Research on incorporating hybrid ASR model to improve KWS-OOV search performance

Zixuan Zhang zz2888
Computer Science Department
Columbia University in the city of New York
New York, U.S.A.
zz2888@columbia.edu

***Abstract*—This paper introduce the basic background and main issues of keyword searching, implements LVCSR-KWS model on TED-LIUM data set, and compares and analyzes the effect of proxy search method and Hybrid model method to solve OOV problem. We simulate full language with full Lexicon, limited language with Limited Lexicon (use proxy search and hybrid-asr model), Let me try it in four different ways. We also explore the impact of increasing model complexity on different situations.**

***Index Terms*—ASR, KWS, OOV, Proxy Search, Hybrid-ASR model**

I. Introduction

A. Background of the task

With the emergence of large-scale voice storage databases, such as TV drama dubbing data, or radio broadcast recording data, etc., people hope to accurately search for keywords in the voice system. Keyword Spotting (KWS) or Keyword Detection describe suck taks that detect a specific set of words from a continues speech. KWS mainly has two problems, keyword spotting and keyword search. In keyword spotting, the keyword is fixed. For example, we always want to know whether a word "up" happened in an audio. Keyword search problem is also known as term detection. The keyword is flexible and we need to locate them in the audio. This task is like 'searching keyword within an article'. But this 'article' is not a text-based but audio-based. However, the KWS is not such a simple task comparing with text matching. Because it is impossible for us to search for a keyword by converting the entire voice data into text and then searching, the cost and time-consuming is unacceptable, and his performance will be very poor. And in the same time, for the same word in text, different situation need to be considered due to the different a ascent or stress. And also, out-of-vocabulary has great impact on a KWS system performance and we will focus on it in this paper.

Luckily, many papers [3] has presented wonderful methods to establish a KWS system. The basic way to conduct this taks was generalized in [1] as $C_{kw} = L(KW)/L_{bkg}$, where C_{kw} is the confidence of this keyword, L(KW) is the likelihood of one keyword happening in a unconstrained utterance, L_{bkg} is the likelihood of utterance without consideration of a keyword. After computing the confidence, we can define a threshold by training to determine which is the keyword. Based on this model, we mainly three methods to conduct KWS task which are Filler Model [2], Query by example and LVCSR KWS. Also, we have end2end neural network models based on totally DNN method, we will not focus on this one in our paper.

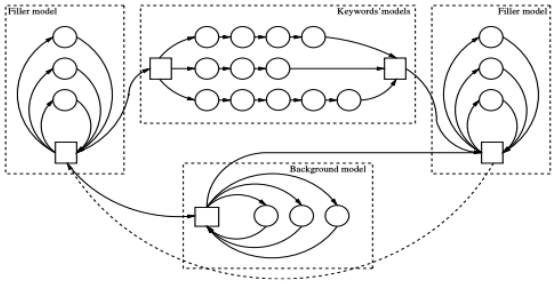

Fig. 1. General KWS Model

The filler model (acoustic KWS model) suggest by [2], it contains two parts (Background Filler Model also known as Garbage Acoustic Model and a Keyword Models). Both are phoneme models but focused on different part. This model can be seen as a frame-wise sequence labelling problem. Keywords and non-keywords will be modeled respectively in this approach. When a token going through the model suggested in the figure, the likelihood of it in two models will be computed, and then we can get the C_{kw} and comparing with the confidence to get the result. However, the problem is this model is not flexible and restricted by the training data. An improving method [4] is to build the keyword system based on phone rather than word. And searching Graph is build with a handcraft phone-level grammar.

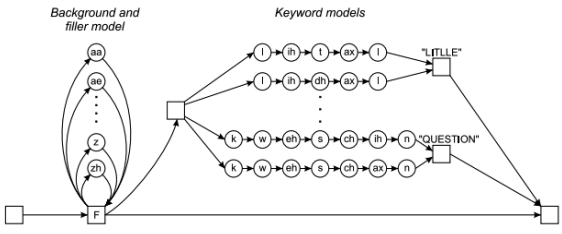

Fig. 2. General Filler Model Overview

Query-by-example method will save the keywords as searching pattern and search them in a speech signal. The keyword can be flexible defined by user recording their own speech. A traditional way to apply this method is to use DTW(Dynamic Time Watching) based method. This method is mainly compute similarity between two sequences of vectors. Firstly, it will convert the queries and target speech into the same representations using acoustic models. Secondly, it will compute the confidence between these two vectors and make the decision. In the KWS, some variants of KWS will be applied to fit for this task including Segmental DTW, Non-segmental DTW, Subsequence DTW and etc. The problem here is its time-cost is big and oversensitive to longer phonetic segments.

