IEICE TRANS. INF. & SYST., VOL.E95–D, NO.9 SEPTEMBER 2012

PAPER

Class-Based N-Gram Language Model for New Words Using Out-of-Vocabulary to In-Vocabulary Similarity

Welly NAPTALI[†*a)], ***Nonmember***, **Masatoshi TSUCHIYA**[††], ***Member***, ***and*** **Seiichi NAKAGAWA**[†], ***Fellow***

SUMMARY Out-of-vocabulary (OOV) words create serious problems for automatic speech recognition (ASR) systems. Not only are they miss-recognized as in-vocabulary (IV) words with similar phonetics, but the error also causes further errors in nearby words. Language models (LMs) for most open vocabulary ASR systems treat OOV words as a single entity, ignoring the linguistic information. In this paper we present a class-based n-gram LM that is able to deal with OOV words by treating each of them individually without retraining all the LM parameters. OOV words are assigned to IV classes consisting of similar semantic meanings for IV words. The World Wide Web is used to acquire additional data for finding the relation between the OOV and IV words. An evaluation based on adjusted perplexity and word-error-rate was carried out on the Wall Street Journal corpus. The result suggests the preference of the use of multiple classes for OOV words, instead of one unknown class.

key words: *out-of-vocabulary, class-based n-gram, language model, adjusted perplexity, speech recognition*

1. Introduction

An ASR system's users tend to speak natural sentences that often contain out-of-vocabulary (OOV) words. OOV words present a serious problem for automatic speech recognition (ASR) in that they introduce two error types into the system. First, OOV words are substituted with in-vocabulary (IV) words and second, the error affects other words nearby. Typically three steps are needed to handle OOV words in an ASR system [1]. The first step determines whether an utterance contains an OOV word, while the second involves recognizing sub-word units contained in the OOV word. The final step involves the recovery, converting the sub-word unit into the corresponding OOV word. There is another approach, that is, registering the OOV words into the ASR's vocabulary. In this paper, we focus on estimating the probability of an OOV word after registering it in the ASR's vocabulary. We simplify the problem as follows: given an OOV word, how do we assign a probability to that word without retraining the whole language model (LM).

Calculating the probability of OOV words is not an easy task, especially in cases where no data or insufficient relevant training data are available. There is always a significant amount of OOV words even when the vocabulary size is very large. A common approach to handling OOV words is to assume a special token <UNK> to represent all unknown words. This unknown word token is treated the same as any other IV word in the probability estimation. This straightforward approach has two shortcomings [7]. First, there is a mismatch in the frequency of OOV words in the training and the test data. Second, the approach ignores their syntactic types and other linguistic information.

Jelinek et al. [8] studied the problem of OOV words by assigning the word into statistically synonymous classes using maximum mutual information and maximum likelihood approach. When encountering a new word during evaluation, the word is added to the vocabulary and the probability is updated according to class-based n-gram LM. Suhm et al. [19] built an OOV-aware language model by mapping all words outside a given vocabulary into the new word class. The proposed LM is no other than a word-based n-gram with the OOV words mapped into classes. Gallwitz et al. [7] proposed the similar approach to Jelinek's but without the need to update the new word into the vocabulary. They assign manually the OOV words into a word class. The work is then followed by Bazzi and Glass [2], suggested multiple classes for OOV words. They used a small number of OOV classes ($8 - 32$ classes) by utilizing parts-of-speech (POS) and agglomerative clustering. Martins et al. [10] proposed a method to handle OOV words without the need for additional data or LM retraining by using POS information. Then, recenly, Lecorve et al. [9] proposed a method utilizing POS and an n-gram similarity to obtain the probability of OOV words.

To handle the OOV words on ASR system, first we have to register the OOV words themselves in the ASR system's vocabulary. The registration could be performed automatically from the related web data of test data domain, or manually added by the user. After registering the OOV words, one may re-train the LM and then perform the ASR, or by making a correction to the ASR's results. Recovering OOV words using web data is not something new. Several approaches have been proposed by some researchers. Methods that use web data to increase the training data for retraining/adapting the LM and registering the vocabulary in the ASR system have been discussed in [14] and [16]. In [15], web data was used to recover the OOV words, but the LM probabilities were obtained from their POS. However, POS has a rather limited number of categories. Furthermore,

Manuscript received November 14, 2011.
Manuscript revised April 17, 2012.

[†]The authors are with the Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi-shi, 441–8580 Japan.
[††]The author is with the Information and Media Center, Toyohashi University of Technology, Toyohashi-shi, 441–8580 Japan.
[*]Presently, with Academic Center for Computing and Media Studies, Kyoto University.
a) E-mail: naptali@slp.cs.tut.ac.jp

DOI: 10.1587/transinf.E95.D.2308

most OOV words are proper nouns.

This paper proposes a framework to estimate the probability of OOV words without retraining the LM using additional data (e.g., data from the World Wide Web (WWW)). Adopting a class-based n-gram LM, we categorize OOV words as IV word classes with similar meanings, then each of the IV words is treated as a singleton class. In this way, for example, a new person's name (OOV word) can be associated with an existing person's name in the IV words that is related to a word class for OOV words. First, documents must be collected to find the class of the OOV word. Then, we find the relation of this OOV word to the IV words using a defined similarity measure. To verify the proposed model, we compare it with the conventional models, in which all OOV words are treated as a single special token <UNK>, the increasing IV size method based on retraining, and some comparable baselines in terms of adjusted perplexity and WER.

In the remainder of this paper, we first give our basic consideration in Sect. 2. Then in Sect. 3, we describe details of the proposed method about how to find the similar IV words to the OOV on observation using WWW data. Section 4 gives the evaluation result on adjusted perplexity, and Sect. 5 gives the evaluation result on WER for automatic speech recognition. The paper ends with our conclusions and future works.

2. Basic Consideration

OOV words can be categorized into two types. *The first type* occurs in the training data (and in the test data). In this case, the new words can be added and the LM is retrained, although it takes much computation time to recalculate all the probability distributions. Moreover, the probability of this new word will likely be unreliable owing to its low frequency of occurrence in the training data. *The second type* includes OOV words that do not occur in the training data, but only in the test data. For this type of OOV word, additional data are needed to provide information about the word. Without any information, there is no way of assigning an appropriate probability to the OOV word. See Fig. 1 for the illustration.

