Boosted Mixture Learning of Gaussian Mixture HMMs for Speech Recognition

Jun Du[1], *Yu Hu*[1], *Hui Jiang*[2]

[1]iFlytek Research, Hefei, Anhui, P. R. China
[2]York University, 4700 Keele Street, Toronto, Ontario M3J 1P3, Canada
unuedjwj@ustc.edu,jadefox@ustc.edu, hj@cse.yorku.ca

Abstract

In this paper, we propose a novel boosted mixture learning (BML) framework for Gaussian mixture HMMs in speech recognition. BML is an incremental method to learn mixture models for classification problem. In each step of BML, one new mixture component is calculated according to functional gradient of an objective function to ensure that it is added along the direction to maximize the objective function the most. Several techniques have been proposed to extend BML from simple mixture models like Gaussian mixture model (GMM) to Gaussian mixture hidden Markov model (HMM), including Viterbi approximation to obtain state segmentation, weight decay to initialize sample weights to avoid overfitting, combining partial updating with global updating of parameters and using Bayesian information criterion (BIC) for parsimonious modeling. Experimental results on the WSJ0 task have shown that the proposed BML yields relative word and sentence error rate reduction of 10.9% and 12.9%, respectively, over the conventional training procedure.

Index Terms: functional gradient, boosted mixture learning, acoustic models, speech recognition

1. Introduction

In the state-of-the-art automatic speech recognition (ASR) systems, we normally use Gaussian mixture HMMs as acoustic models to model basic speech units, ranging from context-independent whole words in small vocabulary ASR tasks to context-dependent phonemes (e.g., triphones) in large vocabulary ASR. Traditionally the HMM-based acoustic models are estimated from available training data using the well-known EM algorithm based on maximum likelihood (ML) criterion. To deal with data sparseness problems in model training, we normally use phonetic decision trees to tie HMM states from different triphone contexts, which leads to the so-called state-tying triphone HMMs. In order to derive a simple closed-form solution, we normally grow the decision trees based on single Gaussian HMMs[7]. After the state-tying structure is determined from the decision trees, a separate "mixture-up" step is performed to gradually increase number of Gaussian mixtures in each HMM state until the optimal performance is achieved. In most today's ASR systems, the "mixture-up" is usually implemented as two steps: i) all existing Gaussians in an HMM state is first randomly split; ii) all split Gaussians are re-estimated based on k-means or EM algorithm. Obviously, we are facing several problems when increasing model complexity in the above-mentioned "mixture-up" strategy. First of all, the random splitting strategy is not optimal in terms of model estimation criterion. For example, there is no guarantee that the newly added Gaussian components from random splitting always increase the likelihood function. Secondly, since the subsequent EM-based re-estimation is sensitive to the initial parameters of randomly split Gaussians, there is no guarantee that the EM-based re-estimation can always converge to any good optimal point when starting from the randomly split Gaussians as initial values.

Recently, the concept of boosting has been widely applied to various pattern classification problems. The basic idea of boosting is to derive and combine a large number of weak classifiers to achieve strong and reliable classifier [1]. Some theoretical work has shown that the boosting algorithms can result in impressive generation performance, which can be attributed to large margin achieved by the boosting algorithms in the training data. More recently, the traditional boosting algorithms have been extended to some learning problems of mixture models [2, 3], which is called boosted mixture learning (BML). The basic idea of BML is to learn mixture models in an incremental and recursive manner. The BML always starts from single mixture model and gradually adds a new mixture component in such a way that it always optimizes a predefined objective function. The essential point of BML is that a new mixture component is calculated in each step according to functional gradient of the objective function so that each new component is always added to the direction that increases the objective function the most. Compared with the traditional random splitting, BML is less sensitive to the initial parameter values and it may probably converge to a better optimal point.

