INTERNATIONAL JOURNAL OF COMPUTERS
Issue 1, Volume 3, 2009

Word Co-occurrence Matrix and Context Dependent Class in LSA based Language Model for Speech Recognition

Welly Naptali, Masatoshi Tsuchiya, and Seiichi Nakagawa

***Abstract*— A data sparseness problem for modeling a language often occurs in many language models (LMs). This problem is caused by the insufficiency of training data, which in turn, makes the infrequent words have unreliable probability. Mapping from words into classes gives the infrequent words more confident probability, because they can rely on other more frequent words in the same class. In this research, we investigates a class LM based on a latent semantic analysis (LSA). A word-document matrix is commonly used to represent a collection of text (corpus) in LSA framework. This matrix tells how many times a word occurs in a certain document. In other words, this matrix ignores the word order in the sentence. We propose several word co-occurrence matrices that keep the word order. By applying LSA to these matrices, words in the vocabulary are projected to a continues vector space according to their position in the sentences. To support this matrices, we also define a context dependent class (CDC) LM. Unlike traditional class LM, CDC LM distinguishes classes according to their context in the sentences. Experiments on Wall Street Journal (WSJ) corpus show that the word co-occurrence matrix works 3.62%-12.72 better than word-document matrix. Furthermore, the CDC improves the performance and achieves better perplexity than the traditional class LM based on LSA. When the model is linearly interpolated with the word-based trigram, it gives improvements about 2.01% for trigram model and 9.47% for fourgram model on relative perplexity against a standard word-based trigram LM.**

***Keywords*— Latent semantic analysis, Language model, Word co-occurrence matrix, Context dependent class**

I. Introduction

A Speech recognition task is to find the corresponding word sequence for a given acoustic signal. Let A be an acoustic input, the corresponding word sequence $\hat{W}$ is a word sequence W that has maximum posterior probability $P(W|A)$ given by the following equation:

$$\hat{W} = \arg\max_{W}(\log P_A(A|W) + \lambda \log P_L(W)), \qquad (1)$$

where $P_A(A|W)$ is the probability of A given W based on acoustic model, $P_L(W)$ is the probability of W based on language model, and λ is a scaling factor (language weight).

Language model (LM) is an important study in automatic speech recognition (ASR) system [1][2]. It assigns a probability to word sequences. The most common LM used in modern ASR systems is word-based n-gram. It is a simple and powerful method based on an assumption that the predicted word (current word) depends on only $n-1$ preceding words. In the case of trigram ($n = 3$), the LM gives the following probability to a word sequence $W = w_1, w_2, ..., w_N$:

$$P_{TRIGRAM}(W) = \prod_{i=1}^{N} P(w_i|w_{i-2}, w_{i-1}). \qquad (2)$$

The parameters of the LM are usually trained from a very large collection of text (corpus). If the corpus is not large enough, word which occurs only few times will have unreliable probability, also known as a data sparseness problem. This is a serious problem and frequently occurs in many LMs. The problem is often solved partially using a good smoothing technique [3][4].

A class LM is another way to avoid data sparseness by mapping words into classes, resulting an LM with less parameters. Class LM gives infrequent words more confidence by relying on another more frequent words in the same class. The simplest class LM is known as class-based n-gram LM [5]. However, this LM has its own problem. Class-based n-gram is hard to recognize different histories, which can be encountered only by increasing number of the context [6].

A common way to improve a class-based n-gram LM is by combining it with a word-based n-gram LM using interpolation method [7] [8]. If two LMs model a different part of the language, the interpolation will leads to further improvements. Another approach is to use a class-based n-gram LM to predict the unseen events, while the seen events are predicted by a word-based n-gram LM. This method is known as word-to-class backoff [9]. But when using both a word-based LM and a class-based LM, the size of parameters will be larger than the previous case which is not good for low resource application.

Word-based n-gram is very good on modeling short-range dependencies but weak on modeling of long-range dependencies. Several attempts have been done to capture the long-range dependencies. There are many methods. The cache model [10] increases the probability for words observed in the history based on an assumption that if a word X occurs, then it is most likely that the word X will occur again in the future. Trigger model [11] incorporates any arbitrary word with the corresponding trigger pairs which are combined in an exponential model. A trigger pair $X \rightarrow Y$ makes the probability of Y augment when X occurs in the document

This research was supported by Global COE program "Frontier of Intelligent Sensing" and MEXT. The views expressed are not necessarily endorsed by the sponsors.

Welly Naptali is with the Electronic and Information Department, Toyohashi University of Technology, Japan (e-mail: naptali@slp.ics.tut.ac.jp)

Masatoshi Tsuchiya is with the Information and Media Center, Toyohashi University of Technology, Japan (e-mail:tsuchiya@imc.tut.ac.jp)

Seiichi Nakagawa is with the Electronic and Information Department, Toyohashi University of Technology, Japan (e-mail: nakagawa@slp.ics.tut.ac.jp)

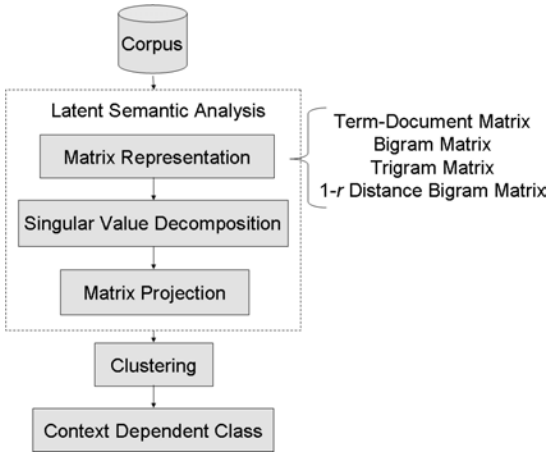

Fig. 1. Model diagram

history. Grammar model [12] is to use syntactical rules for LM. Topic mixture model [13] combined a number of LM trained on documents of various topics. Multi-class model [14] assigns multiple classes to each word by clustering the current word and preceding word separately.

Martin et al. [15] reported some of those language models experiment on Wall Street Journal (WSJ) corpus. The combination of word-based trigram and class-based trigram will improve the performance of word-trigram with 4% relative reduction on perplexity. Using the cache model, the perplexity is getting smaller. It gives 14% relative reduction against the word-based trigram. When the cache model is integrated with the topic adaptation model, it gives 27% relative improvements. Finally when both models are combined with varigram, the perplexity reduced to 31% relatively.

Latent semantic analysis (LSA), which is originally from information retrieval, recently has been successfully used in language modeling to map discrete word into a continuous vector space (LSA space). Bellegarda [16][17] combines the global constraint given by LSA with the local constraint of n-gram language model. The same approach is used in [18][19][20][21][22][23][24] but using Neural Network (NN) as an estimator. Gaussian mixture model (GMM) could also be trained on this LSA space [25]. Instead of a word-document matrix, a word-phrase co-occurrence matrix is used in [26] as a representation of a corpus. The model shows better performance than the clustering method based on the maximisation of the amount of mutual information. However, the model is limited to only the class of the previous word. Our work is somewhat similar to the model with several extensions.

