Meta Neural Networks: Building Neural Networks From Neural Networks

Tom Dörr
tom.doerr@tum.de

Aleksandr Zuev
ge24tav@mytum.de

1. Idea

The current generation of neural networks uses all neurons during inference, which is in stark contrast to how the human brain operates. The brain only activates the regions of the brain that are relevant for interpreting the current input data, instead of activating all available parts of the brain. This contributes to the high efficiency of the brain. To increase efficiency in artificial neural networks, we implement a neural network that consists of multiple smaller neural networks (meta neural network), which more closely resembles the neural networks nature designed.

2. Related work

The concept of Meta Machine Learning (MML) is not new and there are different approaches on how to use multiple classifiers to achieve better results and reduce computation costs. Our work falls into this categorization of MML and can be referred to as Mixture of Experts [3]. Our specialized networks are ”experts” and the Neural Switch is essentially a Gating Network which selects the expert networks. Some previous work focused on enhancing learning algorithms such as Hierarchical mixtures of experts or the EM algorithm [4], but our approach is to assemble a novel architecture using pretrained neural networks. The application of MML using neural networks for image classification showed good results when used with deep mixture of experts via shallow embedding [5]. The mentioned work had some success at the same goal we are aiming for: improving Top-1% accuracy without significant increases in computational cost.

3. Architecture

The meta neural network (MNN) consists of the neural switch and specialized neural networks, which are both just ordinary neural networks. They can be chosen arbitrarily. For our research we decided to use the MobileNetV2 architecture for 10-class classification for all networks, so that we can compare the performance of our MNN architecture to the pretrained MobileNetV2. It's worth emphasizing that we use the 10-class architecture also for Specialized NNs and therefore need a 1-dimensional output probability. We do this by first applying the softmax function, and then pick the element corresponding to the class of this specialized NN.

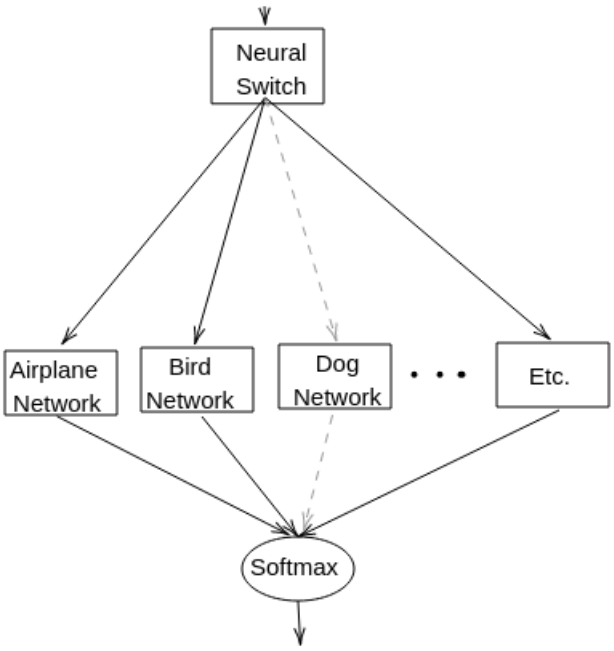

Figure 1. Example for inference in a meta neural networks. The input image is first passed to the neural switch, which then decides what areas to activate. Note that the specialized networks for airplanes and birds are activated, but the one for dogs is not.

Let us consider an example of how the meta neural network runs inference on an image (Fig. 1). The image is first passed to the neural switch network, the resulting probabilities are saved. Instead of using the probabilities to classify the image, they are used to decide which specialized NNs of the MNN to activate by choosing a subset of specialized networks. This task of selecting the specialized networks given the probabilities is accomplished by our switch function, which defines the selection strategy. Our selection strategy is picking those specialized NNs where the probability for the image belonging to the corresponding class is greater than some threshold (we refer to this value as the

switch threshold).

Picking good switch threshold can be non trivial. Keep in mind that the total number of classes is 10, so in the case of equal predictions, all probabilities are 0.1. In the case that 2 classes have equal probabilities and the probabilities for all other classes are zero, the two classes have a probability of 0.5. Therefore, it makes sense to use switch threshold between 0.1 and 0.5 or between 0.1 and 0.33.

After the neural switch decided which specialized networks to activate, the images are passed to the selected networks. To do this efficiently we form a batch for each specialized NN with the corresponding images. Previously saved Switch Network probabilities are multiplied with calculated 1-class probabilities of each selected Specialized NN: since we operate with batches the vector of 10 is constructed so that one of the values is the one predicted by the specialized NN and the other entries are set to equal values so that the sum of vector elements is 1 in order to make it a valid 10-class probability distribution.

The final predictions are made as an argmax operation of the modified probabilities.