In this work, we study how to use BML to learn Gaussian mixture HMMs for speech recognition. As the first step, we only consider the maximum likelihood (ML) estimation criterion in BML, where the objective function of BML is defined as the likelihood function of model parameters. In this paper, we first consider to apply the standard BML algorithm to Gaussian mixture models (GMMs) and then extend it to Gaussian mixture HMMs. Furthermore, several modifications have been proposed to make BML feasible and effective in the HMM framework. Firstly, Viterbi approximation is proposed to obtain state segmentation and BML of HMMs is conducted according to the Viterbi state segmentation. In this way, BML of Gaussian mixture HMMs can be formulated as the same BML problem of GMMs. Secondly, weight decay [4] using power scaling is proposed to deal with the over-fitting problem caused by unbounded sample weights. Thirdly, we propose to update the entire Gaussian mixture model whenever a new component is added to the mixture while only the newly added mixture component is normally updated in each traditional BML step. This is called global updating, which is found to significantly improve recognition performance in speech recognition. Finally, Bayesian information criterion (BIC) [6] is used as the convergence criterion in BML to control the size of model parameters for parsimonious modeling.

2. BML of Mixture Models

First of all, a mixture model $F_K(\mathbf{x})$ is defined as:

$$F_K(\mathbf{x}) = \sum_{k=1}^{K} c_k f_k(\mathbf{x}), \quad c_k \geq 0, \sum_{k=1}^{K} c_k = 1 \tag{1}$$

where K is the mixture number, $\mathbf{x}$ is a feature vector, c_k and $f_k(\mathbf{x})$ are the weight and component of k^{th} mixture, respectively.

Learning of mixture models has been extensively studied in machine learning. The traditional method is based on random splitting and EM-based re-estimation. In this work, we focus on a different method to learn mixture models, which is named as boosted mixture learning (BML). At each stage of BML, a new component (c_k, f_k) is added to the previous mixture model F_{k-1} with $k-1$ mixture components to grow into a new mixture model F_k with k mixture components as follows:

$$F_k(\mathbf{x}) = (1 - c_k)F_{k-1}(\mathbf{x}) + c_k f_k(\mathbf{x}). \tag{2}$$

This procedure is repeated until some convergency condition is met. The key idea of BML is how to derive the new component f_k and its mixture weight c_k in an optimal way.

Table 1: Description of BML procedure

Step 1:	Initialize $F_k(k = 1)$.
Step 2:	For $k = 2, 3, ...$
	$\{c_k^*, f_k^*\} = \arg\max_{c_k, f_k} \mathcal{C}(F_k)$
	Continue to add the new component?
	Yes: $F_k(\mathbf{x}) = (1 - c_k^*)F_{k-1}(\mathbf{x}) + c_k^* f_k^*(\mathbf{x})$
	No: go to Step 3
Step 3:	Output final mixture model F_k

To learn parameters c_k and f_k, we should define an objective function $\mathcal{C}$. If we consider maximum likelihood (ML) estimation, the objective function is defined as log likelihood function of mixture models as follows:

$$\mathcal{C}(F_k) = \sum_{n=1}^{N} \log F_k(\mathbf{x}_n) \tag{3}$$

where N is the number of training samples. Then the general procedure of BML can be described in Table 1. In order to derive each new mixture component and its weight optimally in Step 2, a functional gradient method [2, 3] is used. Assume the objective function $\mathcal{C}(F)$ is viewed as a functional of mixture model F. When a new mixture component f_k is added, hopefully it will increase the objective function as much as possible:

$$\mathcal{C}((1 - \varepsilon)F_{k-1} + \varepsilon f_k) > \mathcal{C}(F_{k-1}) \tag{4}$$

where ε is a small constant. If we use the Taylor series to expand the left hand side of the above equation, we have:

$$\begin{aligned}
& \mathcal{C}((1 - \varepsilon)F_{k-1} + \varepsilon f_k) \\
= \; & \mathcal{C}(F_{k-1} + \varepsilon(f_k - F_{k-1})) \\
= \; & \mathcal{C}(F_{k-1}) + \varepsilon\langle\nabla\mathcal{C}(F_{k-1}), (f_k - F_{k-1})\rangle \\
& +O(||\varepsilon(f_k - F_{k-1})||) \\
\approx \; & \mathcal{C}(F_{k-1}) + \varepsilon\langle\nabla\mathcal{C}(F_{k-1}), (f_k - F_{k-1})\rangle
\end{aligned} \tag{5}$$

