The Dynamics of Functional Diversity throughout Neural Network Training

Lee Zamparo
Service Now
lee.zamparo@servicenow.com

Marc-Étienne Brunet
Service Now
marc.brunet@servicenow.com

Thomas George
MILA
georgeth@mila.quebec

Sepideh Kharaghani
Service Now
sepideh.kharaghani@servicenow.com

Gintare Karolina Dziugaite
Google Research, Brain team
gkdz@google.com

Deep ensembles offer reduced generalization error and improved predictive uncertainty estimates. These performance gains are attributed to functional diversity among the component models that make up the ensembles: ensemble performance increases with the diversity of the components. A standard way to generate a diversity of components is to train multiple networks on the same data, using different minibatch orders, augmentations, etc. In this work, we focus on *how and when* this type of diversity in the learned predictor decreases throughout training.

In order to study the diversity of networks still accessible via SGD after t iterations, we first train a single network for t iterations, then duplicate the state of the optimizer and finish the remainder of training k times, with independent randomness (minibatches, augmentations, etc) for each duplicated network. The result is k distinct networks whose training has been *coupled* for t iterations. We use this methodology—recently exploited for $k = 2$ to study linear mode connectivity—to construct a novel probe for studying diversity.

We find that coupling k for even a few epochs severely restricts the diversity of functions accessible by SGD, as measured by the KL divergence between the predicted label distributions as well as the calibration and test error of k-ensembles. We also find that the number of forgetting events [1] drops off rapidly.

The amount of independent training time decreases with coupling time t however. To control for this confounder, we study extending the number of iterations of high-learning-rate optimization for an additional t iterations post-coupling. We find that this does *not* restore functional diversity.

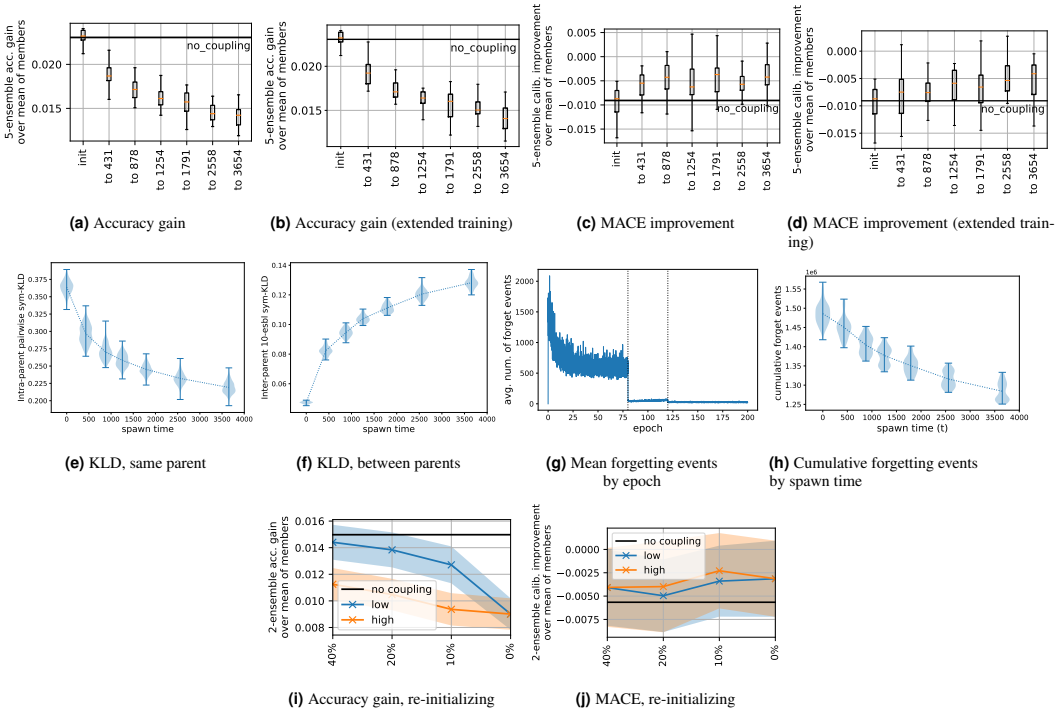
We also study how functional diversity is affected by retraining after reinitializing the weights in some layers. We find that we recover significantly more diversity by reinitializing layers closer to the input layer, compared to reinitializing layers closer to the output. In this case, we see that reinitialization upsets linear mode connectivity. This observation agrees with the performance improvements seen by architectures that share the core of a network but train multiple instantiations of the input layers [2].

1 Functional diversity and neural network training

Preliminaries. We work with neural network models trained on standard image classification datasets. The main results are presented for a ResNet-20 trained on CIFAR-10 but we obtain similar results on a ResNet-50 and a vision transformer on Tiny-Imagenet. Given K individual predictors, we define the *K-ensemble* as the predictor that averages predicted class probabilities over these individual predictors, or *components*. To measure ensemble performance, we use test error as well as *mean average calibration error (MACE)* [3].