To apply LSA, first the corpus must be represented by a mathematical entity called matrix. LSA is usually used together with a word-document matrix [27] to represent the corpus. Its cell contains the frequency of how many times a word occurs in a certain document in the corpus. The order of words in the sentence is ignored and this is why LSA is also called "bag-of-word" method. However, the word order is too important to be ignored. We will show that LSA could also extract the hidden (latent) information behind this word order. With a word co-occurrence matrix, LSA maps a word into a different point in a continuous vector space according to the word's position in the sentence. Then a clustering is applied on each LSA space to get word classes for different word position. Finally, we propose a context dependent class (CDC) LM that can maintain its ability to recognize different histories. All this process is illustrated by Figure 1.

We divide this paper into the several sections. Section 2 gives an introduction about the traditional class-based n-gram LM. In section 3 we review some related works on modeling long-range dependencies. Section 4 gives a brief review about LSA. Section 5 describes how to build the matrix representation to get the projection matrices from words to the LSA vector space. Section 6 introduces our proposed method; context dependent class language model. Section 7 reports all the experiments. This paper is closed with discussion and conclusions.

II. Class-based n-gram

A class-based n-gram LM[5] was proposed to counter a data sparseness problem suffered by word-based n-gram LM. Without loss of generality, let us consider a bigram case and denote C_i for a class of word w_i. Given w_i, w_{i-1} and its class C_i, C_{i-1}, the probability of current word w_i given history w_{i-1} is calculated according to:

$$P_{CLASS}(w_i|w_{i-1}) = P(w_i|C_{i-1}, w_{i-1}, C_i)P(C_i|C_{i-1}, w_{i-1}). \quad (3)$$

Assume that $P(w_i|C_{i-1}, w_{i-1}, C_i)$ is independent on C_{i-1}, w_{i-1}, and $P(C_i|C_{i-1}, w_{i-1})$ is independent on w_{i-1}. Then Equation (3) becomes:

$$P_{CLASS}(w_i|w_{i-1}) = P(w_i|C_i)P(C_i|C_{i-1}), \quad (4)$$

which is known today as a class-based bigram LM.

In general, the probability of word sequence W is defined by:

$$P_{CLASS}(W) = \prod_{i=1}^{N} P(w_i|C_i)P(C_i|C_{i-n+1}, ..., C_{i-2}, C_{i-1}), \quad (5)$$

Instead of words, a class-based n-gram LM estimates parameters for word classes. By mapping words into classes, this model significantly reduces the parameter size. As a tradeoff, the performance of this model is slightly worse compared to word-based n-gram LM. In this work, word classes are determined by clustering word vectors in the LSA space.

III. Long-range Dependencies

n-gram LMs are very powerful in modeling dependencies between words that are very near to each other within the text. However, they fail in modeling long-range dependencies between words because they rely on the history limited to $n-1$ preceding words. Adaptive language modeling is based on the idea of capturing long dependencies in the corpus. A small number of parameters are added to the model to allow the extraction of information from further back in the document history, then the models adapt their expectation and probabilities. Various approaches have been taken to adapt the language model based on the observed text so far, including

the use of a *cache model*, a *trigger model*, or *topic mixture model*.

The cache model was proposed based on idea that words appearing in a document will have an increased probability of appearing again in the same document. Given a history $h = w_{i-n+1}, ..., w_{i-2}, w_{i-1}$, the cache model is defined by the following equation:

$$P_{CACHE}(w_i|h) = \frac{c(w_i \in h)}{length(h)}, \quad (6)$$

where $c(w_i \in h)$ means how many times w_i occur in the history $w_{i-n+1}, ..., w_{i-2}, w_{i-1}$ so far (from the beginning of the document to the preceding word). It is usually used together with the word-based trigram using linear interpolation,

$$P_L \approx \alpha P_{TRIGRAM} + (1-\alpha)P_{CACHE}. \quad (7)$$

There are other variation in the usage of this cache model. For example n-gram caches, instead of a single word w_i, the model use more than one word (n-gram) to detect the occurrence in the history. n-gram cache is defined by,

$$P_{CACHE}(w_i|h) = \frac{c(w_{i-n+1}, \cdots, w_{i-1}, w_i \in h)}{c(w_{i-n+2}, \cdots, w_{i-1}, w_i \in h)}. \quad (8)$$

Another variation is to use n-gram cache only if $w_{i-n+1}, \cdots, w_{i-1}, w_i \in h$ known as conditional n-gram cache. For short documents, the number of words appearing is limited, so the benefit is small.

Trigger model was also introduced by taking advantages of a long-distance document history which is not covered by statistical n-gram model. Similar to the cache model but more general, the trigger model allow to incorporate arbitrary word trigger pairs which are combined in an exponential model. If a word sequence X is significantly correlated with another word sequence Y, then $X \rightarrow Y$ is considered as a trigger pair. It makes the probability of word sequence Y affected when word sequence X occurs in the document history. The selection of these trigger pairs is a complex issue. If w_1 triggers w_2, and w_2 triggers w_3, then w_1 had better trigger w_3. But this is hard if $w_1 \rightarrow w_3$ happens to be a low frequency trigger pair, because it will likely be pruned away by the trigger pair selection algorithm. As a result, different pairs display different behaviour, and the potential of low frequency word triggers is very limited [11]. Another drawback of this LM is that the performance is very similar to that of the cache model, because most of the best triggers are self-triggers (triggers with the same root).

It is known that a language model built for a restricted topic obtains low perplexity. Topic mixture model is based on the idea of producing LM adapted to each particular topic or domain. Corpus to train an LM usually contains many topics which can also include subtopics. So the corpus can be divided into a set of topics K either manually hand-labelled or using a clustering method. The simplest approach for this model is to combine the topic specific LM using linear interpolation [15]

$$P_{TOPIC}(w_i|h) = \sum_k \lambda_k P_k(w_i|h), \quad (9)$$

where P_k refers to the LM probability that was trained on k^{th} topic, and λ_k is the interpolation weight optimized on a held-out data set. Usually one of the mixture is a global model trained on the entire corpus. However, this LM is less practical since it makes smoothing complicated.

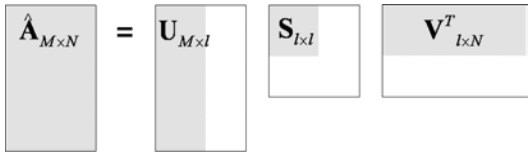

Fig. 2. LSA illustration on matrix decomposition and dimension reduction

Recently, the word trigger concept has been extended into a more systematic framework to handle the trigger pair selection. This is based on a paradigm originally formulated in the context of information retrieval, called latent semantic analysis (LSA). In LSA, the analysis is wider than the long-distance document history. It maps all the words in the corpus to the continuous space, and extracts its semantic relation. Words which have similar semantic meaning will be placed on the continuous space closer. It begins from representing the corpus through a mathematical object called term-document matrix. Then using a Singular Value Decomposition (SVD), the matrix is decomposed and reduced the dimension. Using the matrices obtained from the SVD, every word and document is projected to the continuous space. LSA tries to capture the long-range dependencies and use it together with the word-based n-gram for the short-range dependencies [17]. The probability of this model is given by the following equation:

$$P_{LSA}(w_i|h, \hat{h}) = \frac{P_{NGRAM}(w_i|h)\rho(w_i, \hat{h})}{Z(h, \hat{h})}, \quad (10)$$

where $\hat{h}$ denotes the global document history, $\rho(w_i, \hat{h})$ is a measure of the correlation between the current word and this global LSA history, and $Z(h, \hat{h})$ is a normalization. The language model represents a modified n-gram incorporating large-span semantic information derived through LSA.

