EURASIP Journal on
Audio, Speech, and Music Processing
a SpringerOpen Journal

RESEARCH Open Access

Integrated exemplar-based template matching and statistical modeling for continuous speech recognition

Xie Sun[1] and Yunxin Zhao[2*]

Abstract

We propose a novel approach of integrating exemplar-based template matching with statistical modeling to improve continuous speech recognition. We choose the template unit to be context-dependent phone segments (triphone context) and use multiple Gaussian mixture model (GMM) indices to represent each frame of speech templates. We investigate two different local distances, log likelihood ratio (LLR) and Kullback-Leibler (KL) divergence, for dynamic time warping (DTW)-based template matching. In order to reduce computation and storage complexities, we also propose two methods for template selection: minimum distance template selection (MDTS) and maximum likelihood template selection (MLTS). We further propose to fine tune the MLTS template representatives by using a GMM merging algorithm so that the GMMs can better represent the frames of the selected template representatives. Experimental results on the TIMIT phone recognition task and a large vocabulary continuous speech recognition (LVCSR) task of telehealth captioning demonstrated that the proposed approach of integrating template matching with statistical modeling significantly improved recognition accuracy over the hidden Markov modeling (HMM) baselines for both TIMIT and telehealth tasks. The template selection methods also provided significant accuracy gains over the HMM baseline while largely reducing the computation and storage complexities. When all templates or MDTS were used, using the LLR local distance gave better performance than the KL local distance. For MLTS and template compression, KL local distance gave better performance than the LLR local distance, and template compression further improved the recognition accuracy on top of MLTS while having less computational cost.

Keywords: Gaussian mixture model; Template matching; KL divergence; Dynamic time warping; Large vocabulary continuous speech recognition

1 Introduction

In speech recognition, hidden Markov modeling (HMM) has been the dominant approach since it provides a principled way of jointly modeling speech spectral variations and time dynamics. However, HMM has the shortcoming of assuming the observations being independent within each state, which makes it ineffective in modeling the fine details of speech temporal evolutions that are important in characterizing nonstationary speech sounds [1]. Time derivatives of cepstral coefficients [2] are widely used to supplement time dynamic information to speech feature distributions. Trajectory model [3] introduces time-varying covariance modeling to capture temporal evolutions of speech features. Additionally, approaches like segment models [4,5] and long-contextual-span model of resonance dynamics [6] have been proposed for similar purposes.

Exemplar-based methods have the potential in addressing the deficiency of HMMs and in recent years they have drawn renewed attention in the speech recognition community [7,8], such as sparse representations (SRs) [9] and template matching [10,11]. Template-based methods make direct comparisons between a test pattern and the templates of training data via dynamic time warping (DTW), and potentially they can capture the speech dynamics better than HMMs. Template-based methods were originally used to recognize isolated words or connected digits with

* Correspondence: zhaoy@missouri.edu
[2]Department of Computer Science, University of Missouri, Columbia, MO 65211, USA
Full list of author information is available at the end of the article

good performances [12]. Until recently, template-based methods had been impractical for large tasks of speech recognition, since feature vectors of training templates need to be stored in computer memory. With today's rapid advance in computing power and memory capacity, template-based methods are investigated for large recognition tasks and promising results are reported [10,11,13-18]. However, they are still difficult to use in large vocabulary continuous speech recognition (LVCSR) due to their needs for intensive computing time and storage space. The newly proposed methods, such as template pruning and filtering [19], template-like dimension reduction of speech observations [20], and template matching in the second-pass decoding search [21], are beginning to address this problem. In general, there is a tradeoff between the costs in computation and space and the accuracy in recognition.

Considering the pros and cons of HMMs and template methods, i.e., HMM-based statistical models are effective in compactly representing speech spectral distributions of discrete states but are ineffective in representing the fine details of speech dynamics, while template matching captures well the speech temporal evolutions but demands much larger computational complexity and memory space, it appears plausible to integrate the two approaches so as to exploit their strengths and avoid their weaknesses. In the current work, we propose a novel approach of integrating exemplar-based template matching with statistical modeling. We construct triphone context-dependent phone templates to preserve the time dynamic information of phone units and use phonetic decision trees to generate templates of tied triphone units, which improves the reliability of triphone templates and covers unseen triphones by some triphone clusters. The load on memory storage is reduced by using Gaussian mixture model (GMM) indices to represent the speech frames of the templates. It is worth noting that Gaussian indices were previously used to represent speech frames in speech segmentation [22], speech separation [23], and keyword spotting [24-26]. To facilitate comparison of the templates labeled by GMM indices, we propose the local distances of log likelihood ratio (LLR) and Kullback-Leibler (KL) divergence for DTW-based template matching. To further reduce the costs of memory space and computation, we propose template selection methods to generate template representatives based on the criteria of minimum distance (MDTS) and maximum likelihood (MLTS) and we also propose a template compression method to integrate information from training templates to obtain more informative template representatives. In the recognition stage, the GMMs and the templates are used together by DTW with the proposed local distances. The proposed methods have been applied to lattice rescoring on the tasks of TIMIT [27] phone recognition and telehealth [28] large vocabulary continuous speech recognition, and they have led to consistent error reductions over the HMM baselines.

This paper is organized as follows. In Section 2, we discuss the related work for template-based speech recognition and provide an overview of our proposed system. In Section 3, we describe the proposed methods for template construction, matching, and clustering. In Section 4, we discuss the proposed methods for template representative selection and compression. In Section 5, we present evaluation results on the task of TIMIT phone recognition and the task of telehealth LVCSR. Finally in Section 6, we give our conclusion and discuss future work.

2 Related work and system overview

2.1 Related work

Continuous speech recognition using template-based approaches has gained significant attention over the past several years. In [10], a top-down search algorithm was combined with a data-driven selection of candidates for DTW alignment to reduce search space, together with a flexible subword unit selection mechanism and a class-sensitive distance measure. On the Resource Management task, although the performance of the template matching system fell below the best published HMM results, the word error patterns of the two types of systems were found to be different and their combination was beneficial. In [13], an episodic-HMM hybrid system was proposed to exploit the ability of HMMs in producing high-quality phone graphs as well as the capability of an episodic memory in accessing fine-grained acoustic data for rescoring, where template matching was performed by DTW using the Euclidean distance. This system was evaluated on the 5k-word *Wall Street Journal* (WSJ) task and it showed a comparable performance with state-of-the-art HMM systems. In [18], prosodic information of duration, speaking rate, loudness, pitch, and voice quality was integrated with template matching through conditional random fields to improve recognition accuracy. On the Nov92 20k-word trigram WSJ task, the proposed method improved the state-of-the-art template baseline without prosodic information and led to a relative word error rate reduction of 7%. To make the template-based approach realistic for hundreds of hours of speech training data, a data pruning method was described for template-based automatic speech recognition in [19]. The pruning strategy worked iteratively to eliminate more and more templates from an initial database, and at each iteration, the feedback for data pruning was provided by the word error rate of the current model. This data pruning reduced the database size or the model size by about 30%, and consequently saved the computation time and memory usage in speech recognition. In [21], exemplar-based word-level features were investigated for large-scale speech recognition. These features were combined with the acoustic and language scores of the first-pass model through a segmental conditional random field to rescore word lattices. Since the word lattices helped restrict the search space, the templates were not required to cover the full training data, and the templates were also filtered to a smaller set to reduce computation cost and improve robustness. Experimental results showed that the template-based approach obtained a slightly better performance than the baseline system in Voice Search and YouTube tasks.

Relative to the above-discussed efforts, our approach as proposed in the current work falls into the hybrid category, but our integration of statistical modeling and template representation and matching are tighter, since we not only rescore the lattices generated by the HMM baseline, but we also use the baseline phonetic decision tree (PDT) structures to define the tied triphone templates, representing the template frames by the GMMs and using the LLR and KL distances to measure the differences of speech frames represented in this way. In the aspect of reducing computation and memory costs, we absorb the training data information into template representatives through clustering and estimation, rather than selecting a subset of training data as the templates. On the TIMIT and telehealth tasks, we are able to show statistically significant improvements in phone and word accuracies, respectively, over the HMM baselines.

2.2 System overview

The overall architecture of the proposed template matching method is described in Figure 1. In the training stage, Viterbi alignment is performed on the training data by the baseline model to determine the phone template boundaries; using the PDT-based triphone state tying structures of the baseline system, template clustering is performed to generate tied triphone templates (Section 3.3); using the GMM codebook derived from the baseline model, the template frames are labeled by the GMMs (Section 3.1); template selection and compression are further performed to generate the template representatives (Section 4). In the test stage, the baseline model is first used to perform decoding search on a test speech utterance to generate a word lattice; the test speech frames are labeled by the GMMs in the same way as in training; template matching and best path search are then performed on the word lattice to generate the rescored sentence hypothesis (Section 3.3).