where $\nabla\mathcal{C}(F_{k-1}) = \nabla\mathcal{C}(F)|_{F=F_{k-1}}$ is the functional gradient of the objective function at F_{k-1}. If ε is small enough, high-order item $O(||\varepsilon(f_k - F_{k-1})||)$ can be ignored. By considering both Eq.(4) and Eq.(5), the optimization of objective function, which is equivalent to optimization of the first-order item with the form of inner product in Eq.(5), can be derived as follows:

$$f_k^* = \arg\max_{f_k} \; \langle\nabla\mathcal{C}(F_{k-1}), (f_k - F_{k-1})\rangle. \tag{6}$$

This equation clearly shows that the new mixture component f_k is calculated along the direction of functional gradient where the objective function grows the most. The reason to take the inner product between the functional gradient and the mixture model is to ensure that the new component f_k is calculated in such a way that the new model F_k still falls into the same model space as F_{k-1}.

If we consider the objective function in Eq.(3), it is easy to show that the functional gradient can be calculated as $\nabla\mathcal{C}(F_{k-1}) = \frac{1}{F_{k-1}}$. As a result, Eq.(6) can be re-written as follows:

$$\begin{aligned}
f_k^* &= \arg\max_{f_k} \frac{1}{N} \sum_{n=1}^{N} \frac{f_k(\mathbf{x}_n) - F_{k-1}(\mathbf{x}_n)}{F_{k-1}(\mathbf{x}_n)} \\
&= \arg\max_{f_k} \sum_{n=1}^{N} \frac{f_k(\mathbf{x}_n)}{F_{k-1}(\mathbf{x}_n)}
\end{aligned} \tag{7}$$

Obviously, Eq.(7) is a general form to derive each new mixture component in BML based on the maximum likelihood (ML) estimation criterion. In the following, we consider to apply it to Gaussian mixture models (GMMs), where each mixture component f_k is a multivariate Gaussian distribution with mean vector $\boldsymbol{\mu}_k$ and diagonal covariance matrix $\boldsymbol{\Sigma}_k$ as

$$f_k(\mathbf{x}_n) = \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{8}$$

There is no closed-form solution to solve the optimization problem for GMMs in Eq.(7). In this work, we propose to optimize Eq.(7) iteratively using EM algorithm to search for the optimal component f_k^*. That is, by taking a $\log$ operation, Eq.(7) becomes a log-sum maximization which can be optimized by conventional lower-bound maximization technique using Jensen's inequality. Then the parameters of Gaussian function f_k can be iteratively estimated as follows:

$$w(\mathbf{x}_n) = \frac{f_k(\mathbf{x}_n)}{F_{k-1}(\mathbf{x}_n)} \tag{9}$$

$$\hat{\boldsymbol{\mu}}_k = \frac{\sum_{n=1}^{N} w(\mathbf{x}_n)\mathbf{x}_n}{\sum_{n=1}^{N} w(\mathbf{x}_n)} \tag{10}$$

$$\hat{\boldsymbol{\Sigma}}_k = \frac{\sum_{n=1}^{N} w(\mathbf{x}_n)(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)^\top}{\sum_{n=1}^{N} w(\mathbf{x}_n)} \tag{11}$$

where $w(\mathbf{x}_n)$ denotes sample weights in each iteration, similar to the ones used in the traditional boosting algorithms. The physical meaning of sample weight is that samples with low probability F_{k-1} are given higher weights than the highly likely samples according to F_{k-1}. Hence, the new component f_k focuses on these samples poorly modeled by a simpler distribution F_{k-1}. For initialization, we often set $w^0(\mathbf{x}_n) = \frac{1}{F_{k-1}(\mathbf{x}_n)}$ in the first iteration and then use Eq.(10), Eq.(11) and Eq.(9) to update mean vector, covariance matrix and sample weights iteratively until f_k converges.