The whole approach is based on the idea that specialized networks do not need to waste capacity on recognizing classes that they are not specialized in. Since in this case the classification problem is easier we expect a higher classification accuracy. The switch function should also help to improve accuracy by helping to solve hard classification cases where the probabilities for multiple classes are pretty high. An airplane network will likely be trained on few dog images, because images between those categories are very dissimilar, making the Neural Switch unlikely to activate the airplane network for a dog image. However the categories airplane and bird are more similar, which means that the airplane network will likely receive many bird images and the network can focus on the harder task of distinguishing airplanes from birds. By training the specialized networks in this way, it allows them to focus on the more difficult classification tasks, which we hope improves the Top-1 accuracy of the whole model.

4. Training

4.1. Training Procedure

Since multiple Neural Networks are used in our architecture there are different training sets which are prepared based on the CIFAR10 dataset we started with. The training process consists of 2 stages: Switch Neural Network training and Specialized NNs training.

The Neural Switch is trained on the whole CIFAR10 training set with a 10% subset kept for validation. We use a constant random seed for the results to be reproducible. This part is just normal training of a neural network, the loss used is categorical cross entropy - a natural choice of 10-classes classification.

For specialized NNs, receiving only a portion of all training samples which is determined by Neural Switch and the switch function, the training process is slightly more complicated. At first, in order to perform training of the specialized NNs we need to determine a subset of images which will be passed as an input to each specialized NN. We use the Neural Switch to evaluate all images in the training set and write the output after the softmax layer as probabilities to a file.

Secondly, for each Specialized NN we can determine its training set. The training set is a subset of the original training set where samples are chosen to be passed to this NN by a switch function parameterized with the switch threshold. This way, each NN is trained on the samples which would be passed to it during evaluation of the entire dataset by the whole MNN architecture. Every specialized NN is trained in the same way, the only difference is the target parameter which selects samples from the dataset. Since in this research we use the same architectures for all networks, we start with the weights of the Switch Network as starting point which saves training time.

4.2. Theoretical Advantages

Training this way allows for extreme parallelization. It even enables the training networks with multiple Billion parameters in under three minutes. For example, it takes 2:43 minutes to train a ResNet-50 to 93% accuracy using 16 nodes with InfiniBand (8*V100 with NVLink for each node) [1]. Using 1000 instances of this type for the 1000 ImageNet classes and doing 10% of the training iterations of the initial training for the specialization phase yields an overall training time of 2:59 minutes. For the specialization phase, only few iterations are needed, since the specialized networks are only trained on a subset of the ImageNet data. The overall number of GPUs is 1000 instances * 16 nodes * 8 GPUs = 128000 GPUs. Since a single ResNet-50 network has around 25 million parameters [6], the meta neural network with 1001 of those networks is $\tilde{2}5$ Billion parameters in size.

Furthermore, having completely decoupled training runs for the specialized networks enables the training to happen on multiple kinds of GPUs in multiple datacenters and therefore allows for the use of cheap spot instances around the globe.

5. Evaluation

The process of running inference was described in the section "Architecture" from the algorithmic perspective, now we will focus on the results.

CIFAR10 test set is used for measuring Top-1 accuracy. As for baseline, we have a value of 93.21% for the usual MobileNetV2.

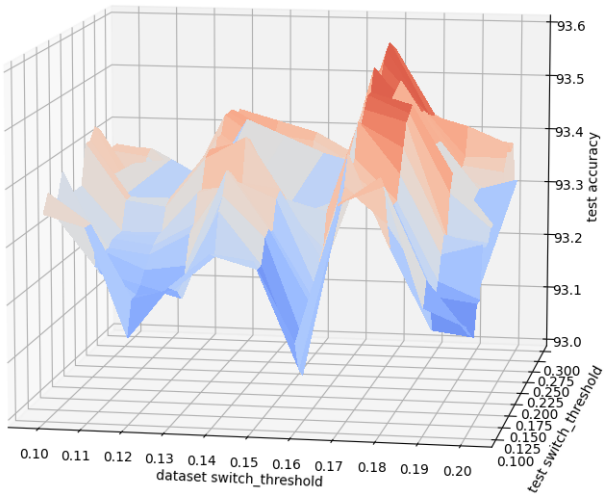

Figure 2. Explored parameter space

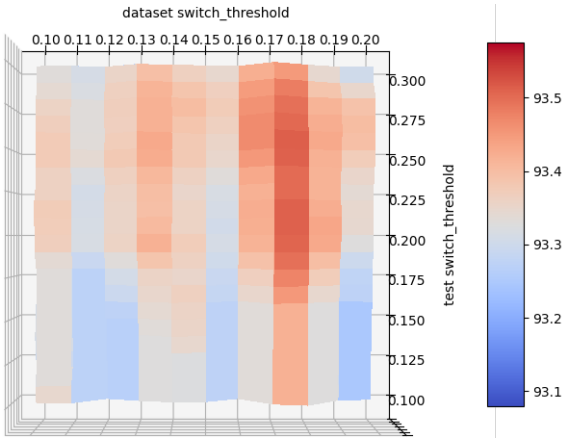

Figure 3. Explored parameter space (top view)

For the training we already have all hyperparameters which work well for training a single MobileNetV2. However, due to the architecture of the MNN we have an additional hyperparameter - the switch threshold which determines the value of the class probability that triggers the specialized NN for this class.

