PHONE BOUNDARY REFINEMENT USING RANKING METHODS

Hung-Yi LO*†, Hsin-Min WANG*
*Institute of Information Science, Academia Sinica, Taipei
†Department of Computer Science and Information Engineering, National Taiwan University, Taipei
{hungyi, whm}@iis.sinica.edu.tw

***Abstract*—The HMM/SVM-based two-stage framework has been widely used for automatic phone alignment. The two-stage method uses SVM classifiers to refine the hypothesized boundaries given by the HMM-based Viterbi forced alignment. However, there are two drawbacks in using the classification model for detecting the phone boundaries. First, the training data contains only information about the boundary and far away non-boundary signal characteristics. Second, the classification model suffers from the class-imbalanced training problem. To overcome these drawbacks, we propose using ranking methods to refine the hypothesized boundaries. We train multiple phone-transition-dependent rankers by using K-means-based and decision-tree-based clustering. Both Ranking SVM and RankBoost are evaluated. The results of experiments on the TIMIT corpus demonstrate that the proposed ranking method outperforms the classification method. The best accuracy achieved is 84.20% within a tolerance of 10 ms. The mean boundary distance is 6.66 ms.**

***Index Terms*—automatic phone segmentation, ranking method, ranking SVM, RankBoost**

I. INTRODUCTION

Annotated speech corpora are indispensable to various areas of speech research, e.g., speech recognition and speech synthesis. Phoneme level annotation is especially important for fundamental speech research. However, the development of a large high-quality, manually labelled speech corpus requires lots of human effort, and is time-consuming. To reduce the human effort and speed up the labelling process, many attempts have been made to utilize automatic phone alignment approaches to provide initial phone segmentation for subsequent manual segmentation and verification [5]–[10], [12].

The most popular method of automatic phone alignment is to adapt an HMM-based phone recognizer to align a phoneme transcription with a speech utterance. Empirically, phone boundaries obtained in this way should contain few serious errors, since HMMs in general capture acoustic properties of phones; however, small errors are inevitable because HMMs are not sensitive enough to detect changes between adjacent phones. Consequently, researchers have proposed a HMM/SVM-based two-stage framework [6], which uses support vector machine (SVM) classifiers to refine the hypothesized boundaries given by the HMM-based Viterbi forced alignment.

In order to provide training data for the SVM classifier, current researches [6], [7], [12] use the feature vectors associated with the true phone boundaries as positive training samples and the randomly selected feature vectors at least 20 ms away from the true boundaries as negative training samples. However, using a classification model for detecting the phone boundaries has two drawbacks. First, the training data contains only information about the boundary and *far away* non-boundary signal characteristics. We expect that the refinement task can be better modelled by treating the instances extracted from the true boundaries as high preference instances, the nearby instances as medium preference instances, and the far away instances as low preference instances. Second, modelling the refinement task as a classification problem will suffer from the *class-imbalanced* training problem, since the training data contains a lot of negative instances but only a limited amount of positive instances. As a result, general classification algorithms will be biased to predict all instances to be negative on such a class-imbalanced dataset, since they are learned to minimize the number of incorrectly classified instances.

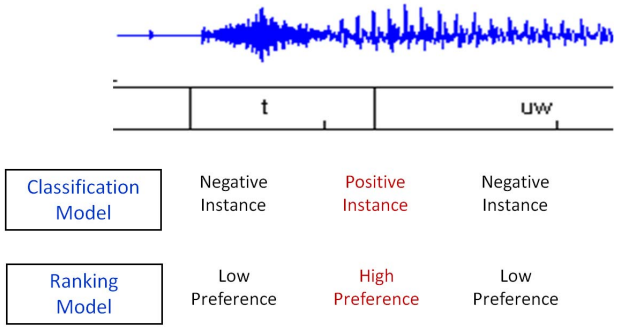

Fig. 1. The difference between using a classification model and using a ranking model for phone boundary refinement.

In this paper, we propose using ranking methods [3], [4] to refine the hypothesized phone boundaries given by the HMM-based Viterbi forced alignment. For each true boundary, we generate a from-high-to-low preference list, i.e., the instance extracted from the true boundary is associated with the highest preference and, for the remaining instances, the preference degree decreases as the distance to the true boundary increases. These preference lists are used to train a ranking model. In our approach, a phone-transition-dependent ranker is applied to detect the true phone transition boundary around each hypothesized boundary given by the initial HMM-

based segmentation. These rankers for detecting boundaries of various phone transitions are trained in advance based on multiple discriminative acoustic features in addition to mel-frequency cepstral coefficients (MFCCs). We conducted automatic phone alignment experiments on the TIMIT speech corpus. The experiment results demonstrate that the proposed ranking method outperforms the classification method. The best accuracy achieved is 84.20% within a tolerance of 10 ms. The mean boundary distance is 6.66 ms.

