Improved Keyword Spotting based on Keyword/Garbage Models

Qiyu Chen , Weibin Zhang , Xiangmin Xu , Xiaofen Xing
South China University of Technology, GuangZhou, China
chenqiyuscut@qq.com, eeweibin@scut.edu.cn, xmxu@scut.edu.cn, xfxing@scut.edu.cn

***Abstract*—We propose two simple methods to improve the performance of a keyword spotting system. In our application, the users are allowed to change the keywords anytime if they want. Thus we focused on phone-based GMM-HMM models since they do not require keyword-specific training data. However, the GMM-HMM based models usually have very high false alarm rate, i.e., a keyword is not present but the system gives a positive decision. We found that we can utilize the uncertainty of the system when a non-keyword is presented. Two simple methods are proposed to incorporate the uncertainty into the confidence measure. Our experiments show that these two methods can substantially reduce the false alarm rate from 75.05% to 5.71%. Meanwhile, the false reject rate increases from 1.04% to 5.71%.**

Keyword: keyword spotting, confidence assessment, GMM, HMM

I. Introduction

Keyword spotting (KWS) is a key component to enable fully hands-free speech understanding experience. It continuously listens to the speech to detect pre-defined keywords to initiate voice input. Since it is designed to be running all the time, the CPU and memory requirements by KWS should be very small.

Some researchers [1], [2], [3] proposed to use a Large Vocabulary Continuous Speech Recognition (LVCSR) system to firstly decode the input speech to generate rich lattices. The output lattices are then searched to detect the presence of pre-defined keywords. Since LVCSR requires a lot of memory and computation, this method is usually used for offline audio indexing and searching.

In recent years, Deep Neural Networks (DNNs) have achieved tremendous success in many machine learning tasks, including speech recognition. Several KWS systems based on DNNs have been proposed. Chen et al. [4] proposed to use DNNs for small-footprint keyword spotting. More advanced models such as Convolutional Neural Networks (CNNs[5]) or Recurrent Neural Networks (RNNs[6], [7]) have recently been proposed to improve the performance. A neural network is trained using keyword specific data to directly predict the keyword(s). Neural Networks (NNs) based methods have greatly improved the performance of KWS system. However, NN-based methods usually need a large amount of keyword specific training data which is very expensive. For example, about 40K training examples of "okay google" were used to train the models in [4]. In addition, new training data need to be collected to re-train the models if the pre-defined keywords are changed.

Another commonly used technique for KWS is the Keyword/Garbage model [8], [9]. A Hidden Markov Model (HMM) is constructed for each keyword. In addition, garbage (or filler) HMMs are constructed to represent all non-keywords. At runtime, a Viterbi decoding is used to find the best path that explains the input speech. A KWS system based on Keyword/Garbage model usually has very low false reject rates. However, the false alarm rates are usually very high, since the garbage model can hardly represents all non-keywords. Zhang et al. [8] proposed to use confusion garbage models to partially solve this problem. Similar pronunciations with pre-defined keywords are carefully selected to join the garbage model to reduce false alarm rate. In this paper, we propose two simple methods to deal with this problem based on phenomenons that we observed in our experiments. If non-keyword speech is fed into the system based on Keyword/Garbage model, usually the decoder is uncertain during decoding, though it might finally reluctantly converge to a certain path that represents a keyword.

The rest of this paper is organized as follows. In Section 2, we describe the methodology. The experimental setup and results are presented in Section 3. Conclusions are presented in Section 4.

II. Methodology

As explained above, we focus on keyword/garbage models in this paper. During decoding, we used the token passing algorithm [10]. Therefore, we will firstly introduce the Keyword/Garbage models, followed by the token passing algorithm. Then we will go on to describe our methodologies.

A. Keyword/Garbage Models

The main problem with keyword spotting is to substantially reduce the false alarm rate (a keyword is not present but the system gives a positive respond) while maintaining very high recognition accuracy at the same time. One way to deal with this problem is to use Keyword/Garbage models [9]. The idea is to absorb all out-of-vocabulary words into the garbage (also called filler) models.