IV. Latent Semantic Analysis

LSA extracts semantic relations from a corpus, and maps them on the l-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 original LSA, this representation matrix is a term-document matrix.

Let $\mathbf{A}$ be a representation matrix with $M \times N$ dimension. SVD decomposes the matrix $\mathbf{A}$ into three other matrices $\mathbf{U}$, $\mathbf{S}$, and $\mathbf{V}$

$$\mathbf{A}_{M\times N} = \mathbf{U}_{M\times k}\mathbf{S}_{k\times k}\mathbf{V}^T_{k\times N}, \quad (11)$$

where $k = \min(M, N)$. Because the solution's dimensionality is too large for computing resources, and the original matrix $\mathbf{A}$ is presumed to be noisy, the LSA matrices ($\mathbf{U}$ and $\mathbf{V}$ matrix) dimension is set to smaller than the original,

$$\hat{\mathbf{A}}_{M\times N} = \mathbf{U}_{M\times l}\mathbf{S}_{l\times l}\mathbf{V}^T_{l\times N}, \quad (12)$$

where $l \ll k$ and $\hat{\mathbf{A}}$ is the best least square fit approximation to $\mathbf{A}$. Dimension reduction is illustrated by Figure 2.

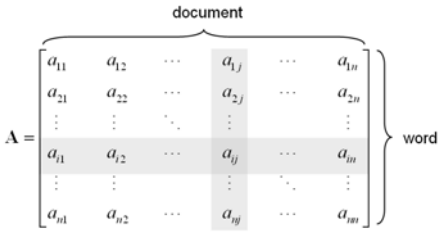

Fig. 3. Term-Document Matrix

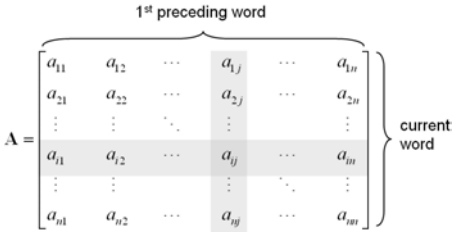

Fig. 4. Bigram Matrix

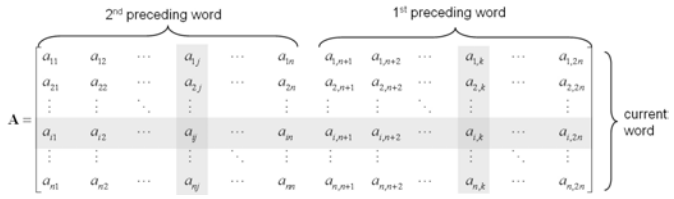

Fig. 5. Trigram Matrix

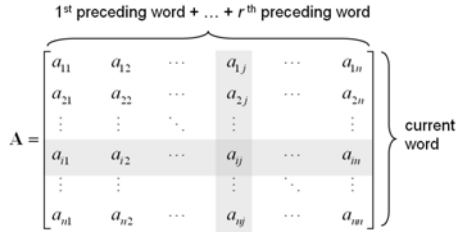

Fig. 6. 1-r Distance Bigram Matrix

The resulting matrix $\mathbf{U}$ is corresponding with the rows of matrix $\mathbf{A}$, and matrix $\mathbf{V}$ is corresponding with the columns of matrix $\mathbf{A}$. These LSA matrices are then used to project the words into the l-dimension LSA vector space. In case of a term-document matrix used as a representation matrix, matrix $\mathbf{U}$ contains information about words while matrix $\mathbf{V}$ contains information about the documents. So matrix $\mathbf{U}$ is used to project words into the LSA space.

Let $\|V\|$ be a size of a vocabulary, each word in the vocabulary can be mapped into an l-dimensional vector space according to the following equation:

$$\mathbf{w}_i = c_i\mathbf{X} \text{ for } 1 \leq i \leq \|V\|, \tag{13}$$

where $\mathbf{X}$ is a projection matrix with $\|V\| \times l$ dimension, and c_i is a discrete vector of word w_i, where the i-th element of the vector is set to 1 and all other $\|V\| - 1$ elements are set to 0. For instance, if $\|V\| = 5$, discrete vector for word w_4 in the vocabulary (c_4) is $(0, 0, 0, 1, 0)$. To make it more easier, a continuous vector for word w_i is represented by the i^{th} row vector of matrix $\mathbf{X}$. So each word w_i has a continuous vector:

$$\mathbf{w}_i = (x_{i1}, x_{i2}, ..., x_{il}) \text{ for } 1 \leq i \leq \|V\|. \tag{14}$$

Since $\mathbf{w}_i$ is a vector which representing word w_i, any familiar clustering method could be applied, and is mapped to the class C_i. The word probability can be approximated according to a class-based LM, such as class-based n-gram [5],

$$\begin{aligned} P_{CLASS}(w_i|w_{i-n+1}, ..., w_{i-1}) \\ = P(C_i|C_{i-n+1}, ..., C_{i-1})P(w_i|C_i), \end{aligned} \tag{15}$$

where C_i is a class of word w_i.

V. Matrix Representation

Originally, LSA uses a word-document matrix to represent a corpus. This matrix ignores the word order in the sentence. In this paper, we tried to keep the order by using co-occurrence of word-word matrix. We propose three kinds of representation matrices; they are bigram matrix, trigram matrix, and 1-r distance bigram matrix.

A. Term-Document Matrix

Term-document matrix or word-document matrix is a matrix where its cell a_{ij} contains occurrence word w_i in document d_j. In other words, the rows of the matrix are corresponding with words and the columns corresponding with the documents.

B. Bigram Matrix

Bigram matrix is a matrix representation where each row represents a current word w_i, and each column represents the 1^{st} preceding word w_{i-1} as illustrated by Figure 4. Each cell a_{ij} is a co-occurrence frequency of word sequence w_jw_i in the corpus. The resulting SVD matrix $\mathbf{U}$ is used to project a current word into the LSA space. While matrix $\mathbf{V}$ is used to project the 1^{st} preceding word.

C. Trigram Matrix

Figure 5 illustrates the trigram matrix. Unlike the trigram matrix defined in [26], in this research the two previous words will not be seen as a phrase, but will be put as independent words in the column. This makes the matrix dimension even smaller. The literature Thus, a trigram matrix is defined as a matrix where each row represents a current word w_i, each column in the first n columns represents the 2^{nd} preceding word w_{i-2}, and each column in the second n columns represents the 1^{st} preceding word w_{i-1}.

