Speaker, Vocabulary and Context Independent Word Spotting System for Continuous Speech

Radu Timofte, Ville Hautamäki, and Pasi Fränti

Speech and Image Processing Unit, Department of Computer Science,
University of Joensuu, Finland
{timofte, villeh, franti}@cs.joensuu.fi

Abstract. Word spotting is a widely known subject in continuous speech recognition and the traditional approaches use either hidden Markov models (HMM) or Gaussian mixture models (GMM). In this paper, we propose a different approach based on language independent phoneme modeling. The proposed system is speaker and vocabulary independent, and it is easy to implement. An equal error rate (EER) of 3.34% and a figure of merit (FOM) of 45.58% are achieved on TIMIT corpus.

Keywords: word spotting, continuous speech recognition, phonetic model, clustering, vocabulary independent, pattern matching

1 Introduction

The goal of a word spotting system is to provide the capability of searching through the audio content based on queries. The query can be formed from a text representation, a phonetic transcription or a speech sample. The search will return the positions of possible occurrences and the corresponding confidence scores.

The easiest way to perform word spotting is to obtain transcription of the speech by large vocabulary continuous speech recognizers (LVCSR), and then performing text retrieval on the transcription. In the application domain of broadcast news clips, this approach has been found sufficient [1].

Unfortunately this approach does not generalize to the problem of searching arbitrary keywords from continuous speech. A phoneme-lattice based approach to solve vocabulary independent audio search has been proposed in [2]. The speech recognition lattice is an intermediate format, usually an acyclic graph, in which the hypotheses for phonemes, sub-words or words are scored. They also show comparative results using phoneme lattices and word lattices. In [3], the authors combine both of these methods in a word/phoneme hybrid model.

A comparison between acoustic keyword spotting, spotting in word lattices generated by a LVCSR and a hybrid approach making use of phoneme lattices generated by a phoneme recognizer can be found in [4]. The most widely used model
in the current approaches is the HMM, using GMMs to model the distribution of the features for each phoneme state.

We introduce a new approach for word spotting based on maintaining a codebook for each phoneme of the language. The scores are computed for each possible start time in the input speech waveform (background speech) based on the codebooks. Those scores are used as input to a dynamic programming algorithm to find the most probable occurrence of the keyword by its phonetic representation.

The word spotting system (see Fig.1) is a three-stage procedure: a training stage, a preprocessing stage and a phonetic search. In the training stage, we create an acoustical model of each phoneme, which are the clustered feature vector sequences. Phonemes are extracted independently from the linguistic context. In the preprocessing stage, we assign scores for each possible phoneme occurrence by creating a phoneme-level representation of the background speech. The phonetic search consists of a pattern matching and computing scores to decide accurately the possible presence of the keyword. Due to the modularity of these stages it is possible to use previously stored phonetic information from the preprocessing stage of an audio recording and to run the phonetic matching to obtain results faster without recalculating the scores. In this case, the system would be suitable for audio-indexing and information retrieval as well.

![Fig. 1. Structure of the three-stage word spotting system.](image)

## 2 Word spotting system

The word spotting system consists of three stages: training the acoustical phoneme models, preprocessing the background speech and phonetic matching of the phonetic form of the keyword (see Fig.1). In phoneme training stage (see Fig.3), we extract features from transcribed speech database, collect phoneme samples (acoustic feature
vector sequences), and create acoustic models for the phonemes using clustering techniques. In the preprocessing stage (see Fig.6), we extract features from the background speech and compute a space of the estimated probabilities based on similarities of the phoneme acoustic models with the background speech. The phonetic matching stage performs phonetic pattern searching of the phonetic form of the keyword in the probability space previously computed. Final decisions are performed by thresholding the obtained scores.