As shown in Fig. 1, the keyword/garbage models consist of multiple path representing the keywords, and a single (as in our experiment) or multiple path (as in [8]) representing the

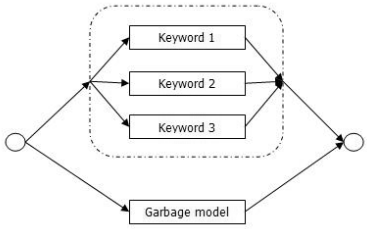

Fig. 1. The keyword/garbage model with a single garbage path.

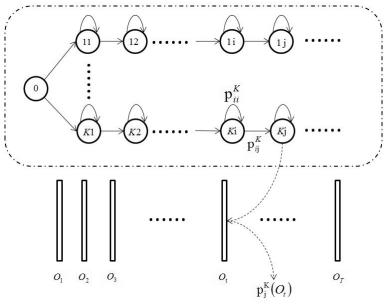

Fig. 2. The structure of the constructed HMM of the keyword/garbage model.

garbage model(s). Usually, the garbage models were simply trained by using all non-keyword speech. A simple decoder will simply find the best path that can best explain the input speech. However, the resulting false alarm rates are still very high since the garbage models cannot represent all non-keywords.

B. Token passing algorithm

The constructed keyword/garbage model is indeed a finite state network as shown in Fig. 2. The decoding algorithm we used to find the best path is the token passing algorithm [10]. Let K be the number of paths and N_k be the number of nodes in the k^{th} path. As can be seen from Fig. 2, there is a *transition cost* a_{ij}^k (i.e. the negative of the $logarithm$ of the transition probability p_{ij}^k of the hidden Markov models in the k^{th} path) associated with each pair of connected states i and j in the k^{th} path. There is also a local cost $b_j^k(o_t)$ associated with each state (i.e. the negative of the $logarithm$ of the state output probability $p_j^k(o_t)$ in the k^{th} path).

Each state of the finite state network is able of holding a moveable *token*. At time t, the token in state kj holds the value $s_j^k(o_t)$ that is the minimum cost alignment between the segment $o_1 \cdots o_t$ of the input speech features and the model starting in state 0 and ending in state kj, i.e.

$$s_j^k(o_t) = \min_i\{s_i^k(o_{t-1}) + a_{ij}\} + b_j^k(o_t) \tag{1}$$

For each input speech frame, the tokens will move one step forward to all connecting states. After all the speech has been fed into the decoder, the decoding algorithm examines the final state, finds the token with the smallest $s_j^k(o_t)$ value, and finally outputs k as the spotted results. k can represent a valid keyword path or a garbage path. For the details of the *token passing* algorithm, we refer the readers to [10]. We will call the output path k as the *decoded path* in the following section.

C. Degree of match

Our preliminary experiments with the above methods show that the false reject rate(i.e. a keyword is present but the system give negative respond) is very low, but the false alarm rate is too high to be used in a real system. An option is to add more garbage models into the system. But this will substantially increase the model complexity and thus requires much more computation and memory. By utilizing the uncertainty of the decoder when non-keyword speech is fed into the decoder, we can substantially reduce the false alarm rate, but (almost) without increasing the computation and memory requirement.

Our first method, which we call it "*degree of match* (DM)", is based on the following phenomenon. We fount that if the input speech is indeed a keyword, for each input speech frame o_t, the best node (i.e. the node with the smallest $b_j^k(o_t)$) usually comes from the *decoded path*. However, if the input speech is a non-keyword, the best nodes for different input frames usually comes from different paths. Lets denote $b^{dec}(o_t)$ as the best node on the *decoded path* that explains o_t (i.e. the node on the decoded path with smallest $b_j^k(o_t), j = 1, 2, \cdots, N_k$), and $b^g(o_t)$ as the global optimal node for o_t (i.e. the node with smallest $b_j^k(o_t), k = 1, 2, \cdots, K, j = 1, 2, \cdots, N_k$). Fig. 3 shows the histogram of the difference between $b^{dec}(o_t)$ and $b^g(o_t)$ if a keyword is present. Fig. 4 shows the histogram of the difference between $b^{dec}(o_t)$ and $b^g(o_t)$ if a non-keyword is present.