3 Template representation, matching, and clustering

3.1 Template representation

We choose the template unit to be context-dependent phone segments, the context being the immediately left and right phones of each phone segment, and we refer the context-dependent templates as triphone templates. We first carry out forced alignments of training speech data

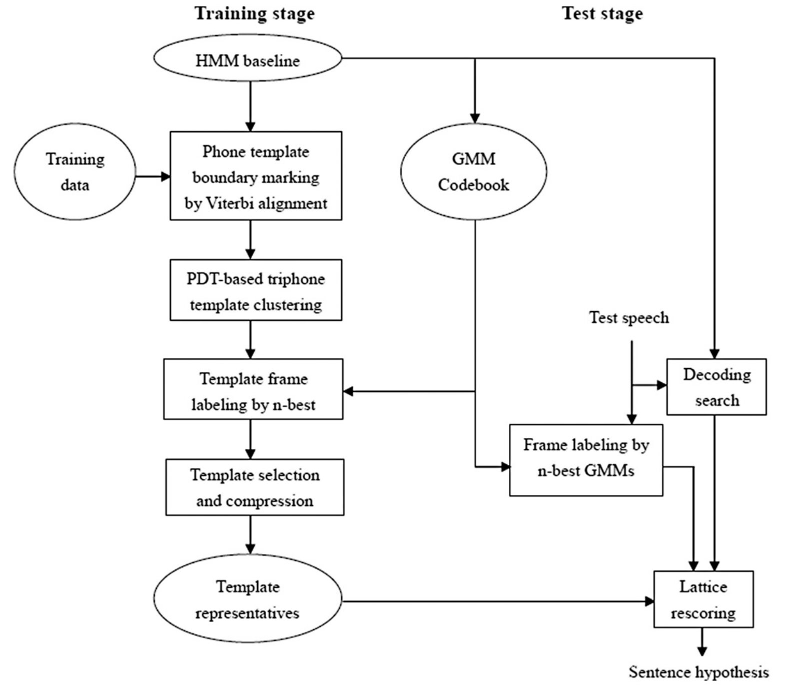
Training stage
Test stage
HMM baseline
Training data
Phone template boundary marking by Viterbi alignment
GMM Codebook
PDT-based triphone template clustering
Test speech
Template frame labeling by n-best
Decoding search
Frame labeling by n-best GMMs
Template selection and compression
Template representatives
Lattice rescoring
Sentence hypothesis

Figure 1 System overview.

with their transcriptions to obtain phone boundaries which define the phone templates. We then use a GMM codebook $\{m_1, m_2, \ldots, m_N\}$ that consists of the GMMs of the phonetic-decision-tree tied triphone states in the baseline HMMs to label the template frames, where N is the total number of GMMs from the HMM baseline. To do so, we compute the likelihood scores of a feature vector or frame (these two terms are used interchangeably with the understanding that a feature vector is normally extracted from a frame of data), $x_t \in R^d$ (d is the dimension of a real-valued feature vector), of a phone template by all GMMs and take the n GMMs that give the top n likelihood scores, $p\left(x_t\middle|m_{1(x_t)}\right)\geq p\left(x_t\middle|m_{2(x_t)}\right)\geq\ldots\geq p\left(x_t\middle|m_{n(x_t)}\right)\geq\ldots$, to label x_t. Each GMM index is also associated with a weight $w_{k(x_t)}$ that is defined to be proportional to the likelihood score $p\left(x_t\middle|m_{k(x_t)}\right)$, with $w_{k(t)} = \frac{p\left(x_t|m_{k(x_t)}\right)}{\sum_{l=1}^{n} p\left(x_t|m_{l(x_t)}\right)}$ and $\sum_{k=1}^{n} w_{k(x_t)} = 1$. A template frame is therefore represented as

$$x_t \rightarrow \left\{ \begin{bmatrix} m_{1(x_t)} \\ \vdots \\ m_{n(x_t)} \end{bmatrix} \begin{bmatrix} w_{1(x_t)} \\ \vdots \\ w_{n(x_t)} \end{bmatrix} \right\}. \tag{1}$$

In general, $n << d$, and hence storing the template frames in GMM indices requires a much smaller space than storing the feature frames for the Templates.

3.2 Template matching

In using DTW to measure the dissimilarity between two speech utterances, the allowed range of speaking rate variations can be specified by local path constraints [12]. Let $d(i, j)$ denote the local distance between the ith and the jth frames of two sequences under comparison and $D(i, j)$ denote the cumulative distance between the two sequences up to the time i and j. The symmetric constraint that we adopt here is defined as

$$D(i,j) = d(i,j) + min\{D(i-1,j), D(i-1,j-1), D(i,j-1)\}. \tag{2}$$

Given a sequence S_x representing a template and a sequence S_y representing a test segment, their average frame distance is calculated as

$$\bar{D}\left(S_x, S_y\right) = \frac{1}{N}\min_{\varnothing}\sum_{k=1}^{N} d\left(\varnothing_{S_x}(k),\quad \varnothing_{S_y}(k)\right), \tag{3}$$

where $\varnothing_{S_x}$ and $\varnothing_{S_y}$ are the warping functions that map S_x and S_y to a common time axis, and N is the warping path length. Considering the fact that in HMM-based decoding search the acoustic score of a test segment is the sum of its frame log likelihood scores (the segment acoustic score is therefore the average frame score scaled by the length of the segment), we define the distance between the template S_x and the test segment S_y in the similar way as

$$\begin{aligned} D\left(S_x, S_y\right) &= L \times \bar{D}\left(S_x, S_y\right) \\ &= \frac{L}{N}\min_{\varnothing}\sum_{K=1}^{N} d\left(\varnothing_{S_x}(k),\quad \varnothing_{S_y}(k)\right) \end{aligned} \tag{4}$$

where L is the length of the test segment S_y. Through scaling the average frame distance by the test segment length, the acoustic scores for different hypotheses of a test speech utterance (which in general consists of many segments) can be directly compared in template matching, as in HMM-based decoding search. Note that without the normalization by N in Equation 3, a template matching score for a speech segment would be affected by the length of the time warping path, which may vary with different templates; on the other hand, if the rescaling by L is not adopted, then the total distance on a decoding path would be dependent on the number of test segments in the path but not the lengths of these segments.

Commonly used local distances, such as Euclidean or Mahalanobis distances, compute the difference between two feature vectors directly [10], and they are thus of a feature-feature type. Let $\boldsymbol{x}$ and $\boldsymbol{y}$ represent two frames under comparison. The Euclidean distance is

$$d_{Euc}(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}-\boldsymbol{y})'(\boldsymbol{x}-\boldsymbol{y}) \tag{5}$$

and the Mahalanobis distance is

$$d_{Mah}(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}-\boldsymbol{y})'\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{y}) \tag{6}$$

with $\boldsymbol{\Sigma}$ as the covariance matrix estimated from training data.

When the template frames are represented by GMM indices, the Euclidean and Mahalanobis distances are no longer suitable. One possibility is to use negated log likelihood (NLL) score as a local distance. Let x_t and $y_{t'}$ be the frames of a test segment and a training template, respectively, and assume that $y_{t'}$ is labeled by a GMM $m_{1\left(y_{t'}\right)}$. The NLL distance is then

$$d_{NNL}\left(x_t, y_{t'}\right) = -\log p\left(x_t|m_{1\left(y_{t'}\right)}\right). \tag{7}$$

When $y_{t'}$ is represented by n GMMs $\left\{m_{1\left(y_{t'}\right)}, \ldots, m_{n\left(y_{t'}\right)}\right\}$ with the weights $\left\{w_{1\left(y_{t'}\right)}, \ldots, w_{n\left(y_{t'}\right)}\right\}$, the NLL distance becomes

$$d_{\mathrm{NNL}}\left(x_t, y_{t'}\right) = -\log\sum_{k=1}^{n} w_{k\left(y_{t'}\right)} p\left(x_t|m_{k\left(y_{t'}\right)}\right). \tag{8}$$

The NLL distance is of the feature-model type, as it does not use the information of the GMM labels on the test segment frames. The proposed log likelihood ratio and KL divergence distances make use of the GMM labels on both the test and the training frames. These two model-model distances are described below.

3.2.1 Log likelihood ratio local distance

Assuming that the test frame x_t is labeled by a GMM $m_{1(x_t)}$ and the training frame $y_{t'}$ is labeled by a GMM $m_{1\left(y_{t'}\right)}$. The LLR local distance between x_t and $y_{t'}$ is then defined as follows:

$$d_{LLR}\left(x_t, y_{t'}\right) = \log \frac{p\left(x_t | m_{1(x_t)}\right)}{p\left(x_t | m_{1\left(y_{t'}\right)}\right)}. \qquad (9)$$

The LLR distance contrasts the fit score of a test frame with its best model against its fit score with the best model of the template frame, and it therefore compares the two frames indirectly through the models. The LLR distance is nonnegative when 1-best GMM is used in frame labeling. When using multiple GMM indices for speech frame representation, the nonnegativity also holds if the weights are kept uniform, but it is not guaranteed if the weights are nonuniform, where the latter is due to the fact that although the GMM scores of the numerator are not smaller than those of the denominator, a skew in the denominator's weights toward some large GMM scores may make the denominator larger than the numerator. On the other hand, since what we really need is the difference of the numerator and denominator log likelihood scores as the dissimilarity between a test frame and a template frame, while a strict-sense log likelihood ratio is not needed here as in statistic hypothesis testing, we can therefore simply take the absolute value of the log likelihood score difference as the distance measurement which is also the rectified LLR given in Equation 10:

$$\begin{aligned} d_{LLR}\left(x_t, y_{t'}\right) &= \left| \log \sum_{k=1}^{n} w_{k(x_t)} p\left(x_t | m_{k(x_t)}\right) \right. \\ &\qquad \left. - \log \sum_{k=1}^{n} w_{k\left(y_{t'}\right)} p\left(x_t | m_{k\left(y_{t'}\right)}\right) \right| \\ &= \left| \log \frac{\sum_{k=1}^{n} w_{k(x_t)} p\left(x_t | m_{k(x_t)}\right)}{\sum_{k=1}^{n} w_{k\left(y_{t'}\right)} p\left(x_t | m_{k\left(y_{t'}\right)}\right)} \right| \end{aligned} \qquad (10)$$

(it is worth mentioning here that although getting a negative log likelihood ratio is a mathematical possibility, it never occurred in the experiments described in Section 5).

