Mixture of Experts - Experiment 1

1. Introduction

   Mixture of experts is a class of model, where output is formed by several experts together. By experts, we refer to machine learning models. The outputs of experts are combined by weighted sum assigned by some gating function. We want to answer the question: Can the jointly learned gate and experts outperform the distinctly learned experts with a pre-designed gate, which we think is ideal. We also check, if the mixture of experts outperforms a single expert. To answer the questions, we describe the architecture of mixture of experts we use in section 2, describe the dataset in section 3, describe what we will plot and print in section 4, and analyze in section 5.

2. Architecture

   The mixture of experts will be implemented on a classification task, so the output data Y will be a vector containing probabilities for each class. The output of mixture of experts is formed by weighted sum of each experts, where the weights are given by a gating network:

$$Y = \sum_i G(X)_i E_i(X)$$

   Where X is the input, $G(X)_i$ is the weight assigned to the expert i, and $E_i(X)$ is the output of expert i, which has same dimension as output Y.

   In this experiment, there are 4 models in total, where one is a single expert baseline with no gating and others have 2 experts with different gating networks: a pre-designed gate, a linear gate, and a 1-hidden-layer gate.

   2.1 Experts

   All experts of all models in the experiment have the same structure, a linear model. This linear model is represented by the following equation:

$$E_i(X) = softmax(W_i X)$$

   Where W_i is the weight matrix to be learned. The matrix multiplication $W_i X$ represents a linearly computed score vector of size being number of classes. The score vector is then transformed into probabilities by the softmax-function. The softmax-function is defined as following:

$$softmax(\boldsymbol{a})_c = \frac{\exp(a_c)}{\sum_d \exp(a_d)}$$

   2.2 Gating networks

   The gating functions varies in 3 models.

The pre-designed gate can be expressed as such: $G(X) =$ $IF\ IsMnist(X)\ THEN\ [1,0]\ ELSE\ [0,1]$. The function $IsMnist$ is related to the datasets in section 3, and it returns true, if the input X is from the dataset Mnist.

The linear gate is same as the linear experts, except it output a 2-dimensional vector instead of number of classes.

The 1-hidden-layer gate looks like this:

```
layer = tf.layers.flatten(X)
layer = tf.layers.dense(layer,64,tf.nn.relu)
Gate = tf.layers.dense(layer, 2, tf.nn.softmax)
```

2 Dataset

The data for classification is the combination of mnist and cifar-10. Both datasets have images with 1 of 10 labels, so the combined dataset will have 20 classes in total, where the first 10 classes are from mnist and the second 10 classes are from cifar-10. Images from both datasets are resized to 32x32 and gray-scaled.

3 What to plot

a.

Firstly, we want to know how well each model performs. We check this by plotting the accuracy-epoch curves for each model and for both training and testing, 8 curves in total. We assume the pre-designed gate gives the upper-bound, because it already separates the classes from mnist and cifar-10, and experts are ideally specialized on those datasets. However, we could see from the plots, are other mixture of experts performing closely to the upper-bound, or in worst case, worse than a single expert.

b.

We also want to know about how the learned gate will separate the data into 2 experts. For this, we define activation rate of expert i conditioned on dataset D:

$$ActivationRate(i|D) = Mean[G(X)_i|X\ in\ D]$$

We plot a table with axis standing for [Mnist, Cifar]x[Expert1, Expert2] and contents for the activation rate.

4 Results and analysis

a.

The Figure 1 shows that jointly trained gate and experts outperforms the pre-designed gate and distinct experts, which outperforms the single expert.

For this to happen, following reasons are considered. First, in computer science problems, the separation is often better, when it separates the problem into similar difficulty, and the pre-designed gate, that separated mnist and cifar datasets are not similar in difficulty. Second, mixture of experts does not separate input, and the experts could cooperate given non-zero weights, but the hard separation does not allow experts to share the uncertain results.

