CSLT TECHNICAL REPORT-20160003 [Saturday 6th February, 2016]

A study of Similar Word Model for Unfrequent Word Enhancement in Speech Recognition

Xi Ma[1,3], Dong Wang[1,2*] and Javier Tejedor[4]

*Correspondence: wang-dong99@mails.tsinghua.edu.cn
[1]Center for Speech and Language Technology, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China
Full list of author information is available at the end of the article

Abstract

abstract here..

Keywords: speech recognition; semantic similarity; language model

The popular n-gram language model (LM) is weak for unfrequent words, particularly words that are out of vocabulary (OOV) or out of language (OOL). Conventional approaches such as class-based LMs pre-define some sharing structures (e.g., word classes) to improve the probability estimation for unfrequent words. However, defining such structures requires prior knowledge, and the context sharing based on these structures is generally inaccurate.

This paper presents a novel similar word model to enhance unfrequent words. In principle, this approach enriches the context of an unfrequent word by borrowing context information from some 'similar words'. Compared to conventional class-based methods, this new approach offers a fine-grained context sharing that enhances each word by borrowing context information from words that match the target word best. Additionally, the new approach is highly flexible since no sharing structures need to be defined prior to LM training. We tested the proposed method with a large-scale Chinese speech recognition system. The experimental results demonstrated that the similar word approach can improve performance on unfrequent words significantly, while keeping the performance on general tasks almost unchanged.

1 Introduction

The language model (LM) plays a crucial role in automatic speech recognition (ASR), by regularizing the decoder to select the most rational word sequences [1,2]. The most popular language modeling approach is based on n-grams, which relies on statistics of word co-occurrences to estimate the probability of a word within a particular short history [3]. Due to its simplicity and efficiency, n-gram LMs have been widely used in modern large-scale ASR systems [4, 5]. There are many free-available tools that can be used to build and manipulate n-gram LMs, e.g., SRILM [6], MITLM [7], IRSTLM [8] and HTKLM [9].

A long-standing problem of the n-gram LM is that unfrequent words cannot be well modeled, especially words that are out-of-vocabulary (OOV) and out-of-language (OOL). For these words, the training text involves little or even no occurrences of the word tokens and their contexts, which makes estimating the n-gram

probabilities difficult. A simple way to deal with this problem is to collect more data and retrain the model; this approach, however, is costly in both time and budget.

Another well-known approach involves various smoothing techniques [10, 11]. In the back-off and interpolation methods [12, 13], probabilities of unfrequent words are estimated with more general contexts (shorter history), and in the discounting method [10, 13], a small probability mass is re-distributed to unfrequent words. Further analysis showed that the smoothing techniques can be derived from a more general Bayesian treatment [14, 15]. Although allowing unfrequent words recognized with certain possibilities, these methods do not consider any semantic and syntactic properties of the target words and therefore the probability estimation is not accurate. Moreover, once the LM has been trained, adding new words is quite difficult, if not impossible.

Another popular approach to dealing with unfrequent words is to define some sharing structures when training LMs. The class-based LM [16–20] belongs to this category, where the main LM models frequent words while unfrequent words are addressed by adding into suitable classes. By this approach, all the words in a particular class share the same token and context in the main LM, and the probability of an unfrequent word is factorized into the probability of the class token in the main LM and the conditional probability of the word in the class. A clear advantage of this approach is that new words can be easily added into classes, and the probabilities of words in a class can be easily adjusted. The disadvantage is that the classes need to be pre-defined, and once the model has been trained, adding new classes is almost impossible. Additionally, for those words that cannot be classified into any class, this approach does not work. The class-based LM can be extended to various subgraph approaches [21–23], where the sharing structure is a nested grammar. A special case of the subgraph LM is the hybrid model proposed in [24], where the subgraph is a free phone loop and is used to recognize OOV words.

