ECE598: Representation Learning: Algorithms and Models Fall 2017

Lecture 8: Mixture of Experts

Lecturer: Prof. Pramod Viswanath *Scribe: Anand Ramachandran, Oct 5, 2017*

8.1 Competitive Learning

The idea of competitive learning as introduced by Jacob, Jordan et al [JJNH91] assumes the existence of multiple "experts", each contemplating the same problem. There is the notion, roughly speaking, that each expert will be able to learn a subset of the problem better than the entire problem. If this is true, then it is better to allow each expert to concentrate on a subset of the problem (or a subspace of the whole problem space), with a mechanism (or "meta" expert) that is aware of this division of labor, so when an instance of the problem is presented, the meta expert will look to the right expert for the solution depending on which expert is better versed in the subspace the problem instance belongs to. The question of how to train the experts and the meta-expert simultaneously becomes relevant in this context. This setup is called the Mixture of Experts.

In [JJNH91] it is further stated that this division of labor, to be realized effectively, needs a learning framework that is competitive as reflected in the learning objective. In other words competitive learning is what facilitates a good partitioning of the problem subspace, allowing the experts to independently learn and concentrate on their specific subproblems.

To place the subsequent discussion on firm ground, we will define the problem in more objective terms. First, we are treating supervised learning problems. We are given an input vector $\vec{x} \in R^N$, and we need to estimate either a $\vec{y} \in R^M$ (regression), or a discrete value for y (classification). Each "expert" is equipped to solve this problem by itself, and the meta-expert combines the outputs of the experts to provide the final estimate. We will use regression as the vehicle for discussion in the sequel. We will assume that there are I learning examples, (X, Y), and K experts.

We will look at two methods of formulating the learning objective, each of which implies an interpretation of the operation of the meta-expert as well. The first one is the following error-function.

$$E_1 = \sum_{i=1}^{I} (\vec{y_i} - \sum_{k=1}^{K} p_{k,i} \vec{o_{k,i}})^2 \tag{8.1}$$

In 8.1, $\vec{y_i}$ is the target for the i-th example, $\vec{o_{k,i}}$ is the prediction from the k-th expert for the i-th example ($\vec{o_{k,i}}$ is a function of $\vec{x_i}$ where $(\vec{x_i}, \vec{y_i})$ is the i-th example), and $p_{k,i}$ is the weight placed on the k-th expert by the meta-expert for the i-th example. $p_{k,i}$ is also a function of $\vec{x_i}$. The objective formulated this way isn't an example of competitive learning, in the sense described above. This may be illustrated as follows. Assume that the experts are given by

$$\vec{o_{j,i}} = \vec{a_j}^T \vec{x_i}, 1 \leq j \leq K \tag{8.2}$$

Then, an attempt to do gradient descent will encounter the following derivative:

$$\frac{\partial E_1}{\partial a_j^p} = -2\sum_{i=1}^{I}(\vec{y_i} - \sum_{k=1}^{K} p_{k,i}\vec{o_{k,i}})p_{j,i}x_i^p \tag{8.3}$$

where the superscript p indicates the p-th component. This is the gradient that will be used to adjust the p-th parameter of the j-th expert, but note that this gradient involves the parameters of all other experts through the terms $(\vec{y_i} - \sum_{k=1}^{K} p_{k,i}\vec{o_{k,i}})$. Thus the experts influence and interfere with each other. This may be seen as a method of co-operation among the experts rather than competition. A closer examination of the predicted value indicates that an expert fills in to cover the residual of the prediction left by the rest of the experts. Yet another intuition was described in class where it was mentioned that the gradient doesn't delineate the deficiency of prediction of each expert, but only considers the total deficiency. Which part of the problem sub-space an expert eventually gains expertise in depends entirely on the meta-expert in this case, since an expert making a worse prediction isn't penalized (or corrected) based on that fact, but based solely on how the meta-expert rates the expert's knowledge in a subspace.