After f_k^* is estimated from the above EM method, the mixture weight c_k^* can be obtained by using the following line search:

$$c_k^* = \arg\max_{c_k \in [0,1]} \mathcal{C}((1 - c_k)F_{k-1} + c_k f_k^*). \tag{12}$$

In practice, the optimal mixture weight c_k^* can be found efficiently by using a grid search in the interval $[0, 1]$.

3. BML of HMMs for Speech Recognition

In this section, we extend the above BML algorithm of GMMs to estimation of Gaussian mixture HMMs in speech recognition. Several techniques have been proposed to make the above BML procedure feasible and effective under the HMM framework.

3.1. Viterbi approximation for state segmentation

Under the HMM framework, the likelihood function can be viewed as a mixture of all possible hidden state sequences. As a result, it is not straightforward to directly apply the BML method in Eq.(7) to HMMs. In this work, we simply accept the Viterbi approximation where the likelihood function is calculated based on the optimal Viterbi path instead of summation over all possible state sequences. In this way, the above BML algorithm of GMMs can be directly used to estimate GMMs for all HMM states independently.

$$\begin{aligned}\mathcal{C}(F) &= \log \sum_{s_0 s_1 \dots s_N} \pi_{s_0} \prod_{n=1}^{N} a_{s_{n-1}s_n} F(\mathbf{x}_n | s_n) \\ &\approx \sum_{n=1}^{N} \log F(\mathbf{x}_n | s_n^*) + C \qquad (13)\end{aligned}$$

where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$ is a set of training samples, $\{\pi_i\}$ and $\{a_{ij}\}$ denote the initial state probabilities and state transition probabilities of HMMs, respectively, and $s_0^* s_1^* ... s_N^*$ denotes the optimal state sequence based on the Viterbi approximation. In above, we use $F(\mathbf{x}_n | s_n^*)$ to represent each state output probability distribution, which is modeled by a GMM in Gaussian mixture HMMs.

Based on Eq.(13), the BML problem of HMMs can be simplified as BML of GMMs as defined in Eq.(3). Given a set of training data, $\mathbf{X}$, we first use an initial HMM to decode the optimal Viterbi paths. Then all the feature vectors are aligned to different HMM states based on the Viterbi paths and the BML method in Section 2 is used to estimate GMMs for all HMM states based on the aligned feature vectors. It is noted that $\{\pi_i\}$ and $\{a_{ij}\}$ are not updated in the BML procedure since they are not critical for performance of speech recognition.

3.2. Initialization of sample weights with weight decay

After state segmentation, GMM parameters of each HMM state can be learned as the BML algorithm in Section 2. But there are several problems when we apply the BML to HMMs. The first problem is initialization of sample weights for each new mixture component using sample weight $w^0(\mathbf{x}_n) = \frac{1}{F_{k-1}(\mathbf{x}_n)}$. In Gaussian mixture HMMs for speech recognition, it is found that dynamic range of F_{k-1} is so large that the initial sample weights, $w^0(\mathbf{x}_n)$, are dominated by only a small number of samples with low probability, which may cause overfitting problem in BML. To deal with this problem, weight decay [4] using power scaling is used to calculate initial sample weights as follows:

$$w^0(\mathbf{x}_n) = \frac{1}{F_{k-1}^{\alpha}(\mathbf{x}_n)} \qquad (14)$$

where α is the exponential scaling factor $0 < \alpha < 1$. It has been observed that weight decay is critical to achieve good performance in speech recognition and typically the value of exponential factor α is not sensitive to different ASR tasks.