Large Vocabulary Continuous Speech Recognition (LVCSR) - lattice [3] is a model for keyword searching has quite good performance. The following experiments in this paper mainly conduced based on LVCSR model and we will have a detailed introduction of this method in Chapter II.LVCSR is one kind of phonetic system, which recognize word based on the sequences of phone. Basic LVCSR has a high error rate, combine the using of lattice can fix this problem because it involving all possible answers. Phonetic speech analytics search preprocesses the audio into the phonemes and encodes the result in a lattice of possibilities. While we search one token, it also will be translated into sequences of phones and compared with the lattices. The advantage of this model is that even a word never happened in training data, but as long as we know its phone, then we can find this word. However, the problem is we might locate too many paths of this word (which is actually wrong).

B. Out of vocabulary(OOV) Issue Introduction

OOV refers to words that are present in the test audio, but we have not encountered during the training process. To be more precise, our pronunciation dictionary does not have the corresponding pronunciation of this word, so we can't say that this word is parsed into the sequence of phones to search in our built index. A more intuitive solution is that we try to provide a large pronunciation dictionary during the training process, so that the OOV situation will be reduced as much as possible. But obviously this approach is not practical, because in real life, we are constantly creating new words, and the appearance of some entity nouns such as names and places, etc., which increases the OOV situation. We cannot collect these every time. The new vocabulary is pronounced, and then the entire model is retrained. At the same time, the existence of OOV has a great impact on the overall performance of KWS. Even if we do not search for OOV vocabulary, the existence of OOV (the uncertainty of OOV phone) in the test file will cause us to be unable to accurately identify the IV vocabulary around him during the decoding process. Generally, our approach is to detect the existence of OOV, and then perform OOV Recovery through a certain model to obtain a new lexicon. Kaldi uses the Proxy Word method to deal with the OOV problem, and we will also try to use a hybrid ASR model to solve the OOV problem, and compare and analyze the results.

C. The overview of this paper

This paper is divided into five parts in total. In the Introduction, we will mainly introduce what is KWS, the common models in KWS and what are the OOV problems we mainly solve. In the second part we talk about the use of the baseline LVCSR model we chose. In the third part, we mainly introduce the improvement plan we put forward on OOV. In the fourth part, we will introduce the data, process and results of our experiment. In the last part, we will analyze our experiment.

Contribution in this paper: First, we implemented a KWS system based on LVCSR on the TED-LIUM data set. Secondly, we analyzed and compared the KWS performance of full lexicon and limited lexicon. Finally, we compared the performance of proxy word and our hybrid ASR model in handling OOV in the case of limited lexicon.

II. Baseline - LVCSR-KWS Model

A. Overview of ASR framework

Generally speaking, an ASR system consists of feature extraction, speech model training, language model training and decoding.

The first is feature extraction. Under normal circumstances, the data we get is waveforms in the time domain. However, in the speech analysis, the frequency domain will contain more useful information, which is more helpful to our model. In order to better represent the sound information, we will extract the features of the sound signal in the frequency domain. The current mainstream extraction methods can be divided into three categories, based on acoustic parameters (MFCC, FBank), based on posteriorgram (GMM, DNN) and based on Bottleneck Feature (DNN, Autoencoder).

Acoustic Model is basically used to exploit the acoustic characteristics os speech sounds. Most ASRs use hidden Markov models (HMMS) to train Acoustic Model. HMM describes a pair of states(X, O), where X is the hidden state that cannot be directly observed, and Y is the observed sequence. The probability of Y occurring can be described as

$$P(O) = \sum_{X} P(O|X)P(X) \tag{1}$$

Since the number of Phones is much less than that of Words, training HMM based on Phones does not require too much data, the result will be relatively better, and fewer parameters are required. In order to integrate the concept of voice context, we generally do not only train a phone, but train it with the phone in front of her and the phone in the back (to become a triphone). Within n of n-gram increasing, the model can consider more context, but becoming more complex to train.