A different way to formulate the objective is as follows.

$$E_2 = \sum_{i=1}^{I}\sum_{k=1}^{K} p_{k,i}(\vec{y_i} - \vec{o_{k,i}})^2 \tag{8.4}$$

Differentiating this objective gives us

$$\frac{\partial E_2}{\partial a_j^p} = -2\sum_{i=1}^{I} p_{j,i}(\vec{y_i} - \vec{o_{j,i}})x_i^p \tag{8.5}$$

In this case, the gradients with respect to the j-th expert are not influenced by the parameters of any other expert. Not only does this prevent an expert from directly interfering with the learning process of another expert, the correction of the parameters of an expert through gradient descent depends on both the meta-expert and the expert itself. This formulation hence agrees better with the description of competitive learning.

It was mentioned that each method implies an interpretation of the operation of the meta-expert. This will be elaborated upon now, further solidifying the notion of competitition vs cooperation. If we assume that E_1 is the actual error of prediction, then it means the meta-expert simply does a weighted sum of the prediction of each expert, where $p_{k,i}$ is the weight for the k-th expert for the i-th example. This reinforces the notion of cooperation among the experts.

If we assume that E_2 is the *expected* error of prediction (which is valid if $p_{k,i}$'s are conditional probabilities, conditioned on $\vec{x_i}$, over a k-nomial distribution), then the meta-expert is a stochastic switch that chooses the k-th expert with probability $p_{k,i}$ for the i-th example. To see this more clearly, note that if the meta-expert is a stochastic switch, then for a specific set of I examples, and a corresponding specific set of predictions by the Mixture of Experts, the actual error accrued through the predictions is given by

$$e_2 = \sum_{i=1}^{I}\sum_{k=1}^{K} 1_{k,i}(\vec{y_i} - \vec{o_{k,i}})^2 \tag{8.6}$$

where e_2 is the actual error for a specific set of I estimates from the mixture of experts, $1_{k,i}$ is the indicator function indicating that the k-th expert was chosen for the i-the example. Then, we see that

$$E_2 = E[e_2|X] \tag{8.7}$$

This, then is an example where the meta-expert chooses one expert over all others, supporting a notion of competition among the experts.

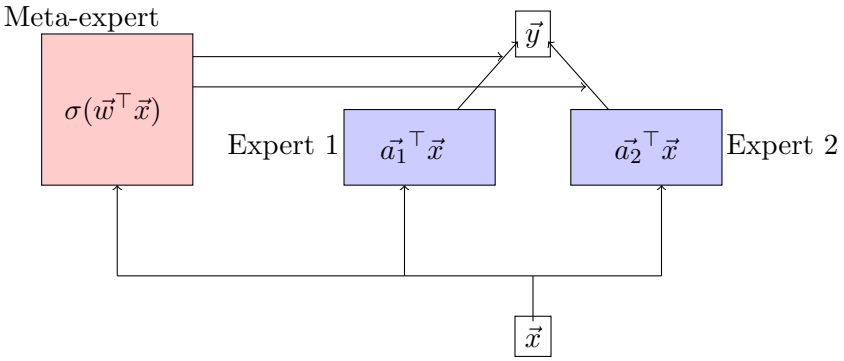

Figure 8.1: The Mixture of Experts[1]

An example of this setup with two experts and a meta-expert is in Figure 8.1. The individual experts are linear functions as discussed above, and the meta-expert uses a softmax function, σ, to weight the opinions of the individual experts (the use of the softmax function will be justified later). Eventhough this seems to only match the E_1 objective readily, it will become clear from later discussions that this can also match the E_2 objective, and that the E_1 and the E_2 objectives can simply be considered two different learning objectives for training the same Mixture of Experts model. That is, the Mixture of Experts model is trained differently when using E_1 compared to when using E_2, but it can be operated identically when used for regression/classification in both cases, and this use is well-justified.