Functional Diversity. It is well known that random initialization, minibatch order, and GPU nondeterminism cause SGD to produce different predictors across runs [4, 5]. We identify different

Bayesian Deep Learning workshop, NeurIPS 2021.



(a) Accuracy gain **(b)** Accuracy gain (extended training) **(c)** MACE improvement **(d)** MACE improvement (extended training)

(e) KLD, same parent **(f)** KLD, between parents **(g)** Mean forgetting events by epoch **(h)** Cumulative forgetting events by spawn time

(i) Accuracy gain, re-initializing **(j)** MACE, re-initializing

Figure 1: *First row:* Diminishing accuracy gain **(a, b)** and calibration error **(c, d)** as measured by MACE (lower is better), as we increase coupling time t in `coupled-to-t` and `coupled-to-t-reset` (named *extended training* in the plot) conditions. *Second row:* **(e)** The sym. KL divergence (computed point-wise and averaged) between components sampled from `coupled-to-t-reset` condition. **(f)** sym. KL divergence between two 10-ensembles that do not share the coupled part in `coupled-to-t-reset` condition. **(g)** Average number of forgetting events per epoch (on the test set) in 10 training runs with different data orders. **(h)** Cumulative forgetting events (on the test set) over the first 50 epochs after coupling. *Third row*: Diminishing accuracy gain **(i)** and calibration error **(j)** as we reinitialize a diminishing percentage of the network's parameters starting from low (blue) or high (orange) layers. The shaded area is ± standard deviation.

regimes of interest based on previous work that introduce the most variability in the learned predictors, based on the disagreement on the test data prediction errors.

We group learned components (neural network predictors) based on the induced distributions on the weights, as we condition on some of the algorithm inputs. We define the following conditions:

- `no-coupling`: components are trained with fully independent training runs for T iterations with the same optimizer settings, but different initializations and mini-batch orders;
- `coupled-order`: as above, but share the same mini-batches;
- `coupled-to-t`: the initialization and the first t mini-batches are shared. From $t+1$ onwards the mini-batches differ. Trained for T iterations total. A special case $t = 0$ means that only the random init is shared;
- `coupled-to-t-reset`: init and the first t mini-batches are shared. At $t+1$, reset the optimiser (momentum and learning rate schedule), then train for another T iterations different mini-batch order. Each component is trained for $T + t$ iterations.

The `coupled-to-t-reset` condition is our main focus in the experimental sections below. Under this condition we eliminate the confounder of a decreased learning time after coupling, while minimally affecting the performance of the *components* in our ensembles.

2 The effect of coupling on ensemble performance

We begin by examining how various coupling conditions affect the performance of K-ensembles.

One basic finding is that the `coupled-order` and `no-coupling` conditions achieve indistinguishable performance in error and MACE. This, in turn, suggests that resampling initializations plays a key role in obtaining the type of diversity among predictors that is useful for improving ensemble accuracy.

Looking closely at Fig. 1a, we see that coupling the first t iterations of training decreases ensemble performance as t increases. One could argue that this decrease may be the result of a decrease in the training time at a high learning rate. To test this hypothesis, we replicate the experiment with `coupled-to-t-reset` condition, thus treating the spawning time t as if it were a new initialization, resetting the learning rate schedule and training time ("extended training").

The results from this replication (Fig. 1b) confirm that increased coupling hurts ensemble performance, even under this new condition with extended training times. Further, we rule out another possibility: that these modified training routines produce predictors with inferior performance (Fig. 2). One may interpret these experiments as suggesting that training neural networks from a pre-trained initialization (i.e., weights obtained after coupling for t iterations), narrows the diversity of functions accessible to SGD and its variants in a way that hurts ensemble accuracy.

In summary, training neural networks from a pre-trained initialization indeed narrows the diversity of functions accessible to SGD and its variants in a way that hurts ensemble accuracy. Increased coupling time means that the components are sampled within the same linearly connected mode. Thus our results provide evidence that two linearly connected modes represent a distinct set of functions.

We next measure the difference of predicted label distributions between components sampled from `coupled-to-t-reset` condition. Fig. 1f shows that, with increased coupling time, the symmetric KL divergence between predicted class probability vectors decreases throughout training, most rapidly at the start of training. Another anticipated effect of the decrease in diversity is that the resulting ensembles themselves should become less similar as the coupling time increases. This is indeed what we observe in Fig. 1f. Each ensemble consists of components trained from a fixed parent, but the parents are not shared between the two ensembles. Thus for different parents, the ensembles formed by their respective children are becoming more dissimilar in their predictions with spawning time.

2.1 Early training dynamics

Toneva et al. [1] define a *forgetting event* of an input (x, y) as an event when a gradient update during learning causes the 0–1 loss on (x, y) to increase from 0 to 1. Based on this definition, an example is assigned a *forgetting score* which is (a lower bound on[1]) the number of forgetting events during the training run. A large number of forgetting events over a training/test set illustrates that the decision boundary varies greatly and is being determined. Fig. 1g demonstrates that the total number of forgetting events (averaged over 10 runs) on the test set drops off rapidly early in training and then plateaus when training at a constant learning rate. Subsequent drops are aligned with learning rate drops.