Since there are a large number of homophones with different words in the language, we need to introduce a language model to solve this problem. The existence of the language model is to introduce $P(W)$, which is the probability of a word appearing in the current context. We can describe

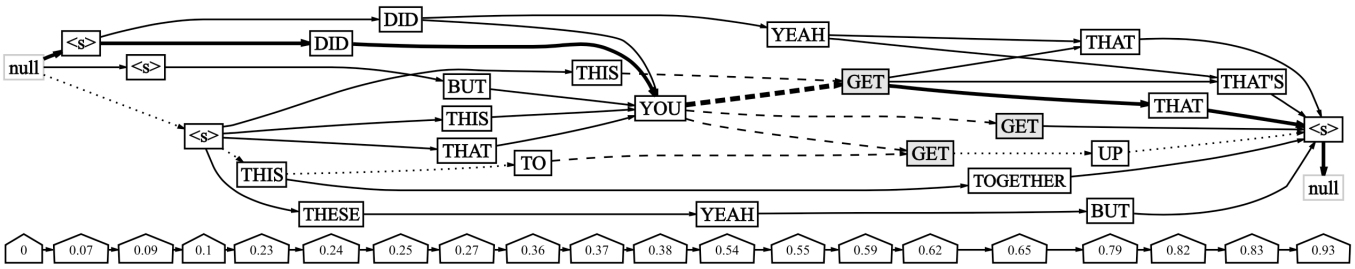

Fig. 3. LVCSR Model Overview

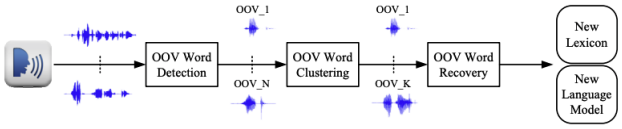

Fig. 4. OOV Detection and Recovery Method

it as $P(w_i) = P(w_i|w_1, w_2, ..., w_{i-1})$. However, there is no possibility for us to calculate $P(w_1, w_2, ..., w_n)$ due to the huge number of parameters. So, in most cases, we will simplify it into n-gram model. For example, the possibility of one word within a 3-gram language model can be described as

$$P(w_i) = P(w_i|w_{i-1}, w_{i-2}) = \frac{C(w_{i-2}, w_{i-1}, w_i)}{C(w_{i-2}, w_{i-1})} \quad (2)$$

, where C is the number of the components showed up in the training data. At the same time, in order to combine the voice model with the language model, we will also introduce a pronunciation dictionary to indicate the phone sequence corresponding to each word.

The subsequent decoding process is our process of predicting the test data. During the training process, we will generate a large HMM Graph and use the training set to train its parameters. Input our test data into HMM Graph, we will get the observation sequence O, and the prediction result will be calculated as:

$$\hat{W} = arg\ max \frac{P(W)P(O|W)}{P(O)} = arg\ max P(W)P(O|W) \quad (3)$$

, where $P(O|W)$ can be calculated from acoustic model and $P(W)$ can be get from langauge model.

B. SGMM-HMM

Our baseline first uses Perceptual Liner Prediction (PLP) to extract sound features. First, we need to process the sound signal and divide it into small frames so that we can grasp the changes in the sound as much as possible, and then calculate the Fast Fourier Transform and Square of magnitude for each frame. Subsequently spectrum is pre-emphasized to approximate the uneven sensitivity of human hearing. Then an Inverse Discrete Fourier Transform is performed to get the auto-correlation coefficients. After applying Linear Prediction analysis and Cepstral analysis, we can get PLP feature.

Subsequently, they use linear discriminant analysis (LDA) and Maximum Likelihood Linear Transform (MLLT). A single fMLLR transform for each training speaker is then estimated and applied for speaker adaptive training (SAT) of triphone GMM-HMM [11].