The last approach to dealing with unfrequent words is to build LMs based on subword units. For instance, characters for English [25], syllables for Chinese [26] and morphologies for Arabic [27], Turkish [28] and Uyghur [29]. This approach is powerful in dealing with OOV words even without specifying their spelling forms, and so is especially useful for agglutinative languages where words can be composed freely by concatenating rich suffixes. Unfortunately, it is difficult to enhance words that need particular emphasis.

Recently, neural LM has gained much attention [30,31]. In contrast to conventional n-gram models that treat each word as an individual discrete symbol, neural LMs represent words as low-dimensional real-valued vectors, and a neural network is used to predict the probability of a word given its history, within the low-dimensional vector space. A fundamental change associated with the vector representation is that similarity among words becomes measurable, which 'allows each training sentence to inform the model about an exponential number of semantically neighboring sentences' [30], leading to a highly compact model which is naturally smooth for unfrequent words. However, training a large scale neural LM is still challenging, and it is not simple to enhance particular words, due to the highly compact neural network structure.

1.1 Motivation

In this paper, we present a flexible and efficient method to enhance unfrequent words, including low-frequency words and OOV/OOL words. The basic idea is that for any word, there must be some words that are similar to it, in terms of either semantic or syntactic similarity. These *similar words* can provide rich context information for the target word so that can be used to estimate the missing context. In fact, if the human language is considered as a concept system governed by some grammars, as argued by the formal symbolic theory proposed by Chomsky [32], any word can find its similar words: in terms of either representing similar concepts or functioning similar roles. This is particularly true for unfrequent words which are mostly name entities and represent variants of concepts that have been well described by other commonly seen words.

We study in this paper how similar words can be used to enhance unfrequent words. Specifically, for a given unfrequent word, some similar words are identified and the contexts of the similar words are *borrowed* to enrich the contexts of the target word. For instance, the word ***Begali*** is rarely observed in training data, but we know that it is a company manufacturing electromechanical devices, and therefore its context might be similar to some famous companies such as ***GE*** or ***Siemens***. This forms a set of similar words {***GE,Siemens***}, by which the n-gram context of ***Begali*** can be enriched by duplicating the contexts of ***GE*** and ***Siemens***.

1.2 Contribution

In previous studies, we have empirically demonstrated that using similar words can improve speech recognition on low-frequency and OOV words [33], and this method can be even extended to deal with words that are in another language (OOL words) [34]. Although promising, these studies are far from ideal. First of all, no theoretical foundation was established for the similar word approach; secondly, the similar words are manually defined, which is not optimal in any sense and is not well-scalable.

The contribution of the present paper is three-fold: (1) we establish a solid theory for the similar word approach, which enables a deep understanding for this method and finds more space to improve; (2) we propose a novel approach to select similar words based on word embedding, a method that has recently attained much success in natural language processing; (3) we conduct a set of experiments based on a large-scale ASR system to validate the proposal.

The rest of this paper is organized as follows. Section 2 discusses some related work, and Section 3 presents the theory of the similar word model. The similar word selection approach based on word embedding is presented in Section 4, which is followed by the implementation details in Section 5. Section 6 presents the experiments and Section 7 concludes the paper.

2 Related work

This work is related to several research directions including smoothing techniques, OOV treatment, LM adaptation, and word embedding. Among these directions, perhaps the most relevant work is various dynamic LMs, whose purpose is to add new words and/or re-weight probabilities of some specific words so that the LM can

be quickly adapted to a particular domain. A major challenge is that the existing LM is a large-scale general model that should keep stable, and any adaptation should be light-weighted in processing cost and does not impact performance of the model on general tasks.