Each cell a_{ij}, for the first n columns $(1 \leq j \leq n)$, is a co-occurrence frequency when the word w_j occurs as the 2^{nd} preceding word of word w_i. For the second n columns, each cell a_{ij} $(n + 1 \leq j \leq 2n)$ is a co-occurrence frequency of word sequence w_jw_i. The resulting SVD matrix $\mathbf{V}$ consists of two different parts. The first n rows are used to project the 2^{nd} preceding word into the LSA space, and the next n rows are used to project the 1^{st} preceding word. Matrix $\mathbf{U}$ is used to project a current word.

D. 1-r distance Bigram Matrix

Different with bigram or trigram matrix, in this matrix we tried to collect the information about the previous word in general by accumulating the co-occurrence of r-distance bigram words. So the column in 1-r distance bigram matrix represents the preceding words of $w_{i-r}, ..., w_{i-1}$ in general as illustrated by Figure 6.

Each cell a_{ij} is the accumulation of co-occurrence word w_i as a current word with w_j appearing from the 1^{st} preceding word to the r^{th} preceding word. The resulting SVD matrix $\mathbf{U}$ is used to project a current word w_i into the LSA space. While matrix $\mathbf{V}$ is used to projecting the preceding words.

VI. Context Dependent Class

Because we use the word co-occurrence matrix to represent the corpus, as a result LSA gives a different projection matrix for different word position. For instance, bigram matrix is decomposed into matrix $\mathbf{U}$ that projects the current word w_i, and matrix $\mathbf{V}$ is a projection matrix for the 1^{st} preceding word w_{i-1}. So instead of calculating probability using Equation (15), we need another formulation that could support different classes. The idea is similar to multi-class composite n-gram in [14]. A simple modification on Equation (15), we define a context dependent class (CDC) language model as

$$\begin{aligned} &P_{CDC}(w_i|w_{i-n+1}, ..., w_{i-1}) \\ &= P(C(w_i, \mathbf{X}_i)|C(w_{i-n+1}, \mathbf{X}_{i-n+1}), ..., C(w_{i-1}, \mathbf{X}_{i-1})) \\ &\quad \times P(w_i|C(w_i, \mathbf{X}_i), \end{aligned} \tag{16}$$

where $C(w_i, \mathbf{X}_i)$ is a class of word w_i obtained from the classification of the LSA space build by projecting all words in the vocabulary using projection matrix $\mathbf{X}_i$. For an unseen n-gram class, we applied *class backoff* to a lower context class.

Equation (16) will change according to what kind of matrix representation which is used to represents the corpus. When using a bigram matrix, the equation becomes:

$$\begin{aligned} &P_{CDC}(w_i|w_{i-1}) \\ &\quad = P(C(w_i, \mathbf{U})|C(w_{i-1}, \mathbf{V}))P(w_i|C(w_i, \mathbf{U})). \end{aligned} \tag{17}$$

In a trigram matrix case, the CDC is calculated as follows:

$$\begin{aligned} &P_{CDC}(w_i|w_{i-2}, w_{i-1}) \\ &\quad = P(C(w_i, \mathbf{U})|C(w_{i-2}, \mathbf{V}_1), C(w_{i-1}, \mathbf{V}_2)) \\ &\qquad \times P(w_i|C(w_i, \mathbf{U})), \end{aligned} \tag{18}$$

where $\mathbf{V} = \begin{bmatrix} \mathbf{V_1} \\ \cdots \\ \mathbf{V_2} \end{bmatrix}$, $\mathbf{V_1}$ is the first n rows and $\mathbf{V_2}$ is the second n rows of matrix $\mathbf{V}$. And when using a 1-r distance bigram matrix case, Equation (16) becomes:

$$\begin{aligned} &P_{CDC}(w_i|w_{i-n+1}, ..., w_{i-1}) \\ &\quad = P(C(w_i, \mathbf{U})|C(w_{i-n+1}, \mathbf{V}), ..., C(w_{i-1}, \mathbf{V})) \\ &\qquad \times P(w_i|C(w_i, \mathbf{U})). \end{aligned} \tag{19}$$

Because the matrix $\mathbf{V}$ contains information about all the preceding words, unlike bigram or trigram matrix, the CDC context could be extended into n-gram context without increasing cost to calculate a matrix.

In [7], language models that model different aspects have been successfully combined with an n-gram language model. Here, the statistical n-gram language model is used to capture the local constraint using linear interpolation

$$P_L \approx \alpha P_{CDC} + (1 - \alpha)P_{NGRAM}, \tag{20}$$

where α is a weight constant.

VII. Experiments

A. Setup

The data taken from Wall Street Journal (WSJ) corpus from year 1987 to year 1989 consists of 37 million words in 86,601 documents. It is divided into training and test set. The vocabulary is used ARPA's official "20o.nvp" (20k most common WSJ words, non-verbalized punctuation). By inserting a beginning sentence, an end sentence, and an out-of-vocabulary (OOV) symbols, the vocabulary size is 19,982 words. More detail is given by Table I.

TABLE I
Experimental data statistics

	#Word	OOV Rate
Training Set	36,754,891	0.0236
Test Set	336,096	0.0243

As a baseline, LSA with word-document matrix is used. The matrix representation was decomposed and reduced using SVDLIBC [1] toolkit with Lanczos method. The LSA dimension was varied from 20, 50, 100, and 200 dimensions. The clustering was conducted by Gmeans[2] toolkit using K-means algorithm with Euclidean distance and various numbers of classes from 500, 1000, 2000, and 4000. For an interpolation model, Katz backoff word-based trigram language model is used. Build using HTK Language Model toolkit[6], *the perplexity of a conventional word-based trigram LM was **111.55***. We maximize the interpolation weight α from 0.1 to 0.9 with step size 0.1.

B. Evaluation Method

To evaluate language model for speech recognition, one may run the recognition experiment and calculate the accuracy. However, it takes the whole speech recognition system. More simple and widely used approach is to calculate its perplexity (PP), as defined by the following equation:

$$PP = 2^{-\frac{1}{N}\log_2 P_L(W)}. \tag{21}$$

Minimising perplexity means maximising the log-likelihood function. Although perplexity is not always agree with the accuracy [28], but it is the first approximation towards better language model. It tells how many word choices during the recognition process. Small number of word choices will make speech recognition system easier to choose the correct word.