OOV words of the first type are relatively easier to deal with than those of the second type, since the number of OOV words is known, whereas the number of OOV words and the OOV rate in the second type are unknown. Without knowing the total number of OOV words and OOV rate, it is hard to conduct an evaluation. Therefore, we made an assumption that the second type of OOV words would follow the first type in terms of evaluation, i.e., the kinds of OOV and OOV rate. Hence, we investigate the first type of OOV words, and then do the same for the second type.

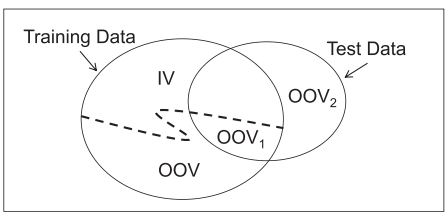


Fig. 1 Two kinds of OOV words.

2.1 Perplexity and Adjusted Perplexity

To evaluate whether an LM is better or worse, we can perform an ASR experiment and compare its word-error-rate (WER). However, this involves the entire ASR system process and consumes computation time. A simpler and more widely used approach is to calculate its perplexity (PP) [13],

$$PP = (P(W))^{-\frac{1}{N}}, \tag{1}$$

where W is a word sequence $W = w_1, w_2, \ldots, w_N$ that includes OOV words (mapped into a unique unknown symbol <UNK>), i.e., $P(w_{OOV}) \to P(\text{<UNK>})$, where w_{OOV} is one of unknown words. Therefore, the perplexity changes according to the vocabulary size. Usually perplexity decreases when the vocabulary size is smaller because of an increased the number of OOV words or OOV ratio. Thus, the perplexity measure should be reported together with the OOV rate. To compare two LMs with different vocabulary sizes, another measure, called adjusted perplexity (APP) , is used. In the APP, $P(w_{OOV}) \to P(\text{<UNK>})\frac{1}{|w_{OOV}|}$. In this paper, we use the APP metric introduced in [20] and [11], defined as follows:

$$APP = (P(W)m^{-o})^{-\frac{1}{N}}, \tag{2}$$

where o is the total number of OOV words in the test data, m is the number of different OOV words in the training data ($m = |w_{OOV}|$), and N is the total number of words (that is, including both IV and OOV words) in the test data. Therefore, adjusted perplexity normalizes the effect of a reduction in OOV rate on the perplexity. In other words, $P(w_{OOV}|\ \text{<UNK>}) = \frac{1}{m}$. For our model, $P(w_{OOV_j}) \to P(C_{OOV_j})\frac{1}{|w_{OOV} \in C_{OOV_j}|}$, where C_{OOV_j} means the word class for w_{OOV_j}.

2.2 Study Case

Handling OOV words using multiple classes is better than mapping it only to one class. Suppose we have a data set consisting of $5,000$ OOV words with OOV rate 5%. Then let us map α of the OOV words (with β occurence) to $UNK1$ class, and the rest to $UNK2$ class.

	# Words	Occurence ratio
$UNK1$	$\alpha \times 5000$	β
$UNK2$	$(1-\alpha) \times 5000$	$1-\beta$

Then the probability of each OOV word can be calculated as follows:

$$P(UNK1) = \frac{5}{100} \cdot \beta \cdot \frac{1}{\alpha \cdot 5000},$$

$$P(UNK2) = \frac{5}{100} \cdot (1-\beta) \cdot \frac{1}{(1-\alpha) \cdot 5000},$$

and the expected total probability of OOV words is given by,

$$P(OOV) = P(UNK1)^{\beta} \cdot P(UNK2)^{(1-\beta)}.$$

If we map the OOV words into one class, then $\alpha = \beta$, resulting $P(OOV) = 10^{-5}$. Note that mapping the OOV words randomly will also cause $\alpha = \beta$. In the other side, mapping the OOV words (non-random) into multiple classes makes $\alpha \neq \beta$, and $P(OOV) > 10^{-5}$. The larger or smaller the ratio of α and β is, the larger $P(OOV)$ becomes. To prove this statement, let γ be an OOV rate, M is the number of OOV words, N is the number of clusters, p_i $(i = 1, 2, \ldots, N)$ is the kind rate of OOV words in each cluster, and q_i $(i = 1, 2, \ldots, N)$ is the occurance rate of OOV words in each cluster. The probability of OOV words is given by:

$$P(OOV_i) = \gamma q_i \frac{1}{M \cdot p_i}.$$

The expected value of $P(OOV_i)$ is given by

$$logP(OOV) = log\frac{\gamma}{M} + \sum q_i log q_i - \sum q_i log p_i.$$

Note that the following expression of inequality is satisfied:

$$-\sum q_i log p_i \geqslant -\sum q_i log q_i.$$

Therefore, $P(OOV$ for multi-classes$) \geqslant P(OOV$ for a single class). This leads to decrease APP.

3. The Proposed Method

A word-based n-gram language model is not adequate for modeling OOV words that are very rare. A class-based n-gram language model is more suitable, since the rare words can rely on other frequent words in the same class. For a given history $h_i = w_{i-n+1} \ldots w_{i-1}$, a class-based n-gram language model is defined as follows:

$$P(w_i|h_i) = P(C_i|C_{i-n+1}, \ldots, C_{i-1})P(w_i|C_i), \quad (3)$$

where C_i is the class of word w_i. Since the IV words are frequent words, we map these words into singleton classes. Then each of the OOV words is mapped into the corresponding class of IV classes. There are two problems to be solved. The first problem is how to find the class of an OOV word (C_{OOV}). We solve this problem by mapping a word into a word vector and using a similarity measure to find the closest word in IV words. The second problem is how to calculate $P(w_i|C_i)$ for OOV words. We assume that all OOV classes are known, and then we calculate $P(w_{OOV}|C_{OOV})$ based on a uniform model:

$$P(w_{OOV}|C_{OOV}) = \frac{1}{|\#\text{Kind of OOV words in } C_{OOV}|}. \quad (4)$$

Note that since IV words use singleton classes, the value of $P(w_i|C_i)$ for IV words is 1.0.