3.2.2 KL divergence local distance

In either the NLL distance or the LLR distance, the feature vector x_t is involved in the distance calculation. Here we consider measuring the local distance between two frames without using the feature vectors. KL divergence is widely used for measuring the difference between two probability distributions [29]. Since the frames are represented by GMM indices, the KL divergence between GMMs becomes a natural choice for indirectly measuring the dissimilarity of two frames. Because there is no closed-form expression for KL distance of GMMs, we use the Monte Carlo sampling method of Hershey and Olsen [30] to compute the divergence from a GMM m_x to a GMM m_y as

$$d\left(m_x \,||\, m_y\right) = \frac{1}{n_s} \sum_{i=1}^{n_s} \log \frac{m_x(x_i)}{m_y(x_i)} \qquad (11)$$

where the x_is are i.i.d. samples generated from the GMM m_x. Since the KL divergence is asymmetric, we further define a symmetric KL distance as

$$d_{KL}\left(m_x, m_y\right) = \frac{1}{2}\left(d\left(m_x \,||\, m_y\right) + d\left(m_y \,||\, m_x\right)\right). \qquad (12)$$

The local distance between the two frame vectors x_t and $y_{t'}$ is then calculated as

$$d\left(x_t, y_{t'}\right) = \sum_{k=1}^{n} \sum_{l=1}^{n} w_{k(x_t)} w_{l\left(y_{t'}\right)} d_{KL}\left(m_{k(x_t)}, m_{l\left(y_{t'}\right)}\right). \qquad (13)$$

3.3 PDT-based template clustering and matching score calculation

Considering the fact that certain triphone contexts may rarely occur or even be missing in a training set, we investigate tying triphone templates into clusters of equivalent contexts to improve the reliability of template matching as well as to handle unseen triphones in recognition. Among many possible clustering algorithms, we decide to utilize the PDT tying structures of the triphone states in the baseline HMMs directly to cluster triphone segments, since the tying structure of a phone state indicates partial similarities among triphone segments. We assume that each phone HMM has three emitting states as commonly used in HTK [31]. For the triphone templates of each monophone, we keep the three tying structures defined by the three emitting states of the corresponding phone HMM and use them jointly in template matching.

Specifically, in matching a test speech segment against a triphone arc in a word lattice, we first identify the three tied triphone clusters by answering the phonetic questions in the PDTs, and for an identified cluster i with k_i templates, we then choose $\sqrt{k_i}$ best-matching templates and average their matching scores for the test segment, and we further average the three scores of the three clusters as the matching score between the speech segment and the triphone arc. Using the square-root rule helps compress the variations of the number of templates k_i used in computing the scores, since the number often vary largely in different triphone clusters. The rule is also analogous to the *K*-

nearest neighbor (KNN) method where K is set as the square root of the training sample size [32].

Figure 2 illustrates the process of computing template matching score for lattice rescoring. It shows a phone lattice and a test speech segment X extracted from a speech utterance according to the start and end time of the phone arc P that has a predecessor phone P_L and successor phone P_R. Figure 3 illustrates the way that the matching scores of X with the three triphone template clusters containing $P_L - P + P_R$ are averaged to one matching score, which is used to replace the original acoustic score in the phone lattice for the phone arc P.

4 Template selection and compression

When the above-described template matching is used for lattice rescoring in LVCSR, the computation and storage overheads are still high. However, certain redundancies in the training templates can be reduced to improve computation and storage efficiency. We propose three methods of template selection and compression to address this problem. In template selection, the goal is to choose a small subset of templates as the representatives for the full set of training templates. In template compression, new GMMs are generated for labeling the frames of the selected template representatives so as to better capture the information in the training Templates.

4.1 Minimum-distance-based template selection

Agglomerative clustering [33] is a hierarchical clustering algorithm and it is widely used in pattern recognition, including speech recognition [34]. For selecting template representatives, we use the agglomerative clustering algorithm to further cluster the templates in each tied triphone cluster at a PDT leaf node, which recursively merges two closest clusters into one cluster until only one cluster is left. Given a distance function $D(C_i, C_j)$ for two clusters, the following procedure describes the algorithm for clustering m templates $\{s_1, s_2,\ldots,s_m\}$ in a leaf node of a PDT:

1. Initialize the template set $Z_1 = \{\{s_1\}, \{s_2\}, \ldots, \{s_m\}\}$ with each template s_i being a cluster.
2. For $n = 2,\ldots,m$: Obtain the new set Z_n by merging the two clusters C_i and C_j in the set Z_{n-1} with the distance $D(C_i, C_j)$ to be the minimum among all existing distinct cluster pairs. Stop the clustering process if the number of clusters in the set Z_n drops below a threshold.

The cluster distance function $D(C_i, C_j)$ is commonly defined by the distance of their elements $D(s_x, s_y)$, and the average distance measure is adopted here [33]:

$$D(C_i, C_j) = \frac{1}{|C_i||C_j|}\Sigma_{s_x \epsilon C_i}\Sigma_{s_y \epsilon C_j} D(s_x, s_y). \tag{14}$$

Note that $D(s_x, s_y)$ is the DTW distance of two templates as defined in Section 3.2, and in this step, the local distance d is the Euclidean distance of two frames.

To select a template representative for a cluster, we use the minimum distance from a template to all other templates in the cluster as the criterion, and therefore the method is called minimum distance template selection (MDTS). Given a cluster C_i, the template-to-cluster distance is defined as follows [33]:

$$D(s_x, C_i) = \sum_{\substack{s_{x'} \in C_i \\ s_x \neq s_{x'}}} D(s_x, s_{x'}), \tag{15}$$

and the template s^* is selected as the representative for the cluster C_i if its distance to the rest of the templates in the cluster is the minimum, i.e.,

$$s^* = \text{argmin}_{s_x \epsilon C_i} D(s_x, C_i). \tag{16}$$

The frames of the selected template representatives are subsequently indexed by their n-best GMMs according to Section 3.1.

4.2 Maximum-likelihood-based template selection

In maximum likelihood template selection, each frame of a cluster center s^* as generated by the MDTS method is relabeled by a set of GMMs that are selected by using a maximum likelihood criterion, so as to make the representative better characterize the templates in each cluster. For maximum likelihood template selection (MLTS), we

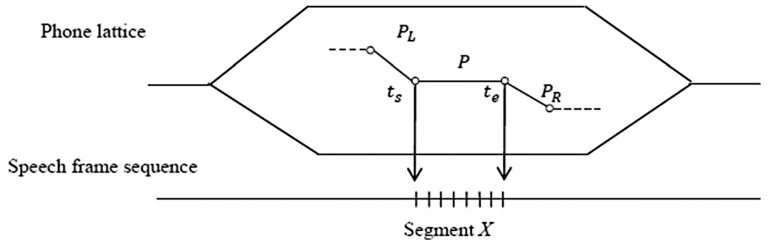

Figure 2 A faction of a phone lattice and a speech segment *X*.

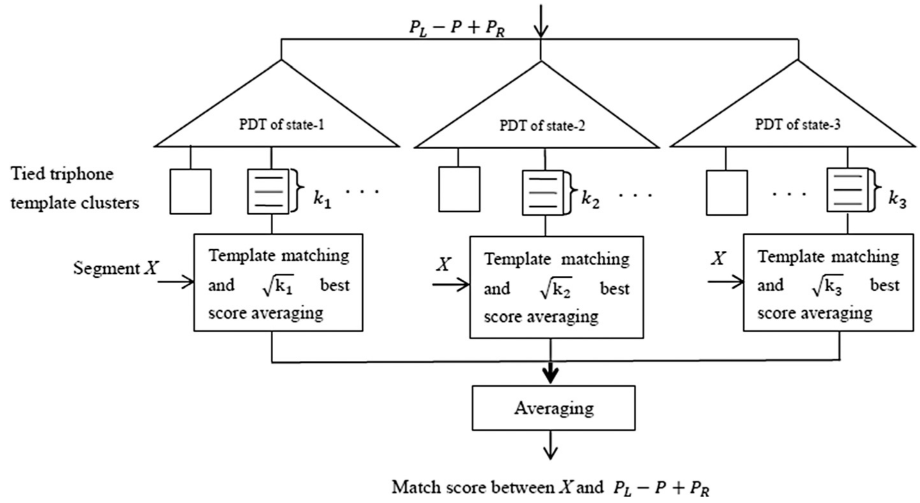
$P_L - P + P_R$
PDT of state-1
PDT of state-2
PDT of state-3
Tied triphone template clusters
k_1
k_2
k_3
Segment X
X
X
Template matching and $\sqrt{k_1}$ best score averaging
Template matching and $\sqrt{k_2}$ best score averaging
Template matching and $\sqrt{k_3}$ best score averaging
Averaging
Match score between X and $P_L - P + P_R$

Figure 3 Using three PDT clustering structures to calculate the template matching score.

use the DTW described in Section 3.2 to align the templates in a cluster C_i to the MDTS-initialized template center s^*. Figure 4 illustrates an outcome of aligning the sequences $s_1,\ldots, s_N$ to s^* in C_i, where the frames $x_{t_1}^{(1)}, \ldots, x_{t_N}^{(N)}$ of the sequences $s_1,\ldots, s_N$, respectively, are aligned to the frame $x_{t_*}^{(*)}$ of the cluster center s^*. The following procedure describes the MLTS method that is applied to relabel $x_{t_*}^{(*)}$ of s^* by using the aligned frames $X = \left\{x_{t_*}^{(*)},\ x_{t_1}^{(1)}, \ldots, x_{t_N}^{(N)}\right\}$:

1. Pool the distinct GMMs which are used to label the frames in X into a local GMM set M.
2. Use the K-medoids algorithm [33] with the KL distance to partition the GMM set M into l clusters M_i, $i = 1,\ldots, l$, where each M_i defines a subset of frames that are labeled by the GMMs in M_i.
3. For $i = 1,\ldots, l$: Use the maximum likelihood criterion to select a GMM of M_i as the cluster center m_i^* for M_i:

$$m_i^* = \text{argmax}_{m_i^j \epsilon M_i}\left(\sum\nolimits_{x\epsilon X_{M_i}} \log p(x|m_i^j)\right) \quad (17)$$

where m_i^j is the jth GMM in M_i.