[1] http://tedlab.mit.edu/∼dr/SVDLIBC
[2] http://www.cs.utexas.edu/users/dml/Software/gmeans.html

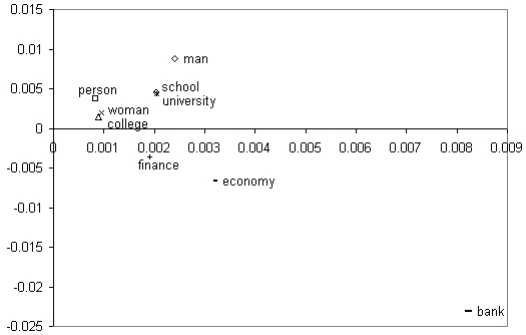

Fig. 7. LSA space of term-document matrix of current word

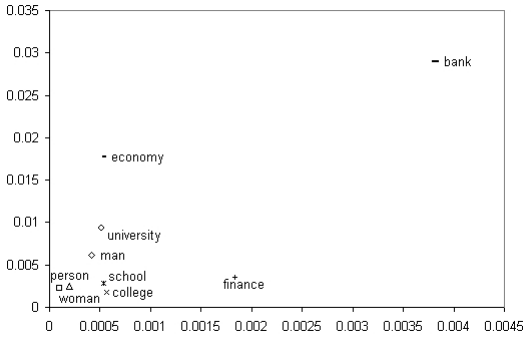

Fig. 8. LSA space of bigram matrix of current word

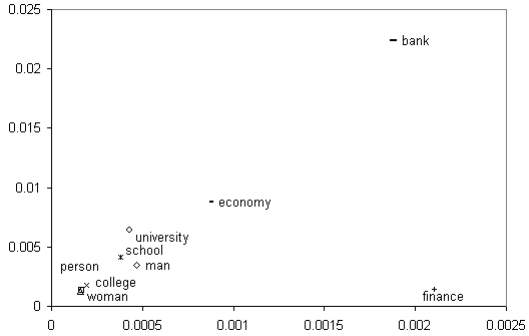

Fig. 9. LSA space of bigram matrix of preceding word

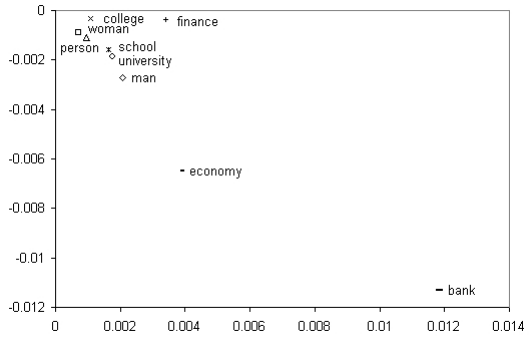

Fig. 10. LSA space of trigram matrix of current word

C. LSA Space

After applying SVD and dimension reduction to a matrix representation, we can project words into a continues vector space using the resulting matrix projection. In this section, we made 2 dimensions plot on nine words where each three of them are closely related. They are *man, woman, person, college, school, university, finance, economy,* and *bank*. With the term-document matrix we can project all those words using $\mathbf{U}$ matrix based on Equation (13). The result can be seen on Figure 7. For the word co-occurrence matrix, there will be more than one plot for each word since the projection matrix is different for a different word position. Word vector *college* as a current word will be different when we compare it to word vector *college* when we refer this word as a preceding word. Thus, it is better to treat this differently. All the plots for LSA with the word co-occurrence matrix can be seen from Figure 8 to Figure 14 for all word positions. From these figures, we may compare word vectors from the word-document matrix with word vectors from the word co-occurrence matrix on how they relate one word to the others. When using word co-occurrence matrix, we will have a different word vectors distribution for different word position in the sentence.

D. LSA-based Class Language Model

To show the performance of the word co-occurrence matrix against word-document matrix, we conducted experiment on these matrices where the probability is calculated with class-based n-gram LM (15). It means that the word co-occurrence matrix will only use its $\mathbf{U}$ matrix after applying LSA. The models are the word-document matrix (TD) as a baseline, the bigram matrix (B), the trigram matrix (T), the 1-2 distance bigram matrix (12DB), the 1-3 distance bigram matrix (13DB), the 1-4 distance bigram matrix (14DB). With 2000 word classes, the result for bigram model is given by Figure 15 in increasing LSA dimension. It shows that by keeping the word order in matrix representation gives improvement on perplexity about 3.62%-12.72% relative. Similar for trigram model (Figure 16), although the trigram matrix gives worse perplexity on 20 dimensions, overall results show that keeping the word order could improve performance. The interpolated model results are given by Figure 17 and Figure 18, respectively. The results for increasing cluster size with fixed 200 LSA dimension can be seen on Figure 19 and Figure 20 for bigram and trigram model, respectively. The interpolated model is also presented in Figure 21 for bigram model, and Figure 22 for trigram model. From all these figures, we can see that the perplexity given by model with the word co-occurence matrix is better than the word-document matrix. LSA extracts latent information that lies within the word order in the word co-occurrence matrix effectively.

E. Context Dependent Class Language Model

Up to this point, the usage of the word co-occurrence matrix is not optimum yet. We only used matrix $\mathbf{U}$ in the model. In the following experiment, we incorporated all LSA matrices of the word co-occurrence matrix in CDC LM. We

