THUEE Language Modeling Method for the OpenKWS 2015 Evaluation

Zhuo Zhang*, Wei-Qiang Zhang*, Kai-Xiang Shen*, Xu-Kui Yang†, Yao Tian*, Meng Cai*, and Jia Liu*
*Tsinghua National Laboratory for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084
Email: zhangzhu14@mails.tsinghua.edu.cn, wqzhang@tsinghua.edu.cn
†Zhengzhou Information Science and Technology Institute, Zhengzhou 450002

Abstract—In this paper, we describe the THUEE (Department of Electronic Engineering, Tsinghua University) team’s method of building language models (LMs) for the OpenKWS 2015 Evaluation held by the National Institute of Standards and Technology (NIST). Due to the very limited in-domain data provided by NIST, it takes most of our time and efforts to make good use of the out-of-domain data. There are three main steps in our work. Firstly, data cleaning has been done on the out-of-domain data. Secondly, by comparing the cross-entropy difference between the in-domain data and out-of-domain data, a part of the out-of-domain corpus which is well-matched to the in-domain one has been selected as the training corpus. Thirdly, the final n-gram LM is obtained by interpolating individual n-gram LMs according to different training corpus and all the training data is further combined to train one feed-forward neural network LM (FNNLM). In this way, we reduce the perplexity on development test data by 8.3% for n-gram LM and 1.7% for FNNLM, and the Actual Term-Weighted Value (ATWV) of the final result is 0.5391.

I. INTRODUCTION

Statistical language modeling based on a large amount of text data has become the main research area of language modeling in these years. Statistic language models (LMs) estimate the distribution of text segments (e.g. words, morphemes, etc.) for the purpose of various language technologies such as keyword spotting. So far, the n-gram models [1] and the neural network models [2] are the most popular LMs.

As we all know, statistical LMs are particularly sensitive to the changes of the text data on which they are trained. Too much training data which is not matched to the in-domain data may get the final LM much worse. Thus, the selection of training data is an important step that can’t be skipped. There are different approaches to address this problem, and among them, comparing the cross-entropy difference of LMs built from in-domain and out-of-domain data, proposed by Moore and Lewis in 2010 [3], has earned widespread acceptance for its good performance.

In the OpenKWS15 evaluation [4], the in-domain corpus is very limited, so we try to select some useful data from the out-of-domain data to help build LMs based on that method described above. In fact, we have built four LMs including the n-gram LMs and the FNNLMs both for the words and the morphemes. This paper tries to clarify the method of building these LMs and data selection is the key point in the method.

But before the data selection, data cleaning on out-of-domain data has been done first in order to improve the data quality. Though this work would cost much time, it is still worth doing in advance for the reason that the data quality would directly impact the performance of LM even of the whole system.

After data selection, there are training texts including selected data and in-domain data. The FNNLM is trained by merging these data into one corpus. As to the n-gram LM, interpolating individual n-gram LMs built from different training sources linearly is a better way to improve its performance.

We have done a series of experiments to help build better LMs for this evaluation by applying above described techniques. And this paper is organized as follows: in Section 2, we describe data processing on out-of-domain data. In Section 3, we describe the data used for training and testing LMs and the experimental setup. In Section 4, we give the results and our discussions. And finally, we conclude in Section 5.

II. DATA PROCESSING

Data processing on out-of-domain data includes data cleaning, data selection and linear interpolation. And the source language is Swahili during this evaluation.

A. Data Cleaning

In this evaluation, NIST provides a large amount of out-of-domain data which is scraped from the web, and this part of data is not suitable for training LMs directly for the reason that it may contains too many characters which don’t belong to the Swahili.

In our work, words and characters contained within the pronunciation lexicon which is generated from the Language Specific Peculiarities (LSP) [4] provided by NIST are all kept, and character strings which consist of English letters even the letters with phonetic symbols are also kept. The other characters such as web URLs and punctuation are deleted. But before deleting punctuation, we have done sentence segmentation based on the its locations.

This work was supported by the National Natural Science Foundation of China under Grant No. 61370034, No. 61273268, and No. 61403224.

The corresponding author is W.-Q. Zhang.
After data cleaning, we count the word frequencies of the out-of-domain data to generate the vocabulary for language modeling. But because of the problem of lower-case letters and capitals, the word counts may not be correct. Thus, words with capital letters are transformed into lower-case ones except for the words in pronunciation lexicon. And about 110 thousand high frequency words are selected as the vocabulary.