Finally, the baseline trained SGMM-HMM over the LDA-MLLT-SAT layer. SGMM is called STRANDED GAUSSIAN MIXTURE HMMS. It's a modification and promotion on simply HMM which introducing the mixture sequences. Suppose the O is still our observation sequence, the X is the hidden state sequence, the M will represent the mixture sequences. Then it can be computed as

$$P(O_t, X_t, M_t) = P(O_t|X_t, M_t)P(X_t|X_{t-1})P(M_t|M_{t-1}, X_t) \quad (4)$$

The representation of SGMM computation is shown in the figure.

C. Proxy Search Model

Our baseline use a way called Proxy keyword generation to handle OOV situation within KWS. The basic idea of proxy keyword is to find a in-vocabulary word which most like the OOV word to represent the OOV word. This procedure has been involved in the WFST (a part of decoding process) as :

$$K' = Project(ShortestPath(K \circ L_2 \circ E \circ (L_1^*)^{-1})) \quad (5)$$

, where K' is the proxy keyword we need to get, K is the OOV keyword we get, E represent the edit-distance transform phones, L_1 is the LVCSR's lexicon and L_2 is a finite state for the pronunciation of the OOV keyword.

III. Model and Method - Hybrid ASR Model

A. Overview of our model Pipeline

In the data preparation part, in order to simulate the limited language situation, we only use 20% of the training set for training, and filter to build a smaller lexicon dictionary. Then we use small data to train our language model and acoustic model. We use the SRILM tool provided in Kaldi to train the language model.

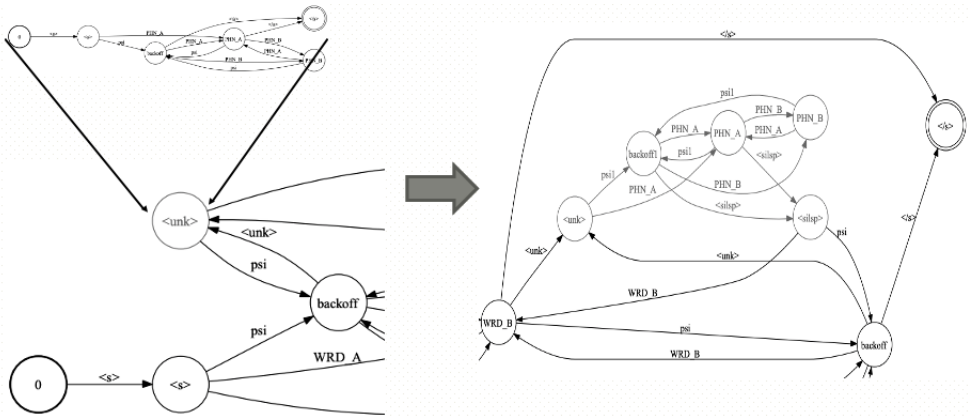

Fig. 5. Hybrid-ASR Method

During the acoustic model training, like Baseline, we first use PLP to extract sound features. Then in order to achieve a smooth training process, we divide the data set into sub-training sets according to the number of 5k, 1w, and 2w, and perform tri-gram HMM model training. Then we used the more common data transformation step, LDA-MLLT-SAT to train the model. LDA stands for Linear Discriminant Analysis, generally used to strengthen the differences between different classifications. Within LDA, our goal is to maximum the formular:

$$
\begin{aligned}
J(w) &= arg\ max_w \frac{w^T S_b w}{w^T S_w w} \\
S_w &= \sum_{\gamma=1}^{N} \sum_{x} (x - u_\gamma)(x - u_\gamma)^T \\
S_b &= \sum_{\gamma=1}^{N} N_\gamma (u_\gamma - u)(u_\gamma - u)^T
\end{aligned}
\tag{6}
$$

, where N is the classes number, S_w is the within class scatter matrix,$S - b$ is the between-class matrix. MLLT is the Maximum Likelihood Linear Transform which always accompany with the LDA [12] [13]. It takes feature space computed from the LDA and derives a unique transformation for each speaker. So, MLLT is also a preparation step for the following SAT(Speaker Adaptive Training) [11], as it minimizes differences among speakers. SAT can further regularize the speaker with the feature vector output by our LDA-MLLT layer. These steps can help us strengthen the sound feature information we need, and make our model more general. Feature space Maximum Likelihood Linear Regression (fMLLR) will use a transformation matrix to transforms the input features from upstream to speaker adapted features and performed as a commonly used estimation techniques [14].