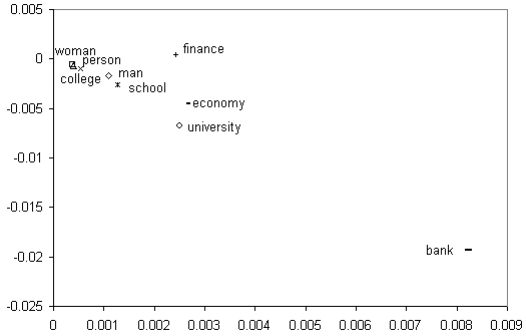

Fig. 11. LSA space of trigram matrix of 1^{st}preceding word

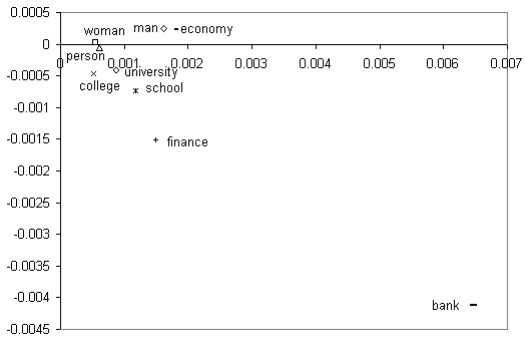

Fig. 12. LSA space of trigram matrix of 2^{st}preceding word

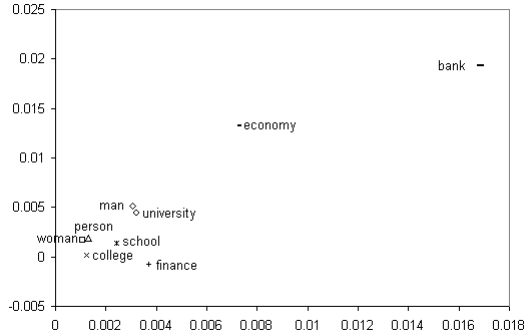

Fig. 13. LSA space of 1-3 distance bigram matrix of current word

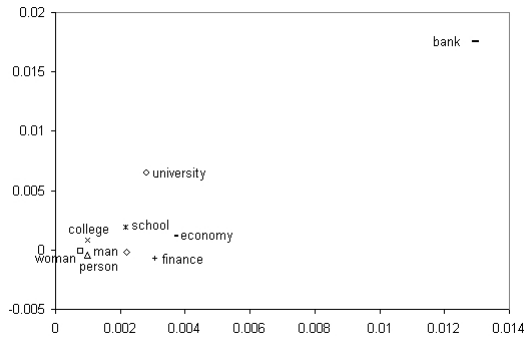

Fig. 14. LSA space of 1-3 distance bigram matrix of preceding word

consider several CDC models; they are CDC with the bigram matrix (CDC-B), CDC with the trigram matrix (CDC-T), CDC with the 1-2 distance bigram matrix (CDC-12DB), CDC with the 1-3 distance bigram matrix (CDC-13DB), CDC with the 1-4 distance bigram matrix (CDC-14DB). As a baseline, a traditional class-based LM with the word-document matrix (TD) is used. Using a fixed 2000 classes, we varied the LSA dimension and show the results in Figure 23 for bigram model and Figure 24 for trigram model. In bigram model, we can see all word co-occurrence matrices give better result than the baseline word-document matrix about 7.88%-13.48% relative. In trigram model, at 20 dimensions, the trigram matrix gives worse performance but after that the performance is increasing. While, the 1-r distance bigram matrix gives better performance in any dimension. Next, we interpolated the model with word-based trigram and show the results in Figure 25 and Figure 26. Although the interpolation gives larger impact to the baseline, but the word co-occurrence matrix model still gives better perplexity about 0.6%-1.68% relative. This is caused by keeping the word order on word co-occurrence matrix. So a combination with word-based trigram which has strong local constraint will impact more on a model which only captures global constraint. Next, we tried to show the model's behaviour on increasing the number of clusters. We set a fixed 200 dimensions LSA space and various number of classes from 500 to 4000. The results are given by Figure 27 for bigram model and Figure 28 for trigram model. The interpolation model's results are also given in Figure 29 and Figure 30. In these figures we can also see that the proposed word co-occurrence matrix has better peformance than the original word-document matrix. These results also indicate that CDC performs better than the traditional class-based language model.

A similar conclusion is also valid for fourgram model on CDC with the 1-r distance bigram matrix (CDC-1rDB) shown by Table II. The perplexity improvements are around 7.09%-10.52% for the stand-alone model, i.e., class-fourgram LM, and 2.41%-2.69% for the interpolated model against LSA based class language model with word-document matrix (TD).

TABLE II
FOURGRAM MODEL OF CDC WITH 200 LSA DIMENSIONS

	Stand-alone	Interpolated	Stand-alone	Interpolated
	2000 classes		4000 classes	
TD	161.51	104.73	131.95	103.68
CDC-12DB	146.23	102.21	122.59	101.05
CDC-13DB	144.53	101.94	122.43	100.99
CDC-14DB	144.53	101.91	122.52	101.07

VIII. DISCUSSION

In 1993, Kneser and Ney [29] proposed a class mapping algorithm with the objective function is to maximize the likelihood, thus it will minimize the perplexity. The algorithm starts from a given number of class, then iteratively mapped a word into a class which will gives the highest likelihood. This is repeated until some threshold or some number of

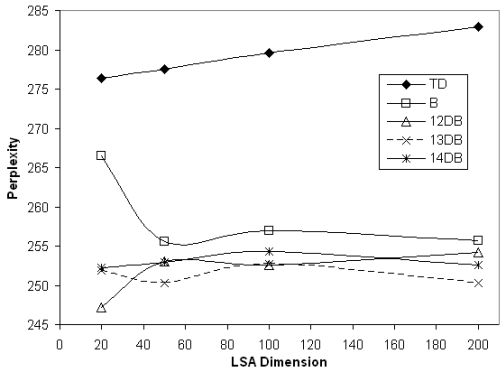

Fig. 15. Bigram model on increasing LSA dimension with 2000 number of classes

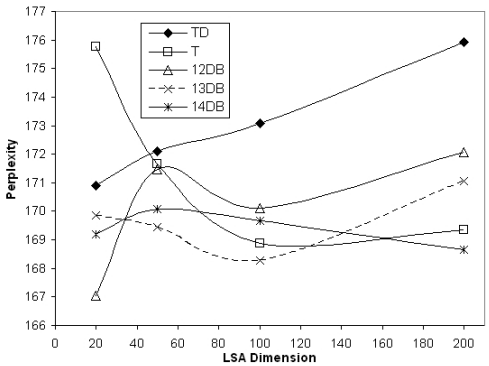

Fig. 16. Trigram model on increasing LSA dimension with 2000 number of classes

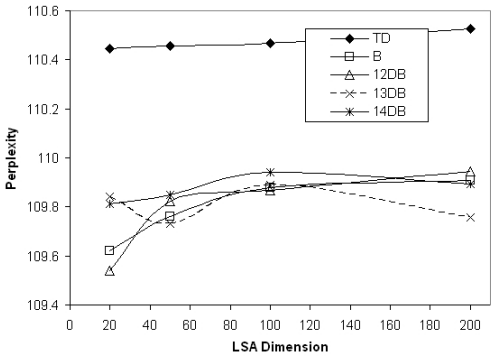

Fig. 17. Interpolated bigram model on increasing LSA dimension with 2000 number of classes

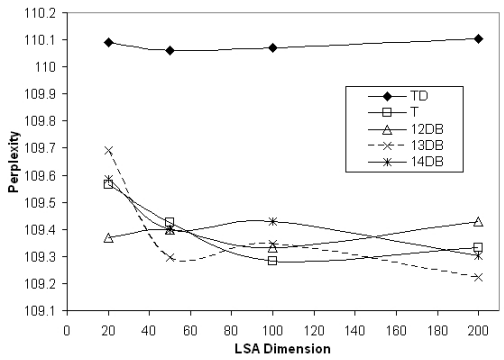

Fig. 18. Interpolated trigram model on increasing LSA dimension with 2000 number of classes

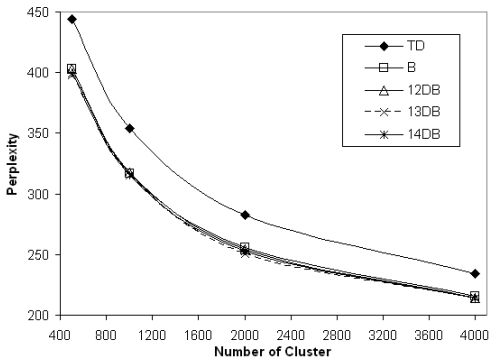

Fig. 19. Bigram model on increasing cluster size with 200 LSA dimensions

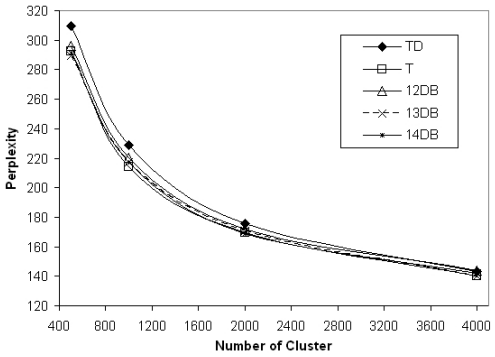

Fig. 20. Trigram model on increasing cluster size with 200 LSA dimensions

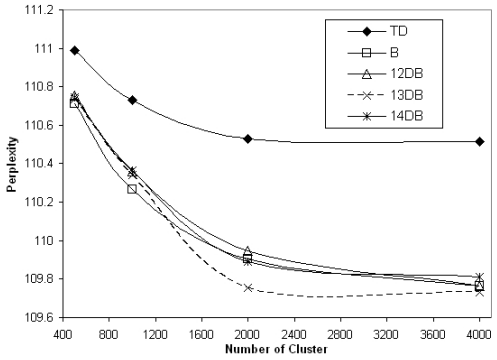

Fig. 21. Interpolated bigram model on increasing cluster size with 200 LSA dimensions