2.1 Feature extraction

As features in this system we used both the formants and the mel frequency cepstral coefficients (MFCC). Both features were extracted using Praat scripts [5]. The features are extracted at a frame rate of 100 Hz. For formants, we used Burg algorithm from Praat with a window length of 25 ms, pre-emphasis threshold at 50 Hz, 5500 Hz maximum frequency and 5 formants at maximum. For MFCC, we used the following settings: 12 coefficients, 15 ms analysis window, 10 ms time step, 100 mel position of the first filter with a distance between filters of 100 mel.

2.2 Phoneme similarity

The distance measures used between phonemes that are represented by different length feature vector sequences are proportional centered distance (PCD) and symmetric dynamic time warping (DTW). The proposed PCD distance is a Euclidean type distance function, and DTW is an alignment method. The calculation of the first distance has a linear time complexity. On the other hand, the second distance has a very good accuracy although quadratic time complexity.

![Fig. 2. Pseudocode for the proportional centered distance (PCD) where \( m \) and \( n \) are the lengths of the sequences \( X \) and \( Y \) \((m > n)\).](image)

The SED in Fig.2 stands for standard Euclidean distance. The idea behind PCD is to provide a measure that works for sequences with different lengths. We divide the sequences into two halves, and find how many points from the longer sequence \( Y \)
correspond to one point in the shorter sequence $X$. The distances are computed from the end points to the center, and the remaining of the division between $n$ and $n$ will be the number of points on $Y$ that correspond to the middle point of $X$.

In the symmetric case, dynamic time warping alignment method for two sequences $X$ and $Y$ (of lengths $n$ and $m$) can be recursively computed as in Eqn. (1). The recursion is typically solved using dynamic programming.

$$D_{ij} = SED(X_i, Y_j) + \min \{D_{i-1,j} ; D_{i,j-1} ; D_{i-1,j-1} \} \text{ where } i = 1, n \text{ and } j = 1, m$$

Given a start and an end point of a sequence, the purpose of DTW is to determine in an optimal manner the warping function that provides the best time alignment between the two sequences. The DTW distance score is defined as the Euclidean distance of the optimal alignment. In our case, the score is normalized by dividing it by the sum of the sequence lengths ($n+m$) in order the distance to be independent from the lengths of the sequences.

2.3 Phoneme clustering

After feature extraction we identify the samples for each phoneme ($[aa]$, $[ae]$, $[zh]$) from the training speech material according to their phonetic transcriptions. The samples are sequences of the feature vectors of different lengths. Based on the sets of the samples for each phoneme we perform clustering (see Fig. 3). We implemented two clustering algorithms: several iterations of a random clustering and a k-means algorithm repeated for a number of randomly initial codebooks.

The random clustering algorithm is suitable for both DTW and PCD distances, but the k-means algorithm is used only with PCD distance because for DTW we do not have a good method to compute the cluster centroids.

![Fig. 3. Generation of the phoneme codebooks from the phonetic transcriptions.](image)

In the clustering we want to have representative sequences of the features for each phoneme. The distance between a phoneme and a codebook is computed with the
Eqn.(2) where either PCD or DTW can be used as the distance function between two phonemes \((\text{dist}((p_{h_i}, s_k)))\).

\[
DPCB(p_{h_i}, cb_j) = \min\{\text{dist}((p_{h_i}, s_k)) | s_k \in cb_j\}
\]  

In the k-means algorithm (see Fig.5) with PCD, we project each phoneme from the entire set of samples to the longest length from the entire set. The projection (see Fig.4) is done similarly with the PCD computation (see Fig.2).