Finally, we will use SGMM (our baseline approach) and DNN to train the model output by fMMLR, and then combine with our Hybrid-ASR model in the following decoding process to decode through SGMM and DNN. Our hybrid-ASR system will make the HMM decoding graph transform from $H \circ C \circ L \circ G$ to $H \circ C \circ (L_{subword} \cup L_{word}) G_{subword} \circ G_{word}$.

B. Hybrid-ASR Search Model

Our Hybrid-ASR Model is essentially the combination between word model and subword model. Within a word model, the language model graph will be trained with word as its node. However, to fit for the open-vocabulary requirement, it has to include tag ¡unk¿ as the annotation of unknown-word. This model has a good performance for in-vocabulary word, however bad for out-of-vocabulary word due to lack knowledge. Then a sub-word model (also called sub-unit model) was proposed to use phones to train the language model rather than the word. Because the size of phones is limited (39 in English), we seems to have the full knowledge of the whole language. Although we have way to recognize the out-of-vocabulary, but the overall performance is bad. Proxy word in our baseline is to find the most similar in vocabulary word to represent the ¡unk¿ tag. And we hope to use the word model as the background model and replace the ¡unk¿ tag part with subword model and generated a huge graph (as indicated in figure).

IV. Experiments and Result

A. Dataset and Preparation

The data set we used is TED-LIUM corpus [1] published in 2014 (also known as TEDLIUM-2). The data contains in this data set are high-quality recorded audio extract from public talks from TED website. This version of data set mainly have 207 hours of talks (including 141 hours male talks and 66 hours female talks) Containing total 2.6 M different words and 1514 unique talks. In addition, this data set also provide a 159848 phone dictionary.

I split the data set into train, dev and test set according to the official definition. Because these data are long talks, which is not suitable for modle training, we split them into segments by 180 seconds. is The Table. I present some important characteristic of these three sub data set. The unique speakers number is calculate according to the unique speaker numbers

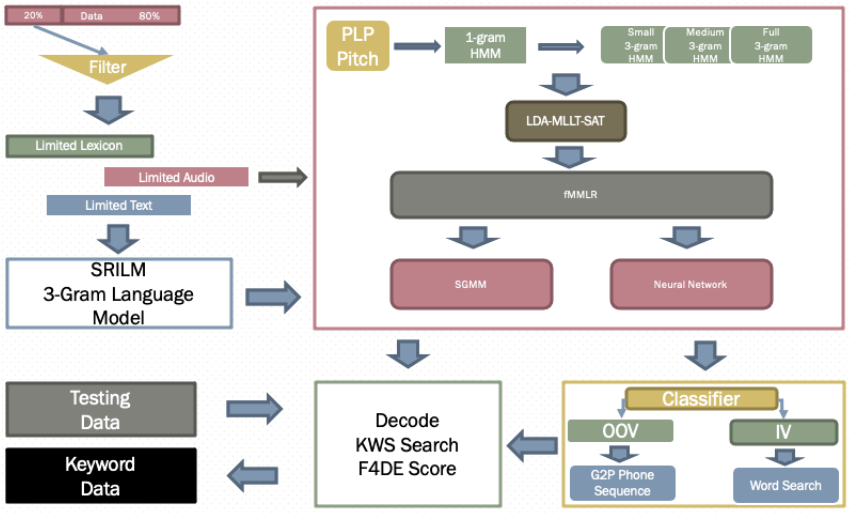

Fig. 6. Hybrid-ASR Model Overview

in spk2utt file. The number of words are the total number (repeatedly) showed up in the whole set.

TABLE I
CHARACTERISTIC OF SPLIT DATA SET

Characteristic	Train	Dev	Test
Number of talks	1495	8	11
Unique Speakers Numbers	5079	38	59
Number of Segments	92973	507	1155
Number of words	2222151	17783	27500

To imitate a limited language and OOV situation, we split the data set by 5 segments per speaker. And generate a small dataset which only has 20% data of the original dataset with 24410 different words. In the same time, we limited the lexicon dictionary, so we can have more out-of-vocabulary situation when we testing. The limited lexicon only have 24406 words' phones rather than the original 160000.