8.2 The model for the meta-expert

For both E_1 and E_2 definitions, it is acceptable to assume that $p_{k,i}$ is a multinomial probability distribution conditioned on $X = \vec{x_i}$. A handy function for defining a multinomial distribution, given a vector is the softmax function as defined below, which satisfies the required properties, and whose parameters may be trained as necessary.

$$p_{k,i} = P[\epsilon = k|X = \vec{x_i}] = \frac{e^{\vec{w_{k,i}}^T \vec{x_i}}}{\sum_{l=1}^{K} e^{\vec{w_{l,i}}^T \vec{x_i}}} \tag{8.8}$$

Here we introduced a new random variable ϵ representing the behaviour of the meta-expert, which was described as a stochastic switch before.

8.3 The probabilistic expert model for regression

Equations 8.7, 8.8 provide a probabilistic interpretation for the operation of the meta-expert, but we haven't yet provided an interpretation for the operation of the individual experts and that of the entire mixture.

The central assumption is that fixing X fixes a distribution for Y, and allows the Mixture of Experts to act as a generative model for Y given X. This distribution may be further decomposed into two components, one describing the behaviour of the meta-expert (ϵ) conditioned on X and the other describing the behaviour of each individual expert given X (or the behaviour of the mixture given X and ϵ). We have described the first component in Equation 8.8, so we will describe the second below.

We assume that the k-th expert may be considered a generative model for some random variable O_k when X is fixed (to $\vec{x_i}$, say). This distribution may be described as a Gaussian distribution (we are not restricted to Gaussians; [JJ94] assumes an exponential family), with mean value $\vec{o_{k,i}}$, which so far we treated as the prediction of the k-th expert when $X = \vec{x_i}$, and a coveriance matrix $\tau_{k,i}$ (which wasn't introduced before). From the behaviour of the meta-expert, which is a stochastic switch, we can see that this Gaussian that describes the conditional distribution of O_k given $X = \vec{x_i}$, also represents the conditional distribution of Y given $X = \vec{x_i}$ and that expert k is chosen. That is,

$$P[Y = \vec{y_i}|X = \vec{x_i}, \epsilon = k] = \mathcal{N}(\vec{\mu} = \vec{o_{k,i}}, \Sigma = \tau_{k,i}) \tag{8.9}$$

where $\mathcal{N}(\vec{\mu}, \Sigma)$ defines a Gaussian distribution with mean $\vec{\mu}$ and covariance matrix Σ. We may next assemble the conditional distribution of Y given X as follows.

$$P[\vec{y_i}|X = \vec{x_i}] = \sum_{k=1}^{K} p_{k,i}\mathcal{N}(\vec{\mu} = \vec{o_{k,i}}, \Sigma = \tau_{k,i}) \tag{8.10}$$

Note that the minimum mean-square error estimate for Y given $X = \vec{x_i}$ would be $E[Y|X = \vec{x_i}]$ which is

$$E[Y|X = \vec{x_i}] = \sum_{k=1}^{K} p_{k,i}\vec{o_{k,i}} \tag{8.11}$$

Incidentally, Equation 8.1 tries to minimize the mean-square error, and also assumes the exact output estimation as given in Equation 8.11. And we argued that it may be insufficient for the purposes of learning!

There is no fallacy here however, nor are we claiming to have found a way to estimate Y given X that is better than the conditional expectation. That is because competitive learning isn't specifying how to predict Y given X, but rather specifying how to better learn the conditional distribution of Y given X. Put another way, the statement that conditional expectation minimizes mean-squared error assumes that a probability distribution is already determined, and is not a statement regarding how to determine that probability distribution. After we learn the model by whatever means (and we still maintain that competitive learning is better suited), there is nothing that prevents us from

estimating Y using Equation 8.11. In regression problems, this is indeed (atleast) one of the ways that is adopted [YWG12]. From this point of view, it may be noted that irrespective of the use of E_1, or E_2, the resultant Mixture of Experts provides a weighted sum of the individual experts' outputs when used in regression. Thus E_1 and E_2 simply provide a different learning objective for training the same Mixture of Experts model. The output of the Mixture of Experts model, when using E_2, is simply interpreted as the *minimum mean-square error estimate* for the target random variable (in this case, Y).