A popular dynamic LM approach is the class-based LM [16]. This approach clusters similar words into classes where the probabilities of the members can be freely adjusted and new words can be added dynamically. It has been reported that class-based LMs may deliver better performance than word-based LMs on low-frequency words [19]. In speech recognition, class-based LMs are often implemented in the form of finite state transducers (FST) [35, 36], where the word LM (including tokens representing classes) is compiled to a *host FST* and the classes are compiled into *nested FSTs*. The host and nested FSTs are then composed into a complete decoding graph. This composition is highly efficient and can be performed online. Similar approaches were proposed in [21–23], where various dynamic composition methods were proposed, and simple word classes were extended to any complicated grammars. In a previous study [37], we investigated this FST-based word class composition and found that it did improve performance for unfrequent words, even if the n-gram contexts around the nested FSTs are simply ignored.

The similar word approach presented in this paper is similar to the class-based LM in the sense that both enhance unfrequent words by context sharing. The difference, however, is also obvious: in the similar word method, this sharing is word-specific, which means that every unfrequent word has its unique class of words to share (the set of similar words). Additionally, the similar word approach does not need to pre-define any class – it is based on a regular word-based LM without any class token involved, and all the sharing is performed on-the-fly based on a similar pair model (see Section 4). This leads to a very flexible framework: it does not require any specific LM re-training and can deal with any word even though its class is hard to identify.

This work is also closely related to the neural LM approach [30, 31]. These approaches rely on word similarity to enhance context sharing. The difference is that the sharing scheme of neural LMs is a very compact model due to its layer-wised neural structure, which leads to strong co-adaptation among words and reduced flexibility when adjusting probabilities for particular words. The similar word approach does not suffer from this problem, as it is based on n-gram models which involve little co-adaptation.

Note that this work is an extension of our previous studies [33,34]. In those studies, we experimentally demonstrated that the similar word approach can be successfully applied to enhance unfrequent words, even those words in another language. This paper extends those studies by establishing the theory background for this approach and proposing a similar word selection method based on word embedding.

3 Similar word model

This section establishes the theory background for the similar word method. Briefly, we derive a Bayesian formulation that treats vocabulary words as latent variables and factorizes the probabilities of unfrequent words into two components that are modeled by the original LM and a word pair model respectively. The similar word approach is then interpreted as marginalizing over the latent vocabulary words.

3.1 Context sharing in class-based LM

Let $\tilde{w}$ represent an unfrequent word for which many realistic contexts are missing in the training data. Our goal is to estimate the probability $P(\tilde{w}|C_k)$ of the word at any possible context C_k, where k indexes the context. With an n-gram LM, C_k is a word history, i.e., a sequence of words whose length is less than n. Estimating $P(\tilde{w}|C_k)$ can be rewritten as:

$$\begin{aligned} P(\tilde{w}|C_k) &= \sum_v P(\tilde{w}, v|C_k) && (1)\\ &= \sum_v P(v|C_k)P(\tilde{w}|v, C_k) && (2)\\ &= \sum_v P(v|C_k)P(\tilde{w}|v), && (3) \end{aligned}$$

where v is an 'information variable' that has been well modeled by the LM via $P(v|C_k)$, and is closely related to the target word $\tilde{w}$ where the relation is modeled by the conditional probability $P(\tilde{w}|v)$. Note that we have assumed that $\tilde{w}$ is independent of C_k conditioned on v, which simplifies the model and is reasonable if the target word is known little by the LM.

It should be highlighted that (3) is a general form and the information variable v can be defined in any arbitrary way. Specifically, if it represents a word class, the above formulation defines the class-based LM. In this case, the class token v generally occurs often and hence the class probability $P(v|C_k)$ can be readily estimated. Additionally, the class-member probability $P(\tilde{w}|v)$ can be either defined manually or estimated by a simple statistical model (e.g., unigram). Since both the two components can be well estimated, the formulation (3) results in a robust estimation for $P(\tilde{w}|C_k)$, i.e., the probability for the unfrequent word $\tilde{w}$, even if though is never seen in the training data.