TABLE II
CHARACTERISTIC OF LIMITED TRAINING DATA SET

Characteristic	Train	Dev	Test
Number of talks	1495	8	11
Unique Speakers Numbers	5079	38	59
Number of Segments	24705	507	1155
Number of words	662017	17783	27500

B. Keyword Preparation

We created two keyword search lists, one IV vocabulary and one OOV vocabulary. According to the limited training data and limited lexicon dictionary, we perform word frequency statistics on all the words appearing in the dev and test data set, and divide the word frequency into three categories: high frequency, medium frequency, and low frequency. At the same time, we compared word frequency with limited lexicon, and selected 2k words as the IV database according to the ratio of high frequency: intermediate frequency: low frequency = 4:4:2 from the words contained in limited lexicon. Then, the vocabulary that is no longer in the limited lexicon is selected as the OOV vocabulary according to the same ratio of 363 words.

C. Evaluation Method

Our evaluation method is proposed by NIST in 2013's evaluation [7], Keyword Occurrence Scoring, which evaluates system accuracy based on a temporal alignment of reference keywords to hypothesized keywords. Within this method, the performance metric we used is called Term Weighted Value(TWV) (term in TWV means keyword). It is calculated as:

$$\begin{aligned} TWV(\theta) &= 1 - [P_{miss}(\theta) + \beta P_{False}(\theta)] \\ P_{Miss}(\theta) &= \frac{\sum_{kw=1}^{K} \frac{N_{Miss}(kw,\theta)}{N_{True}(kw)}}{K} \\ P_{False}(\theta) &= \frac{\sum_{kw=1}^{K} \frac{N_{FA}(kw,\theta)}{N_{NT}(kw)}}{K} \end{aligned} \tag{7}$$

where $P_{miss}(\theta)$ describe the missing detection probability, $P_{False}(\theta)$ describe the false detection probability, β is the term-weighted-sum factor, kw represent the i_{th} referenced keyword from 1 to K (total number of keywords), $N_{property}$ represent the number of keywords which own the specific property.

With the overview of what TWV metric is, to be specific, we will use ATWV (Actual TWV, using TWV test the hardness decision for 'YES' occurrences in the system) and MTWV(Maximum TWV, the maximum TWV value we found during the test) for our system.

In our model, we make use the testing tool provide by NIST in Kaldi which require the installation of F4DE. To fulfill the testing procedure, we need to prepare three format file as the entry of this testing software. One ECF file describe

the overview information of testing dataset, including signal duration, version and language, one kwlist file contains the keyword you want to search in a xml format and one rttm file describe every token's begin and end time within the recording. These files have been prepared by tester manually. To make life easier, I create a simple script under /local to prepare these files automatically.

D. Experiments Description

First, we use our Baseline, that is, the SGMM+HMM model to train and decode on the full language TEDLIUM data set, and analyze the 2,360 keywords we generated. In this case, we have almost no OOV situation and look at the performance of the model.

Subsequently, we still use our Baseline to train the model in limited language and limited lexicon, and analyze our keyword search performance. At this time, we have a large number of OOV situations, and we do not have the ability to deal with these OOVs, so we use proxy search to deal with OOV situations and view the results.

Next, we use Baseline to train in a limited language, but provide an extended lexicon file. The OOV situation is mainly due to the lack of lexicon dictionary. We want to see how the expansion of lexicon dictionary will affect the model.

Finally we introduce our hybrid-asr model. First, we use G2P tools to predict OOV phones and generate a larger lexicon dictionary. Then when training the language model, we replace the with the phones generated by our G2P prediction. Next, we will train in limited language and limited lexicon. And compare and analyze the performance of ASR and KWS with other models.

E. Results and Analysis

Firstly, we will analysis the ASR performance for fMLLR and SGMM2 under 4 situations. Firstly, we look into the ASR performance of fMLLR. We use three metrics WER, SER and Confusion Pair to evaluate their performances.