8.4 Application to classification problems

So far, we discussed how the Mixture of Experts functions in regression problems. We will briefly touch upon how it functions in classification problems next.

The adaptation from regression to classification is quite straightforward. In this case, the Mixture of Experts is still assumed to be a generative model for Y given X. The difference is that this distribution is discrete. Such a distribution is multinomial and is described by a stochastic vector of dimensionality Q for Q-way classification.

As before, the k-th expert produces a distribution conditioned on X, of a random variable O_k, which becomes the conditional distribution of the entire Mixture of Experts when conditioned on two factors : X and ϵ. Note that the individual expert also does Q-way classification. Being a multinomial distribution, there is no independent dispersion factor associated with each expert. Dispersion factor in the case of our regression treatment is the covariance matrix for the individual expert's Gaussian distribution [JJ94].

Overall for the classification problem, we have

$$P[Y = q|X = \vec{x_i}, \epsilon = k] = o^q_{k,i} \tag{8.12}$$

where $1 \leq q \leq Q$ and the superscript indicates the q-th component of a vector. And,

$$P[Y = q|X = \vec{X_i}] = \sum_{k=1}^{K} P[\epsilon = k|X = \vec{x_i}] o^q_{k,i} \tag{8.13}$$

The model for $P[\epsilon = k|X = \vec{x_i}]$ was described in Equation 8.8. Since $o^{\vec{}}_{k,i}$ should describe a multinomial distribution itself, it can be a neural network whose final layer describes the softmax function.

8.5 Training using Expectation Maximization (EM)

When there are hidden variables and observed variables and one may posit a joint probability distribution for the hidden and observed variables, one may invoke the EM machinery. The EM formulation casts a problem attempting to maximize the log likelihood of the observed data into a problem attempting to maximize the expected log joint distribution of the observed data and

hidden data. This expected log joint distribution is the Q function. Lets say that the observed data is $\mathcal{D}$ and the hidden data is Z, then for a parameter set Θ, we have

$$Q(\Theta, \Theta^t) = E_{Z|\mathcal{D},\Theta^t}\Big[logP[\mathcal{D}, Z|\Theta]\Big] \tag{8.14}$$

where the superscript t is a specific realization of the parameter set, or the parameter set after iteration t of EM. Computing the Q step is called the E-step. The next step is called the M-step which determines a Θ^{t+1} that maximizes Q.

$$\Theta^{t+1} = argmax_\Theta Q(\Theta, \Theta^t) \tag{8.15}$$

Dempster et al [DLR77] showed that an improvement in Q leads to an improvement in the log-likelihood of the observed data as well. It was seen in a previous lecture that EM attempts to maximize the lower bound of the log-likelihood of the observed data.

To apply EM to the Mixture of Experts, we need to first define the hidden variables. This is readily done, by assuming that ϵ, the random variable that describes the meta-expert's operation is the hidden variable. The i-th "complete" example is hence $(\vec{y_i}, \vec{\epsilon_i})$. Note that we have left out $\vec{x_i}$ from this. This is because $\vec{x_i}$ isn't assumed to be part of the data, but part of the model. This is in keeping with the formulation of Model of Experts as a generative model for Y given $X = \vec{x_i}$. We will define the nuts and bolts of EM for Mixture of Experts next.

The joint distribution of the complete data is,

$$P[Y = \vec{y_i}, \epsilon = k|X = \vec{x_i}, \Theta] = P[Y = \vec{y_i}|X = \vec{x_i}, \epsilon = k, \Theta]P[\epsilon = k|X = \vec{x_i}, \Theta] \tag{8.16}$$

To reduce clutter, we will use the following notation from now, $P[Y = \vec{y_i}|X = \vec{x_i}, \epsilon = k, \Theta] = P_{\vec{y_i}|\vec{x_i},k}$, $P[Y = \vec{y_i}, \epsilon = k|X = \vec{x_i}, \Theta] = P_{\vec{y_i},k|\vec{x_i}}$, $P[\epsilon = k|X = \vec{x_i}, \Theta] = P_{k|\vec{x_i}}$. When the parameter set Θ^t is involved, we will use P^t to indiciate the probabilities.