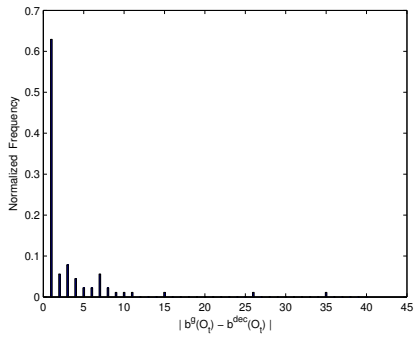

Fig. 3. The histogram of the difference between $b^{dec}(o_t)$ and $b^g(o_t)$ if a keyword is present.

The proposed DM tries to utilize the above information in the decoding in order to reduce the false alarm rate. To illustrate the strategy of DM, lets firstly define the following variables:

$$\begin{aligned} b^k(o_t) &= \min_j\{b_j^k(o_t)\}, j = 1, 2, \cdots, N_k \\ b^g(o_t) &= \min_k\{b^k(o_t)\}, k = 1, 2, \cdots, K \\ m_t^k &= \begin{cases} m_{t-1}^k + 1, & b^k(O_t) > b^g(o_t) + d \\ m_{t-1}^k, & else \end{cases} \\ m_0^k &= 0 \end{aligned} \tag{2}$$

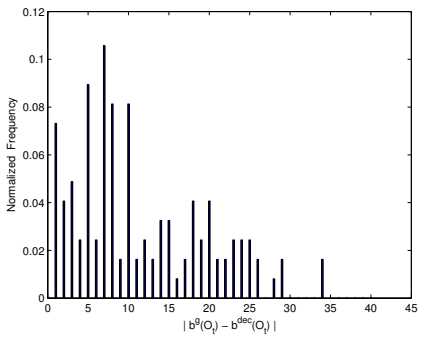

Fig. 4. The histogram of the difference between $b^{dec}(o_t)$ and $b^{g}(o_t)$ if a non-keyword is present.

In other words, m_t^k denotes the number of times that the best node (with a tolerance defined by d) does not comes from the k^{th} path. Every time the *token passing* algorithm finishes the decoding and finds the best path k^*, the DM strategy will further check $m_T^{k^*}$ to determine the final result using the following mechanism

$$\begin{cases} output \quad k^*, & m_T^{k^*}/T < thrd_{dm} \\ non-keyword, & else \end{cases} \tag{3}$$

where $thrd_{dm}$ is another hyper parameter. Note that the DM strategy can be used together with the keyword/garbage model. Thus if the output path k^* represents a garbage model, the output is still a non-keyword.

D. Degree of stability

Our second strategy, which is called "*degree of stability* (DS)", tries to explore the stability of the global optimal path in order to reduce the false alarm rate. Lets define $s^k(o_t) = \min_j\{s_j^k(o_t)\}, j = 1, 2, \cdots, N_k$. We found that if a keyword is indeed present, the global optimal path (i.e. the path with the token that has the smallest $s_j^k(o_t)$) quickly converges to the correct one, as shown in fig. 5. If a non-keyword is present, the best token (i.e. the token with the smallest $s_j^k(o_t)$ globally) usually comes from different paths at different time, as shown in fig. 6.

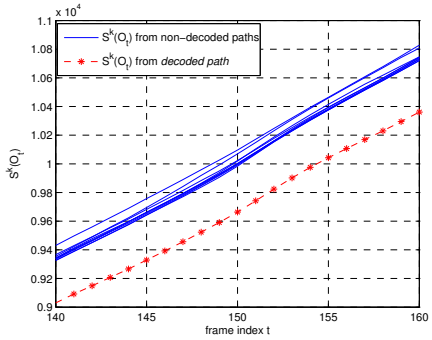

Fig. 5. $s^k(o_t)$ of different paths increase with t increases. The global optimal path quickly converges to the correct one if a keyword is indeed present.