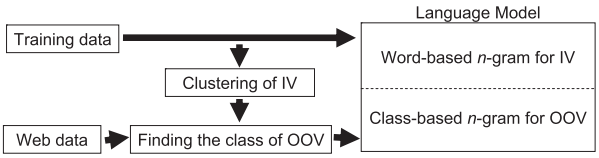


Fig. 2 Training procedure of Proposed Method.

IV words:	OOV words:
$w_1 \rightarrow C_1$	$w_{OOV_1} \rightarrow C_{n+1} = corr\{w_1, w_2, \ldots, w_n\}$
$w_2 \rightarrow C_2$	$w_{OOV_2} \rightarrow C_{n+2} = corr\{w_1, w_2, \ldots, w_n\}$
$\vdots$	$\vdots$
$w_n \rightarrow C_n$	$w_{OOV_m} \rightarrow C_{n+m} = corr\{w_1, w_2, \ldots, w_n\}$
	$w_{OOV_{m+1}} \rightarrow$<UNK>
	$w_{OOV_{m+2}} \rightarrow$<UNK>
	$\vdots$
	$w_{OOV_{m+k}} \rightarrow$<UNK>

Fig. 3 Word to class mapping process.

The overall training procedure is illustrated by the chart in Fig. 2. In this phase, a web data is required to make OOV classes. The mapping process of IV words and OOV words are illustrated by Fig. 3. Where n is the number of IV words, $m + k$ is the total number of OOV words, m is the number of registered OOV words that have additional data, k is the total number of OOV words that is not registered to the recognition vocabulary (without any additional information, there is no way to assign an appropriate probability to the OOV word. Therefore, for such OOVs, we use the unknown class.), and $corr\{\}$ means "corresponds to one of the classes of".

3.1 Clustering of IV Words

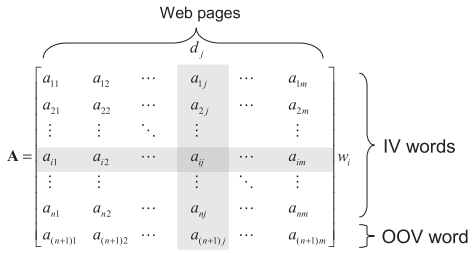
Before we find the classes of OOV words, first we need to make classes of IV words. Note that these classes will not be used to map the IV words in Eq. (3), but to be used to map the OOV words through the similar IV words. In this paper, we classify IV words based on the semantic similarity using latent semantic analysis (LSA) [3]. LSA extracts semantic relations from a corpus, and maps them on a low dimension vector space. The discrete indexed words are projected into an LSA space by applying singular value decomposition (SVD) to a matrix that representing a corpus (representation matrix). In the LSA space, any familiar clustering method could be applied to make semantic classes. In this paper, we used vector quantization with cosine distance.

3.2 Clustering of OOV Words

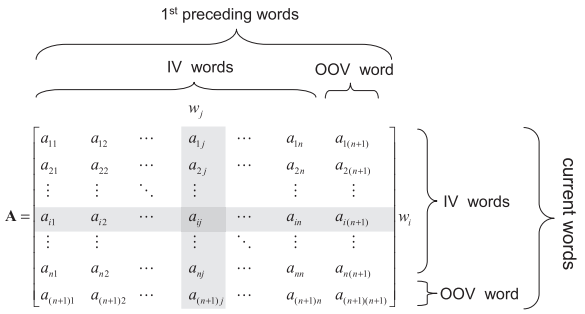
Each word w_i can be represented by the following word vector:

$$\mathbf{w_i} = \mathbf{A}^T \mathbf{c_i}, \quad (5)$$

where $\mathbf{A}$ is the matrix representation and $\mathbf{c_i}$ is the discrete vector of word w_i, where the i-th element of the vector is



(a) Term-document matrix.



(b) Bigram matrix.

Fig. 4 Representation matrix A.

set to 1 and all other elements are set to 0. We use a term-document matrix and a word co-occurrence matrix such as bigram or 1-4 distant bigram matrix [12] as matrix representation to model the relation between words. Term-document matrix **A** is a matrix where its cell a_{ij} contains frequency of word w_i in the document d_j. The observed OOV word w_{OOV} is inserted in the last row of this matrix (see Fig. 4 (a)). Word co-occurrence matrix cell a_{ij} records how many times word w_i occurs after w_j. In other words, its row corresponds with the current words and its column with the preceding words. Unlike the term-document matrix, the OOV word w_{OOV} is also inserted in the last column of the matrix (see Fig. 4 (b)). We normalize the matrix according to a term frequency (tf):

$$tf(i,j) = \frac{a_{ij}}{\sum_k a_{k_j}}. \tag{6}$$

After each word has been represented by a word vector, a similarity measure between an OOV word and any IV word is obtained to find out which IV word can represent the observed OOV word. Here, we used cosine similarity:

$$\cos(\mathbf{w_i}, \mathbf{w_{OOV}}) = \frac{\mathbf{w_i}.\mathbf{w_{OOV}}}{|\mathbf{w_i}||\mathbf{w_{OOV}}|}. \tag{7}$$

By sorting these scores in descending order for all $w_i \in IV$, we can determine the class to represent w_{OOV}. However, we found that several functional words appeared as the closest matching words to these OOV words. To avoid this, an inverse document frequency (idf) weight is applied to (7). The idf weight used in this work is defined as follows:

$$idf(w_i) = \log\left(\frac{dimension(\mathbf{w_i})}{\#Nonzero\ cell}\right). \tag{8}$$

This idf weight is calculated from the training corpus (using the same type of matrix representation), not from the corpus retrieved from the WWW. A small web corpus will be only resulting unreliable idf weight, and will not be effective on eliminating the functional words. Thus, the similarity between an IV word w_i and an OOV word w_{OOV_j} is defined by:

$$Sim(\mathbf{w_i}, \mathbf{w_{OOV_j}}) = \cos_W(\mathbf{w_i}, \mathbf{w_{OOV_j}}) * idf_T(\mathbf{w_i}), \tag{9}$$

where subscripts W and T indicate that the values are calculated using data collected from the WWW and from the available training set, respectively. Finally, the class of OOV word w_{OOV_j} can be decided using the following criterion:

$$C_{OOV_j} = C_{\arg\max_{w_i \in IV} Sim(\mathbf{w_i}, \mathbf{w_{OOV_j}})}, \tag{10}$$

where C_{w_i} denotes the word class of w_i to classify OOV. After obtaining the OOV word classes for all OOV words in the training data, we mapped all the words to their classes and build a class-based n-gram model. Note that each of IV words is mapped to its singleton class.