The principle behind the model (3) is that the unfrequent word $\tilde{w}$ shares context of v through $P(v|C_k)$. For the class-based LM, all the words in the same class share the same context $P(v|C_k)$, which is often too aggressive. For example, ***Siemens*** and ***Goldman Sachs*** are both company names, but they are in different domains, and therefore should appear in different word contexts. The consequence of sharing the diverse companies with the same class is that the context of the class ***company name*** is over rich, leading to a high perplexity for the words in the class. A natural idea is to define fine-grained classes, e.g., 'electronic companies' and 'financial companies', but this will cause much human effort, and the manual definition is not necessarily accurate and optimal.

3.2 Similar word model

Referring to the formulation (3), if the information variable v is defined as the regular words in the LM vocabulary $\mathcal{D}$, the model becomes even simpler:

$$\begin{aligned} P(\tilde{w}|C_k) &= \sum_{w\in\mathcal{D}} P(\tilde{w}, w|C_k) && (4)\\ &= \sum_{w\in\mathcal{D}} P(w|C_k)P(\tilde{w}|w), && (5) \end{aligned}$$

where $P(w|C_k)$ is just the LM, and $P(\tilde{w}|w)$ is a *word pair model* which models the similarity between the target unfrequent word $\tilde{w}$ and any vocabulary word w. Note that (5) is a Bayesian formulation that treats w as a latent variable and $P(w|C_k)$ as a prior distribution over w. The probability estimation for the unfrequent word $\tilde{w}$ is cast to marginalizing the product of the prior and the word-pair probability over the latent vocabulary word w. We refer to the formulation (5) as the *similar word model* in this paper.

Compared to the class-based LM, this new formulation holds several advantages. Firstly, no classes need to be manually pre-defined, which saves much human effort. Secondly, $P(w|C_k)$ has been modeled by the original LM, so re-training for a 'host LM' is not required. Thirdly, the word pair model $P(\tilde{w}|w)$ defines a word-specific context sharing that is more accurate than class-based sharing. For example in the case of ***Siemens*** and ***Goldman Sachs***, ***Siemens*** is more similar to ***Samsung*** while ***Goldman Sachs*** is more similar to ***Morgan Stanley***. By defining similar words, ***Siemens*** shares contexts with ***Samsung*** and ***Goldman Sachs*** shares contexts with ***Morgan Stanley***, which is certainly more accurate than putting them in a single 'company name' class.

We highlight that even with very fine-grained classes, the class-based LM can hardly beat the similar word model. This is not because defining detailed classes is costly and prone to errors, but lies in the fact that any explicitly defined class leads to inaccurate context sharing. The similar word model does not define any class; instead, it models the context by *all* the vocabulary words, where the contribution of each word is specified by the word pair probably $P(\tilde{w}|w)$. This can be regarded as an extreme case of the class-based LM where a unique class is constructed for each particular target word. Although the class members of different target words are almost the same, the probability distribution over the members is derived from $P(\tilde{w}|w)$, so the context sharing is unique for each word. This particular definition of classes avoids the awkward context over-sharing with the conventional class-based LM.

For a realistic LM, only some of the vocabulary words gain significant probability $P(w|C_k)$ for a particular context C_k. This subset of words is denoted by $\mathcal{D}(C_k)$. Additionally, for a particular target word $\tilde{w}$, only a small set of words results in significant word pair probabilities $P(\tilde{w}|w)$. These words form a *similar word set* of $\tilde{w}$, denoted by $\mathcal{S}(\tilde{w})$. For example, $\mathcal{S}($***Siemens***$)=\{$***GE, Samsung, Nokia***$\}$ and $\mathcal{S}($**Goldman Sachs**$) = \{$ ***Morgan Stanley, Merrill Lynch***$\}$. In practice, $\mathcal{S}(\tilde{w})$ can be selected by either placing a threshold on $P(\tilde{w}|w)$ or choosing the n-best words according to $P(\tilde{w}|w)$. By this definition, the similar word model (3) is simplified by considering only the similar words that appear in the context C_k, given by:

$$P(\tilde{w}|C_k) \quad = \quad \sum_{w \in \mathcal{D}(C_k) \cap \mathcal{S}(\tilde{w})} P(w|C_k) P(\tilde{w}|w). \tag{6}$$

