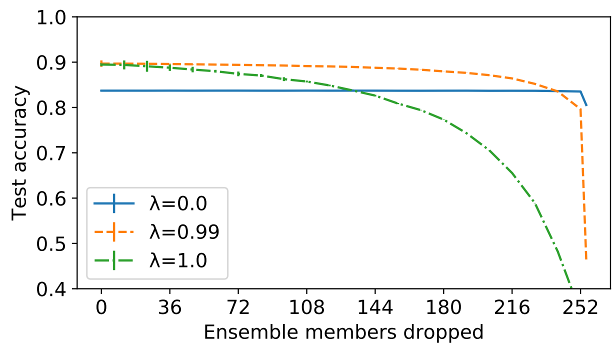


Fig. 7. Independent training ($\lambda = 0$) gives robust ensembles; however, many members are redundant. End-to-End training ($\lambda = 1$) gives brittle ensembles. Joint Training with $\lambda \approx 1$ (but strictly less than 1) retains the accuracy whilst adding robustness.

Figure 7 analyses the 256×4H ensemble, dropping a random subset of networks for each test example in Fashion-MNIST, averaged over 20 repeats. Independent training ($\lambda = 0$) builds a highly robust ensemble, but which could also be seen to be redundant—adding more networks does not increase accuracy. End-to-End training ($\lambda = 1$) achieves higher test accuracy, but the ensemble is not at all robust. Joint Training with $\lambda = 0.99$ achieves the same accuracy as End-to-End with all members, but can maintain accuracy even if more than 50% of the networks are dropped. A similar behaviour is observed with the 64×16H architecture, in Figure 8.

Robust ensembles: why does this occur? When training independently, the networks have no "idea" of each others' existence—so solve the problem as individuals—and hence the ensemble is highly robust to removal of networks.

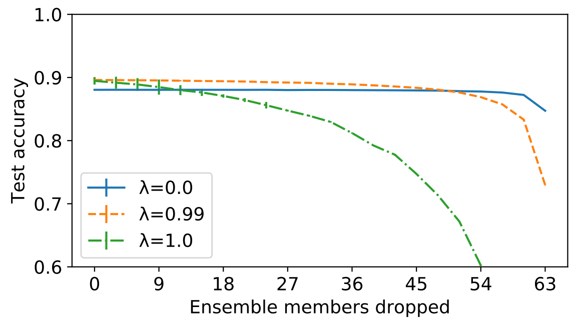


Fig. 8. Robustness with the 64×16H architecture. The same robustness appears, but relative benefit is less, as the independently trained ensemble can perform better.

When we *End-to-End* train the *exact same architecture*, all networks are effectively "slave" components in a larger network. They cannot directly target their own losses, so individual errors would not be expected to be very low. However, as a system, they work together, and achieve low ensemble error. The problem is that the networks *rely* on the others too much—so when removing networks, the performance degrades rapidly. This is illustrated in Figure 9—we see individual network loss rapidly increasing as we approach End-to-End training (i.e. $\lambda = 1$).

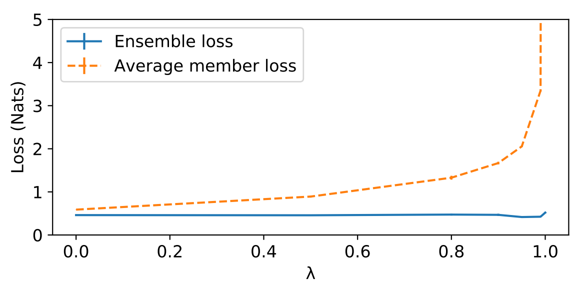


Fig. 9. 64×16H: ensemble vs. average loss. The average grows sharply as $\lambda \to 1$, a symptom of the individuals relying on others to correct mistakes.

The "sweet spot" is a small amount of *joint training* with $\lambda = 0.99$, which ensure the networks cannot rely on each other completely, but are still working together. This is highly reminiscent of *Dropout*, where one of Hinton's stated motivations was to make each *"hidden unit more robust and drive it towards creating useful features on its own without relying on other hidden units to correct its mistakes."* (Srivastava et al., 2014, Section 2). This similarity is not a coincidence, and is discussed in the next section.

6 Discussion & Future Work

The primary question for this paper was *"when does E2E ensemble training fail?"*, with the intention to explain conflicting results in the literature, e.g. Lee et al. (2015); Dutt et al. (2020). The answer turns out to be intimately tied with the over-parameterization (or not) of individual networks in the ensemble. Dutt et al. (2020); Lee et al. (2015) concluded that End-to-End training of ensembles leads to poor performance—however, only very large state-of-the-art models were studied. Our results agree with theirs *exactly*, in situations where ensemble members are over capacity—Table 2 reproduces the results of Dutt et al. (2020, Table 2)—but, the advantage of E2E emerges with lower capacity individuals, where there are also interesting and potentially useful dynamics in-between Independent/E2E training.

Relations to prior work. The investigation turned up links to literature going back over 20 years, in terms of ensemble diversity (Heskes, 1998). As mentioned, we presented results primarily on classification problems—however, the framework presented in Section 2 applies generally for targets following any exponential family distribution. For the special case of a Gaussian, and when we re-arrange the loss as (5), the gradient is *exactly* equivalent to a previously proposed method, Negative Correlation Learning (NCL) (Liu and Yao, 1999; Brown et al., 2005b). Thus, an alternative view of the loss is as a generalization of NCL to arbitrary exponential family distributions, where the Categorical distribution assumption here can be seen as *managing diversity in classification ensembles*, echoing the title of Brown et al. (2005b).