4. Experiments on Adjusted Perplexity

The experiment was conducted on WSJ corpus from year 1987 to 1989. The training corpus contains 39,962,779 words from 85,445 documents (**WSJTRAIN**) and the test corpus contains 365,730 words from 809 documents (**WSJTEST**). We made a reasonal assumption that any OOV or the same context appears at any additional web data. Therefore, the web data used is not necessarily the same. The main point is to get an additional information about the OOV and its relation to the IV words from resources other than training data. First, we conducted a preliminary experiment to validate our proposed method using WWW to find the similar OOV words from IV words. Then we perform evaluation on OOV words that occur in the training data.

4.1 World Wide Web as Additional Data

Using the most frequent 19,981 words as vocabulary, the OOV rate for training and testing corpus are 2.47% and 2.57%, respectively. There are 144,599 OOV kinds in the training set, and 6,102 OOV kinds in the test set. We collected at most 100 web page addresses for each OOV (using the OOV word itself as a keyword) from search engine Google† collected from December 2008 to February 2009. Note that not all OOV words exist in the WWW. We require at least one occurrence for the class estimation of OOV word. A common stemming procedure to remove affix may solve some of the problem.

From the collected addresses, we retrieve the web page with the depth of 1. Some web pages could not be retrieved for some reasons (e.g., request time out), and some web pages are not parseable (e.g., flash, AJAX). However, these problems could easily be solved using a programming tech-

†http://www.google.com

Table 1 Similarity results. The bold type face means semantically related IV word to the corresponding OOV word.

OOV \ Representation matrix	term-doc	bigram	1-4dbigram
fogelman (proper name)	**cantor**, philanthropy, armenians, diablo, lifestyle, yu, accredited, polar, ne, partisans	**mcdonough**, hello, stimulation, lyrics, awesome, undertake, hi, donation, transforming, whoever	interestingly, stimulation, believer, fatigue, strangers, meridian, im, gentle, oaks, **mcdonough**
troll (proper name)	denim, **russ**, dragon, caves, wow, sailor, im, ne, coloring, brad	statue, blade, hassle, virtual, pot, sexes, caretaker, wreckage, breath, chair	cave, tribes, finnish, edit, creature, im, capitals, preacher, thread, ah
kurokawa (proper name)	pavilion, empires, earthquakes, flats, ashes, organisms, **gogh**, ecology, prix, classmate	retrospect, **alexandria**, acutely, generalized, molded, sorted, catherine, roses, purposely, villain	**browns**, comics, chapel, acutely, roses, picasso, retrospect, oracle, refreshing, residency
sharer (proper name)	softball, retarded, preacher, sprayed, quincy, raped, motorcycles, unannounced, starving, gravel	partnership, stranger, agent, **greene**, midst, telescope, ensuing, safest, narrator, steward	**steward**, narrator, shave, mates, flashed, telescope, sails, tug, ladder, essay

Table 2 Baseline PP and APP on WSJTRAIN for various vocabulary sizes.

IV vocab size	PP	APP	APP (OOV only)	#OOVs	#Kind OOVs	OOV rate (%)
1k	25.0	488.4	533,707	9,502,909	163,580	23.78
5k	37.8	110.9	1,139,373	3,444,977	159,580	8.62
20k	**38.9**	**51.5**	**2,371,657**	**903,782**	**144,580**	**2.26**
21k	38.9	50.6	2,419,397	854,171	143,580	2.14
25k	38.8	48.0	2,616,946	693,520	139,580	1.74
30k	38.7	45.8	2,822,796	550,975	134,580	1.38
40k	38.5	43.1	3,163,710	372,152	124,581	0.93
60k	38.3	40.7	3,601,489	198,550	104,580	0.50
100k	38.2	39.0	3,733,217	71,972	64,580	0.18
165k	38.1	38.1	-	0	0	0

nique. The retrieved web page can not be used for a language model directly. Several steps are necessary; cleaning HTML tags, removing boilerplate (e.g., canned text, includes navigation bars, page headers, link list, disclaimers and copyright statements, and advertisement) and other unwanted material. NCleaner toolkit [4] does this task automatically. After cleaning the web data, we can build matrix representation. The resulting matrix will be very sparse, since it was built only based on a web corpus. Thus, the similarity measure calculation will not be a burden for computational resources. Table 1 shows the 10-closest IV words for 4 OOV words using similarity measure defined in Eq. (9) with a term-document matrix, bigram matrix, and 1-4 distance bigram matrix, respectively.

We can assume that the class of the 1^{st} closest word is not always suitable to represent the class of the OOV word. In this case, we take the N-closest words and vote for the classes of these words based on their occurrence to represent the class of the OOV word. Another approach is to use the averaged word vectors to represent each class, and then to calculate the similarity between each of these classes (instead of the IV words) and the OOV word. In this paper, we will use the 1^{st} candidate or 1^{st} closest word as the suitable closest word.

4.2 OOV Words that Occurs in the Training Data

In this section, we experimented with the training data using a limited vocabulary size and made an evaluation on the same data. An ideal source for additional data is to use WWW. However, we used an English Wikipedia[†] dump data on the 30^{th} of January 2010, because the number of OOV words is larger, and since there is a day limitation when requesting search results from a public search engine. The data is 5.6 GB file consisting of $9,541,307$ pages of articles or documents. For the rest of this section, this Wikipedia data will be referred as web data.

4.2.1 WSJTRAIN

We begin the experiment using the same training and test data, i.e., WSJTRAIN, to avoid the effect of backoff for unseen event. The baseline perplexity and adjusted perplexity on the training data for various vocabulary sizes are given in Table 2. The baseline is a class-based 3-gram LM where each of IV words is mapped into a singleton class and OOV words are mapped into <UNK>. The proposed model is a class-based 3-gram where each IV word is mapped into a singleton class, and OOV word is mapped into a similar IV word class, while the rest of words (UNK words) is mapped to <UNK> class.