### B. Data Selection

As is mentioned above, data cleaning simply deletes the obvious wrong characters and cannot make the out-of-domain data suitable for language modeling. Our key work for making use of out-of-domain data is data selection. We choose the algorithm proposed by Moore and Lewis [3] as our method to do the data selection. The principle is described as follows:

First, let \( I \) be the in-domain corpus, and \( O \) the out-of-domain one to be selected. \( O_1 \) is a part of \( O \) and well-matched to the \( I \). Given the sentences \( s \) drawn from \( O \), \( P(O_1|s,O) \) will represent the posterior probability of \( s \) belonging to \( O_1 \). So, this probability can indicate the degree of similarity between \( s \) and \( I \).

Based on the Bayes rules, the probability \( P(O_1|s,O) \) can be expressed as

\[
P(O_1|s,O) = \frac{P(s|O_1,O)P(O_1|O)}{P(s|O)} \tag{1}
\]

Because \( O_1 \) is a subset belonging to \( O \), there is,

\[
P(s|O_1,O) = P(s|O_1) \tag{2}
\]

At the meantime, we assume the distribution of \( O_1 \) is similar with \( I \), so,

\[
P(s|O_1) \approx P(s|I) \tag{3}
\]

Thus, we can get following formula:

\[
P(O_1|s,O) = \frac{P(s|I)P(O_1|O)}{P(s|O)} \tag{4}
\]

As to the probability \( P(O_1|O) \), it indicates the prior probability of matched data \( O_1 \), and its value just influences the accuracy of classification which is all the same to every sentence \( s \) to be selected. That means this probability \( P(O_1|O) \) has no effect on the sort of similarity between sentences. So we should only focus on the ratio of \( P(s|I) \) and \( P(s|O) \).

For convenience, we change division to subtraction by working in the log domain. Hence, we get new formula:

\[
\log(P(s|I)) - \log(P(s|O)) \tag{5}
\]

Define \( H_O(s) \) and \( H_I(s) \) as the cross-entropy of \( s \) according to LMs trained on the out-of-domain data \( O \) and the in-domain data \( I \) separately:

\[
H_O(s) = \frac{1}{L} \log P(s|O) \tag{6}
\]

\[
H_I(s) = \frac{1}{L} \log P(s|I) \tag{7}
\]

where, \( L \) is the length of the sentence \( s \).

Moore and Lewis put forward to using the difference of cross-entropy \( H_I(s) - H_O(s) \) as the new metric to select segments of data instead of \( \log(P(s|I)) - \log(P(s|O)) \). Since, without length-normalization, \( \log(P(s|I)) - \log(P(s|O)) \) correlate strongly with the length of \( s \), and it would select sentences according to their lengths. Thus, it would fail to select sentences which are similar with in-domain data. By comparing the difference of cross-entropy, the length \( L \) almost has no effect on the process of selecting data, and we can score the sentences correctly, so that the data selection will be done.

### C. The Use of Selected Data

After the data selection, the selected data and in-domain data are used as our training data. For building FNNLMs, merging these training data to train one LM is enough. But for n-gram LMs, we can build LMs according to different sources of data, and then interpolate these LMs linearly to get the final model. As for the interpolation parameter, which are tuned to make the perplexity on development data set best, we can use SRI Language Modeling Toolkit [5] with the “Compute-best-mix” option to compute in an iterative process. This approach is based on the Expectation Maximization Algorithm (EM) [6]. For the evaluation, the performance of n-gram LM can be further improved by linear interpolation.

## III. Experiment

We do the experiments in OpenKWS15 evaluation and get the final LMs to help this evaluation.

### A. Data Preparation

The source language is Swahili during the OpenKWS15 evaluation. Our all training data, development test data are provided by NIST [4].

The length of training data is 80 hours, and it’s divided into three parts. The first 40-hour part is untranscribed training pool, and we can get 1-best decoded text data from this part; the second 30-hour is transcribed training audio pool, and 3 hours of transcriptions named Very Limited Language Pack (VLLP) is selected from the middle of conversation sides from this part; the rest 10-hour data is transcribed “Tuning Set” audio pool which is used for parameter setting for evaluation conditions, and only 3-hour data of this part is selected to be transcribed. Thus, VLLP, the decoded text data, Tuning Set, these three parts are considered as the in-domain data.