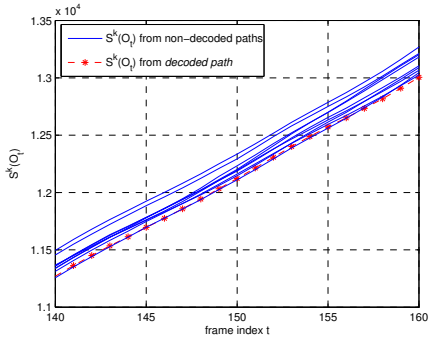

Fig. 6. $s^k(o_t)$ of different paths increase with t increases. The global optimal path hardly converges if a non-keyword is present.

The proposed DS method utilize the above information to reduce the false alarm rate. Let

$$\begin{aligned} &s^k(o_t) = \min_j\{s_j^k(o_t)\}, j = 1, 2, \cdots, N_k \\ &s^{-k}(o_t) = \min_m\{s^m(o_t)\}, m = 1, 2, \cdots, K \ and \ j \neq k \\ &y_t^k = y_{t-1}^k + |s^k(o_t) - s^{-k}(o_t)| * \omega \\ &y_0^k = 0 \end{aligned} \tag{4}$$

where ω is a hyper parameter. As with the DM strategy, every time the *token passing* algorithm finishes the decoding and finds the best path k^*, the DS strategy will further check $y_T^{k^*}$ to determine the final result using the following mechanism

$$\begin{cases} output \quad k^*, & y_T^{k^*}/T < thrd_{ds} \\ non-keyword, & else \end{cases} \tag{5}$$

where $thrd_{ds}$ is another hyper parameter.

III. Experiments and Results

A. Data

We used a corpus of about 1600 hours of transcribed Mandarin speech as the training data. The KWS system was then evaluated on another separate data set, which was recorded from 20 speakers (10 male speakers and 10 female speakers). The evaluating data set contains 2000 utterances, where 1000 of them consist of 10 keywords and the other 1000 utterances are non-keywords. We further divided the evaluating data set into a development set (10%) and a testing set (90%). The number of utterances in both the development and testing data sets for different classes of keywords/non-keywords is balanced.

B. Experimental Setup

Mel Frequency Cepstrum Coefficient(MFCC)[11] was used as the features. Triphone models based on decision-tree tying were trained by using all the training data. The same data (i.e. all the training data) were then used again to train the garbage model, a hidden Markov model (HMM) with 3 hidden states. To restrict the computation and memory usage, we did not use a lot of Gaussian component for each HMM state. Specifically,

we used 24 Gaussian components for each state of all the HMMs. We then used the trained models to construct a finite state network similar to the one shown in Fig. 2.

To evaluate the effectiveness of the proposed methods, we compared the following KWS systems in our experiments. 1) A KWS system with only the garbage models (GM). 2) A KWS system with the garbage models and the proposed DM method (GM+DM). 3) A KWS system with the garbage models and the proposed DS method (GM +DS). 4) A KWS system with only the proposed DM and DS methods (DM+DS). 5) A KWS system with the garbage models, the proposed DM and DS methods (GM+DM+DS).

We used the Equal Error Rate (EER) as the evaluation criteria.

C. Experimental Results

The equal error rates of different models are shown in Table 1. When we only used the garbage models (GM), the false reject rate (FRR) of the system is only 1.04%, but the false alarm rate (FAR) is as high as 75.05%. Therefore, KWS system based only on the garbage models can hardly be used in real application. Since the KWS system with the garbage model does not have any hyper parameters to tune to get the EER. We did not include its EER in Table 1.

TABLE I
THE EER OF DIFFERENT SYSTEMS EVALUATED IN THIS PAPER.