In this research, we only take into consideration using $20,000$ IV words and $1k$, $5k$, $10k$, and $20k$ OOV words (first type). While the rest follows the traditional approach, i.e., by mapping into a special token <UNK>, and we will refer to these as UNK words. The numbers of classes for OOV

[†]http://en.wikipedia.com

Table 3 Statistics of OOV, and UNK on WSJTRAIN for the Proposed Method (+20*k* IV).

OOV vocab size	#OOVs	#Kind OOVs	#UNKs	#Kind UNKs
1k	47,622	960	856,160	143,620
5k	199,342	4,736	704,440	139,844
10k	333,206	9,431	570,576	135,149
20k	488,152	18,008	415,630	126,572

words are 100 and 200. Note that these numbers should be optimized in the future. Table 3 shows the statistics for this model. Note that there is the difference between *OOV vocab size* and *the number of kind OOVs*, it is caused by the OOV words without additional data (i.e., it does not appear on the web) and it is mapped to the unknown class.

Table 4 (a) gives the adjusted perplexity of the proposed model for each parameter. Compared to the baseline with 20,000 vocabulary, the adjusted perplexity is getting better when we treat OOV words in multiple classes, instead of one class. A small differences on adjusted perplexity is caused by the large number of IV words in the calculation, compared to the number of OOV words. Therefore, we also calculated the adjusted perplexity where the samples are only when the OOV words appear as the current word in a word sequence (APP (OOV only) in Table 4).

To further validate our proposed model, we compare it with four other baselines; they are:

- LSA baseline: A similar approach with the proposed model, different similarity and without using web data. The model uses the available LSA projection matrix to map OOV words into IV classes (described below in detail).
- TRAIN baseline: A similar approach with the proposed model, the same similarity but without using web data. Instead, the model used training data split in documents, and used a simple indexing to retrieve documents that consisted of OOV words on observation. Note that TRAIN database contains all OOV words in this evaluation.
- RANDOMCLUSTER baseline: A similar approach with the proposed model, different similarity and without using web data. Instead, the model used a random approach to map OOV words into IV classes during training, then are used by the defined classes during testing. Note that this random cluster is consistent with the training data and test data (see Table 5 (a)). Therefore, this clustering method is impractical for new OOV words in test corpus, because we have no means for classifying the new word into one of pre-defined clusters.
- RANDOM baseline: A similar approach with the proposed model, without using similarity and web data. Instead, this model used a completely random approach to map OOV words into IV classes during training and testing. Different to RANDOMCLUSTER baseline, the approach used no information taken from the OOV classes during training, in other words, the cluster is not

Table 4 Adjusted perplexity on WSJTRAIN (+20*k* IV).

(a) Proposed method

OOV vocab size	APP 100 IV Classes	APP 200 IV Classes	APP (OOV only) 100 IV Classes	APP (OOV only) 200 IV Classes
0k	51.5	51.5	2,371,657	2,371,657
1k	50.9	50.9	1,787,706	1,748,026
5k	49.6	49.4	954,245	870,700
10k	48.7	48.4	612,479	526,917
20k	47.9	47.5	409,558	328,474

(b) LSA baseline method

OOV vocab size	APP 100 IV Classes	APP 200 IV Classes	APP (OOV only) 100 IV Classes	APP (OOV only) 200 IV Classes
1k	50.9	50.8	1,784,771	1,742,419
5k	49.6	49.4	939,074	850,345
10k	48.6	48.3	594,578	507,993
20k	47.8	47.4	394,205	315,471

(c) TRAIN baseline method

OOV vocab size	APP 100 IV Classes	APP 200 IV Classes	APP (OOV only) 100 IV Classes	APP (OOV only) 200 IV Classes
1k	50.9	50.9	1,792,445	1,753,818
5k	49.6	49.4	946,755	865,875
10k	48.7	48.4	604,244	523,758
20k	47.9	47.5	400,608	324,894

(d) RANDOMCLUSTER baseline method

OOV vocab size	APP 100 IV Classes	APP 200 IV Classes	APP (OOV only) 100 IV Classes	APP (OOV only) 200 IV Classes
1k	50.9	50.8	1,760,705	1,707,698
5k	49.5	49.2	899,359	795,909
10k	48.5	48.1	558,831	458,433
20k	47.6	47.1	360,581	271,127

(e) RANDOM baseline method

OOV vocab size	APP 100 IV Classes	APP 200 IV Classes	APP (OOV only) 100 IV Classes	APP (OOV only) 200 IV Classes
1k	51.7	51.7	2,453,470	2,501,994
5k	52.3	52.3	2,475,521	2,594,257
10k	52.7	52.9	2,504,126	2,682,534
20k	53.3	53.5	2,651,975	2,910,058

Table 5 Example of word class mapping.

(a) RANDOMCLUSTER method

Training Phase	Test Phase
$w_1 \rightarrow CLASS1$	$w_1 \rightarrow CLASS1$
$w_2 \rightarrow CLASS2$	$w_2 \rightarrow CLASS2$
$w_3 \rightarrow CLASS2$	$w_3 \rightarrow CLASS2$
$w_4 \rightarrow CLASS1$	$w_4 \rightarrow CLASS1$
$w_5 \rightarrow CLASS2$	$w_5 \rightarrow CLASS2$

(b) RANDOM method

Training Phase	Test Phase
$w_1 \rightarrow CLASS1$	$w_1 \rightarrow CLASS2$
$w_2 \rightarrow CLASS2$	$w_2 \rightarrow CLASS2$
$w_3 \rightarrow CLASS2$	$w_3 \rightarrow CLASS1$
$w_4 \rightarrow CLASS1$	$w_4 \rightarrow CLASS2$
$w_5 \rightarrow CLASS2$	$w_5 \rightarrow CLASS1$

consistent with the training and test (see Table 5 (b)). This leads to use wrong class in the test phase.