3.3. Partial and global updating in BML

In the traditional BML, when a new mixture component f_k is added to the mixture model, we first estimate a new mixture component as in Eq.(7) and then the mixture weight is estimated from a separate line search process as in Eq.(12). In this section, we propose an alternative method to estimate each mixture component and its weight. As in [5], we directly apply the EM algorithm to optimize the original log likelihood function only with respect to the new mixture component f_k and weight c_k while F_{k-1} are assumed to be constants. For GMMs, it can be easily derived that mixture weight c_k, mean vector and covariance matrix of f_k are estimated iteratively as follows:

$$\begin{aligned} w(\mathbf{x}_n) &= \frac{f_k(\mathbf{x}_n)}{c_k f_k(\mathbf{x}_n) + (1 - c_k) F_{k-1}(\mathbf{x}_n)} && (15) \\ \hat{c}_k &= \frac{1}{N} \sum_{n=1}^{N} c_k w(\mathbf{x}_n) && (16) \\ \hat{\boldsymbol{\mu}}_k &= \frac{\sum_{n=1}^{N} w(\mathbf{x}_n) \mathbf{x}_n}{\sum_{n=1}^{N} w(\mathbf{x}_n)} && (17) \\ \hat{\boldsymbol{\Sigma}}_k &= \frac{\sum_{n=1}^{N} w(\mathbf{x}_n)(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)(\mathbf{x}_n - \hat{\boldsymbol{\mu}}_k)^\top}{\sum_{n=1}^{N} w(\mathbf{x}_n)} && (18) \end{aligned}$$

In this work, this estimation method is named as *partial EM*. Compared with the re-estimation based on the functional gradient method from Eq.(9) to Eq.(11), the updating formula for mean vector and covariance matrix are the same and the main difference is the estimation formula of sample weights. Comparing the estimation formula Eq.(9) with Eq.(15), it is easy to see that sample weights in partial EM have much smaller dynamic range due to normalization in Eq.(15). As a result, it may lead to robust and reliable estimation of the new component. In partial EM, we simply initialize each mixture weight c_k as $1/k$.

Our experimental results show that in each stage of BML, if only the newly-added mixture component is updated, the convergence of recognition performance is quite slow. Therefore, similar to the re-estimation in partial EM, additional EM-based re-estimation can be applied to re-estimate all mixture components in F_k, which is called *global EM* in this work. It has been shown that the additional global EM step can significantly improve performance of Gaussian mixture HMMs in speech recognition.

3.4. BIC for parsimonious modeling

BML is an incremental and recursive learning process where only one new mixture component is added in each iteration. In this section, we consider to use Bayesian information criterion (BIC) to select the optimal number of mixture components. The BIC criterion has been widely used as a popular model selection criterion and it can be viewed as a regularized likelihood function as follows:

$$\text{BIC}(k) = \mathcal{C}(F_k) - \frac{\lambda}{2} * M_k * \log(N) \qquad (19)$$

where $\mathcal{C}(F_k)$ is the conventional log likelihood function defined in Eq.(3). M_k is the number of parameters used in mixture model F_k. In our BML procedure, we first run BML to gradually increase the number of mixture components until certain point. At last, we use the BIC criterion to roll back model size

Table 2: Performance (word error rate and sentence error rate) comparison on the WSJ-5k test set.

WER(%)	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8
HTK	10.98	9.88	8.59	8.03	7.08	5.98	5.34	5.14
1-pass BML	N/A	7.51	6.18	5.75	5.70	5.34	4.99	4.84
2-pass BML	N/A	6.61	5.62	5.68	5.14	4.86	4.63	4.58
+BIC	Avg. 6.6 Gaussians per state, WER is 4.58%							

SER(%)	K=1	K=2	K=3	K=4	K=5	K=6	K=7	K=8
HTK	69.1	67.0	64.6	63.3	56.1	47.9	46.1	44.6
1-pass BML	N/A	55.5	49.4	47.3	46.4	42.7	40.3	41.2
2-pass BML	N/A	50.0	44.6	45.8	42.4	40.0	38.2	38.8
+BIC	Avg. 6.6 Gaussians per state, SER is 38.8%							

and select the optimal value of k which maximizes the BIC criterion in Eq.(19). By doing so, we can typically reduce model size significantly for parsimonious modeling.