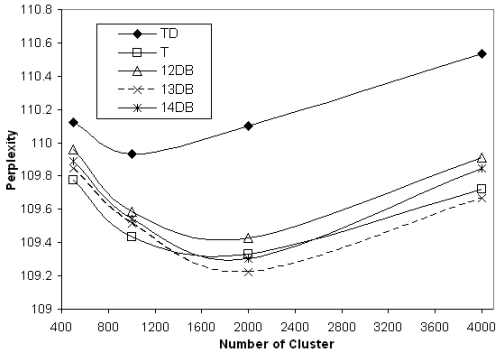

Fig. 22. Interpolated trigram model on increasing cluster size with 200 LSA dimensions

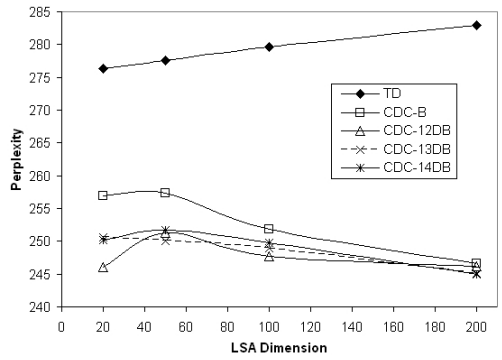

Fig. 23. Bigram model of CDC on increasing LSA dimension with 2000 number of classes

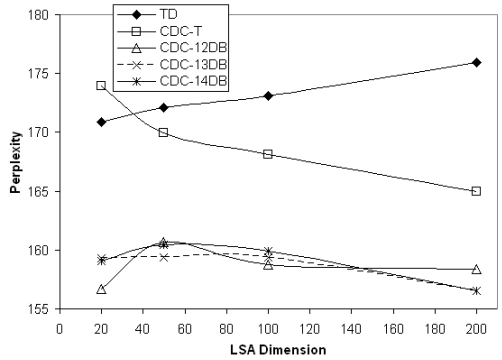

Fig. 24. Trigram model of CDC on increasing LSA dimension with 2000 number of classes

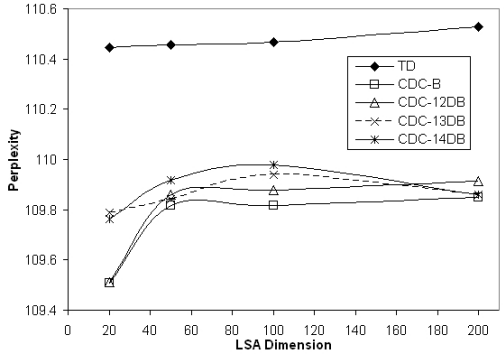

Fig. 25. Interpolated bigram model of CDC with word-based trigram on incrasing LSA dimension with 2000 number of classes

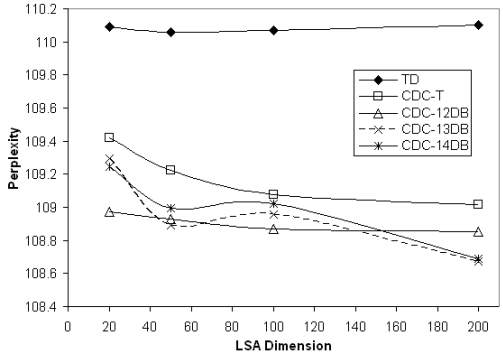

Fig. 26. Interplated trigram model of CDC with word-based trigram on incrasing LSA dimension with 2000 number of classes

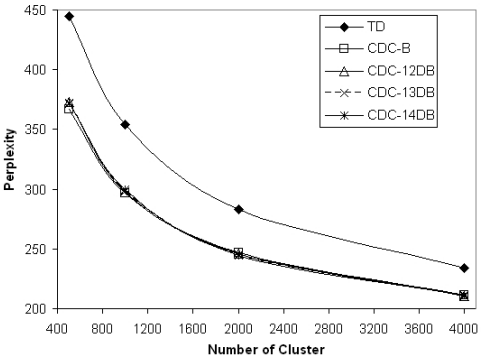

Fig. 27. Bigram model of CDC on increasing cluster size with 200 LSA dimensions

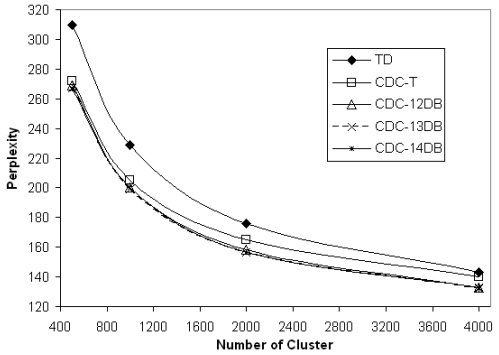

Fig. 28. Trigram model of CDC on increasing cluster size with 200 LSA dimensions

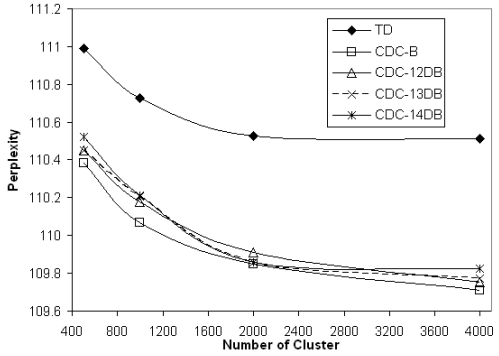

Fig. 29. Interpolated bigram model of CDC with word-based trigram on increasing cluster size with 200 LSA dimensions

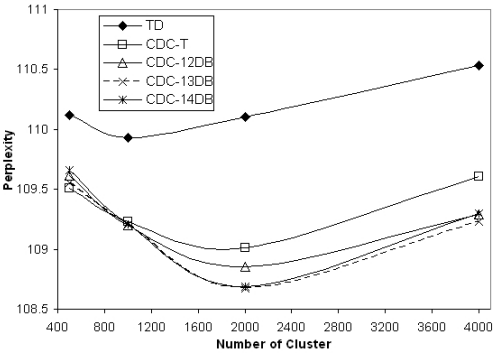

Fig. 30. Interplated trigram model of CDC with word-based trigram on increasing cluster size with 200 LSA dimensions

iteration has been reached. The initial class is usually formed by randomly distributed words among all classes or placing all words in the first class. This algorithm is implemented in the HTK LM Toolkit [6].