4. For $i = 1,\ldots, l$: Calculate the weight w_i for each GMM cluster center m_i^*, which is proportional to the likelihood of X evaluated by m_i^*, i.e., $p\left(X\middle|m_i^*\right)$:

$$w_i = \frac{p\left(X\middle|m_i^*\right)}{\sum_{k=1}^{l} p\left(X\middle|m_k^*\right)} = \frac{e^{\sum_{x\epsilon X} \log p\left(x\middle|m_i^*\right)}}{\sum_{k=1}^{l} e^{\sum_{x\epsilon X} \log p\left(x\middle|m_k^*\right)}}. \quad (18)$$

After the relabeling, the frame x_t is represented by m_i^* and w_i, $i = 1,\ldots, l$. The MLTS procedure is applied to each frame of s^*. The resulting representation of s^* has a form similar to what is described in Section 3.1, with the difference that the best-fitting n GMMs of the baseline HMMs are used to label a frame in Section 3.1, but here the template frames that are aligned to a frame of the MDTS representative are used to select a set of l GMMs to relabel the frame of the representative.

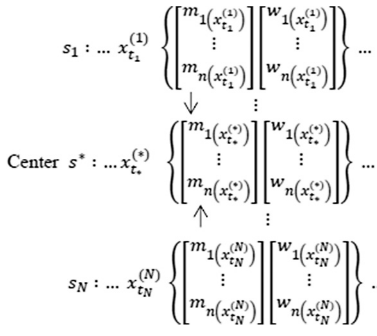

Figure 4 An alignment of the sequences s,..., s_N to s^*.

4.3 Template compression

The template compression method aims at taking in more information from the original templates for the template representatives. For each frame of a template representative, instead of selecting only one GMM and excluding the rest of the GMMs for a cluster M_i as in MLTS, here we merge the original GMMs in each cluster M_i into a new GMM and use the l new GMMs from the l clusters M_i, $i = 1,\ldots, l$ to relabel the frame. To reduce the negative effect of outlier templates, for each GMM m_i^j in a cluster M_i, we calculate its distance to the cluster center m_i^* based on the KL distance $d_i^j = d\left(m_i^j, m_i^*\right)$. From the distances d_i^j

of M_i, the mean $\bar{d}$ and the standard deviation σ are computed. If a GMM m_i^j is t times standard deviation away from $\bar{d}$, i.e.,

$$|d_i^j - \bar{d}| > t\sigma \tag{19}$$

then it is considered an outlier and is removed from the merging process. Suppose that after removing the outliers, there are n_G GMMs left in M_i. We first pool the component Gaussian densities from the n_G GMMs and normalize the weight of each Gaussian component by n_G. We then merge the pooled Gaussian components according to the criterion of minimum entropy increment. The entropy increase due to merging two Gaussian components $f_i \sim N(\mu_i, \Sigma_i)$ and $f_j \sim N(\mu_j, \Sigma_j)$ into $N(\mu, \Sigma)$ is defined as [35]:

$$\Delta E\left(f_i,\ f_j\right) = \log|\Sigma| - \frac{w_i}{w_i + w_j}\log|\Sigma_i| - \frac{w_j}{w_i + w_j}\log\left|\Sigma_j\right| \tag{20}$$

where w_i and w_j are the normalized mixture weights for f_i and f_j. The mean μ, covariance Σ, and mixture weight w of the newly generated Gaussian component are defined as

$$\begin{aligned} \Sigma &= \frac{w_i}{w_i + w_j}\Sigma_i + \frac{w_j}{w_i + w_j}\Sigma_j + \frac{w_i w_j}{\left(w_i + w_j\right)^2}\left(\mu_i - \mu_j\right)\left(\mu_i - \mu_j\right)' \\ \mu &= \frac{w_i}{w_i + w_j}\mu_i + \frac{w_j}{w_i + w_j}\mu_j \\ w &= w_i + w_j. \end{aligned} \tag{21}$$

The Gaussian components are merged iteratively until the number of components in M_i is below a preset threshold. The remaining Gaussian components are used to construct a new GMM, and the new GMM is used as one of the l GMMs to label the corresponding frame of the template representative.

The flowcharts of the above-discussed three template selection and compression methods are given in Figure 5. As shown in the figure, the three methods share the same template representatives that are selected from the original GMM-labeled templates. While MDTS stops here, MLTS reselects the GMM labels for the representative frames, and template compression generates new GMMs and uses them to relabel the frames of the template representatives. As are shown in the experimental results of Section 5, the refinements on the GMM labels make the template representatives more effective, and when coupled with a proper local distance they allow using only a small fraction of template representatives in lattice rescoring with little performance loss.

5 Experimental results

We performed speaker-independent phone recognition on the task of TIMIT [27] and speaker-dependent large vocabulary speech recognition on the task of telehealth captioning [28]. The experimental outcomes were measured in phone accuracy and word accuracy, respectively, for TIMIT and telehealth through aligning each phone or word string hypothesis against its reference string by using the Levenshtein distance [31].

5.1 Corpora

The TIMIT training set consisted of 3,696 sentences from 462 speakers and the standard test set included 1,344 sentences spoken by 168 speakers. The telehealth task included spontaneous speech from five doctors and the vocabulary size was 46,000. A summary of the Telehealth corpus is given in Table 1, where the word counts from the transcription texts are also listed. For a detailed description of this task, please refer to [28].

5.2 Experimental set up and lattice rescoring

For both tasks of TIMIT and telehealth, the speech features consisted of 13 MFCCs and their first- and second-order time derivatives, and crossword triphone acoustic models were trained by using HTK toolkit. In calculating a KL distance between two GMMs [30], 10,000 Monte Carlo simulation data samples were generated.

For the TIMIT dataset, the set of 39 phones was defined as in [36], and a phone bi-gram language model (LM) was used (trained from the TIMIT training speech transcripts). The HMM baseline was trained with the GMM mixture sizes of 24; and 1,189 GMMs were extracted for template construction. The total original triphone templates were 152,715 in the training set. Phone lattices were generated for each test sentence by HTK. The average number of nodes per lattice was in the order of 850, and the average number of arcs was in the order of 2,350.

For the telehealth task, speaker-dependent acoustic models were trained for five healthcare provider speakers Dr. 1 to Dr. 5. In the baseline acoustic model, each GMM included 16 Gaussian components and on average, 1,905 GMMs were extracted from the baseline HMMs of each of the five doctors. The average number of triphone templates was 181,601 per speaker for the five doctors. Trigram language models were trained on both in-domain and out-of-domain datasets, where word-class mixture trigram language models with weights obtained from a procedure of forward weight adjustment were used [37]. For each test sentence, word lattices including phone boundaries were generated by HTK. The average number of nodes per lattice was in