First, by comparing the results of TABLE III and TABLE IV, it can be seen that the effect of SGMM2 is significantly better than that of MLLR. At the same time, we can find that limiting the large amount of training data has little impact on the results of ASR, while extending Lexicon can significantly improve THE performance of ASR. At the same time, the influence of data amount on the model will increase with the increase of model complexity. However, both our Proxy Search model and hybrid ASR model have little effect on improving ASR performance. Even the performance of hybrid model in ASR is relatively poor.

TABLE III
ASR PERFORMANCE FOR FMLLR UNDER DIFFERENT SITUATION

fMLLR	WER	SER	Confusion Pair
Full Lang + Full lex	23.1%	94.9%	2061
Lim Lang + Lim Lex Proxy	25.9%	96.6%	2372
Lim Lang + Ext Lex Proxy	23.5%	94.9%	2119
Lim Lang + Lim Lex Hybrid	26.1%	96.1%	2348

TABLE IV
ASR PERFORMANCE FOR SGMM2 UNDER DIFFERENT SITUATION

SGMM2	WER	SER	Confusion Pair
Full Lang + Full lex	17.5%	90.7%	1476
Lim Lang + Lim Lex Proxy	20.2%	91.3%	1790
Lim Lang + Ext Lex Proxy	18.0%	91.7%	1522
Lim Lang + Lim Lex Hybrid	20.2%	93.5%	1748

Then we start to analysis the performance for different models and different situations. Firstly, we compare the performance of under fMLLR for In-Vocabulary keywords. The metrics we used inlcuding ATWV, STWV, MTWV and OTWV all comes from the TWV we introduced in the last subsection.

We combine the IV and OOV keywords together to search. There is no doubt that the effect of Full Language with Full Lexicon is the best. Secondly, the effect of extend lexicon is very similar to full language. The model using SGMM is better than the fMMLR model on an equal footing. Proxy search results are generally worse than hybrid results. This conclusion can be seen in both fMLLR and SGMM models. However, we can see that the gap between Proxy Search and hybrid model is gradually narrowing in SGMM model. We wonder if Proxy Search is more suitable for complex models. Therefore, after further training of DNN model, proxy Search performs better than Hybrid model in THE task of KWS.

TABLE V
KWS PERFORMANCE UNDER FMLLR

fMLLR	ATWV	STWV	MTWV	OTWV
Full Lang + Full lex	0.7949	0.8798	0.7949	0.8467
Lim Lang + Lim Lex Proxy	0.7674	0.8708	0.7674	0.8251
Lim Lang + Ext Lex Proxy	0.7925	0.8796	0.7934	0.8478
Lim Lang + Lim Lex Hybrid	0.7793	0.8797	0.7803	0.8365

TABLE VI
KWS PERFORMANCE UNDER SGMM2

SMGG2	ATWV	STWV	MTWV	OTWV
Full Lang + Full lex	0.8737	0.9459	0.8737	0.9208
Lim Lang + Lim Lex Proxy	0.8501	0.9435	0.8501	0.9053
Lim Lang + Ext Lex Proxy	0.8663	0.9430	0.8685	0.9146
Lim Lang + Lim Lex Hybrid	0.8544	0.9461	0.8544	0.9074

V. CONCLUSION AND FUTURE WORK

A. Conclusion

First, I used LVCSR to build a KWS system on the TEDLIUM data set, used PLP to extract features, and enhanced audio features through LDA-MLLT-SAT. Then we trained the fMMLR model, and trained the SGMM model and DNN model on this basis. And compare the results of the three models respectively. It can be seen that the effect of DNN is better than that of SGMM, and the effect of fMMLR model is the worst. But the gap is not big.

Secondly, the data preparation of KWS in Kaldi is very complicated, and there is no very convenient code. We provide a script that only requires a data set to generate keywords (including IV and OOV) and related files for you.

Then we used G2P to generate phonemes for OOV, built a subword graph and combined them to form our hybrid ASR model. At the same time, I also provide scripts to train G2P models and synthesize hybrid models. We compared the effects of proxy search, hybrid model and extend lexicon on the results of KWS in the case of limited language.