The remainder of this paper is organized as follows. In Section 2, we describe feature extraction. In Section 3, we briefly review the SVM-based classification method. Then, we describe ranking methods in Section 4. The phone transition clustering methods are introduced in Section 5. The experiments and results are detailed in Section 6. Finally, the conclusions are drawn in Section 7.

II. FEATURE EXTRACTION

In the HMM-based segmentation, each frame of the speech data is represented by a 39 dimensional MFCC-based feature vectors comprised of 12 MFCCs and log energy, plus their delta and delta-delta coefficients. In the refinement stage, each frame is represented by a 45 dimensional feature vector consisting of the above 39 MFCC-based coefficients, plus zero crossing rate, bisector frequency [8], burst degree [8], spectral entropy, general weighted entropy [11], and subband energy.

For each hypothesized boundary, the feature vectors of the left and right frames next to it, together with the symmetrical Kullback-Leibler distance (SKLD) and the spectral feature transition rate (SFTR) between the two feature vectors, are concatenated to form a 92 dimensional augmented vector. The augmented vectors are used as features to cluster the phone transitions and as the input vectors to the classification model and the ranking model.

III. THE CLASSIFICATION MODEL

In this paper, we use SVM as the classification model. The training process of SVM aims to maximize the margin (also considered as minimizing a regularization term) and minimize the training error at the same time. The objective function is of the form:

$$\begin{array}{lrcl} \min\limits_{\boldsymbol{w},b,\xi} & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} & + & C\sum\limits_{i=1}^{m}\xi_i, \\ \text{s.t.} & y_i(\boldsymbol{w}^T\phi(\boldsymbol{x}_i)+b) & \geq & 1-\xi_i \\ & \xi_i & \geq & 0, i=1,\ldots,m, \end{array} \tag{1}$$

where ϕ is a function that maps the input data into a higher dimensional space and C is a tuning parameter. The classification score is $f(\boldsymbol{x}) = \boldsymbol{w}^T\phi(\boldsymbol{x}) + b$.

The classification-based phone boundary refinement proceeds as follows. For each initial boundary detected by the HMM-based segmentation, several hypothesized boundaries around it are identified first; then each of which is examined by a SVM classifier; and finally the most likely boundary (the one with the largest classification score) is selected to replace the initial boundary.

As we have mentioned, using a classification model for phone boundary detection has two drawbacks. First, the training data contains only information about the boundary and *far away* non-boundary signal characteristics. We expect that the refinement task can be better modelled by treating the instances extracted from the true boundaries as high preference instances, the nearby instances as medium preference instances, and the far away instances as low preference instances. Second, modelling the refinement task as a classification problem will suffer from the *class-imbalanced* training problem, since the training data contains a lot of negative instances but only a limited amount of positive instances. As a result, general classification algorithms will be biased to predict all instances to be negative on such a class-imbalanced dataset, since they are learned to minimize the number of incorrectly classified instances.

IV. THE RANKING ALGORITHMS

Unlike a classification algorithm, a ranking algorithm aims to find a scoring function $f : \mathcal{X} \to \mathbb{R}$, where $f(\boldsymbol{x}_i) > f(\boldsymbol{x}_j)$ means that instance $\boldsymbol{x}_i$ is preferred to $\boldsymbol{x}_j$. Note that the actual numerical values of f are not important; only the order is of interest. Given a training dataset, the scoring function can be learned by learning-to-rank algorithms. The learning-to-rank algorithms can be divided into three categories: pointwise, pairwise, and listwise algorithms. We adopt the pairwise method in this work. For the pairwise method, the training data is represented by ordered pairs $\{\boldsymbol{x}_i \succ \boldsymbol{x}_j\}$ and the learning object is to minimize the number of mis-ordered pairs. In the following subsections, we describe two pairwise learning-to-rank algorithms, namely Ranking SVM [4] and RankBoost [3], used in this paper.