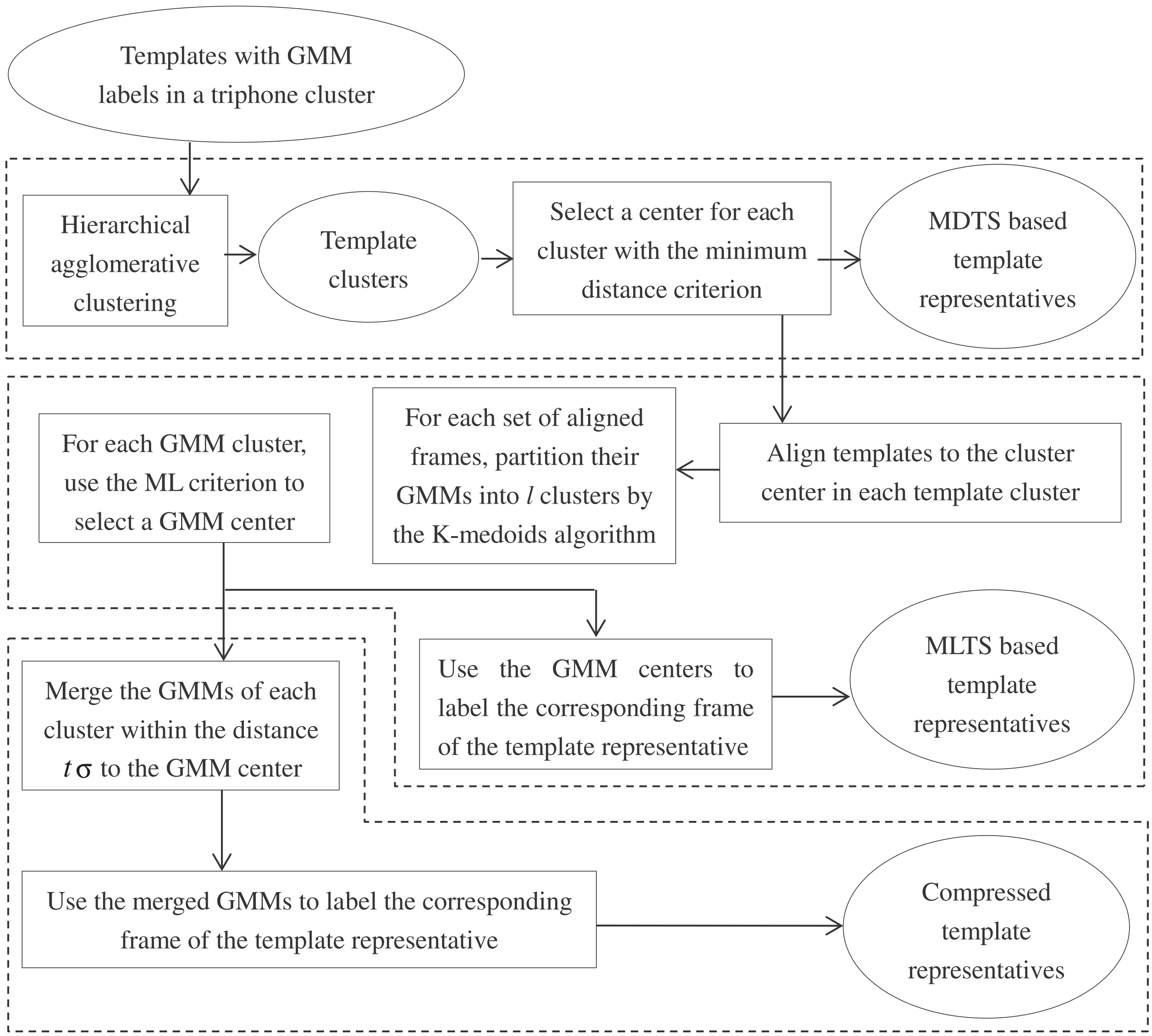
Templates with GMM labels in a triphone cluster
Hierarchical agglomerative clustering
Template clusters
Select a center for each cluster with the minimum distance criterion
MDTS based template representatives
For each GMM cluster, use the ML criterion to select a GMM center
For each set of aligned frames, partition their GMMs into l clusters by the K-medoids algorithm
Align templates to the cluster center in each template cluster
Merge the GMMs of each cluster within the distance $t\sigma$ to the GMM center
Use the GMM centers to label the corresponding frame of the template representative
MLTS based template representatives
Use the merged GMMs to label the corresponding frame of the template representative
Compressed template representatives

Figure 5 Flowcharts of MDTS, MLTS, and template compression.

the order of 700, and the average number of arcs was in the order of 1,950.

In rescoring a lattice, the acoustic score of each phone arc in the lattice was replaced by its corresponding triphone template matching score, where the distance score of Equation 4 was negated to become a similarity score. By using the acoustic similarity scores and the original language model scores, the best path with the largest sum of acoustic and language model log scores was searched on the lattice using dynamic programming to produce the rescored sentence hypothesis.

Table 1 Datasets used in the telehealth task: speech (min)/text (no. of words)

	Training set	Test set
Dr. 1	210/35,348	19.3/3,248
Dr. 2	200/39,398	29.8/5,085
Dr. 3	145/28,700	12.1/3,988
Dr. 4	180/39,148	14.3/2,759
Dr. 5	250/44,967	27.8/6,421
Total	985/187,561	103.3/21,501

5.3 TIMIT phone recognition task

On the TIMIT task, we provide a detailed account of the factors in the proposed template matching methods that affect the rescoring performance, including local distances, number of GMMs employed for frame labeling, template selection, compression methods and their interactions with the local distances, and the percentage of selected template representatives. We also examine the patterns of phone error reduction and look at the cost-performance tradeoffs.

5.3.1 Local distances

In Figure 6, we compare the phone recognition performances by using the HMM baseline and the template-matching-based lattice rescoring with the local distances of Mahalanobis, NLL, LLR, and KL divergence. Except for the baseline and the Mahalanobis distance, each frame of a template or a test speech segment was labeled by 1GMM index. The HMM baseline had the phone accuracy of 72.72%. In template matching, the Mahalanobis and NLL local distances improved the baseline by merely 0.11% and 0.12% absolute, respectively, but the LLR and KL distances improved the HMM baseline by 1.30% and 0.96% absolute, respectively. The LLR distance gave higher phone accuracies than the KL distance did. This may be attributed to the fact that the KL divergence measures the difference between GMM distributions but not directly the difference between feature vectors, whereas the LLR distance contrasts the likelihood scores of two sets of GMMs for each test frame, and therefore it reflects the characteristics of the test frame more closely. Giving the superiority of the proposed LLR and KL distances, we only use these two local distances in the subsequent experiments.

5.3.2 Number of GMMs for frame labeling

In Figure 7, we show the effects of using different numbers of GMMs ($n = 1$, $n = 3$, $n = 5$, and $n = 7$) in labeling each frame of the templates. For both LLR and KL distances, the accuracy performance peaked when five GMMs were used for frame labeling, and phone accuracies of 74.51% and 74.26% were achieved for LLR and KL distances with absolute improvements of 1.79% and 1.54%, respectively, over the HMM baseline of 72.72%. The results confirmed the advantage of using multiple GMMs for frame labeling over using single GMM, as the former induced smaller quantization errors than the latter. However, using too many GMMs to represent a frame could increase confusion and reduce efficiency. We conducted significance tests on the performance difference between the '5GMMs' case and the HMM baseline. Let x_i and y_i be the phone recognition accuracy of the ith test sentence for the baseline and a proposed method, respectively. Let $t_i = y_i - x_i$ and denote the sample mean and sample variance of t_i as $\bar{t}$ and s^2 with the sample size m. The Student's t test statistic is $T = \bar{t}/(s/\sqrt{m})$. In the TIMIT standard test set, $m = 1{,}344$ and $t_{m-1,1-0.05} = 1.65$ for one-sided test. For the LLR and KL local distances, we obtained $T > t_{m-1,1-0.05}$, and therefore our proposed template matching methods using the LLR and KL distances improved TIMIT phone recognition accuracy significantly over the HMM baseline at the significance level of 0.05. We also used two-fold cross-validation on the test set to automatically select the number of GMMs for frame labeling, and the case of 5GMM was selected in each validation set. Therefore, the result of the 5GMM case in Figure 7 also represents an open test performance. In the subsequent experiments, five GMMs were used for labeling each frame.

5.3.3 Template selection and compression

The performances of template selection and compression exhibited a dependency on the local distance measures. Here we discuss how the three methods of (1) MDTS, (2) MLTS, and (3) template compression performed when using the LLR and KL distances and show the results in Figure 8, where the number of template representatives were kept to be 20% of the total templates for the three cases (further details are discussed in Section 5.3.5). In template compression, the threshold t in Equation 19 was set to 2 for removing GMM outliers, and the number of Gaussian components in each merged GMM was 24, the same as the GMM mixture size in the baseline HMMs, with a total of 749 newly generated GMMs for the template representatives. In MDTS,

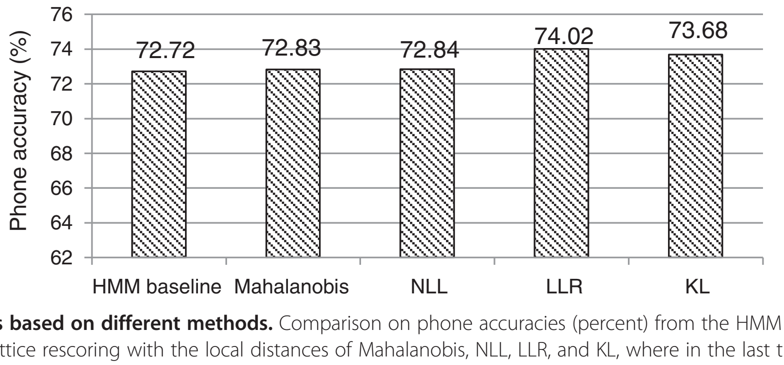

Figure 6 Phone accuracies based on different methods. Comparison on phone accuracies (percent) from the HMM baseline and the template-matching-based lattice rescoring with the local distances of Mahalanobis, NLL, LLR, and KL, where in the last three cases 1GMM was used in labeling each frame vector.