The out-of-domain data to be selected is the text data scraped from the web (202WebText) which is also provided by NIST.

Counting the word frequencies of in-domain and out-of-domain data, approximate 110 thousand words are selected as the vocabulary for language modeling.

In addition, there is another 10-hour data as the development test data, which is used only for development testing and should not be incorporated into training material.
B. Experimental Setup

Based on the algorithm described in Section 2, we use the XenC toolkit [7] to accomplish data selection. We build two LMs according to the in-domain and out-of-domain data, and use this toolkit to compute the cross-entropy difference between them. Then, a new corpus will be sorted according to degree of similarity with in-domain data.

After that, we choose different numbers of sentences from the new sorted corpus to build LMs by MITLM toolkit [8]. At meantime, we have selected the same number of sentences from the previous out-of-domain data randomly and build LMs in that way. By comparing the perplexity on development test data, we can test the effectiveness of the method of selecting data.

When we get the part which is the most similar with in-domain data, we merge this part, the VLLP data and the decoded untranscribed text data to train one FNNLM with variance regularization to reduce the computational complexity [9].

But we have taken another method to build n-gram LM. we combine n-gram LMs built from different amounts of selected data with n-gram LMs built from the VLLP and the decoded untranscribed text data separately using SRI Language Modeling Toolkit [5] to find out which amount is the best to be selected. We should pay attention to the use of in-domain-data , that the Tuning Set can not be used to train LM for interpolation.

What’s more, we merge all the VLLP, the decoded untranscribed text data and the selected data into one new big corpus to train an n-gram LM. That is used to test and verify whether combining different n-gram LMs according to different sources data is better than using all source data together to train one LM.

IV. RESULTS AND DISCUSSIONS

For data selection, Fig. 1 shows the comparison between the method we used and the randomly selecting way. We select different numbers of sentences from out-of-domain data to train LMs, and test their performance on development test data. According to this figure, the perplexity on development test data decreases with the increase of randomly selected data. But as to the method we have used for data selection, the perplexity decreases first and then increases slowly. When an LM is built only from the VLLP corpus, its perplexity is 697.856 which is much smaller than points on the two curves in Fig. 1. It suggests that the out-of-domain data has low correlation with the in-domain data. So, when we use more data to train LMs without any operations on data selection, the performance of LMs could be better, as we can see the changes of the upper curve in Fig. 1.

With regard to the lower curve, using about 50 thousand sentences selected according to their cross-entropy differences for language modeling could get the smallest perplexity of 1513.346. When the amount of data increases, the LM gets worse, which means excessive data is useless for building LMs. A reasonable explanation might be that after data selection, these 50 thousand sentences are the most similar with the in-domain data, and if we increase the amount of training data, it would lead to the result that the correlation between training corpus and in-domain data gets lower.

So we just merge these 50 thousand sentences with the VLLP data and the decoded untranscribed text data to train the FNNLM. The result is that perplexity is reduced by 1.7% from 582.04 to 572.207.

For n-gram language modeling, we interpolate individual LMs linearly to get the final one. So, we would not only use the 50 thousand sentences to train LM. Fig. 2 shows the results of another experiment to help determine how many sentences should be selected. In Fig. 2, the upper curve gives
the performance of merging different amounts of selected data, the VLLP data and the decoded untranscribed text data into a new big corpus to train only one LM. With the increase of selected data, the perplexity gets worse. The reason for this phenomenon is that simply adding selected data into training corpus make the correlation between the whole training data and in-domain data lower. After all, the selected data is out-of-domain data and can not be equal to the in-domain data.

As to the lower curve in Fig. 2, it’s apparent that its perplexity is much better than the upper one. This curve indicates another method of building LM. We combine different LMs built from different sources corpus, including the selected data, the VLLP data, and the decoded untranscribed text data, by using SRI Language Modeling Toolkit option “Compute-best-mix” [5]. From this curve, not only can we know that using this method would get much better LM, but also the approximately suited amount of selected data could be determined. That is, selecting two million sentences from out-of-domain data to help build LM almost get the lowest perplexity.