The approach that we proposed in this research is performed in a different way. A semantic approach performed by LSA maps similar words into a continuous vector space closely to each other. Then the clustering method will group these words into semantic classes. The class-based LM performed using HTK on 2000 classes gives perplexity 132.38 for trigram model and 121.46 for fourgram model. These results are comparable to the CDC LM method with 200 LSA dimensions and 4000 classes proposed here, which gives perplexity 132.46 for trigram model and 122.43 for fourgram model. This fact proves the validness of our proposed method.

The linear interpolation of CDC LM with word-based trigram LM give improvements around 2.01% for trigram model, and 9.47% for fourgram model against the word-based trigram LM perplexity. Obviously, these results can still be improved by optimizing the available parameters, such as using another distance in the clustering, or changing the clustering method itself. We are also looking forward to use another word extraction method, such as Probabilistic LSA (PLSA) [30] or Latent Dirichlet Allocation (LDA) [31].

IX. Conclusion

In this paper, we demonstrated that word co-occurrence matrix has better performance than the traditional word-document matrix in LSA framework. One of the reason is because the proposed word co-occurrence matrix keeps word order unlike word-document matrix. We also showed that the CDC LM improve its perplexity and also give performance better than the traditional class-based n-gram LM based on LSA.

References

[1] R. Thangarajan, A. M. Natarajan, and M. Selvam, "Word and triphone based approaches in continuous speech recognition for tamil language," *WSEAS Transaction on Signal Processing*, vol. 4, no. 3, pp. 76–85, 2008.

[2] A. Žgank, T. Rotovnik, and M. S. Maučec, "Slovenian spontaneous speech recognition and acoustic modeling of filled pauses and onomatopoeas," *WSEAS Transaction on Signal Processing*, vol. 4, no. 7, pp. 388–397, 2008.

[3] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transaction on Acoustic, Speech and Signal Processing*, vol. 35, no. 3, pp. 400–401, March 1987.

[4] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *International Conference on Acoustics, Speech, and Signal Processing, 1995*, vol. 1, 1995, pp. 181–184.

[5] P. Brown, V. Pietra, P. deSouza, J. Lai, and R. Mercer, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18, pp. 467–479, 1992.

[6] S. Yung, 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.

[7] S. Broman and M. Kurimo, "Methods for combining language models in speech recognition," *Interspeech*, pp. 1317–1320, September 2005.

[8] Y. Wada, N. Kobayashi, and T. Kobayashi, "Robust language modeling for a small corpus of target tasks using class-combined word statistics and selective use of a general corpus," in *Systems and Computers in Japan*, vol. 34, no. 12, 2003, pp. 92–102.

[9] T. Niesler and P. Woodland, "Combination of word-based and category-based language models," in *Proceeding ICSLP96*, 1997, pp. 1779–1782.

[10] R. Kuhn and R. de Mori, "A cache based natural language model for speech recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 14, 1992, pp. 570–583.

[11] R. Rosenfeld, "A maximum entropy approach to additive statistical language modeling," in *Computer Speech and Language*, October 1996.

[12] D. Jurafsky, C. Wooters, G. Tajchman, J. Segal, A. Stolcke, E. Fosler, and N. Morgan, "Using a stochastic context free grammar as a language model for speech recognition," in *IEEE ICASSP*, 1995, pp. 189–192.

[13] R. M. Iyer and M. Ostendorf, "Modeling long distance dependencies in language: Topic mixtures versus dynamic cache model," in *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 1, January 1999, pp. 236–239.

[14] H. Yamamoto, S. Isogai, and Y. Sagisaka, "Multi-class composite n-gram language model for spoken language processing using multiple word clusters," in *ACL 2001: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2001, pp. 531–538.

[15] S. C. Martin, J. Liermann, and H. Ney, "Adaptive topic-dependent language modelling using word-based varigrams," in *Proceeding Eurospeech*, Rhodes, Greece, 1997, pp. 1447–1450.

[16] J. R. Bellegarda, "A multi-span language modelling framework for large vocabulary speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 6, pp. 456–457, September 1998.

[17] ——, "Latent semantic mapping," *IEEE Signal Processing Magazine*, pp. 70–80, September 2005.

[18] Y. Bengio and R. Ducharme, "A neural probabilistic language model," in *Neural Information Processing Systems*, vol. 13, 2001, pp. 932–938.

[19] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," in *Journal of Machine Learning Research*, vol. 3, 2003, pp. 1137–1155.

[20] F. Blat, M. Castro, S. Tortajada, and J. Senchez, "A hybrid approach to statistical language modeling with multilayer perceptrons and unigrams," in *Proceedings of the 8th International Conference on Text, Speech and Dialogue*, 2005.

[21] A. Emami, P. Xu, and F. Jelinek, "Using a connectionist model in a syntactical based language model," in *ICASSP*, 2003, pp. I:272–375.

[22] H. Schwenk and J. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition," in *ICASSP*, 2002, pp. I:765–768.

[23] ——, "Neural network language models for conversational speech recognition," in *ICSLP*, 2004, pp. 1215–1218.

[24] ——, "Building continuous space language models for transcribing european languages," in *Eurospeech*, 2005, pp. 737–740.

[25] M. Afify, O. Siohan, and R. Sarikaya, "Gaussian mixture language models for speech recognition," *ICASSP*, vol. 4, pp. 29–32, April 2007.

[26] S. Terashima, K. Takeda, and F. Itakura, "A linear space representation of language probability through svd of n-gram matrix," in *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, vol. 86, 2003, pp. 61–70.

[27] T. Rishen, L. Perkins, S. Yenduri, F. Zand, and S. Iyengar, "Augmentation of a term/document matrix with part-of-speech tags to improve accuracy of latent semantic analysis," *WSEAS Transactions on Computers*, vol. 5, no. 6, pp. 1361–1366, 2006.

[28] D. Klakow and J. Peters, "Testing the correlation of word error rate and perplexity," *Speech Communication*, vol. 38, no. 1-2, pp. 19 – 28, 2002.

[29] R. Kneser and H. Ney, "Improved clustering techniques for class-based statistical language modelling," in *Proceedings of the European Conference on Speech Communication and Technology*, 1993, pp. 973–976.

[30] T. Hofmann, "Probabilistic latent semantic analysis," in *In Proc. of Uncertainty in Artificial Intelligence, UAI'99*, 1999, pp. 289–296.

[31] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, p. 2003, 2003.

Welly Naptali received his B.E. of Mathematics degree from Bandung Institute of Technology, Indonesia, in 2004 and an M.E. of Informatics degree from Toyohashi University of Technology, Japan, in 2008. He is now a Ph.D. student at Toyohashi University of Technology.

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 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.