Unsupervised query by example spoken term detection using features concatenated with Self-Organizing Map distances

Haiwei Wu[1], *Ming Li*[2], *Zexin Cai*[3], *Haibin Zhong*[4]

[2,3]Duke Kunshan University, Kunshan, China
[1]School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China
[3]Jiangsu Jinling Science and Technology Group Limited, China

wuhw3@mail2.sysu.edu.cn, ming.li369@dukekunshan.edu.cn, caizx3@mail2.sysu.edu.cn, zhonghb86@126.com

Abstract

In the task of the unsupervised query by example spoken term detection (QbE-STD), we concatenate the features extracted by a Self-Organizing Map (SOM) and features learned by an unsupervised GMM based model at the feature level to enhance the performance. More specifically, The SOM features are represented by the distances between the current feature vector and the weight vectors of SOM neurons learned in an unsupervised manner. After fetching these features, we apply sub-sequence Dynamic Time Warping (S-DTW) to detect the occurrences of keywords in the test data. We evaluate the performance of these features on the TIMIT English database. After concatenating the SOM features and the GMM based features together, we achieve an improvement of 7.77% and 7.74% on Mean Average Precision (MAP) and P@10 on average.

Index Terms: unsupervised learning, query by example, self-organizing map

1. Introduction

Query by example spoken term detection (QbE-STD) [1, 2] is a task that searching queries from a set of audio archives. Query examples and test utterances are matched based on their acoustic features. Nowadays while the supervised QbE-STD methods have already achieved very promising results, the performances of unsupervised QbE-STD methods still need to be improved. Compared to other spoken term detection tasks, unsupervised QbE-STD focuses on the searching task without the requirement of any knowledge or transcripts about the target language which is very suitable for applications on low resource languages. In recent years, this area attracts more and more attention. There are various types of acoustic features being used in this task, for example, spectral features [3], frame clustering features [4], GMM posteriorgrams [4], features extracted by a deep neural network (DNN) [5, 6, 7] and so on. Basically, these features try to capture information on the phonetic-like units [8] so that keyword examples and test utterances can be represented in a more informative way. The method to extract features has a great impact on the final searching results. So, in order to further improve the performance of unsupervised QbE-STD, in this paper, we strengthen the commonly used GMM based features by concatenating the features extracted by Self-Organizing Map (SOM) [9] at the feature level.

The Self-Organizing Map [9, 10] (SOM) is an unsupervised data-analysis method introduced by T. Kohonen. The idea was first inspired by neuron-biological learning paradigms. And now this method is widely used in many areas [11, 12, 13] such as data clustering and visual representation. In many data exploration tasks related to linguistics and natural science, SOM is widely employed and in specific areas such as bioinformatics and massive textual database, it achieves very promising results. In addition, Self-Organizing Map could be easily trained using a small amount of data.

Explorations have been done by many researchers on Self-Organizing Map in speech areas such as speech recognition [14], speaker verification [15], phoneme recognition [16] and gender classification [17]. In these research, SOM is thought to be able to represent some attributes of speech signals.

SOM is relevant to the basic idea of classic vector quantization (VQ) [18], which represents data using a set of models. The SOM is a two-layer network constructed by an input layer and an output layer. The output layer is a grid having many output nodes. Each node of the output layer has a weight vector and can be viewed as a cluster. In the process of training, each input feature vector is automatically classified to a specific output node, called the best match unit. All the weight vectors are updated depending on their distances between the best match unit. The topology structure is flexible and well expresses the differences and relationships between nodes. Those nodes of the output layer in SOM could be considered as the phoneme-like units learned in an unsupervised manner [16]. It is in this context that we explore the feasibility of using SOM features for unsupervised QbE-STD.

GMM based features are commonly used to represent spoken units. In this paper, we totally extract three types of GMM based features.

The first type of feature is extracted by an unsupervised diagonal GMM model which is efficient and robust on small-scale data.

As is mentioned in [19, 20], Dirichlet Process Gaussian Mixture Model (DPGMM) is thought to be a better way to represent phonetic units, so we use DPGMM to extract our second basic features.

Besides the features extracted by diagonal GMM and DPGMM, we also investigate features extracted by an unsupervised DNN model. We first cluster all our MFCC feature vectors in training set by DPGMM and then use both the cluster labels and features to train a neural network. For the reason that we do not use any transcripts of data, it is still unsupervised. Some research [6] recently revealed that this method is helpful for the final performance.