4 Word pair model by word embedding

The central task of the similar word approach presented in the previous section is to estimate the word pair model $P(\tilde{w}|w)$, where $\tilde{w}$ is the target unfrequent word

and w is any vocabulary word in the original LM. Once this has been obtained, the probability of the target word $P(\tilde{w}|C_k)$ can be easily computed following (6). In principle, the word pair model $P(\tilde{w}|w)$ can be estimated by any approach using any data resource, which offers great flexibility and is a major advantage of the similar word model. We start from describing the manually defined word pair model that was used in our previous studies, and then present a novel model based on word embedding.

4.1 Manually defined word pair model

In our previous studies [33, 34], the similar word set $S(\tilde{w})$ was manually defined for each word $\tilde{w}$ according to some prior knowledge (e.g., ***Siemens*** is similar to ***Samsung***), and the members in the similar word set are treated equally. This corresponds to define a binary $P(\tilde{w}|w)$, given by:

$$P(\tilde{w}|w) = \begin{cases} 1 & \text{if } w \in S(\tilde{w}) \\ 0 & \text{if } w \notin S(\tilde{w}) \end{cases}.$$

This approach, although very simple, is highly effective in our study, attributed to the rich knowledge associated with the word pairs defined by people. The disadvantage is also evident: the probability value is binary so the relative importance of the similar words cannot be represented; moreover, the similar words need to be defined by hand, which is not feasible for large scale tasks. These shortcomings can be addressed by the word embedding technique in an elegant way, as presented in the next section.

4.2 Word embedding

The word pair model $P(w_i|w)$ describes the relevance of the target word $\tilde{w}$ and the vocabulary word w, and hence learning $P(\tilde{w}|w)$ can be cast to learning the relevance of the two words. A very powerful approach that can be used to learn word relevance is based on various word embedding algorithms.

Word embedding represents words as low-dimensional continuous *word vectors* so that relevant words are located close to each other in the vector space. The 'relevance' could be measured in various senses, including semantic meanings, syntactic roles, sentimental polarities, or any others depending on the model objectives [38–40]. Compared to the conventional categorical or one-hot representations, word vectors posses significant advantage in semantic representation, model training and inference, as well as generalizability across domains and languages. Word embedding has attained remarkable success in a multitude of text processing tasks [41, 42].

A popular word embedding approach is based on the skip-gram model proposed by Mikolov, which can be used to learn word vectors from raw text without resorting to any external knowledge [43]. Briefly, the model tries to optimize vectors of the words in such a way that a word is close to its context words measured by the inner product of their vectors. This model can be seen as a neural network where the input is the one-hot representation of the focused word w, denoted by e_w.

This one-hot input is projected to its word vector c_w, by looking up an embedding matrix U. This word vector, c_w, is then used to predict the vectors of its left and right C neighboring words. Given a word sequence $w_1, w_2 ... w_N$, the training process maximizes the following objective function with respect to the embedding matrix U which is in fact composed of vectors of all the words in the vocabulary:

$$\mathcal{L}(U) = \frac{1}{N} \sum_{i=1}^{N} \sum_{-C \leq j \leq C, j \neq 0} logP(w_{i+j}|w_i), \tag{7}$$

where

$$P(w_{i+j}|w_i) = \frac{\exp(c_{w_{i+j}}^T c_{w_i})}{\sum_w \exp(c_w^T c_{w_i})}. \tag{8}$$

Note that the denominator of (8) involves summation over all possible words so the computation is costly. Efficient algorithms such as hierarchical softmax and negative sampling have been proposed [38]. With these efficient algorithms, embedding 150,000 words with a training text involving 1000 million tokens can be completed in two hours.