In Fig. 1h, we visualize the forgetting scores summed over the test set, averaged over components sampled from `coupled-to-t-reset` condition. Note that for any coupling time t, the training time over which the forgetting events are summed over is the same (from coupling epoch t until the end of training). However, since the number of forgetting events drops off rapidly during the initial epochs of training, we also see a rapid drop in cumulative forget events with coupling time. We hypothesize that in the regions of interest (around the test points), the decision boundary is somewhat determined early in training and has smaller fluctuations with further training.

3 Layer reinitialization as a way to restore functional diversity

In Section 2 we showed how any non-trivial amount of coupling hurts ensemble performance. Here we ask whether reinitializing a fraction of a partially trained network's weights (after coupling) and then proceeding with training multiple copies with different minibatch orders can restore functional diversity improving ensemble accuracy and calibration.

We reinitialize a fraction of the weights of network trained up to time t. The weights to reinitialize are chosen by one of the following two ways: *low*, meaning that we start with the weights closest to the input, with increasing percentage we move in the direction of the forward pass; *high*, meaning

[1]The forgetting events are only tracked when the example appears in the minibatch during training, and thus the final *forgetting score* is actually a lower bound on the total number of forgetting events.

that we start with the weights closest to the output, and with increasing reinitialization percentage we travel down the backward pass.

Fig. 1i reveals that reinitializing low layers restores significantly more diversity accessible via SGD. However, one needs to reinitialize nearly half of the network ($\approx 40\%$) in order to approach the same ensemble accuracy as under `no-coupling` condition. As seen in Fig. 1j, the effect on model calibration seems to be insignificant. Interestingly, reinitializing the weights also restores the error barrier [6] on the linear path between trained components (Fig. 3).

Our observations also shed some light on why multi-input-multi-output (MIMO) architecture, proposed by [2] works well for nearly matching ensemble performance: MIMO has independent low layers, that are effectively trained with different minibatch orders.

References

[1] M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon. "An empirical study of example forgetting during deep neural network learning". *arXiv preprint arXiv:1812.05159* (2018).

[2] M. Havasi, R. Jenatton, S. Fort, J. Z. Liu, J. Snoek, B. Lakshminarayanan, A. M. Dai, and D. Tran. "Training independent subnetworks for robust prediction". In: *International Conference on Learning Representations*. 2020.

[3] M. P. Naeini, G. Cooper, and M. Hauskrecht. "Obtaining well calibrated probabilities using bayesian binning". In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015.

[4] M. Hardt, B. Recht, and Y. Singer. "Train faster, generalize better: Stability of stochastic gradient descent". In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by M. F. Balcan and K. Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1225–1234.

[5] P. Toulis, D. Tran, and E. Airoldi. "Towards Stability and Optimality in Stochastic Gradient Descent". In: *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*. Ed. by A. Gretton and C. C. Robert. Vol. 51. Proceedings of Machine Learning Research. Cadiz, Spain: PMLR, Sept. 2016, pp. 1290–1298.

[6] J. Frankle, G. K. Dziugaite, D. Roy, and M. Carbin. "Linear mode connectivity and the lottery ticket hypothesis". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 3259–3269.

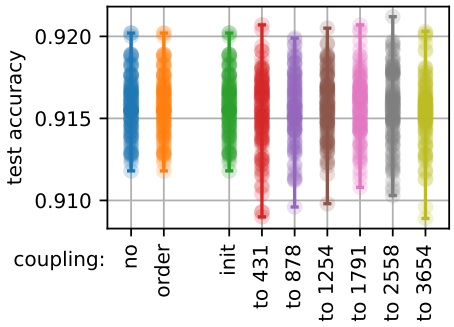


Figure 2: Individual accuracy of all models and conditions (including spawn times). Confirms that we do not observe any performance gap in distribution between coupled init and coupled-to-t for later values of t.

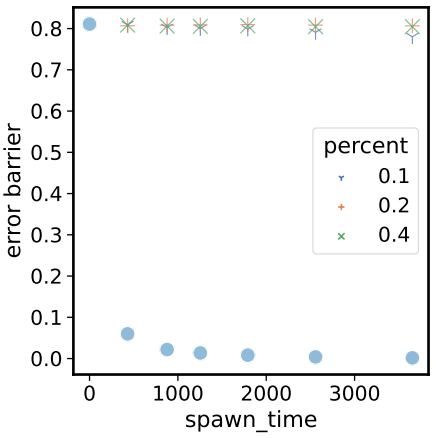


Figure 3: Average error barrier between two child models with a different amount of parameters reinitialized at coupling time t. The blue dots represent no reinitialization. We observe that reinitializing even a small percentage of the weights at time t results in a large expected error barrier between models sharing most of their weights at t.