In this work, we try to concatenate SOM features and each of the three types of features mentioned above. After combination, we use sub-sequence DTW(S-DTW) [21] which is widely used in pattern matching to align our example sequences and sequences in the test collection. Experimental results show that the concatenation of SOM features and GMM based features could effectively improve the results.

The rest of the paper is organized as follows. Section 2 describes the proposed methods in detail. Experimental results are presented in Section 3 while conclusion are provided in Section 4.

2. Methods

2.1. SOM feature extraction

In this section, we will introduce the extraction of Self-Organizing Map [9] (SOM) features. Firstly, we should specify a group of units which are used to represent the phonetic-like units. In our case, we set the shape of the output layer as a square ($N \times N$ units). One of the advantages of SOM is that its topology structure has the ability [10] to well express the relevance or differences between clusters or phonetic-like units. Then on this set of units, we need a neighborhood function [11] to decide the scale of weight updating. Usually, the neighborhood function should be symmetrical and for every neuron, the value to update depends on its distance between the winner neuron. In every training iteration, one of the output layer neurons will be selected as the winner neuron [9]. When a neuron wins in a step, its output of neighborhood function will be set to 1, and the result should decrease as the distance between the current neuron and the winner neuron increases. A widely used neighborhood function here is the step function (1).

$$h_{kc^t(x)}(t) = exp(-\frac{dist^2(k, c^t(x))}{2\sigma^2(t)}) \qquad (1)$$

$c^t(x)$ is the winner neuron, x is the weight of winner neuron and k is each of the other neurons in the output layer. From the equation above, we can see the function meets the requirement, and $\sigma(t)$ will decrease over time to reduce the intensity [11]. Each neuron of the output layer is represented by a model which is also called weight vector [11]. The weight vectors have the same dimension as the feature vectors. The aim of SOM algorithm is to update these weight vectors in a way that they can both represent the input signals and preserve the information between each neuron with its topological structure. The SOM training algorithm [9] is introduced as follows.

1. Extract the MFCC vectors of each training utterances.
2. Find out the best match unit by the distances of weight vectors and input vectors. K is the set of output neurons. $m_k(t)$ is a weight vector. $c^t(x)$ is the best match unit.

$$c^t(x) = \underset{K}{argmin}\, ||x - m_k(t)||^2 \qquad (2)$$

3. All the weight vectors are updated by the following equation. $\epsilon(t)$ is learning rate.

$$m_k(t+1) = m_k(t) + \epsilon(t) h_{kc^t(x)}(t)(x - m_k(t)) \quad (3)$$

We can see that, during the training process, we select the unit that is closest to the current feature vector as the best match unit [10]. And then we update every weight vector as equation (3). After SOM model is trained, we get a list of weight vectors. We can compute the distances between an input feature vector and weight vectors, and group them as our new features. Figure 3 shows the network structure of SOM.

The algorithm above is easy to implement while the mathematical proof is much more complex and is provided in [11].

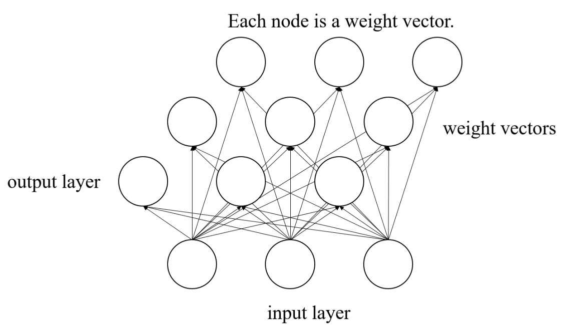

Figure 1: *Self-Organizing Map network.*

2.2. GMM based feature extraction

In this section, some unsupervised feature extraction methods are introduced. These methods are all based on GMM. Instead of using classic GMM model to extract features, we use diagonal GMM, DPGMM and unsupervised DNN labeled by DPGMM. Diagonal GMM is a faster version of GMM which needs less memory and less training data, it can be trained more efficiently than GMM.

And for DPGMM, it is a more suitable and feasible choice to represent speech data, especially for the scenarios of low resource language according to [19, 20].

In recent years, using trained GMM model to label data features for the subsequent DNN modeling becomes a popular method [8, 9, 10]. We first train a DPGMM to cluster our training set. For each vector, the component with the largest DPGMM probability would be chosen as the label.

For diagonal GMM and DPGMM, we use the posteriorgrams as features. While for DNN, we fetch the output of final softmax layer as features.