4. Experiments

The proposed BML algorithms have been evaluated in a large vocabulary ASR task using the WSJ0 database. In the WSJ0 task, the training set is the standard SI-84 set, consisting of 7133 utterances from 84 speakers (about 12 hours speech data in total). Evaluation is performed on the standard Nov'92 non-verbalized 5k close-vocabulary test set (WSJ-5k), including 330 utterances from 8 speakers. For the baseline system, we use the HTK to build standard state-tying cross-word triphone HMMs [7], which includes a total number of 2774 tied-states. The feature vectors are 39-dimensional MFCC features (including delta and delta-delta features) after cepstral mean normalization processing in sentence level. A standard trigram language model is used in evaluation.

For the BML configurations, we set the exponential factor α of weight decay to 0.05. The parameter λ for BIC is set to 0.98. The initial single Gaussian HMMs are trained using HTK procedure. And the initial HMM state probabilities and state transition probabilities are not updated in the BML stage. For each boosting stage of BML, firstly we initialize the new mixture component using functional gradient based sample weights with weight decay in Eq.(14) and set initial new mixture weight to $1/k$. Then re-estimation of partial EM is used to refined both the new mixture component and weight. Finally, global EM is applied for all mixture components in current mixture model.

4.1. Experimental results on WSJ0 task

We compare recognition performance of HMMs from different training procedures in terms of word error rate and sentence error rate on the WSJ-5k test set. As shown in Table 2, K denotes the mixture number of GMM in each tied HMM state. "HTK" stands for the HTK-trained baseline system which uses the conventional random splitting and EM-based re-estimation during the model training process. "1-pass BML" means Viterbi state segmentation is regenerated using currently updated HMMs in each step. "2-pass BML" represents Viterbi state labels are regenerated by using the best HMM (with $K = 8$) from "1-pass BML" and then the same BML training produce is repeated without regenerating the state labels. "+BIC" means BIC is applied to "2-pass BML" to reduce model size for parsimonious modeling. It is observed that for both word error rate and sentence error rate, the proposed BML procedures significantly outperform the traditional "HTK" procedure, especially when the number of Gaussians is small. By comparing "1-pass BML" and "2-pass BML", we can see that the precision of the state labels has a significant impact on recognition performance for BML. So HMMs used to generate state segmentation should be refined as much as possible. Using BIC, the good recognition performance is maintained even though the model size has been significantly reduced. Finally, our BML procedure yields the relative word and sentence error rate reduction of 10.9% and 12.9%, respectively, compared with "HTK" procedure. Meanwhile, relative reduction of 17.5% in model size (from 8 Gaussians to averaged 6.6 Gaussians per tied-state) can be achieved by using BIC without any loss in recognition performance.

5. Conclusion

In this paper, we have presented a novel boosted mixture learning (BML) framework based on maximum likelihood (ML) criterion for Gaussian mixture HMMs in speech recognition. The Viterbi approximation has been accepted for state segmentation to extend the BML of GMMs into Gaussian mixture HMMs. Several techniques have been proposed to improve performance of BML in speech recognition, such as weight decay to initialize sample weights to avoid overfitting, combining partial updating with global updating of parameters and using BIC for parsimonious modeling. Experimental results on the WSJ0 task have shown that the proposed BML method yields significantly better performance than the conventional HMM training procedure.

6. References

[1] L. Mason, J. Baxter, P. Bartlett, and M. Frean. "Boosting algorithms as gradient descent in function space." *NIPS 11*, 1999.

[2] V. Pavlovic, "Model-based motion clustering using boosted mixture modeling." *Proc. of CVPR*, 2004, pp. 811-818.

[3] M. Kim and V. Pavlovic, "A recursive method for discriminative mixture learning." *Proc. of ICML*, 2007, pp. 409-416.

[4] S. Rosset, "Robust boosting and its relation to bagging." *Proc. of ACM SIGKDD*, 2005, pp. 249-255.

[5] G. McLachlan. "Finite mixture models." *John Willey & Sons, Inc.*, 2001.

[6] G. Schwarz, "Estimating the dimension of a model." *Annals of Statistics*, Vol. 6, No. 2, pp. 461-464, 1978.

[7] P. Woodland, J. Odell, V. Valtchev, and S. Young, "Large vocabulary continuous speech recognition using HTK." *Proc. of ICASSP*, 1994, Vol. 2, pp. 125-128.