4.3 Word pair model based on word embedding

Looking at the objective function (7) of the skip-gram model, it can be seen that the optimization goal of the word embedding is just learning the word pair probabilities required by the similar word model. This means that the two models (skip-gram and similar word models) are fully consistent, and the word vectors inferred by the skip-gram model can be readily used to compute the word pair probability $P(\tilde{w}|w)$, following the formulation in (8).

In practice, we firstly train the skip-gram model and infer the vectors of all the words in the training data, except those words that occur very rare (less than 5 times in our experiments). The data used for training the LM can be used to train the skip-gram model, though any data is fine. Particularly, if there are many new words to enhance, it would be good to collect some additional data that involve these new words so that their vectors can be reliably inferred.

For an unfrequent word $\tilde{w}$, if it has been embedded in the previous step, there would be no difficulty to enhance its probability by (6) and (8). If $\tilde{w}$ is a new word and has not been embedded, its word vector has to be firstly derived. A simple way is to average the vectors of the words in the left and right contexts of the target word. This method is efficient in computation and is consistent to the skip-gram objective function: the training objective is to let the target word predict the context words, so the best estimation of the vector of a word is the mean vector of its context words.

As mentioned already, in order to save computation, only a limited set of similar words is considered in the similar word model. In our study, the n-best approach is adopted to select the similar words, formulated by:

$$\mathcal{S}(\tilde{w}) = \{w_1, w_2, ...w_n; \quad w_j^T \tilde{w} \geq w_k^T \tilde{w}, for \quad \forall j < k\}. \tag{9}$$

It should be noted that there are diverse embedding methods, e.g., [30,31,38–40]. Any of them can be used to compute the word pair probabilities as long as the word relevance can be easily derived from the resultant word vectors. In fact, in our study the negative sampling approach was used for the sake of fast training [43]. Since a slightly different objective function is used, the word pair probabilities are not perfectly computed by (8); however, we found in experiments that the simple exponential form in (8) provides a good approximation.

Moreover, the word embedding methods described above are mostly based on raw text. In fact, the embedding approach can be extended to exploit various information resources, particularly knowledge graphs such as WordNet or Freebase, e.g., [44–46]. Compared to raw text, knowledge graphs provide cleaner information, and hence may produce word vectors with higher quality. This is particularly useful for words that cannot find sufficient occurrences in the raw text. A shortage of knowledge-graph embedding, however, is that the word coverage is often limited. Recent research shows that knowledge graphs and text data can be combined to deliver better embedding [47–50].

Although word embedding is a good choice for the word pair model, we highlight that any model or knowledge form is valid, as long as it can estimate $P(\tilde{w}|w)$ in a reasonable way. This reveals a big advantage of the similar word approach: it can utilize rich information from heterogeneous resources and use very flexible rules and models. In contrast, the class-based LM is largely limited by the information they can utilize, e.g., word unigrams.

A particular advantage of the similar word approach associated with the capability of utilizing heterogeneous information is that it can deal with foreign words in a straightforward manner. In principle, any foreign word can find a set of similar words in the host language through a dictionary or machine translation [51]. Therefore the similar pair approach described in this section can be simply applied. We will demonstrate in Section 6 that the similar word approach can deal with foreign words fairly well.

5 FST-based Implementation

In practice, several alternative ways can be taken to implement the similar word model. For example, it can be used to enrich the conventional n-gram LM by updating the probabilities of the target words and adding new n-grams if necessary. Another way is to change the decoder so that new probabilities of the target words can be picked up when propagating partial paths. In this study, we choose an FST-based implementation, which is clear in concept, easy to implement, and simple to be incorporated in modern ASR systems, as most of them are based on the FST framework.