2.3. Acoustic features matching

Acoustic pattern matching can be computed in many variants of dynamic time warping (DTW) algorithm. DTW is used to align two sequences and evaluate the similarity by the cost. In this paper, we employ sub-sequence DTW (S-DTW) [21] algorithm for our unsupervised QbE task. S-DTW is first introduced as a modification of the classic DTW algorithm to find the matching sub-sequences of utterances in test collection between query examples. This method is widely used in sequences retrieval tasks. We denote a query example sequence as $X := (x1; x2 ::: xN)$ and a test sequence as $Y := (y1; y2 ::: yM)$. S-DTW is an algorithm to find the begin and end of the sequence in Y that optimally matches the keyword example sequence X as equation (4) shows.

$$(b^*, e^*) := \underset{(b,e):1\leq b\leq e\leq M}{argmin} (DTW(X, Y(b : e))) \qquad (4)$$

b is the begin of the sequence and e is the end. S-DTW algorithm is implemented mainly in two steps [21]. First, the accumulated cost matrix is modified so that it allows the process to consider all points as beginning in order to find the optimal one. It is more flexible compared to the classic DTW whose starting point is fixed. The second step also differs from the classic DTW implementation. It allows the optimal matching sub-sequence to finish in any ending time of the sequence following the equation (5).

$$e^* := \underset{e\in[1:M]}{argmin}\, D(N, e) \tag{5}$$

D returns cost value given a coordinate in cost matrix. Like the classic DTW, after finishing finding the best matching point, a traceback dynamic programming process is computed to find the best path. It is also very important to decide the distance function between vectors when calculating the cost of DTW algorithm. Many distance can be used here, in our experiment, we choose Euclidean distance (6) by default.

$$d = ||x - y|| = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2} \tag{6}$$

3. Experiment

3.1. Dataset

We investigate the performance of our system on the TIMIT English speech corpus. The dataset consists of a training set of 4620 utterances and a whole test set of 1680 utterances. We extract MFCC features of the training set to train the GMM based models and Self-Organizing Map. As for keywords, we choose 50 keywords and our target is to find out the occurrences of these keywords in the test set of 1680 utterances. There are totally more than 300 queries to find. Each keyword we choose consists of 3 to 5 syllables. For each query, a correct hit is accumulated if an utterance in the retrieval contains the query so that we can evaluate the performance of our system.

All our MFCC features are extracted by the python_feature_speech toolkit [22]. We set window size as 25ms and shift length as 10ms. Two metrics are used to evaluate our performance. The first one is mean average precision (MAP), which shows the average precision for correct hits in retrieval. The second metric is P@10, which counts the correct hits of top 10 and calculates the precision.

3.2. Experimental Setup

3.2.1. Overview of the system

Our system is mainly constructed by two parts, including feature extraction and feature matching. Our features are extracted from GMM based models and SOM. After we obtain these features, we use S-DTW to find the keywords in test utterances. We compare the performances of basic diagonal GMM features, DPGMM features, unsupervised DNN features to our concatenated features. The whole system is illustrated in Figure 2.

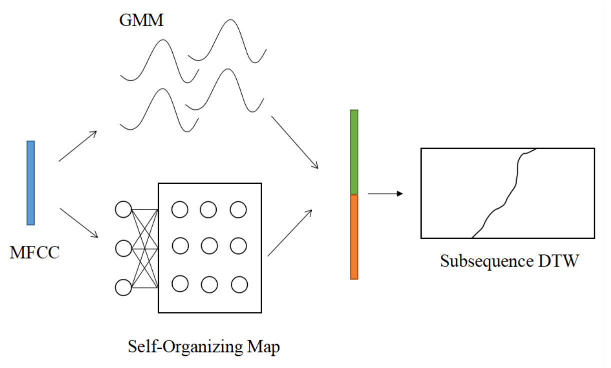

Figure 2: *Structure of our system.*

3.2.2. GMM based feature extraction

We train two types of GMM model, the DPGMM and the diagonal GMM model using MFCC features. For MFCC, we use MFCC results in 39-dimensional feature vectors. Number of components for the diagonal GMM and number of max components for the DPGMM are set to 144. The number of max iteration is set to 100, it can stop at any time when it converges.

We use these two models to extract posteriorgrams for query examples and the test data. Besides, we use trained DPGMM model to label the training set and then use the labels and features to train a DNN model. The structure of our DNN is 1024×4–40-1024-144. Epoch is set to 20. We take the output of the final softmax layer as features. Keras [23] is used to train the unsupervised DNN model.

Diagonal GMM features, DPGMM features and unsupervised DNN features are our basic GMM based features, they are also used to build our baselines.

3.2.3. SOM feature extraction

Later, we trained SOM for extracting some new features to improve our basic features. SOM has several options to set. After some experiments, we find a relatively proper value to set.