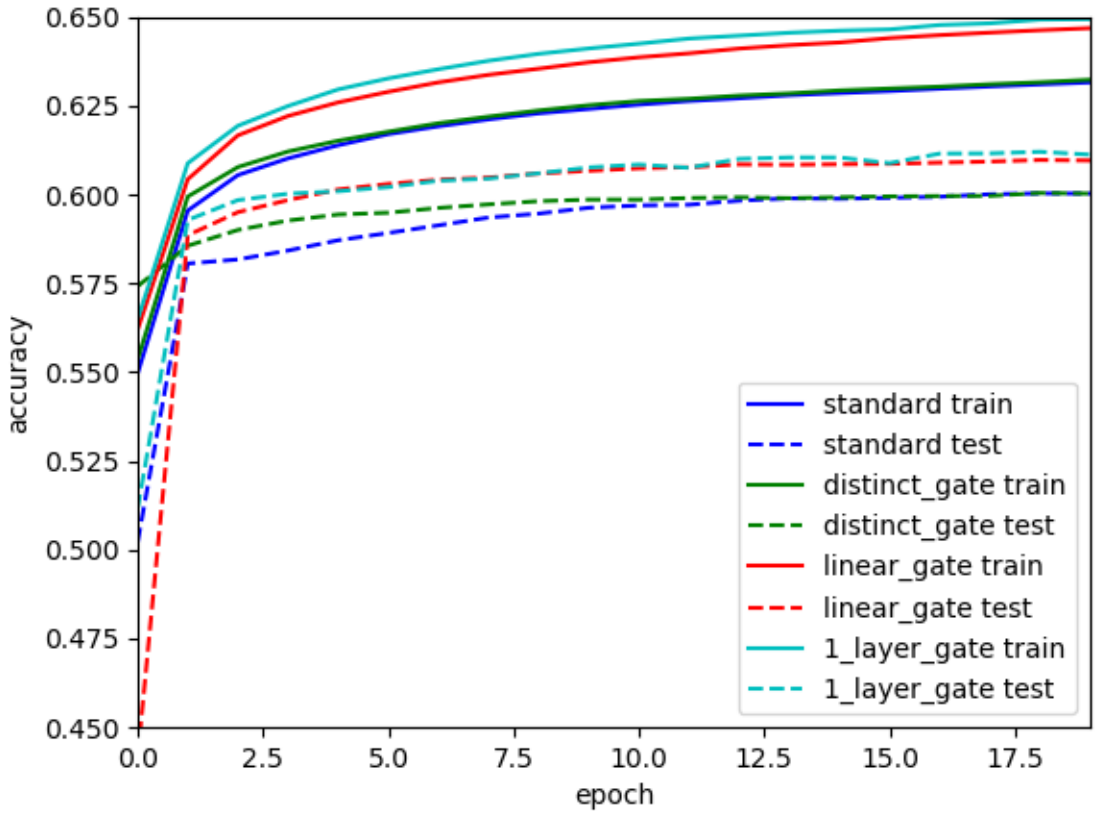

Figure 1. The accuracy curves of different models. Blue is the single expert, green is the pre-designed gate, red is the linear gate, cyan is 1-hidden-layer gate. Solid line is on train data, dashed line on test data.

b.

The Figure 2 shows the activation rate of different models defined in 4.b. In both linear gate and 1-hidden layer gate, they assign almost all cifar images to an expert. The separation is not balanced in the amount of data, nor in the difficulty discussed in section a. Since it does outperform pre-designed gate, it might be balanced in the sense of objective function: Learning a clear classification of a simple image from subset of mnist is objectively more valuable than learning a fuzzy classification of a difficult image from superset of cifar.

Model		Train Expert 1	Train Expert 2	Test Expert 1	Test Expert 2
Pre-designed gate	Mnist	1.00	0.00	1.00	0.00
	Cifar	0.00	1.00	0.00	1.00
Linear gate	Mnist	0.63	0.37	0.63	0.37
	Cifar	0.01	0.99	0.01	0.99
1-hidden layer gate	Mnist	0.45	0.55	0.44	0.55
	Cifar	1.00	0.00	1.00	0.00

Figure 2. The table of activation rate.

5 Discussion

Overall, in this simple configurations, mixture of experts performs better than a single expert. This work can be extended in several directions.

Measuring and plotting could be extended to see how each input is assigned to an expert, are the weights frequently close to 0 and 1 or are they concentrated close to the activation rate. Way around, we could not see enough either, what inputs does different experts have, do inputs of different experts belong to different and small subsets of classes, or are inputs from same class separated into different experts.

This architectures of experts and gating networks are too simple yet for stating that highly non-linear experts and gating networks could be jointly learned. The number of experts, the different complexities and architectures between experts, should also be considered. Be aware that overcomplex architectures on mnist and cifar datasets might cause worse cases than the case in section 5.b. To avoid this, learning strategy, on gating could be tested as well.