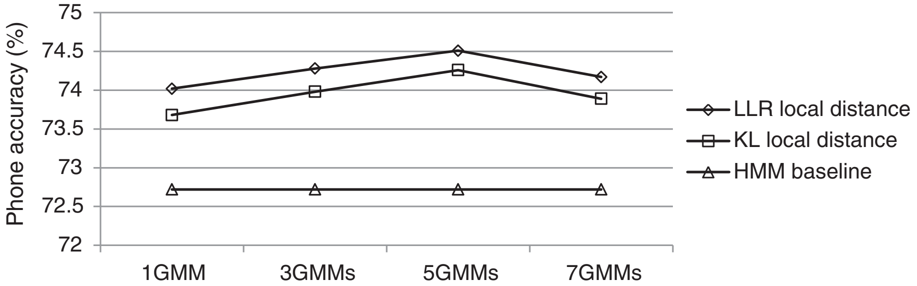

Figure 7 Lattice rescoring phone accuracy (percent) using different numbers of GMM indices for frame representation. Using multiple GMMs such as 3, 5, and 7 to label each frame can get better performance than using one single GMM. For both LLR and KL distances, the accuracy performance peaked when five GMMs were used for frame labeling.

the phone accuracies were 73.82% and 72.70% for the LLR and KL distance, respectively, and in MLTS, the phone accuracies were 74.05% and 73.07% for the LLR and the KL distance, respectively. Relative to MLTS, template compression increased absolute phone accuracy by 0.27% with the KL distance and it decreased absolute phone accuracy by 0.40% with the LLR distances. Several points worth noting in Figure 8 are discussed below.

First, MDTS worked well with the LLR distance but poorly with the KL distance, and vice versa for MLTS and template compression. In MDTS, the template representative frames were labeled in the same way as the test frames, i.e., by the best-fit GMMs of the baseline model, and in this case, a better outcome of LLR than KL is consistent with what was shown in Figure 6 for using all templates. In MLTS, however, the selected template representative frames were relabeled by GMMs to maximize the likelihood of the aligned template frames, and template compression went further by generating new GMMs from the baseline GMMs and used the new GMMs to relabel the representative frames. Because in MLTS or template compression the template representative frames were no longer labeled by the best-fit GMMs, the LLR distance that contrasted the model-frame fit became ineffective in comparison with the KL distance that measures the distance between GMMs.

Second, relative to using all of the original templates as discussed in Section 5.3.2, using 20% template representatives that were selected by MLTS with the KL distance slightly decreased phone accuracy by 0.21% (from 74.26% to 74.05%), but using the template representatives selected by MDTS with the LLR distance significantly decreased phone accuracy by 0.69% (from 74.51% to 73.82%). This difference may be explained by the fact that MDTS simply selects a cluster center as a template representative, but MLTS further refines the GMM indices of each template representative frame to maximize the likelihood of the aligned frames in the corresponding cluster. In this way, MLTS absorbs more information from the training data into the template representatives than MDTS, and so fewer template representatives are needed in MLTS than in MDTS.

Third, with the KL distance, template compression further improved the performance over MLTS, where by using 20% template representatives, phone accuracy was actually improved by 0.06% over the case of using all templates (from 74.26% to 74.32%). This indicates that the new GMMs were more effective in labeling the

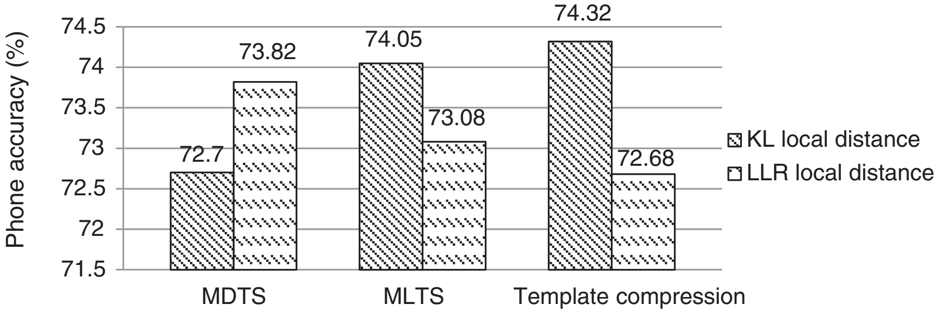

Figure 8 Phone accuracies (percent) for methods of template selection and compression with KL and LLR local distances. The three methods of template selection and compression interact with the LLR and KL local distances in different ways, and therefore each selection or compression method has its most compatible local distance. Here the number of template representatives was kept to be 20% of total templates.

Table 2 Phone accuracies (percent) from using different outlier threshold values for the compressed template representatives

Threshold $t\sigma$	1σ	2σ	3σ	∞
Accuracy (%)	73.95	74.32	73.42	70.89

template representative frames, and the exclusion of the outlier GMMs was helpful, too.

In summary, MDTS worked well with LLR distance, and MLTS and template compression worked well with KL distance. Using the respectively compatible local distances and fixing the selection percentage at 20%, template compression performed the best, MLTS the next, and MDTS the last. Specifically, the accuracy gains over the HMM baseline were 1.6% absolute by template compression with KL, 1.33% by MLTS with KL, and 1.1% by MDTS with LLR. We also conducted the Student's t test on the performance differences between each of the three methods (with respectively compatible distance) and the HMM baseline, and the three methods all significantly improved phone accuracy over the baseline at the level of $\alpha = 0.05$.

5.3.4 Evaluation on the outlier threshold t

In Table 2, we show how the threshold value t of Equation 19 for removing the GMM outliers affected the recognition performance, where the template selection method was MLTS with the KL distance, and 20% template representatives were selected. Among the four t values studied here, it is observed that $t = 2$ gave the best phone accuracy performance. Also note that when $t = \infty$, all GMMs in a cluster were used to generate compressed templates, where the existence of outliers degraded the accuracy performance significantly. Accordingly, the threshold $t = 2$ was used in all the template compression experiments.

5.3.5 Evaluation on the number of template representatives in template selection methods

In Figure 9, we show how the percentages of template representatives selected from the total templates affect phone accuracies for MDTS and MLTS with their respectively compatible distances. The number of GMM clusters l in MLTS was set to 5, corresponding to using five GMMs to label each frame of a template representative. It is seen from the two curves that with the percentage varied from 100% down to 1%, the phone accuracies decreased for both methods. When 100% templates were used, i.e., without template selection, LLR distance performed better than KL distance, as discussed in Section 5.3.1 and Section 5.3.2. When less than 80% templates were used, MLTS performed better than MDTS since the MLTS templates generalized better than MDTS templates, as discussed in Section 5.3.3. For MDTS, when the selection percentage reduced from 100% to 60%, the phone accuracy dropped rapidly by 0.55% (from 74.51% to 73.96%), and when the selection percentage reduced from 60% to 20%, the phone accuracy reduced slowly by 0.14% (from 73.96% to 73.82%). In contrast, for MLTS, with the selection percentage reduced from 100% to 20%, the phone accuracy went down gradually by 0.21% (from 74.26% to 74.05%). Moreover, both curves went down rapidly when the selection percentage was further reduced below 20%. From Figure 9, we conclude that MLTS is more robust to using a small percentage of template representatives, and the selection percentage of 20% is a reasonable compromise between accuracy performance and computation and storage cost.

5.3.6 Phone accuracy analysis

In order to better understand the effect of the proposed template matching methods, we compare the patterns of TIMIT phone accuracies from using the methods of all templates with the KL and LLR local distances against that of the HMM baseline. Table 3 provides the phone accuracies of the five broad phone classes (vowels, semivowels, stops, fricatives, and nasals) and the accuracy of silence for the HMM baseline and template matching. In Figure 10, we plot the absolute phone accuracy changes

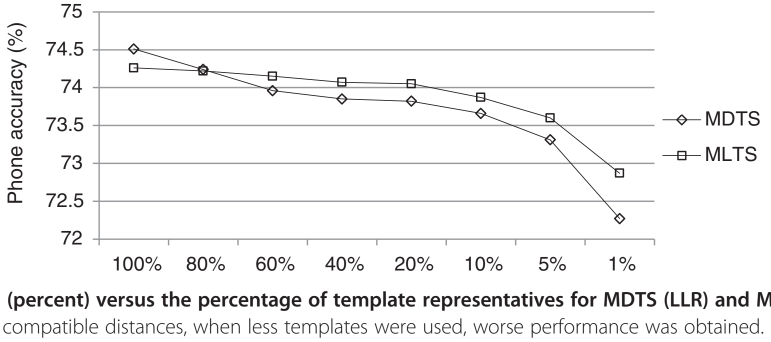

Figure 9 Phone accuracies (percent) versus the percentage of template representatives for MDTS (LLR) and MLTS (KL). For MDTS and MLTS with their respectively compatible distances, when less templates were used, worse performance was obtained. MLTS is more robust to using a small percentage of template representatives, and the selection percentage of 20% is a reasonable compromise between accuracy performance and computation and storage cost.

Table 3 Phone accuracies (percent) of vowels, semivowels, stops, fricatives, nasals, and silence

	Vowels	Semivowels	Stops	Fricatives	Nasals	Silence
HMM baseline	63.48	72.47	73.65	74.83	72.03	86.02
KL distance	68.30	76.52	73.45	72.37	72.53	85.48
LLR distance	68.32	76.85	75.65	71.95	72.66	85.57

of template matching against the HMM baseline. For the vowel class, the KL- and LLR-based template matching produced absolute phone accuracy gains of 4.82% and 4.84%, respectively, and for the semivowel class, the absolute accuracy gains were 4.05% and 4.38%, in the same order. For the stop class, template matching using the LLR distance made an absolute gain of 2.0% while using the KL distance did not help. For the fricative class, phone accuracies were decreased 2.46% and 2.88% by the KL- and LLR-based template matching, respectively. For the nasal class, there were small phone accuracy gains, and for silence, there were small accuracy degradation by template matching, but both changes were small and insignificant.