A. Ranking SVM

The objective function for Ranking SVM is of the form:

$$\begin{array}{lrcl} \min\limits_{\boldsymbol{w},\xi} & \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} & + & C\sum\limits_{i,j}\xi_{i,j}, \\ \text{s.t.} & \boldsymbol{w}^T\phi(\boldsymbol{x}_i) & \geq & \boldsymbol{w}^T\phi(\boldsymbol{x}_j)+1-\xi_{i,j} \\ & \xi_{i,j} & \geq & 0, \end{array} \tag{2}$$

where ϕ is a function that maps the input data into a higher dimensional space, $\{\boldsymbol{x}_i \succ \boldsymbol{x}_j\}$ is an ordered pair and C is a tuning parameter the same as that in the general SVM form. In other words, Ranking SVM aims to minimize mis-ordered pairs and the regularization term $\frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}$. The trade-off between the training error and the regularization is controlled by the parameter C. When using linear Ranking SVM, the term $\phi(\boldsymbol{x})$ is replaced by $\boldsymbol{x}$. The optimization problem can be solved by using the decomposition algorithm used in classification SVM. Note that the number of constraints and the slack variable ξ depend on the number of ordered pairs in the training data. The computational complexity sometimes becomes an important practical issue of Ranking SVM. The output score for the input vector $\boldsymbol{x}$ is $f(\boldsymbol{x}) = \boldsymbol{w}^T\phi(\boldsymbol{x})$. The procedure of the ranking-based phone boundary refinement is the same as that of the classification-based refinement. That

is, the frame with the largest score will be selected to replace the initial boundary.

B. RankBoost

RankBoost finds a highly accurate ranker by combining many weak rankers, despite that each of them is only moderately accurate. The typical weak ranker is a threshold function:

$$h(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x}^j > \theta \\ 0 & \text{if } \boldsymbol{x}^j \leq \theta, \end{cases} \quad (3)$$

where $\boldsymbol{x}^j$ is the j-the element of $\boldsymbol{x}$. The RankBoost algorithm iteratively trains many weak rankers. The training procedure maintains a weight distribution matrix D_t over $\mathcal{X} \times \mathcal{X}$ in each iteration and determines the parameters θ and j in the ranker h_t to minimize a weighted pairwise mis-ordered error r according to D_t,

$$r = \sum_{\boldsymbol{x}_i, \boldsymbol{x}_j} D_t(\boldsymbol{x}_i, \boldsymbol{x}_j)(h_t(\boldsymbol{x}_j) - h_t(\boldsymbol{x}_i)). \quad (4)$$

After each iteration, the weight is updated by

$$D_{t+1}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{D_t(\boldsymbol{x}_i, \boldsymbol{x}_j) \exp(\alpha_t(h_t(\boldsymbol{x}_j) - h_t(\boldsymbol{x}_i)))}{Z_t}, \quad (5)$$

where $h_t(\boldsymbol{x}_i)$ is the prediction score of ranker h_t on the instance $\boldsymbol{x}_i$, $\{\boldsymbol{x}_i \succ \boldsymbol{x}_j\}$ is an ordered pair, Z_t is a normalization factor to make D_{t+1} a distribution. The parameter α_t is calculated by $\frac{1}{2}\ln(\frac{1+r}{1-r})$. The output score for the input vector $\boldsymbol{x}$ is $f(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x})$.

The ranking methods in general require a higher training time complexity than the classification methods. However, there is an efficient solution for training RankBoost [3]. Its time complexity depends on the number of instances but not on the number of ordered pairs.

C. Generation of Ordered Pairs for Training

The training data for both Ranking SVM and RankBoost is represented by a set of correct-ordered pairs of instances. For each true boundary, we first extract feature vectors from a speech segment centered at it. Let the instances in the segment be $(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{T-1}, \boldsymbol{x}_T, \boldsymbol{x}_{T+1}, \ldots, \boldsymbol{x}_N)$, where $\boldsymbol{x}_T$ is extracted from the true boundary. Then, we generate four *ordered* ranking lists for this segment:

1) $(\boldsymbol{x}_T, \boldsymbol{x}_{T-1}, \ldots, \boldsymbol{x}_2, \boldsymbol{x}_1)$
2) $(\boldsymbol{x}_T, \boldsymbol{x}_{T+1}, \ldots, \boldsymbol{x}_{N-1}, \boldsymbol{x}_N)$
3) $(\boldsymbol{x}_{T-1}, \boldsymbol{x}_{T-2}, \ldots, \boldsymbol{x}_2, \boldsymbol{x}_1)$
4) $(\boldsymbol{x}_{T+1}, \boldsymbol{x}_{T+2}, \ldots, \boldsymbol{x}_{N-1}, \boldsymbol{x}_N)$

Each ordered ranking list can produce multiple ordered pairs by coupling the first instance in the list with every one of the remaining instances in the same list. For example, the first list produces $\{\boldsymbol{x}_T \succ \boldsymbol{x}_{T-1}\}, \{\boldsymbol{x}_T \succ \boldsymbol{x}_{T-2}\}, \ldots$, and $\{\boldsymbol{x}_T \succ \boldsymbol{x}_1\}$. Then, the ordered pairs are gathered together to form the training set. We have tried to generate more ordered ranking lists for each segment, e.g., the lists starting from $\boldsymbol{x}_{T-2}$ and $\boldsymbol{x}_{T+2}$. However, the computational time increases substantially but the prediction performance does not improve.