**Fig. 4.** Pseudocode for projecting sequence \(X\) (length \(n\)) to a sequence \(Y\) (length \(m, m > n\)).

\[
\begin{align*}
\text{step} &= \left\lfloor \frac{m/2}{n/2} \right\rfloor; \\
D &= 0; \\
&\quad \text{for } i = 1 \text{ to } \left\lfloor n/2 \right\rfloor \text{ do} \\
&\quad \quad \text{for } j = (i-1)*\text{step}+1 \text{ to } i*\text{step} \text{ do} \\
&\quad \quad \quad Y_j = X_i; \\
&\quad \quad Y_{m-j+1} = X_{n-i+1}; \\
&\quad \quad \text{for } j = 1 \text{ to } m-\text{step}*n \text{ do} \\
&\quad \quad \quad Y_{[n/2]*\text{step}+j} = X_{[n/2]*j+1};
\end{align*}
\]

**Fig. 5.** Pseudocode for the k-means clustering algorithm with the restriction of running at most \(T\) iterations. Here \(N\) is the number of phonemes to be clustered, \(M\) is the number of clusters, \(p_i\) is the cluster of the phoneme \(X_i\), \(C\) is the codebook and \(c_j\) are the centroids from the codebook. \(MAE\) stands for mean absolute error.
2.4 Score computation

In the preprocessing stage, we create hypotheses for each phoneme, for each start frame and each possible phoneme length in the background speech.

![Diagram](image)

Fig. 6. Computing distances for each phoneme codebook starting with frame \( j \) and the possible phoneme of length \( k \).

Starting with frame \( j \) we compute distances between each phoneme codebook and the possible phoneme that starts from this particular position in the background speech with a fixed length (see Fig.6). After computing the scores for a fixed length for the possible phoneme, we are interested to abstract those scores. For this purpose, we sort the scores in an ascending order. Phoneme with the minimum score will get the probability estimate 1 and the rest according to their ranks:

\[
\text{probability}(p_{h_i}) = 1 - \frac{\text{rank}(p_{h_i}) - 1}{\text{number Phonemes}}
\]

We replace the distances with the assigned probabilities. Finally, we will have a 3 dimensional space: the 1st dimension is for the codebook/phoneme, the 2nd is the length of the possible phoneme and the 3rd is the starting frame in the input speech.

2.5 Phonetic matching

As input the phonetic matching algorithm take the previous matrices of probabilities constructed on the background speech and the keyword phonetic transcription (base form). If the keyword is in another format we need to convert it to its base form:

\[
W = ph_1 ph_2 ... ph_n
\]

The result of the phonetic matching for a start time (or frame) \( t_s \) is the estimated probability:

\[
P(W, t_s) = \max \left\{ \prod_{i=1}^{n} p(ph_i, t_{s_i}, l_i) \mid t_{s_i} = t_{s_{i-1}} + l_{i-1} \right\}
\]
A phoneme length can vary between fixed values. \( p(ph_i, t_s, l) \) is the probability that the phoneme \( ph_i \) from the keyword phonetic transcription is of length \( l \) at the time/frame position \( t_s \) in the background speech. This probability is extracted from the computed values at the preprocessing stage.

To compute \( P(W, t_s) \) we use recursive definition, which can be solved by exhaustive search using dynamic programming in the space of all solutions (see Fig. 7). The time complexity of the algorithm is \( O(n^2 \beta^2) \).

Fig. 7. Problem definition of the phonetic pattern matching.

### 3 Experimental setup

#### 3.1 Speech database

We evaluate our system using the TIMIT corpus (see table 1), which consists of two parts: train data and test data. The acoustic model was trained on the train data excluding the SA1 and SA2 files.

**Table 1.** Structure of the TIMIT corpus.

<table>
<thead>
<tr>
<th></th>
<th>TRAIN</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of sentences</td>
<td>3696</td>
<td>1344</td>
</tr>
<tr>
<td>No. of uttered words</td>
<td>30132</td>
<td>11025</td>
</tr>
<tr>
<td>No. of distinct words</td>
<td>4891</td>
<td>2373</td>
</tr>
<tr>
<td>No. of male speakers</td>
<td>438</td>
<td>112</td>
</tr>
<tr>
<td>No. of female speakers</td>
<td>192</td>
<td>56</td>
</tr>
<tr>
<td>No. of dialect regions</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Total speech time</td>
<td>~188 minutes</td>
<td>~69 minutes</td>
</tr>
</tbody>
</table>
3.2 Keyword selection

First we retain for test words with at least 4 letters and that are not substrings of other words in the speech material. After that we renounce at those that do not have 1 occurrence exactly in the entire test material. The final set has 862 keywords as summarized in Table 2.