1. Fixed 0.5 for learning rate.
2. 20000 iterations, which is sufficient for our system.
3. 20^2 to 23^2 output neurons.
4. Fixed 0.3 for σ.

We extract SOM features by calculating the distances between the input vector and weight vectors of the model. The performance is influenced by the shape of the output layer. We apply a range of units from 20^2 to 23^2 to find out the optimal choice. σ is set to 0.3 by default. For the reason that it has little impact on the our results, we do not further discuss it. After training, we are able to extract the SOM features. Figure 3 illustrates the distance matrix for a single input vector. Each neuron has its own color. Lighter color represents smaller distance, darker represents larger. We can find out that neurons of light color are close to each other, which means phonemes-like units represented by these neurons are relatively similar to the senone structure of decision tree. The SOM learns both the distribution and physical topology with the training algorithm [15]. The numerical value of features extracted by SOM may be relatively larger than posteriorgrams. For this reason, we apply normalization on both features, which is also able to avoid some extreme values of some dimension. And then, we combine both features mentioned above in a proper proposition. After that, we apply S-DTW algorithm to find examples in the test utterances.

Table 1: *Experimental results on TIMIT database with diagonal GMM features and SOM features*

Features	P@10	MAP
D-GMM	0.222	0.3216
D-GMM+SOM20	0.236	0.3367
D-GMM+SOM21	**0.240**	**0.3607**
D-GMM+SOM22	0.228	0.3210
D-GMM+SOM23	0.232	0.3206
D-GMM+SOM21×23	0.231	0.3201

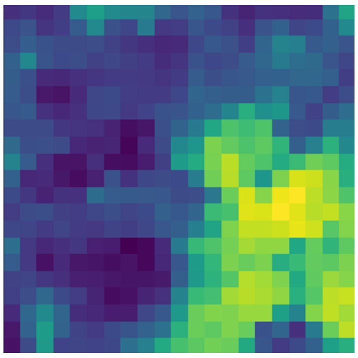

Figure 3: *Distance matrix of an input vector.*

Table 2: *Experimental results on TIMIT database with DPGMM features and SOM features*

Features	**P@10**	**MAP**
DPGMM	0.282	0.4398
DPGMM+SOM20	0.304	0.4512
DPGMM+SOM21	**0.308**	**0.4655**
DPGMM+SOM22	0.298	0.4626
DPGMM+SOM23	0.296	0.4511
DPGMM+SOM21×23	0.296	0.4495

Table 3: *Experimental results on TIMIT database with unsupervised DNN features and SOM features*

Features	**P@10**	**MAP**
DNN	0.306	0.4511
DNN+SOM20	**0.324**	0.4711
DNN+SOM21	0.318	**0.4751**
DNN+SOM22	0.322	0.4691
DNN+SOM23	0.318	0.4652
DNN+SOM21×23	0.310	0.460

3.3. Results

From the experimental results, we can observe that by concatenating new SOM features to current basic features, both P@10 and MAP are improved. We successfully improve 12.16%, 5.84%, 5.32% for diagonal GMM, DPGMM, unsupervised DNN features on MAP and 8.11%, 9.22%, 5.89% on P@10. SOM features are able to represent the speech signal and strengthen the widely used features for QbE-STD task. We have three types of baseline feature of our system. The performance of diagonal GMM features is shown in Table 1, GMM features in Table 2, and unsupervised DNN in Table 3. At the very beginning, we tried out some rectangle-shape structures for Self-Organizing Map. What we found is that the square-shaped structures are able to achieve better results. We can have the result intuitively that square-shaped structures could better represent the phonetic-like units than other shapes. The symmetrical topology of the square-shaped structures may be the reason.

Next, we further investigated the size of SOM to find out the best configuration. Size of near 20×20 or 21×21 gets a relatively better result than other sizes.

It is worth mentioning that training the Self-Organizing Map takes little time. For SOM with the output shape of 20×20 units, it takes only several minutes for 20000 iterations on a standard desktop PC which is more efficient than other feature extraction methods.

4. Conclusion

Unsupervised QbE-STD is a task to search keywords in audio utterances without any information about language and transcripts. So, it is of great significance to find out proper features to represent the speech signal. GMM based features are widely used in this task. Based on these features, we consider concatenating some new unsupervised features extracted by Self-Organizing Map to further enhance the performance. In the results, without any help of other data or heavy computation, results on both MAP and P@10 are improved. We can draw a conclusion that SOM is able to extract effective and complementary features for the unsupervised QbE-STD task.