V. PHONE TRANSITION CLUSTERING

Ideally, we should be able to train a ranker or classifier for each type of phone transition. However, this is not feasible because the training data is always limited. Maintaining a balance between the available training data and the model's complexity is critical to the training process. Furthermore, since many phone transitions have similar acoustic characteristics, we can partition them into clusters so that the training data can be shared and the phone transitions with little training data can be covered by the rankers or classifiers of the categories they belong to. We implement phone transition clustering in two ways: by K-means clustering and by decision-tree-based clustering.

A. K-means-based Clustering

The K-means-based phone transition clustering is described as follows:

1) For each specific phone transition case, we gather all the augmented vectors associated with the human-labelled phone boundaries, and compute the mean vector.
2) For each one of the four phone transition classes, namely *sonorant to non-sonorant*, *sonorant to sonorant*, *non-sonorant to sonorant*, and *non-sonorant to non-sonorant*, we apply the K-means algorithm to cluster the phone transitions according to their mean vectors. Note that only the phone transitions with enough instances are considered in this step. The number of clusters is determined according to the cross-validation accuracy that the resulting rankers or classifiers achieve in the training data.
3) We assign the phone transitions, which are ignored in Step 2 due to sparse instances, to the nearest clusters according to the distances between their mean vectors and the cluster centers.

B. Decision-tree-based Clustering

The drawback of K-means clustering is that it can not cover phone transitions that do not occur in the training data. In contrast, decision-tree-based clustering can generalize to unseen phone transitions and take advantage of linguistic knowledge during clustering. Here, all the questions have the form "Is the left phone of the transition a member of the set **X** and the right phone a member of the set **Y**?" The sets X and Y range from broad phonetic classes, such as sonorant, stop, and unvoiced classes, to distinct phonemes, such as {r} and {s}. In total, 397 phonetic sets are used.

VI. EXPERIMENTS

A. Experiment setup

Our experiments were conducted on the TIMIT acoustic-phonetic continuous speech corpus. TIMIT contains a total of 6,300 sentences, comprised of 10 sentences spoken by each of 630 speakers from 8 major dialect regions in the United States. The TIMIT suggested training and testing sets contain 462 and 168 speakers, respectively. We discard the dialect sentences (SA1 and SA2 utterances) and utterances with phones shorter

than 10 ms. The resulting training set and testing set contain 3,696 sentences and 1,312 sentences, respectively.

The acoustic models for HMM-based segmentation consist of 50 context-independent phone models, each represented by a 3-state continuous density HMM (CDHMM) with a left-to-right topology. Each frame of the speech data is represented by a 39-dimensional feature vector comprised of 12 MFCCs and log energy, and their delta and delta-delta coefficients. The frame width is 20 ms and the frame shift is 5 ms. Utterance-based cepstral variance normalization (CVN) is applied to all the training and testing speech. The acoustic models were trained on the training speech according to the human-labelled phoneme transcriptions and boundaries by the Baum-Welch algorithm using the ML criterion with 10 iterations. Then, the Minimum Boundary Error (MBE) [6] discriminative training approach was applied to manipulate the above ML-trained HMMs with 10 more iterations.

By using the cross-validation on the TIMIT training data, the number of K-means-based phone transition clusters is 46. As a result, 46 classifiers or rankers are used. The cluster number for the decision-tree-based clustering is 151. In the refinement phase, given the boundary of each phone transition obtained by the HMM-based segmentation, 5 hypothesized boundaries extracted every 5 ms around the initial boundary within ± 10 ms will be examined by SVM, Ranking SVM or RankBoost. We compare the proposed ranking approaches with the HMM$_{MBE}$-based segmentation and the classification approaches. We exploit two classification models: linear SVM implemented by the LIBLINEAR package [2] and nonlinear SVM with Gaussian kernel implemented by the LIBSVM [1] package. We also exploit two ranking models: RankBoost and linear Ranking SVM modified from LIBLINEAR. We do not use nonlinear Ranking SVM because of its high computational complexity. However, RankBoost has the nonlinear (more specifically speaking, piecewise linear) prediction capability.