Table 2. Information about keywords selected for testing.

<table>
<thead>
<tr>
<th></th>
<th>Number of keywords</th>
<th>Length average (phonemes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-vocabulary (INV)</td>
<td>238</td>
<td>7.03</td>
</tr>
<tr>
<td>Out-of-vocabulary (OOV)</td>
<td>624</td>
<td>7.96</td>
</tr>
<tr>
<td>All keywords (INV+OOV)</td>
<td>862</td>
<td>7.7</td>
</tr>
</tbody>
</table>

4 Results

For testing our system, we considered a positive match as a case when the computed score corresponds to the starting position of the keyword, or the score is computed for a possible word that ends at the keyword ending position and the length is between 90% and 120% of the keyword. The rest of the matches are considered as negative matches. We use the transcriptions for information about keyword occurrence. After computing the best scores for each start frame we split the entire set of scores into segments of 8 times of the number of phonemes in the keyword. This is chosen based on the average number of frames per phonemes in the training material. All the experiments were computed on a notebook: Centrino 1.4GHz, 512Mb RAM, Windows XP.

4.1 Evaluation criteria

In the evaluation, we use Receiver Operating Characteristics (ROC) curves [6]. The basic ROC graph use true positive rates (tpr) defined as number of positives correctly classified reported at the total number of positives and false positive rates (fpr) defined as number of negatives correctly classified reported at the total number of negatives.

As the first evaluation criterion, we measure the word-spotting accuracy by Figure of Merit (FOM) as proposed in [7] and then defined by NIST (National Institute of Standards and Technology) as the average of the detection/false-alarm curve over the range [0..10] fa/kw/h (false alarms per keyword per hour). For each keyword in the test material, we use linear interpolation for a lower approximation of FOM so that FOM equals to 0 at 0 fa/kw/h if the keyword is not spotted with 0 false alarms. The ROC curve is plotted for detection rate or tpr and number of fa/kw/h. The FOM for one set of keywords will be represented by the average value of all keyword FOMs computed separately.
The second evaluation criterion we used is Equal Error Rate (EER). It is defined as the error rate at the point where the false rejection rate (1 - tpr) and the false acceptance rate (fpr) are equal. The EER for one set of keywords will be represented by the average value of all keyword EERs computed separately.

4.2 Comparison of the features

For the comparison, we fixed the minimum phoneme length at 1 frame and the maximum at 12 frames. The acoustic models are computed using a random clustering algorithm. We set the codebook size to 5. The keyword test set is obtained by random selection of 34 keywords from the test material. We use the first three formants (F1, F2, F3) and the first 12 MFCC coefficients.

Table 3. Comparison of the formants and the MFCC features.

<table>
<thead>
<tr>
<th>Similarity measure</th>
<th>FOM(%)</th>
<th>EER(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Formants</td>
<td>MFCC</td>
</tr>
<tr>
<td>PCD</td>
<td>10.42</td>
<td>42.77</td>
</tr>
<tr>
<td>DTW</td>
<td>12.84</td>
<td>39.72</td>
</tr>
</tbody>
</table>

We observe a big gap between the results obtained using formants and those with MFCC (see Table 3). Using MFCC, the keyword spotting system performs almost three to four times better than using formants (see Fig.8). The results indicate that the formants are not sufficient for accurate keyword spotting but wider representation of the spectrum, such as MFCC features, should be used.

![Fig. 8. Comparative word detection results (ROC curve) between the formants and MFCC, for 34 keywords set with PCD.](image)

4.3 Comparison of design parameters

For the tests, we use the first 12 MFCC coefficients and the set of 862 keywords selected previously. Each codebook contains 5 phoneme samples.
The first experiment aims at finding out a proper maximum phoneme length setting for the phonetic matching algorithm. From Fig. 9 we can see that the FOM reach the maximum (45.58%) for maximum phoneme length of 12, and the best EER (3.02%) is achieved for maximum phoneme length of 13.