Note that examples for $P_{\vec{y_i}|\vec{x_i},k}$, and $P_{k|\vec{x_i}}$ were given before in Sections 8.3 and 8.2 respectively. These distributions are hence known in terms of the parameters of the Mixture of Experts. As an exercise of this notation, consider the distribution of the Mixture of Experts presented as follows

$$P_{\vec{y_i}|\vec{x_i}} = \sum_{k=1}^{K} P_{\vec{y_i}|\vec{x_i},k} P_{k|\vec{x_i}} \tag{8.17}$$

For EM, the posterior distribution of the hidden data given the observed data is needed as well, since the expectation is taken over this distribution. This maybe written as

$$P^t_{k|\vec{y_i},\vec{x_i}} = \frac{P^t_{\vec{y_i}|k,\vec{x_i}} P^t_{k|\vec{x_i}}}{P^t_{\vec{y_i}|\vec{x_i}}} = \frac{P^t_{\vec{y_i}|k,\vec{x_i}} P^t_{k|\vec{x_i}}}{\sum_{l=1}^{K} P^t_{\vec{y_i}|l,\vec{x_i}} P^t_{l|\vec{x_i}}} \tag{8.18}$$

Note that $P^t_{\vec{k}|\vec{y_i},\vec{x_i}}$ may be computed from the parameter set Θ^t when $(\vec{x_i}, \vec{y_i}, k)$ are given.

We next compute the Q function following Equation 8.14.

$$Q(\Theta, \Theta^t) = E_{\epsilon|Y,X,\Theta^t}\left[logP[Y,\epsilon|X,\Theta]\right] = \sum_{i=1}^{I}\sum_{k=1}^{K} P^t_{k|\vec{y_i},\vec{x_i}} \log P_{\vec{y_i}|\vec{x_i},k} P_{k|\vec{x_i}} \tag{8.19}$$

Q sums over an enumeration of the hidden variable and $\vec{y_i}, \vec{x_i}$ are known examples. Hence Q is a function only of the parameter set Θ and one may use some known optimization techniques to maximize it to accomplish the M-step.

Note also that Q nicely separates out into two parts, one exclusively involving $P_{\vec{y_i}|\vec{x_i},k}$ that deals with each expert and the other involving $P_{k|\vec{x_i}}$ that deals with the meta-expert. In addition, taking derivative of Q with respect to the parameters of any one expert doesn't mix parameters of multiple experts together, since each expert's distribution enters Q conditioned on the fact that that expert is used in producing the output. In other words each expert's parameters are confined to a summand of the form $\log P_{\vec{y_i}|\vec{x_i},k}$, a term that contains only parameters from the k-th expert. Thus the EM formulation successfully conserves the notion of competitive learning.

8.6 Hierarchical Mixture of Experts, Deep Mixture of Experts, and Sparsely-Gated Mixture of Experts

The Mixture of Experts model has been extended to a few other configurations. The configuration that keeps most of the characteristics of the original formulation is the Hierarchical Mixture of Experts [JJ94].

In this formulation, every expert in a regular Mixture of Experts model is assumed to be itself a Mixture of Experts. This recursive decomposition can be continued any number of times, with the result that we have a tree-like structure in which the leaf nodes are the individual experts, and the intermediate nodes are all meta-experts. This model can also be trained using the EM algorithm above, just that the structure of the meta-experts is more complicated now. In the regular Mixture of Experts model, the k-th expert would be chosen based on a single stochastic decision. In the Hierarchical Mixture, a specific expert at the leaf would be chosen based on a sequence of decisions - every meta-expert in the path from the leaf expert to the root of the tree structure should make the appropriate decision. This has the effect that the decision that chooses a specific leaf expert goes from being a scalar random variable ϵ to a vector random variable $\vec{\epsilon}$. Thus, a simple change in notation would be enough to reformulate the Q function.