B. Experiment results

TABLE I
THE PERCENTAGE OF PHONE BOUNDARIES CORRECTLY PLACED WITHIN DIFFERENT TOLERANCES WITH RESPECT TO THEIR ASSOCIATED HUMAN-LABELLED PHONE BOUNDARIES.

Methods	Mean Boundary Distance (ms)	Accuracy %	
		$< 10ms$	$< 20ms$
HMM	7.14	81.57	93.73
Linear SVM$_{KM}$	6.84	83.51	93.85
Linear SVM$_{DT}$	6.89	83.44	93.79
RBF SVM$_{KM}$	6.75	84.00	94.33
RBF SVM$_{DT}$	6.83	83.70	94.12
Linear RankSVM$_{KM}$	6.72	83.89	94.17
Linear RankSVM$_{DT}$	6.76	83.90	94.01
RankBoost$_{KM}$	6.66	84.20	94.14
RankBoost$_{DT}$	6.66	84.13	94.11

Table 1 shows the percentage of phone boundaries correctly placed within different tolerances with respect to their associated human-labeled phone boundaries. The second row represents the results of the MBE-trained HMM forced alignment. The next four rows show the performance of the classification-based refinement based on the initial boundaries given by the MBE-trained HMM forced alignment. The last four rows show the performance of the ranking-based refinement. Linear SVM$_{KM}$ uses K-means-based clustering and Linear SVM$_{DT}$ uses decision-tree-based clustering. The other classification and ranking methods follow the same naming rule. Some observations can be drawn from Table 1. First, we observe that the ranking methods outperform the classification methods. The linear Ranking SVM methods (denoted by RankSVM) are comparable to the nonlinear classification SVM methods in terms of mean boundary distance and are better than the linear classification SVM methods. The nonlinear RankBoost methods perform better than the linear Ranking SVM methods in most of the evaluation metrics. Second, we observe that K-means-based clustering performs consistently better than decision-tree-based clustering when combined with all prediction models, although the difference is not significant. The best accuracy achieved is 84.20% within a tolerance of 10 ms. The mean boundary distance is 6.66 ms.

VII. CONCLUSIONS

Learning to rank is an active research topic in machine learning and information retrieval. In this paper, we have presented a ranking-based boundary refinement approach to refine the hypothesized phone boundaries given by the HMM-based Viterbi forced alignment. We have described how to generate the training instance pairs for training the ranking SVM and RankBoost. The results of experiments on the TIMIT corpus show that the proposed ranking-based approach outperforms the conventional classification-based approach in phone boundary refinement.

VIII. ACKNOWLEDGEMENT

This work was supported by the National Science Council of Taiwan under Grant: NSC 97-2221-E-001-022-MY3.

REFERENCES

[1] C.-C. Chang and C.-J. Lin. LIBSVM: a library for support vector machines, 2001. Available: http://www.csie.ntu.edu.tw/ cjlin/papers/libsvm.pdf.

[2] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[3] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[4] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[5] J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan. Phoneme alignment based on discriminative learning. In *Proc. Interspeech*, 2005.

[6] J.-W. Kuo, H.-Y. Lo, and H.-M. Wang. Improved hmm/svm methods for automatic phoneme segmentation. In *Proc. Interspeech*, 2007.

[7] K. S. Lee. MLP-based phone boundary refining for a tts database. *IEEE Trans. on Speech and Audio Processing*, 14:981–989, 2006.

[8] C.-Y. Lin, J.-S. R. Jang, and K.-T. Chen. Automatic segmentation and labeling for mandarin chinese speech corpora for concatenation-based TTS. *Computational Linguistics and Chinese Language Processing*, 10(2):145–166, 2005.

[9] H.-Y. Lo and H.-M. Wang. Phonetic boundary refinement using support vector machine. In *Proc. ICASSP*, 2007.

[10] S. S. Park and N. S. Kim. On using multiple models for automatic speech segmentation. *IEEE Trans. on Audio, Speech, and Language Processing*, 15:2202–2212, 2007.

[11] J.-L. Shen, J.-W. Hung, and L.-S. Lee. Robust entropy-based endpoint detection for speech recognition in noisy environments. In *Proc. ICSLP*, 1998.

[12] T. G. Toledano, L. A. H. Gomez, and L. V. Grande. Automatic phonetic segmentation. *IEEE Trans. on Speech and Audio Processing*, 11:617–625, 2003.