Methods	EER
GM+DM	16.94%
GM+DS	12.82%
DS+DM	7.81%
GM+DS+DM	5.71%

From Table 1, we can also see that both the DM and DS methods can help substantially reduce the EER. Even a system with only the DM and DS methods (i.e. without garbage model) can achieve the EER of 7.81%. Combing the GM, DS and DM achieve the best result.

The FFR-FAR curve of the KWS system with GM+DM+DS is plotted in Fig. 7. We can tune the parameters to meet the requirements of different applications.

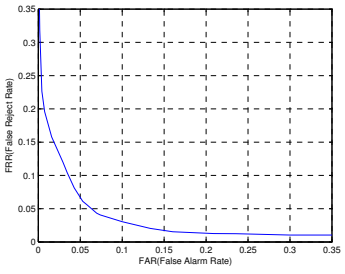

Fig. 7. The FFR-FAR curve of the GM+DS+DM system.

We haven't collect enough data to train dedicated DNNs to compare with the methods proposed in this paper. From their FFR-FAR curve in [4], we estimate that the EER is about 4.5% which is slightly better than our methods. However, our methods do not require a large number of keyword specific data to train the DNNs (e.g. about 40k number of utterances of "okay google" were used to train the DNNs). The users can easily change the keywords according to their own habit.

IV. CONCLUSIONS

In this paper, we present two new methods to substantially reduce the false alarm rate of a keyword spotting system based on the keyword/garbage models. The proposed methods are based on the observed uncertainty during decoding when non-keywords are present to the system. The experiments demonstrate the effectiveness of the proposed methods.

V. ACKNOWLEDGEMENT

This work is supported in part by the Science and Technology Planning Project of Guangdong Province, China (2014B010111003 and 2014B010111006), the Guangzhou Key Lab of Body Data Science (201605030011), the Fundamental Research Funds for the Central Universities (2015ZJ009) and the National Natural Science Foundation of China (61601187).

REFERENCES

[1] Weintraub, Mitchel, "LVCSR log-likelihood ratio scoring for keyword spotting." in *Acoustics, Speech, and Signal Processing. ICASSP-95, International Conference on,* 1995:297–300.

[2] Motlicek, Petr and Valente, Fabio and Szoke, Igor, "Improving acoustic based keyword spotting using LVCSR lattices." in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on,* 2012:4413–4416.

[3] Chen, I-Fan and Ni, Chongjia and Lim, Boon Pang and Chen, Nancy F and Lee, Chin-Hui, "A novel keyword+ LVCSR-filler based grammar network representation for spoken keyword search." in *ISCSLP, 9th International Symposium on,* 2014:192–196.

[4] Chen, Guoguo and Parada, Carlos and Heigold, Georg, "Small-footprint keyword spotting using deep neural networks." in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on,* 2014:4087–4091.

[5] Sainath, T and Parada, Carolina, "Convolutional neural networks for small-footprint keyword spotting." in *Proc. Interspeech,* 2015.

[6] Li, KP and Naylor, JA and Rossen, ML, "A whole word recurrent neural network for keyword spotting." in *Acoustics, Speech, and Signal Processing, ICASSP-92,IEEE International Conference on,* 1992:81–84.

[7] Fernández, Santiago and Graves, Alex and Schmidhuber, Jürgen, "An application of recurrent neural networks to discriminative keyword spotting." in *Artificial Neural Networks–ICANN,* 2007:220–229.

[8] Zhang, Shilei and Shuang, Zhiwei and Shi, Qin and Qin, Yong, "Improved mandarin keyword spotting using confusion garbage model." in *Pattern Recognition (ICPR), 20th International Conference on,* 2010:3700–3703.

[9] Manos, Alexandros Sterios, "A study on out-of-vocabulary word modelling for a segment-based keyword spotting system." in *Massachusetts Institute of Technology,* 1996.

[10] Young, SJ and Russell, NH and Passing, JHS ThorntonToken, "A Simple Conceptual Model for Connected Speech Recognition Systems." in *Cambridge University,* 1989.

[11] Imai, Satoshi, "Cepstral analysis synthesis on the mel frequency scale." in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'83,* 1983:93–96.