The Dropout connection? We observed a qualitative similarity between ensemble robustness, and the motivations behind Dropout (Srivastava et al., 2014). It is interesting to note that Negative Correlation Learning Liu and Yao (1999) can be proven equivalent to a stochastic dropping of ensemble members during training (Reeve et al., 2018), i.e. Dropout at the network level. With some investigation we have determined that their result depends critically on the Gaussian assumption and hence holds only for NCL, not more generally for Joint Training loss with any exponential family. Despite this, the observations in Section 5 do suggest a connection, worthy of future work.

Should we always E2E train Multi-Branch networks? The End-to-End training methodology treats the ensemble as if it were a single multi-branch architecture. A similar view could be taken of *any* multi-branch architecture, e.g. ResNeXt (Xie et al., 2017) that it is either an ensemble of branches, or a single system. We have found that if ensemble members are over capacity, then one network can 'dominate' an ensemble, leading to poor performance overall. This raises the obvious open question, of whether it is also true for sub-branches in general branched architectures like ResNeXt.

Ill-conditioning? As $\lambda \to 1$, the condition number of the Hessian for the JT Loss tends to infinity, as we show in the supplementary material. Therefore there may be cases where E2E training with first-order optimization methods (e.g. simple SGD) performs poorly, and where second-order methods behave markedly differently. We leave this question for future study.

7 Conclusions

We have presented a detailed analysis of the question *"When does End-to-End Ensemble training fail?"*. This is in response to the recent trend of end-to-end training being used more and more in deep learning, and specifically for an ensemble (Lee et al., 2015; Furlanello et al., 2018; Dutt et al., 2020). Our strategy for this was to study a convex combination of the likelihood-based losses for independent and E2E training, blending the gradient slowly from one to the other.

Our answer to the question is that End-to-End training tends to underperform when member networks are significantly over-parameterized, though it is possible to diagnose whether this will happen by examining the relative network training errors in the first few epochs. In this case we suggest alternative classical methods such as Bagging, or Independent training. Further, we conjecture that this may have implications for general multi-branch architectures—should we always train them E2E, or perhaps individually?

The space between independent and E2E turned out to be a rich source of quite unexpectedly complex dynamics, generating robust ensembles, with links to Dropout, general multi-branch deep learning, and early literature on ensemble diversity. We suggest the book is not yet closed and perhaps there is much more to learn in this space.

Acknowledgements

The authors gratefully acknowledge the support of the EPSRC for the LAMBDA project (EP/N035127/1).

Bibliography

Anastasopoulos, A. and Chiang, D. (2018). Leveraging translations for speech transcription in low-resource settings. *arXiv:1803.08991*.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24(2):123–140.

Brown, G., Wyatt, J., Harris, R., and Yao, X. (2005a). Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5 – 20.

Brown, G., Wyatt, J. L., and Tiňo, P. (2005b). Managing diversity in regression ensembles. *Journal of Machine Learning Research*, 6:1621–1650. `dl.acm.org/citation.cfm?id=1046920.1194899`.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

Dutt, A., Pellerin, D., and Quénot, G. (July 2020). Coupled ensembles of neural networks. *Neurocomputing*, 396:346–357.

Furlanello, T., Lipton, Z. C., Tschannen, M., Itti, L., and Anandkumar, A. (2018). Born again neural networks. *arXiv:1805.04770*.

Glasmachers, T. (2017). Limits of end-to-end learning. *arXiv preprint arXiv:1704.08305*.

Heskes, T. (1998). Selecting weighting factors in logarithmic opinion pools. In *NIPS*, pages 266–272. The MIT Press.

Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv:1503.02531*.

Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., and Weinberger, K. (2018). Multi-scale dense networks for resource efficient image classification. In *ICLR*. `openreview.net/forum?id=Hk2aImxAb`.

Huang, G., Liu, Z., v. d. Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *CVPR*, pages 2261–2269.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *NIPS*.

Kuncheva, L. I. and Whitaker, C. J. (2003). Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207.

Lee, S., Purushwalkam, S., Cogswell, M., Crandall, D., and Batra, D. (2015). Why m heads are better than one: Training a diverse ensemble of deep networks. *arXiv preprint arXiv:1511.06314*.

Liu, Y. and Yao, X. (1999). Ensemble learning via negative correlation. *Neural networks*, 12 10:1399–1404.

Reeve, H., Mu, T., and Brown, G. (2018). Modular dimensionality reduction. In *European Conference on Machine Learning*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, pages 1929–1958. `jmlr.org/papers/v15/srivastava14a.html`.

Veit, A., Wilber, M. J., and Belongie, S. (2016). Residual networks behave like ensembles of relatively shallow networks. In *Advances in neural information processing systems*, pages 550–558.

Welling, M. (2007). Product of experts. *Scholarpedia*, 2(10):3879. revision #137078.

Wolpert, D. H. (1992). Stacked generalization. *Neural networks*, 5(2):241–259.

Xie, S., Girshick, R., Dollár, P., Tu, Z., and He, K. (2017). Aggregated residual transformations for deep neural networks. In *CVPR*, pages 5987–5995.

Zhao, L., Wang, J., Li, X., Tu, Z., and Zeng, W. (2016). On the connection of deep fusion to ensembling. Technical Report MSR-TR-2016-1118.

Zhu, S., Dong, X., and Su, H. (2019). Binary ensemble neural network: More bits per network or more networks per bit? In *CVPR*, pages 4923–4932.