Detailed experiment results are given in Table I. From the results, we could know that using the method described in this paper can make the perplexity of LM built from in-domain data decrease by 8.3%. And from Table I, using all web data to build LM by linear interpolation gets similar perplexity compared with using selected data only. This is because, during the interpolation process, to get the best perplexity, the weighted parameter of all web data is much smaller than the parameter of selected data, and the in-domain data gets bigger weight at the same time. Doing data selection has another advantage, where the size of the final LM would be much smaller, and would save the computational time and the storage resources.

At last, we give the final results of the word error rate (WER) of the systems with different data for training acoustic models (AMs) and the LMs in Table II. There are two groups of contrasts in this table. Results of the first row and the second row have shown that using selected data to train LMs can improve the WER by 0.6 percent on Tuning Set and 0.8 percent on development test data, and the AMs of these two rows are trained using VLLP only. Results of the third row and the fourth row have shown that using decoded text data for semi-supervised training LMs can improve the WER by 0.6 percent on Tuning Set and 0.3 percent by development test data, and the AMs of these two rows are trained using VLLP and decoded text data for semi-supervised training.

The improvements of the whole system have further demonstrated our method of building LMs for this evaluation is good. The THUEE OpenKWS15 keyword search system includes 12 sub-systems [10, 11]. Among them, 11 sub-systems developed based on Kaldi [12] have used the final morphemes and words n-gram LMs, and the rest one developed based on HTK [13] has used the morphemes and words FNNLMs. The morpheme LMs are trained to deal with the out-of-vocabulary (OOV) problem, and the method of these two morpheme LMs are the same as it used in word LMs’ training. And the ATWV of the final result is 0.5391 in the evaluation.

V. CONCLUSION

Experiments on the OpenKWS15 evaluation have demonstrated the effectiveness of our proposed methods for building LMs. Making good use of out-of-domain data plays an important role in building LMs in this evaluation. For the language modeling techniques, data selection has been used in many situations successfully. In this evaluation, it helps us determine which part can be used for building LMs both for FNNLMs and n-gram LMs. By interpolating n-gram LMs built from different sources, we can further improve the performance of the n-gram LMs.

In the future, we would like to build recurrent neural network (RNN) LMs [14] and long short-term memory (LSTM) LMs [15] for their good performance and interpolate them with our LMs to find out whether it could get further improvements.

REFERENCES


<table>
<thead>
<tr>
<th>Data sources</th>
<th>Perplexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLLP</td>
<td>597.856</td>
</tr>
<tr>
<td>Decoded untranscribed text</td>
<td>549.168</td>
</tr>
<tr>
<td>Selected data</td>
<td>2451.399</td>
</tr>
<tr>
<td>All web data</td>
<td>3099.715</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data sources</th>
<th>Perplexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLLP + decoded text</td>
<td>1208.763</td>
</tr>
<tr>
<td>a) Merging all data</td>
<td>1208.763</td>
</tr>
<tr>
<td>b) Linear interpolation</td>
<td>479.594</td>
</tr>
<tr>
<td>VLLP + decoded text + selected data</td>
<td>1533.638</td>
</tr>
<tr>
<td>a) Merging all data</td>
<td>1533.638</td>
</tr>
<tr>
<td>b) Linear interpolation</td>
<td>481.633</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data sources</th>
<th>WER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>All web data</td>
<td>63.0</td>
</tr>
<tr>
<td>VLLP</td>
<td>63.0</td>
</tr>
<tr>
<td>AM:VLLP</td>
<td>62.4</td>
</tr>
<tr>
<td>AM:VLLP+selected data</td>
<td>61.7</td>
</tr>
<tr>
<td>AM:VLLP+decoded text</td>
<td>61.1</td>
</tr>
<tr>
<td>AM:VLLP+decoded text + selected data</td>
<td>59.5</td>
</tr>
</tbody>
</table>

TABLE I

Perplexity on Development Test Data According to Different Source for Building LMs. The Notation A) Indicates the Method That Merging All Data Sources Into One Corpus to Train LM. The Notation B) Indicates the Method That Combining Individual LMs by Linear Interpolation.

TABLE II

The WER of Systems on Tuning Set and Development Test Data with Different Data Sources for Training AMs and LMs. The Full Systems Are Trained with Gaussian Mixture Models, Multilingual Bottleneck Features and Data Augmentation [10].