![Fig. 9. FOM and EER variations of the matching algorithm as a function of the maximum phoneme length. The entire set of keywords and the DTW distance are used.](image)

From Fig. 10 we can see that our word spotting system performs better with the DTW distance than with the PCD distance in terms of higher FOM. A lower EER is also achieved with the DTW distance (see Table 4). The main observation here is that the better codebooks are obtained the better will be the matching results. The codebooks were computed with the time and iterations constraints set up so that the processing time would not exceed 10 hours.

![Fig. 10. Comparison (ROC curve) between different clustering and similarity measures in terms of FOM. The entire set of keywords and the maximum phoneme length of 12 are used.](image)

The performance of our word spotting system is directly influenced by the length of the keywords (measured as the number of phonemes), (see Fig. 11). The longer the keywords the better is the performance in terms of FOM and EER. This trend is more clearly visible when more data is used (OOV and INV+OOV) whereas the smaller
INV set with only few keywords in the length range [10…13], which explains the high variation in the results in this part. From these graphics we can also conclude a less significant difference between INV and OOV keywords performances, than is shown in the Table 5 where due to the relatively reduce number of keywords and unbalance in keywords lengths over the INV and OOV sets is a big difference in favor of OOV keywords.

Table 4. Search accuracy (FOM) and equal error rates (EER) of the different clustering methods and similarity measures. The entire set of keywords and the maximum phoneme length of 12 are used.

<table>
<thead>
<tr>
<th></th>
<th>PCD K-means clustering</th>
<th>PCD Random clustering</th>
<th>DTW Random clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOM (%)</td>
<td>42.68</td>
<td>38.99</td>
<td>45.58</td>
</tr>
<tr>
<td>EER (%)</td>
<td>3.98</td>
<td>4.57</td>
<td>3.34</td>
</tr>
</tbody>
</table>

Fig. 11. FOM and EER variations of the matching algorithm as a function of keywords length in phonemes for INV, OOV and combined INV+OOV keywords using the DTW distance and the maximum phoneme length of 12.

Table 5. Comparison between word spotting results for in- and out-of-vocabulary keywords separately and jointly. The DTW distance and maximum phoneme length of 12 are used.

<table>
<thead>
<tr>
<th>Keywords</th>
<th>FOM (%)</th>
<th>EER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>INV</td>
<td>38.62</td>
<td>4.17</td>
</tr>
<tr>
<td>OOV</td>
<td>48.23</td>
<td>3.02</td>
</tr>
<tr>
<td>INV+OOV</td>
<td>45.58</td>
<td>3.34</td>
</tr>
</tbody>
</table>
5.3 Overall comparison

From our tests, the best result in terms of FOM (45.58%) is obtained with the DTW distance and a maximum phoneme length of 12 frames (120 ms in our case). The corresponding EER is less than 3.34%.

The performance of the system varies depending on the length in phonemes of the searched words. In general, longer search words tend to provide better results. Moreover, our word spotting system performs similarly both in the case of in-vocabulary and out-of-vocabulary keywords.

6 Conclusions and further work

We have presented a new approach in keyword spotting. Our system introduces a new model for the phonemes, namely clustered feature vector sequences. We have also presented a scoring mechanism and a dynamic programming algorithm for phonetic pattern matching in the space of estimated probabilities. The presented system is speaker, vocabulary and language independent assuming that all phonemes of the language in question have been modeled.

The performance of the system, 45.58% FOM and less than 3.34% EER, is comparable with other results reported in literature and is a good starting point for further improvements. The phonetic pattern matching algorithm is also fast (over 50 times faster than the background speech length without thresholds). The parameters affecting the system are the performance of the clustering algorithm, the maximum phoneme length and the number of phonemes in the keyword phonetic transcription.

References