It is not surprising that the template-based methods produced the largest positive impact on semivowels (largest relative phone error reduction). Semivowels are transient sounds and templates can capture their trajectory information better than HMM. Similarly, some vowel sounds are nonstationary, such as diphthongs or vowels in strong coarticulation. Stops, having the closure and burst pattern, are nonstationary as well and often have short durations, and they are difficult to model by HMM but can be better represented by templates, as reflected in the accuracy gain by the LLR-based template matching. Fricatives are noise like and without clear trajectory patterns, and their boundaries are also difficult to determine, making template-based methods not as effective as HMMs.

5.3.7 Computation time and memory overhead

We first compare the storage space costs of the conventional and the proposed template representation methods, assuming a speech feature vector is 39 dimensional as in the baseline HMM. In conventional template methods that use Mahalanobis local distance, a speech frame is represented by a 39-dimensional vector (float), while in the proposed method a frame is labeled by n GMM indices (short integer) and n −1 weights (float). On a 32-bit machine and with $n = 5$ in our experiments, the proposed method used 26 $(5 \times 2 + (5–1) \times 4)$ bytes per frame versus the conventional method of 156 bytes per frame, which amounts to an 83% saving in storage space. For the TIMIT dataset, there were 152,715 phone templates and the average length of a phone template was eight frames (with the frame shift of 10 ms), giving a total of 1,221,720 frames and an overhead for template storage of 30.3 MB. In template selection, the memory overhead was around 6.1 MB when 20% templates were selected to be the representatives. In template compression, the memory overhead for template storage was the same as in template selection. However, since there were 749 new GMMs for labeling the frames of the template representatives, there was an extra memory overhead of 5.4 MB.

In Table 4, we provide a comparison on the per-frame computational time for the proposed template-matching-based lattice rescoring and the HMM baseline. The computation time was divided into two parts. One part was on test-frame labeling which used GMMs from the HMM

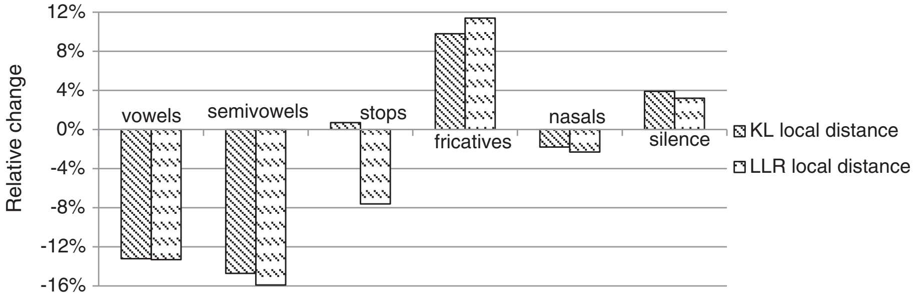

Figure 10 Phone accuracy change due to template-matching-based rescoring with respect to HMM baseline. For the vowel and semivowel classes, the KL- and LLR-based template matching obtained better performance than HMM baseline. For the stop class, template matching using the LLR distance got better phone accuracy than HMM baseline while using the KL distance did not help. For the fricative class, the phone accuracies were worse than HMM baseline for both KL- and LLR-based template matching. For the nasal class and silence, the changes between template-based methods and HMM baseline were not significant.

Table 4 Computational overhead (percent) per frame using all templates, template selection, and template compression for TIMIT phone recognition

	All templates	Template selection	Template compression
Test frame labeling overhead	40.0	40.0	22.4
Rescoring overhead	22.0	4.4	4.4
Overall computational overhead	62.0	44.4	26.8

baseline and the time was proportional to the total number of GMMs extracted from the HMMs. The other part was the rescoring time which calculated the DTW matching scores between a test segment (time marked by a phone arc on the phone lattice) and templates in a template clusters (specified by the PDTs of the phone unit). The more templates were in a template cluster, the longer the rescoring time. Since the KL distances between the GMMs were pre-calculated and the likelihood scores used in LLR distance were obtained in the test frame labeling, the time for rescoring was mainly consumed on determining the warping path in DTW, and hence for the LLR and KL distances, the rescoring times were similar (we therefore omit the local distance in Table 4). Relative to the decoding time per frame of the HMM baseline, when all templates were used, the test per-frame labeling overhead was 40% and the rescoring overhead was 22%, and hence the overall computational overhead per frame was 62.0%. In template selection, since only 20% template representatives were used, the rescoring time was reduced to 1/5 of the all-template case, and the computational overhead became 44.4%.In template compression, the number of new GMMs that were merged from the baseline GMMs was about 63% of the baseline GMMs (749 vs. 1,189), the time consumed for test frame labeling also decreased, and the computation overhead was reduced to 26.8%. Based on these numbers, we conclude that by using template representatives with a selection percentage of 20%, the costs in computation time and storage space were greatly reduced.

5.4 Large vocabulary speech recognition task

Based on the outcomes of the TIMIT phone recognition task, we only report the telehealth results for the following three cases of template matching: (1) all templates with LLR distance, (2) MLTS with KL distance, and (3) template compression with KL distance, where the cases 2 and 3 used 20% templates as the representatives and word accuracy was averaged over the five doctors. In template compression, the number of Gaussian components in each new GMM was 16, the same as the GMMs of the baseline; the average number of GMMs generated for the compressed template representatives was 1,048 per doctor (the baseline was 1,905 GMMs per doctor). The HMM baseline was trained using crossword triphone models, with an average word accuracy of 78.43%. In Table 5, we compare the recognition word accuracies between the HMM baseline and the template-based methods. In case 1, the average word accuracy was 80.03%, which is an absolute gain of 1.6% over the baseline. In case 2, the average word accuracy was 79.40%, which is an absolute gain of 0.97% over the baseline. In case 3, the word accuracy was 79.70%, which is an absolute gain of 1.27% over the baseline. Again, we conducted a Student's t test on the word accuracy gain (averaged over the five doctors) obtained by each of the three cases over the baseline and found the performance gain in every case to be statistically significant at the level of $\alpha = 0.05$.

In Table 6, the average computation cost of the five doctors is given for the three cases. In comparison with the TIMIT phone recognition task, even though there were more GMMs to be used for test frame labeling and more templates in template clusters, the computation overhead did not increase much, especially for template selection and template compression. In addition, the memory overhead for all five doctors was around 236.2, 47.2, and 73.2 MB for using all templates, selected template representatives, and compressed template representatives, respectively. Therefore, the template-based methods, especially MLTS and template compression, are affordable for LVCSR.

5.5 Discussion

So far we have shown that representing the template frames by GMMs and using the local distance measures

Table 5 Comparison of word accuracies (percent) between the HMM baseline and the template-based methods

Speakers (no. of words)	Dr. 1 (3,248)	Dr. 2 (5,085)	Dr. 3 (3,988)	Dr. 4 (2,759)	Dr. 5 (6,421)	Average
Baselines	72.14	82.50	84.00	74.20	79.32	78.43
All templates (LLR)	73.53	84.22	85.98	75.74	80.67	80.03
MLTS (KL)	73.22	83.39	84.87	75.35	80.15	79.40
Template compression (KL)	73.55	83.61	85.21	75.71	80.39	79.70

Word accuracies (%) for HMM baselines, LLR-based all templates, and KL-based MLTS and template compression for five doctors in the telehealth task.

Table 6 Average computation overhead (percent) per frame of the five doctors

	All templates (LLR)	MLTS (KL)	Template compression (KL)
Test frame labeling overhead	43.3	43.3	23.3
Rescoring overhead	26.7	5.4	5.4
Overall computational overhead	70.0	48.7	28.7

Computational overhead per frame using LLR-based all templates and KL-based MLTS and template selection for five doctors in the telehealth task.

of LLR or KL significantly improved the accuracy performance over our HMM baselines, and the proposed methods are much more effective than the conventional template matching methods where the template frames use the original speech features. A question naturally arises is how would the proposed template-matching methods interact with an underlying acoustic model from which the GMMs are derived and the phone or word lattices are generated, and of particular interest is that as a baseline HMM system improves, whether the performance gain we have observed by the proposed template matching methods can still hold. This is a relevant issue since a baseline HMM system can be improved by using more advanced training methods and better features. Recently, a major advance has been made in using deep neural networks (DNNs) with many hidden layers for speech acoustic modeling, where the resulting DNNs learn a hierarchy of nonlinear feature detectors that can capture complex statistical patterns for speech data. For example, context-independent, pretrained DNN/HMM hybrid architectures have achieved competitive performance in TIMIT phone recognition [38], context-dependent DNN/HMM has led to large improvements to several public domain large speech recognition tasks [39], and dumping features from deep convolutional neural networks to train GMM/HMM-based systems achieved higher accuracy performance than DNN/HMM hybrid architectures in several tasks [40].

We have investigated this issue in [41] on the TIMIT phone recognition task by performing lattice rescoring with the proposed template-matching methods on top of progressively better HMM baselines, where the test set was the same as discussed in Section 5.1. The HMM baseline system employed discriminative training, neural-network-derived phone posterior probability features, as well as ensemble acoustic models, etc. We observed that with the baseline system phone accuracy raised to 73.25%, 75.66%, 76.51%, and 77.97%, the template-matching-based lattice rescoring delivered consistent performance gains and gave phone accuracies of 74.74%, 77.27%, 77.96%, and 79.55%, respectively, where the phone accuracy of 79.55% was among the best reported results on the TIMIT continuous phoneme recognition task. For the sake of space, we omit the details of these baseline systems. For further information, please refer to [41]. The consistent performance gains support the notion that template matching improves recognition accuracy through a mechanism different from HMM. This is in agreement with the observation in [10] that the template matching system and the HMM system behave differently in word error patterns. Since our template-based methods are compatible with the GMMs trained from neural-network-derived features, it is reasonable to expect that our methods can take advantage of and add value to the advancements in this research direction.