First, let us perform the LSA baseline. From IV clustering, we obtained matrix projection that was used to project IV words into a vector space. Then, we used this projection matrix to map OOV words into the same vector space according to

$$w_{OOV} = \mathbf{w}_{OOV}\mathbf{V}\mathbf{S}^{-1}, \quad (11)$$

where $\mathbf{w}_{OOV}$ is an OOV word matrix corresponding to the documents in the training data and $\mathbf{V}$ and $\mathbf{S}$ are document and singular matrices, respectively, obtained from LSA. After projecting OOV words into a vector space, for each OOV word, the distance against class centroids was calculated and assign the OOV word to an IV class with the nearest centroid. Table 4 (b) gives the adjusted perplexity. Compared to the proposed model performance in Table 4 (a), there is no significant differences on adjusted perplexity. These results suggest that obtaining OOV information from the web data is as reliable as obtaining OOV information from the training data.This is a good sign for modeling the second type of OOV words. The conclusion is also supported by TRAIN baseline, where its adjusted perplexity is given by Table 4 (c).

Finally, RANDOMCLUSTER and RANDOM baselines are performed using Fisher-Yates shuffle [5]. Fisher-Yates shuffle is an algorithm for generating a random permutation of a finite set. The results are given by Tables 4 (d), and 4 (e), respectively. RANDOMCLUSTER performs the best amongs other baseline. It also performs better than our proposed method. The reason is because RANDOMCLUSTER is actually a partial random method, i.e., it uses the OOV class relation in the training and test phases (see Table 5 (a). For instance, OOV word *hitchcock* is mapped to class C_{39} randomly during training, and during the test the word *hitchcock* will also be mapped into class C_{39}. However, when applied this to handle the OOV words that do not occur in the training data, the assignment of OOV classes will become completely random. Thus, the performance of RANDOMCLUSTER will most likely perform as worse as RANDOM baseline. While using our proposed model to handling the OOV words that do not occur in the training data is expected to give results as good as handling the OOV words that occur in the training data (Table 4 (a)).

Note that LSA baseline, TRAIN baseline, and RANDOMCLUSTER except for the proposed method are not available for the second type of OOV.

4.2.2 WSJTEST

The experiment is ended by making an evaluation on WSJTEST data (open test data). Using web data, all OOV words in the training data are mapped to word classes. For 145*k* model (144, 580 in Table 2), only 60*k* OOV words have additional data in the web, therefore we trained OOV classes using only 60*k* OOV words and the rest 85*k* OOV words were treated as <UNK>. The adjusted perplexity is given by Table 6. It shows that modeling the OOV words using multiple classes lead to better adjusted perplexity than the baseline with a single OOV class.

Table 6 Adjusted perplexity on WSJTEST for the proposed method (+20*k* IV).

OOV vocab size	APP 100 IV Classes	APP 200 IV Classes	APP (OOV only) 100 IV Classes	APP (OOV only) 200 IV Classes	OOV rate (%)
1k (1k)	149.3	149.3	3,119,956	3,089,913	2.39
5k (5k)	148.4	148.2	2,499,368	2,446,090	1.99
10k (9k)	147.8	147.6	2,193,484	2,168,025	1.66
20k (18k)	147.5	147.3	2,024,062	1,991,909	1.27
145k (60k)	148.6	148.4	2,652,141	2,607,767	0.75
Baseline	149.8		3,426.387		2.52

5. Experiments on ASR

5.1 Setup

Using the HTK toolkit [22], we trained acoustic models for American English using 49, 190 utterances from the Wall Street Journal (WSJ) corpus for the years 1987-1989. The resulting feature vector is 39-dimensional, and consists of 12 MFCCs plus the 0^{th} ceptral, together with their first and second deviation coefficients, normalized using ceptral mean subtraction. We also used the CMU pronunciation dictionary[†] and LOGIOS lexicon[††], containing 39 phonemes without lexical stress. The HMM models are initialized based on TIMIT phonetic transcriptions. Cross-word triphones are learned by tied-state triphones based on a decision tree. There are 16 Gaussians for non-silent states and 32 for the silent state. For more details, please refer to [21]. The language model is similar to the previous section (Sect. 5.2). For the decoder, we used the in-house large vocabulary continuous speech recognition system, SPOJUS++ (SPOken Japanese Understanding System) [6].

We selected test data from the WSJ collection, consisting of 553 sentences (10, 957 words). Our used test dataset is relatively difficult set, because we selected the dataset in such a way that each sentence or utterance contained at least one OOV word from a vocabulary of 20, 000 words. The OOV rate and APP are 6.26% and 359.6 for a 20*k* vocabulary size, respectively. We denote this test set as **WSJASR**. The APP, correct (CORR), and WER are given in Table 7 for various vocabulary sizes. The WER and OOV rate for this test dataset using a word-based trigram and ignoring the OOV words are also given in the table. Note that in this experiment we focused on a vocabulary size of 20, 000 (IV words) with a WER of 27.3%. Registering OOV words in the ASR system vocabulary improves the WER, as can be seen in Table 8 by adding 1*k*, 5*k*, 10*k*, and 20*k* words, respectively. Note that this baseline is for a class-based 3-gram, where all the IV words are mapped

[†]http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[††]http://www.speech.cs.cmu.edu/tools/lextool.html

Table 7 Adjusted perplexity and WER of baseline on WSJASR (%).

Vocab size	OOV rate (%)	# OOVs in test	# Kind OOVs in test	PP	APP	Del	Ins	Sub	CORR	WER
20k	**6.26**	**686**	**595**	**164.3**	**359.6**	**3.2**	**4.1**	**20.0**	**76.8**	**27.3**
21k	5.89	645	559	169.6	354.0	3.2	3.9	19.5	77.2	26.7
25k	4.52	495	424	191.1	335.8	3.3	3.2	18.1	78.6	24.6
30k	3.26	357	302	212.1	318.1	3.4	2.8	17.2	79.4	23.4
40k	2.04	212	178	239.5	304.2	3.4	2.1	16.2	80.4	21.7
60k	0.64	67	49	274.2	295.4	3.7	1.6	15.2	81.2	20.4
100k	0.19	20	15	289.8	296.0	3.8	1.4	15.1	81.1	20.4
165k	0.11	12	9	297.8	297.8	3.9	1.3	14.8	81.3	20.0

Table 8 WER of baseline by registering OOV words with 20k IV words on WSJASR (%) (1 class).

OOV Vocab Size for registration	APP	Del	Ins	Sub	CORR	WER
1k	359.6	3.3	3.8	19.6	77.1	26.7
5k	359.6	3.4	3.3	18.5	78.1	25.2
10k	359.6	3.4	2.9	17.8	78.8	24.2
20k	359.6	3.6	2.3	16.9	79.6	22.7

Table 9 Adjusted perplexity of the proposed method on 100 and 200 IV classes on WSJASR (+20k IV words).