It can be seen that the effect of extend lexicon is the best, because it is the most correct analysis of vocabulary pronunciation. However, since new words are born every day, it is unrealistic to blindly expand lexicon to cover all vocabulary, and it will increase the computing time of the model. In the process of predicting OOV pronunciation, proxy search replaces OOV with existing words. When the vocabulary is extremely poor, the effect of the model decreases. In the hybrid model, the subword model is a graph composed of phonemes. The prediction result of phonemes generated by G2P is better than the result of proxy search using IV instead. But because IV search is more suitable for word model, it performs very poorly in subword model. We use the hybrid model to replace the data in the lattice graph with the subword model to achieve better search results.

The size of the Lexicon affects the performance of the model more, but as the model becomes more complex, the amount of data becomes more important. On simple models, hybrid model has better effect than proxy search. However, with the increase of model complexity, the effect of Proxy search beats hybrid model on DNN.

B. Future Work

Our experiment is not enough. First of all, due to computing power, we have not fully completed our exploration of DNN, and only conducted simple 4-layer DNN training on two of the models. In the later period, we can continue to explore whether there is a more significant improvement after using TDNN. Secondly, the features used in the model are also a good point of exploration. According to recent papers, the effect of fBank is better than other feature extraction methods, especially when using deep learning. Finally, we can further explore the specific influence of the amount of lexicon on the model effect. According to experiments, we know that the effect of expanding lexicon on the KWS effect is significant, and both proxy search and hybrid model rely on the existing lexicon for prediction. We can explore the impact of expanding different amounts of lexicon on the improvement of the two models in stages. It allows us to make a balance in expanding the cost of lexicon and ensuring the efficiency of the model.

REFERENCES

[1] Rousseau, Anthony & Deléglise, Paul & Estève, Yannick. (2014). Enhancing the TED-LIUM Corpus with Selected Data for Language Modeling and More TED Talks.

[2] Szöke I., Schwarz P., Matějka P., Burget L., Karafiát M., Černocký J. (2005) Phoneme Based Acoustics Keyword Spotting in Informal Continuous Speech. In: Matoušek V., Mautner P., Pavelka T. (eds) Text, Speech and Dialogue. TSD 2005. Lecture Notes in Computer Science, vol 3658. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11551874_39

[3] P. Motlicek, F. Valente and I. Szoke, "Improving acoustic based keyword spotting using LVCSR lattices," 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012, pp. 4413-4416, doi: 10.1109/ICASSP.2012.6288898.

[4] Sun M, Snyder D, Gao Y, et al. Compressed Time Delay Neural Network for Small-Footprint Keyword Spotting[C]//Interspeech. 2017: 3607-3611.

[5] F. Itakura, "Minimum prediction residual principle applied to speech recognition," in IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 23, no. 1, pp. 67-72, February 1975, doi: 10.1109/TASSP.1975.1162641.

[6] G. Chen, O. Yilmaz, J. Trmal, D. Povey and S. Khudanpur, "Using proxies for OOV keywords in the keyword search task," 2013 IEEE Workshop on Automatic Speech Recognition and Understanding, 2013, pp. 416-421, doi: 10.1109/ASRU.2013.6707766.

[7] OpenKWS13 Keyword Search Evaluation Plan

[8] Weintraub M . LVCSR log-likelihood ratio scoring for keyword spotting[C]// International Conference on Acoustics. IEEE, 1995.

[9] G. E. Dahl, T. N. Sainath and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 8609-8613, doi: 10.1109/ICASSP.2013.6639346.

[10] Y. Zhao and B. Juang, "Stranded Gaussian mixture hidden Markov models for robust speech recognition," 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012, pp. 4301-4304, doi: 10.1109/ICASSP.2012.6288870.

[11] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," Computer Speech and Language, vol. 12, pp. 75–98, 1998.

[12] R. O. Duda, P. E. Hart, and David G. Stork, "Pattern classification," in Wiley, November 2000.

[13] R. Gopinath, "Maximum likelihood modeling with Gaussian distributions for classification," in Proc. IEEE ICASSP, 1998, vol. 2, pp. 661–664.

[14] M.J.F. Gales,Maximum likelihood linear transformations for HMM-based speech recognition,Computer Speech & Language,Volume 12, Issue 2,1998,Pages 75-98,ISSN 0885-2308