6 Conclusions

In this paper, we have presented a novel approach of integrating template matching with statistical modeling for continuous speech recognition. The approach inherits the GMMs and the PDT state tying structures from the baseline HMMs and is therefore easily implemented. Generating template representatives and representing the frames by GMM indices make the approach extendable to LVCSR task. Based on our experimental results from the tasks of TIMIT phone recognition and telehealth LVCSR, we conclude that the proposed method of integrating template matching and statistical modeling has significantly improved the recognition performance over our HMM baselines, and the proposed template selection and compression methods have also largely saved computation time and memory space over using all templates with small losses in accuracy performance. Although in the current work we used the basic acoustic modeling techniques to train our HMM baselines, the proposed template matching methods can take advantage of and add value to more advanced GMM/HMM systems, and as such they are promising for further improving the state-of-the-art speech recognition.

Competing interests

The authors declare that they have no competing interests.

Disclosures

The work described in this paper was conducted during the first author's Ph.D. study at the University of Missouri-Columbia, USA.

Author details

[1]Nuance Communications, Inc, 1 Wayside Rd, Burlington, MA 01801, USA.
[2]Department of Computer Science, University of Missouri, Columbia, MO 65211, USA.

Received: 3 September 2013 Accepted: 14 January 2014
Published: 1 February 2014

References

1. M Ostendorf, V Digalakis, OA Kimball, From HMMs to segment models: a unified view of stochastic modeling for speech recognition. IEEE Trans on SAP **4**(5), 360–378 (1996)
2. S Furui, Speaker-independent isolated word recognition using dynamic features of speech spectrum. IEEE Trans SAP **ASSP-34**(1), 52–59 (1986)
3. H Gish, K Ng, Parametric trajectory models for speech recognition. Proc of ICSLP1 , 466–469 (1996)
4. J Glass, A probabilistic framework for segment-based speech recognition. Computer Speech and Language **17**(2–3), 13–152 (2003)
5. G Zweig, P Nguyen, A segmental CRF approach to large vocabulary continuous speech recognition, in *IEEE Workshop on Automatic Speech Recognition & Understanding* (, Merano, 2009), pp. 152–157
6. L Deng, D Yu, A Acero, A long-contextual-span model of resonance dynamics for speech recognition: parameter learning and recognizer evaluation, in *Proceedings of IEEE Workshop on Automatic Speech Recognition and Understanding* (San Juan, 2005), pp. 145–150
7. K Demuynck, D Seppi, H van Hamme, D van Compernolle, Progress in example based automatic speech recognition, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Prague, 2011), pp. 4692–4695
8. TN Sainath, B Ramabhadran, D Nahamoo, D Kanevsky, D van Compernolle, K Demuynck, JF Gemmeke, JR Bellegarda, S Sundaram, Exemplar-based processing for speech recognition. IEEE Signal Process. Mag. **29**, 98–113 (2012)
9. TN Sainath, B Ramabhadran, S Nahamoo, S Kanevsky, A Sethy, Exemplar-based sparse representation features for speech recognition, in *Proceedings of I NTERSPEECH 2010* (Makuhari, 2010), pp. 2254–2257
10. M de Wachter, M Matton, K Demuynck, P Wambacq, R Cools, D van Compernolle, Template-based continuous speech recognition. IEEE Trans ASLP **15**(4), 1377–1390 (2007)
11. X Sun, Y Zhao, Integrate template matching and statistical modeling for speech recognition, in *Proceedings of INTERSPEECH 2010* (Makuhari, 2010), pp. 74–77
12. L Rabiner, B Juang, *Fundamentals of Speech Recognition* (Prentice Hall, Englewood Cliffs, 1993)
13. S Demange, D van Compernolle, HEAR: an hybrid episodic-abstract speech recognizer, in *Proceedings of INTERSPEECH 2009* (Brighton, 2009), pp. 3067–3070
14. L Golipour, D O'Shaughnessy, Phoneme classification and lattice rescoring based on a k-NN approach, in *Proceedings of INTERSPEECH 2010* (Makuhari, 2010), pp. 1954–1957
15. K Demuynck, K Demuynck, D Seppi, D van Compernolle, P Nguyen, G Zweig, Integrating meta-information into exemplar-based speech recognition with segmental conditional random fields, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Prague, 2011), pp. 5048–5051
16. S Sundaram, JR Bellegarda, Latent perceptual mapping: a new acoustic modeling framework for speech recognition, in *Proceedings of INTERSPEECH 2010* (Makuhari, 2010), pp. 881–884
17. X Sun, Y Zhao, New methods for template selection and compression in continuous speech recognition, in *Proceedings of INTERSPEECH 2011* (Florence, 2011), pp. 985–988
18. D Seppi, K Demuynck, D van Compernolle, Template-based automatic speech recognition meets prosody, in *Proceedings of INTERSPEECH 2011* (Florence, 2011), pp. 545–548
19. D Seppi, D Van Compernolle, Data pruning for template-based automatic speech recognition, in *Proceedings of INTERSPEECH 2010* (Makuhari, 2010), pp. 901–904
20. S Sundaram, J Bellegarda, Latent perceptual mapping with data driven variable-length acoustic units for template-based speech recognition, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Kyoto, 2012), pp. 4125–4128
21. G Heigold, P Nguyen, M Weintraub, V Vanhoucke, Investigations on exemplar-based features for speech recognition towards thousands of hours of unsupervised, noisy data, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Kyoto, 2012), pp. 4437–4440
22. J Ming, Maximizing the continuity in segmentation- a new approach to model, segment and recognize speech, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Taiwan, 2009), pp. 3849–3852
23. J Ming, R Srinivasan, D Crookes, A Jafari, CLOSE—a data-driven approach to speech separation. IEEE Trans ASLP **21**(7), 1355–1368 (2013)
24. A Garcia, H Gish, Keyword spotting of arbitrary words using minimal speech resources, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, (ICASSP), vol. 1* (Atlanta, 2006), pp. 123–127
25. T Hazen, W Shen, C White, Query-by-example spoken term detection using phonetic posteriorgram templates, in *IEEE Workshop on Automatic Speech Recognition & Understanding* (Merano, 2009)
26. Y Zhang, J Glass, Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams, in *IEEE Workshop on Automatic Speech Recognition & Understanding* (Merano, 2009)
27. L Lamel, R Kassel, S Seneff, Speech database development: design and analysis of the acoustic-phonetic corpus, in *Proceedings of the DARPA Speech Recognition Workshop*, 1989
28. Y Zhao, X Zhang, R Hu, J Xue, X Li, L Che, R Hu, L Schopp, An automatic captioning system for telemedicine, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Toulouse, 2006), pp. 957–960
29. S Kullback, Letter to the editor: the Kullback–Leibler distance. Am. Stat. **41**(4), 338–341 (1987)
30. JR Hershey, PA Olsen, Approximating the Kullback–Leibler divergence between Gaussian mixture models, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol 4* (Hawaii, 2007), pp. 317–320
31. S Young, G Evermann, M Gales, T Hain, D Kershaw, X Liu, G Moore, J Odell, D Ollason, V Valtchev, P Woodland, *The HTK Book* (Cambridge University Engineering Department, Cambridge, 2009)
32. R Duda, P Hart, D Stork, *Pattern Classification*, 2nd edn. (Wiley, New York, 2009)
33. S Theodoridis, K Koutroumbas, *Pattern Recognition*, 3rd edn. (Academic Press, San Diego, 2006)
34. A Sankar, F Beaufays, V Digalakis, Training data clustering for improved speech recognition, in *Proceedings of EUROSPEECH* (Madrid, 1995)
35. Y Li, L Li, A greedy merge learning algorithm for Gaussian Mixture Model, in *Third International Symposium on IITA*, 2009, pp. 506–509. vol. 2, Nanchang, 21–22 November 2009
36. KF Lee, HW Hon, Speaker-independent phone recognition using hidden Markov models. IEEE Trans ASSP **37**(11), 1641–1648 (1989)
37. X Zhang, Y Zhao, L Schopp, A novel method of language modeling for automatic captioning in telemedicine. IEEE Trans ITB **11**(3), 332–337 (2007)
38. A Mohamed, G Dahl, G Hinton, Acoustic modeling using deep belief networks. IEEE Trans ASLP **20**(1), 14–22 (2012)
39. F Seide, G Li, X Chen, D Yu, Feature engineering in context-dependent deep neural networks for conversational speech transcription, in *Proceedings of IEEE Workshop on Automatic Speech Recognition & Understanding* (Hawaii, 2011), pp. 24–29
40. TN Sainath, A Mohamed, B Kingsbury, B Ramabhadran, Deep convolutional neural networks for LVCSR, in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Vancouver, 2013), pp. 8614–8618
41. X Sun, X Chen, Y Zhao, On the effectiveness of statistical modeling based template matching approach for continuous speech recognition, in *Proceedings of INTERSPEECH* (Florence, 2011), pp. 2163–2166

doi:10.1186/1687-4722-2014-4
Cite this article as: Sun and Zhao: **Integrated exemplar-based template matching and statistical modeling for continuous speech recognition.** *EURASIP Journal on Audio, Speech, and Music Processing* 2014 **2014**:4.