OOV vocab size	PP		APP	
	100 IV classes	200 IV classes	100 IV classes	200 IV classes
1k	170.6	170.3	356.7	356.1
5k	195.7	195.5	346.3	345.9
10k	221.5	220.0	339.2	336.8
20k	253.6	251.0	337.0	333.6
Baseline (1 class)	164.3		359.6	

Table 10 WER of the proposed method with 20k IV words on WSJASR (%).

(a) 100 IV classes

OOV Vocab Size for registration	Del	Ins	Sub	CORR	WER
1k	3.3	3.8	19.6	77.1	26.7
5k	3.4	3.1	18.3	78.3	24.8
10k	3.5	2.8	17.5	79.0	23.9
20k	3.7	2.2	16.8	79.5	22.7

(b) 200 IV classes

OOV Vocab Size for registration	Del	Ins	Sub	CORR	WER
1k	3.3	3.8	19.6	77.1	26.7
5k	3.1	3.1	18.3	78.3	24.9
10k	2.8	2.8	17.5	79.0	23.8
20k	3.6	2.1	16.6	79.8	22.4

to a singleton class, and all the OOV words are mapped to an <UNK> class, where each OOV's probability is given by $P(\text{<UNK>}) \times \frac{1}{|\#OOV|}$.

5.2 First Type of OOVs

For the proposed method, the adjusted perplexity is given in Table 9. Similar to the previous section, the proposed method achieves better adjusted perplexity compared with the baseline. The adjusted perplexity also improves with an increase in the number of OOV words registered on observation. The WER for the proposed method can be seen

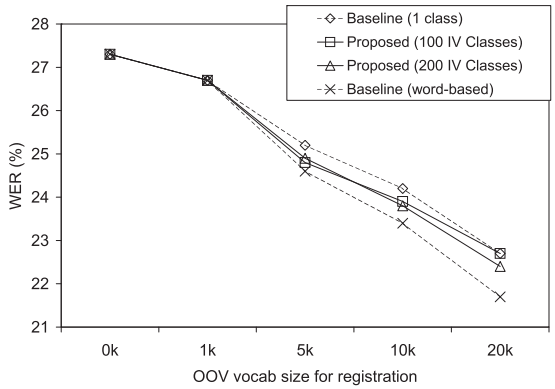


Fig. 5 WER of the Proposed Method on WSJASR. Baseline (word-based) is a word-based 3-gram with OOV words mapped to a single class.

in Tables 10 (a) and 10 (b). Similar to the adjusted perplexity, the proposed method also performs better than the baseline in Table 8 in terms of WER. Incorporating OOV words yields an improvement of at most 18.0% relative to the WER against a model without OOV words. The proposed method also yields at most 1.7% relative improvement on WER against a model with a single <UNK> class (see Fig. 5), where the row "40k" in Table 7 corresponds to the mark "×" at the right corner in Fig. 5.

Although the proposed method performs worse than a word-based LM, which must retrain the LM by using all training data including OOV words, the proposed method does not require any retraining, and is therefore, more effective for the second type of OOV words, but we can not use the word-based method.

5.3 Second Type of OOVs

In this subsection, we add the second type of OOV words into the experiment. As we can see in Table 7, there are 9 kinds of OOV words (12 occurrences) that does not appear on the training data. Using the same offline version of Wikipedia that was explained in Sect. 5.2, we are only able to get the relation of 3 OOV words to the IV word classes. Therefore, we used different web data in recognition phase to obtain information about the OOV. We used the online version of Wikipedia on April 2011. Using at most 100 articles, we got the relation of 6 OOV words (8 occurrences). The rest 3 OOV words (4 occurrences) of the second type were mapped to <UNK> class. We performed the experiments by taking into account of these OOV words (we will

Table 11 WER of the proposed method by incorporating the second type of OOV words on WSJASR (%) (+20k IV words).

Model	Del	Ins	Sub	CORR	WER
1k OOV1 + 9 OOV2 (1 class)	3.3	3.8	19.6	77.1	26.7
1k OOV1 + 6 OOV2 (100 classes)	3.3	3.8	19.5	77.2	26.6
1k OOV1 + 6 OOV2 (200 classes)	3.3	3.8	19.5	77.2	26.6
20k OOV1 + 9 OOV2 (1 class)	3.6	2.2	16.8	79.6	22.6
20k OOV1 + 6 OOV2 (100 classes)	3.7	2.2	16.7	79.6	22.6
20k OOV1 + 6 OOV2 (200 classes)	3.6	2.1	16.6	79.8	22.3

refer these as OOV2 words, and OOV1 for the first type of OOV words) to some of our proposed model in the previous subsection, Sect. 5.2, where the IV size is 20k, the OOV sizes (the first type of OOV words) are 1k and 20k, and the number of classes are 100 and 200. The results can be seen in Table 11. Adding the OOV2 words to the system's vocabulary and mapping the words to the <UNK> class did not change the results (see the first rows in Tables 10 and 11). Finding the relation between the OOV2 words with the IV words lower the WER 0.1% absolute for 1k model and 0.3% absolute for 20k model. Statistical significance was investigated on the latter experiment according to Strik et al. [17], [18] by using a combination of the Number of Errors per Sentence (NES) metric and the Wilcoxon Signed Rank (WSR) test. The improvement is statistically significant at the 0.14% level (p 2-tailed). Note that the proposed method also recovered the errors in the surrounding words by the correct recognition of unknown words. Among 12 occurrences of OOV2, 5 occurrences were able to recovered. Overall, from 426 OOV words (486 occurrences), the 20k model baseline recovered 234 OOV words (268 occurrences), and the our 20k model recovered 268 OOV words (301 occurrences).

6. Conclusions and Future Works

We have shown that the proposed method without retraining the LM is better than the baseline methods. The results suggest the advantages of using multiple classes for OOV words, instead of one unknown class. In this paper, we experimented only on a certain vocabulary size, the number of observed OOV words, and the number of OOV classes. For the class estimation of OOV words, we only required at least one occurrence of OOV word in the additional data, which could result inadequate similarity. For future work, we aim to relax these limitations and optimize all three parameters to achieve a better OOV model.