Note that switch threshold is used twice in our setup: the first time to select samples for the specialized NN datasets and the second time to select specialized NNs to activate during evaluation. Initially, they intended to be equal, however, while changing the first one (denoted later as "dataset switch threshold") requires to train specialized NNs again, changing the second (denoted later as "test switch threshold") can be easily done without additional actions. So, we consider them as different parameters and explore the behaviour of MNN.

We've explored the resulting accuracy for models with the dataset and test switch thresholds in ranges from 0.1 to 0.2 with step 0.01 and from 0.1 to 0.3 with step 0.01 respectively (Fig. 2 and 3). We present our findings: (i) these parameters are important and significantly affect the resulting accuracy; (ii) the choice of dataset switch threshold is more important since for the good value it gives overall better results with a wider range of test switch threshold value; (iii) given that changing dataset switch threshold requires retraining model finding the good value for dataset switch threshold is a hard problem.

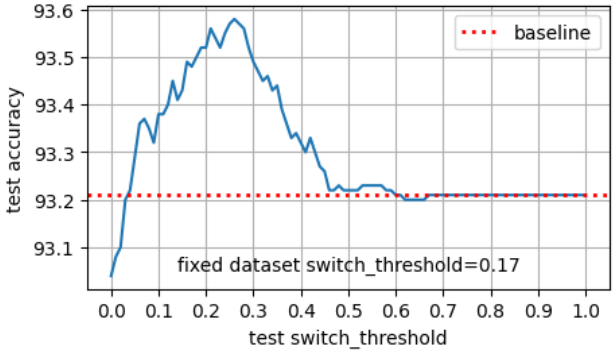

Figure 4. Explored test switch threshold for a fixed dataset switch threshold

Next, we focused on the evaluation, i. e. test switch threshold for a fixed dataset switch threshold of 0.17 (which gave the best results). This 1-dimensional parameter space meets our expectations (Fig. 4): (i) for the test switch threshold of 1 test accuracy is equal to baseline since in this case no specialized NNs are used; (ii) for the test switch threshold of 0 the accuracy is worse than baseline since all specialized NNs are always used, but they are not trained for this setup (they have different purpose); (iii) the best value lies in the interval from 0.1 to 0.33, so activating some number N of specialized NNs such that $1 < N < 10$ is the right approach.

6. Summary

We explored a new architecture based on separate NNs as blocks for a bigger Neural Network. The gain in Top-1 accuracy so far is not significant, but it shows that the approach is viable. The influence of the switch threshold on accuracy confirms our intuition behind the neural switch, but more research is needed to study dataset switch thresholds and different switch functions. Currently it would be quite problematic to find a good value for the dataset switch threshold: it is expensive to test on very large neural networks and it is unclear whether the thresholds found using small models work well with large ones. Possible solution can include examining Neural Switch probabilities for the training set to find patterns on probability vectors for choosing specialized NNs, using a weighted loss to ensure the neural switch gives better probabilities rather than focusing on the classification of 1 class.

7. Further research

There are 3 directions for research concerning this MNN architecture: exploring switch functions, exploring the properties of dataset and enhancement of the training pipeline.

1. One possible switch function that might improve performance is the following. Using running ratio of sorted probabilities to determine possible confusions: e. g. for given sorted vector [0.4 0.3 0.1 0.03 0.03 0.03 0.03 0.03 0.03 0.02] the running ratio [(1) 1.33 3 3.33 1 1 1 1 1 1.5] and threshold = 3.1 we can pick specialized NNs which corresponds to [0.4 0.3 0.1]. This would directly refer to probabilities which are close to each other by clustering them into high and low probabilities.

2. It might be beneficial to look at the saved probabilities - what is a particular feature of probability vectors of misclassified images? If the maximum probability is big we want to enforce more uniform probabilities of Neural Switch, if they are already similar we want to use a better switch function.

3. Concerning the training pipeline, using weighted batch for training specialized NNs can help to focus on the subset which is the hardest to predict (hard mining).

References

[1] DAWNBench. https://dawn.cs.stanford.edu/benchmark/ImageNet/train.html. Second entry in the list.

[2] Tim Dettmers. https://www.youtube.com/watch?v=8Fp9m4fNDQ4.

[3] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[4] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

[5] Xin Wang, Fisher Yu, Lisa Dunlap, Yi-An Ma, Ruth Wang, Azalia Mirhoseini, Trevor Darrell, and Joseph E Gonzalez. Deep mixture of experts via shallow embedding. In *Uncertainty in Artificial Intelligence*, pages 552–562. PMLR, 2020.

[6] Wolframcloud.com. https://resources.wolframcloud.com/NeuralNetRepository/resources/ResNet-50-Trained-on-ImageNet-Competition-Data.

8. Appendix

8.1. Interview Tim Dettmers

The idea is inspired by what Tim Dettmers said on the Chai Time Data Science Show [2].