5. Acknowledgement

This research was funded in part by the National Natural Science Foundation of China (61773413), the Natural Science Foundation of Guangzhou City (201707010363), the Science and Technology Development Foundation of Guangdong Province (2017B090901045), the National Key Research and Development Program (2016YFC0103905).

6. References

[1] A. Mandal, K. P. Kumar, and P. Mitra, “Recent developments in spoken term detection: a survey,” *International Journal of Speech Technology*, vol. 17, no. 2, pp. 183–198, 2014.

[2] A. S. Park and J. R. Glass, “Unsupervised pattern discovery in speech,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 1, pp. 186–197, 2008.

[3] P. Yang, C.-C. Leung, L. Xie, B. Ma, and H. Li, “Intrinsic spectral analysis based on temporal context features for query-by-example spoken term detection,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[4] Y. Zhang and J. R. Glass, “Unsupervised spoken keyword spotting via segmental dtw on gaussian posteriorgrams,” in *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on.* IEEE, 2009, pp. 398–403.

[5] H. Kamper, M. Elsner, A. Jansen, and S. Goldwater, “Unsupervised neural network based feature extraction using weak top-down constraints,” in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on.* IEEE, 2015, pp. 5818–5822.

[6] Y. Zhang, R. Salakhutdinov, H.-A. Chang, and J. Glass, “Resource configurable spoken query detection using deep boltzmann machines,” in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on.* IEEE, 2012, pp. 5161–5164.

[7] L. Badino, C. Canevari, L. Fadiga, and G. Metta, “An auto-encoder based approach to unsupervised learning of subword units,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on.* IEEE, 2014, pp. 7634–7638.

[8] T. J. Hazen, W. Shen, and C. White, “Query-by-example spoken term detection using phonetic posteriorgram templates,” in *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on.* IEEE, 2009, pp. 421–426.

[9] T. Kohonen, “The self-organizing map,” *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[10] V. Chaudhary, R. Bhatia, and A. K. Ahlawat, “A novel self-organizing map (som) learning algorithm with nearest and farthest neurons,” *Alexandria Engineering Journal*, vol. 53, no. 4, pp. 827–831, 2014.

[11] M. Cottrell, M. Olteanu, F. Rossi, and N. Villa-Vialaneix, "Theoretical and applied aspects of the self-organizing maps," in *Advances in Self-Organizing Maps and Learning Vector Quantization*. Springer, 2016, pp. 3–26.

[12] M. Wickramasinghe, K. Gunawardana, J. Rajapakse, and D. Alahakoon, "Investigating individual game-play patterns using a self-organzing map," in *Information and Automation for Sustainability (ICIAfS), 2012 IEEE 6th International Conference on*. IEEE, 2012, pp. 203–208.

[13] M. Ohta, Y. Kurosaki, H. Ito, and H. Hikawa, "Effect of grouping in vector recognition system based on som," in *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on*. IEEE, 2016, pp. 1–8.

[14] R. Venkateswarlu and R. V. Kumari, "Novel approach for speech recognition by using selforganized maps," in *Emerging Trends in Networks and Computer Communications (ETNCC), 2011 International Conference on*. IEEE, 2011, pp. 215–222.

[15] P. Das and U. Bhatacharjee, "Robust speaker verification using self organizing map," in *Computing, Communication and Networking Technologies (ICCCNT), 2014 International Conference on*. IEEE, 2014, pp. 1–4.

[16] X. Dong, *Phoneme-based speech recognition using self-organizing map.*, 2004.

[17] P. Partila, J. Tovarek, and M. Voznak, "Self-organizing map classifier for stressed speech recognition," in *Machine Intelligence and Bio-inspired Computation: Theory and Applications X*, vol. 9850. International Society for Optics and Photonics, 2016, p. 98500A.

[18] R. Gray, "Vector quantization," *IEEE Assp Magazine*, vol. 1, no. 2, pp. 4–29, 1984.

[19] H. Chen, C.-C. Leung, L. Xie, B. Ma, and H. Li, "Parallel inference of dirichlet process gaussian mixture models for unsupervised acoustic modeling: A feasibility study," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[20] ——, "Unsupervised bottleneck features for low-resource query-by-example spoken term detection." in *INTERSPEECH*, 2016, pp. 923–927.

[21] X. Anguera and M. Ferrarons, "Memory efficient subsequence dtw for query-by-example spoken term detection," in *Multimedia and Expo (ICME), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1–6.

[22] https://github.com/jameslyons/python_speech_features.

[23] F. Chollet *et al.*, "Keras," https://keras.io, 2015.