Acknowledgments

This research was supported in part by the Global COE Program "Frontiers of Intelligent Sensing" from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

[1] I. Bazzi and J.R. Glass, "Modeling out-of-vocabulary words for robust speech recognition," ICSLP 2000, pp.401–404, 2000.
[2] I. Bazzi and J.R. Glass, "A multi-class approach for modelling out-of-vocabulary words," ICSLP 2002, pp.1613–1616, 2002.
[3] J.R. Bellegarda, J.W. Butzberger, Yen-Lu Chow, N.B. Coccaro, and D. Naik, "A novel word clustering algorithm based on latent semantic analysis," Proc. Acoustics, Speech, and Signal Processing, vol.1, pp.172–175, May 1996.
[4] S. Evert, "A lightweight and efficient tool for cleaning web pages," 6th International Conference on Language Resources and Evaluation, pp.3489–3493, 2008.
[5] R. Fisher and F. Yates, Statistical tables for biological, agricultural and medical research, 3rd ed. Oliver and Boyd, London, 1948.
[6] Y. Fujii, K. Yamamoto, and S. Nakagawa, "Large vocabulary speech recognition system: SPOJUS++," 11th WSEAS International Conference on Multimedia Systems and Signal Processing (MUSP '11), pp.110–118, March 2011.
[7] F. Gallwitz, E. Noeth, and H. Niemann, "A category based approach for recognition of out-of-vocabulary words," ICSLP 1996, vol.1, pp.228–231, 1996.
[8] F. Jelinek, R. Mercer, and S. Roukos, "Classifying words for improved statistical language models," ICASSP 1990, pp.621–624, 1990.
[9] G. Lecorve, G. Gravier, and P. Sebillot, "Automatically finding semantically consistent n-grams to add new words in LVCSR systems," Proc. ICASSP, pp.4676–4679, May 2011.
[10] C. Martins, A. Teixeira, and J. Neto, "Automatic estimation of language model parameters for unseen words using morpho-syntactic contextual information," Interspeech 2008, pp.1602–1605, 2008.
[11] S. Nakagawa and H. Akamatsu, "A new computation method of perplexity for text corpus including unknown words – new adjusted perplexity," Acoustic Society of Japan Autumn Meeting 1998, no.2-1-13, pp.63–64, Sept. 1998. (in Japanese).
[12] W. Naptali, M. Tsuchiya, and S. Nakagawa, "Word co-occurrence matrix and context dependent class in lsa based language model for speech recognition," North Atlantic University Union (NAUN) International Journal of Computers, vol.3, no.1, pp.85–95, 2009.
[13] H. Ney, S. Martin, and F. Wessel, "Discriminative training on language model," Corpus-Based Methods in Language and Speech Processing, pp.174–207, Kluwer, The Netherlands, 1997.
[14] T. Ngyz, M. Ostendorfy, M.-Y. Hwangy, M. Siuz, I. Bulykoy, and X. Leiy, "Web-data augmented language models for mandarin conversational speech recognition," IEEE International Conference on Acoustics, Speech and Signal Processing, pp.589–592, March 2005.
[15] S. Oger, V. Popescu, and G. Linares, "Using the world wide web for learning new words in continuous speech. recognition tasks: Two case studies," Proc. Speech and Computer 2009, pp.76–81, June 2009.
[16] C. Parada, A. Sethy, M. Dredze, and F. Jelinek, "A spoken term detection framework for recovering out-of-vocabularywords using the web," Interspeech 2010, pp.1269–1272, Sept. 2010.
[17] H. Strik, C. Cucchiarini, and J.M. Kessens, "Comparing the recognition performance of csrs: In search of an adequate metric and statistical significance test," Proc. ICSLP 2000, pp.740–744, Beijing, China, 2000.
[18] H. Strik, C. Cucchiarini, and J.M. Kessens, "Comparing the performance of two CSRs: How to determine the significance level of the differences," Proc. Eurospeech, vol.3, pp.2091–2094, Aalborg, Denmark, 2001.
[19] B. Suhm, M. Woszczyna, and A. Waibel, "Detection and transcription of new words," EUROSPEECH 1993, pp.2179–2182, 1993.
[20] J. Ueberla, "Analysing a simple language model: Some general conclusions for language models for speech recognition," Comput. Speech Lang., vol.8, no.2, pp.153–176, 1994.
[21] K. Vertanen, "Baseline WSJ acoustic models for HTK and Sphinx: Training recipes and recognition experiments," Technical report, Cavendish Laboratory, University of Cambridge, 2006.
[22] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, The HTK book (for HTK version 3.3). Cambridge, 2005.

Welly Naptali received his B.Sc. of Mathematics degree (with honors) from Bandung Institute of Technology, Indonesia, in 2004. Received his M.E. and Dr. of Informatics degree from Toyohashi University of Technology, Japan, in 2008 and 2011, respectively. He is now a research fellow at Kyoto University. From July 2004 to September 2005, he worked at the Research Consortium of Optimization on Gas and Oil Transmission and Distribution Pipeline Network (RC-OPPINET), Indonesia. His research interests is in automatic speech recognition and natural language processing.

Masatoshi Tsuchiya received his B.E., M.E., and Dr. of Informatics degrees from Kyoto University in 1998, 2000, and 2007 respectively. He joined Computer Center of Toyohashi University of Technology, which was reconstructed to Information Media Center in 2005, as an assistant professor in 2004. His major interest in research is natural language processing.

Seiichi Nakagawa received Dr. of Eng. degree from Kyoto University in 1977. He joined the faculty of Kyoto University in 1976 as a Research Associate in the Department of Information Sciences. From 1980 to 1983, he was an Assistant Professor, from 1983 to 1990, he was an Associate Professor, and since 1990, he has been a Professor in the Department of Information and Computer Sciences, Toyohashi University of Technology, Toyohashi. From 1985 to 1986, he was a Visiting Scientist in the Department of Computer Sciences, Carnegie-Mellon University, Pittsburgh, USA. He received the 1997/2001 Paper Award from the IEICE and the 1988 JC Bose Memorial Award from the Institution of Electro, Telecomm. Engrs. His major interests in research include automatic speech recognition/speech processing, natural language processing, human interface and artificial intelligence. He is a Fellow of IPSJ.