$$Q(\Theta, \Theta^t) = E_{\vec{\epsilon}|Y,X,\Theta^t}\left[logP[Y,\vec{\epsilon}|X,\Theta]\right] = \sum_{i=1}^{I}\sum_{\vec{k}} P^t_{\vec{k}|\vec{y_i},\vec{x_i}} \log P_{\vec{y_i}|\vec{x_i},\vec{k}} P_{\vec{k}|\vec{x_i}} \tag{8.20}$$

Note that $\log P_{\vec{y_i}|\vec{x_i},\vec{k}}$ still refers to the output distribution of a single expert, conditioned on X. In addition, $P_{\vec{k}|\vec{x_i}}$ maybe decomposed into the product of individual conditional distribution of the meta-experts in the path from the leaf (where our expert of interest resides) to the root of the tree, since each meta-expert is independent of another expert when conditioned on the input, $\vec{x_i}$.

While the Hierarchical Mixture of Experts attempts a more sophisticated division of the problem space into sub-problems, the Deep Mixture of Experts attempts to **compose** multiple Mixtures of Experts for applications in classification problems. That is, the output of one Mixture of Experts is

the input to another Mixture of Experts. In addition, the meta-expert goes from being a softmax function to a rectification unit. This has the effect that all experts which are not "selected" by the rectifier are completely invisible to the later stages in the network, and do not impact the final prediction. Another way to view this is that a Deep Mixture of Experts is a Deep Neural Network where certain parts of the network are turned off or on based on the input, thus dynamically changing the network structure.

For two-levels of composition, the Deep Mixture of Experts looks as below. Assume that $\vec{x}$ is the input for a classification problem. Then,

$$\vec{y} = \sum_{m=1}^{M} p_m^1(\vec{x}) o_m^1(\vec{x}) \tag{8.21}$$

$$\vec{z} = \sum_{n=1}^{N} p_m^2(\vec{y}) o_n^2(\vec{y}) \tag{8.22}$$

$$\vec{w} = softmax(\vec{z}) \tag{8.23}$$

Here, $\vec{o}$s are the outputs of experts and $\vec{p}$s are the decisions of meta-experts. The outputs of the first layer of Mixture of Experts defined by the o^1 are passed to the second level of Mixture of Experts after rectification by p^1. This network is trained using Stochastic Gradient Descent. However, there is a risk that if the initial examples favour a certain subset of experts, then the training process can continue to favour them for the rest of the examples. To avoid this, certain precautions (in the form of some hard heurstics) may be adopted to uniformize the selection of networks in the beginning of the training phase. Details may be found in [ERS13].

The sparsely gated mixture of experts is another take on dynamically altering the structure of a Deep Neural Network based on the input. In this case, the meta-expert uses a specialized form of softmax function instead of rectification to "turn-off" certain experts. The softmax function examines its inputs and keeps the top-k values while setting the rest of the inputs to $-\infty$, effectively renormalizing the meta-expert to consider only k experts at a time. The sparse gated mixture of experts also runs the risk of favouring certain experts disproportionately. While the Deep Mixture of Experts adopts heuristics to impose hard restrictions on the selection of experts, the sparse-gated mixture of experts adds a noise term to the input of the meta-expert, to combat this problem. Details are in [SMM+17].

---

[1] Figure 8.1 was incorporated for release as lecture notes. It is based on latex code provided by Ashok Vardhan (TA for the course).

Bibliography

[DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1977.

[ERS13] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. *CoRR*, 2013.

[JJ94] Michael I. Jordan and Robert A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Comput.*, 1994.

[JJNH91] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. *Neural Comput.*, 1991.

[SMM+17] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *CoRR*, abs/1701.06538, 2017.

[YWG12] S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty years of mixture of experts. *IEEE Transactions on Neural Networks and Learning Systems*, Aug 2012.