5.1 Finite state transducer

Formally, a finite state transducer (FST) is defined as a 6-tuple $(Q, \delta, \Sigma, \Gamma, l, F)$, where Q represents states, and δ represents transitions between states. Σ and Γ are the set of input and output symbols associated to the transitions, respectively. The set of initial states is denoted by l, and the set of final states is denoted by F. An FST can be represented as a graph, where the nodes represent states and

arcs represent transitions [52]. In speech and language processing, weights are often assigned to transitions and final states so the cost of traversing in the graph can be computed, which leads to a weighted FST [53–55]. All the FSTs we mention in this paper are weighted. A simple FST graph used to convert phones to words is illustrated in Fig 1.

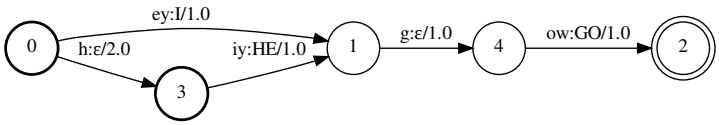

Figure 1: An FST example for converting phone sequences to word sequences.

FST is a powerful representation for probabilistic sequential models. In fact, each path of an FST describes a 'generation' process which accepts an input sequence and produces an output sequence, for which the probability is read from weights of the transitions on the path. It is therefore feasible to associate the probability predicted by a probabilistic model to the weights of the transitions of an FST, so that the FST becomes an equivalent representation of the probabilistic model. For example, an n-gram language model can be converted to an FST where a state represents a particular word history and each transition represents a possible subsequent word that is specified by the output symbol[1]. The weights associated to the transitions are then read from the n-gram LM, with some transformation applied if necessary, e.g., negative logarithm for FSTs defined in the log semiring [53, 56].

5.2 FST-based speech recognition

Modern speech recognition relies on various statistical models, including hidden Markov models (HMMs), lexica, decision trees, and n-gram LMs. All these models can be converted to FSTs, and each FST represents the correlation between a low-level input sequence (e.g., phones) and a high-level output sequence (e.g., words). More importantly, the FSTs that represent different levels of statistical models can be composed to form a single compositional FST that converts the primary input (e.g., HMM states) to top-level output (e.g., sentences). The compositional FST can be further optimized by standard FST operations, including determinization, minimization, and weight pushing. This produces a very compact and efficient graph that enables very simple and fast decoding. This FST-based architecture has been widely adopted by most modern ASR systems.

A typical graph building process was described in [55] and it can be roughly represented as follows:

$$\pi_{\epsilon}(min(det(H \circ (det(C \circ det(L \circ G))))), \tag{10}$$

[1]In fact, an LM FST is essentially a finite state acceptor (FSA) where the input and output symbols are identical.

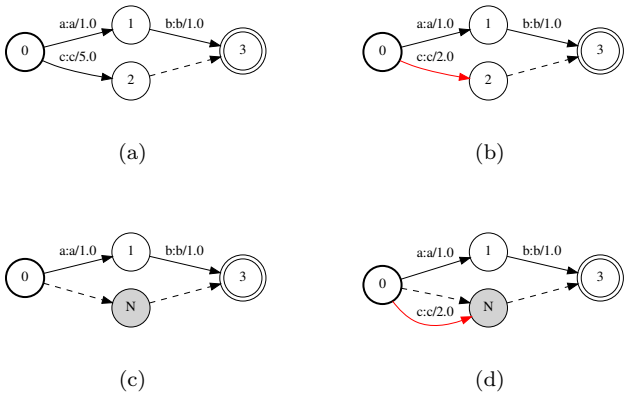

Figure 2: Naive word enhancement, where the four pictures show the LM FST (a) before enhancement (target word exists); (b) after enhancement (target word doesn't exist); (c) before enhancement (target word not exists); (b) after enhancement (target word does not exist).

where H, C, L, and G represent the HMM, the decision tree, the lexicon and the LM (or grammar in grammar-based recognition) respectively, and $\circ$, '*det*', and '*min*' denote the three FST operations: composition, determinization, and minimization. Note that in order to deal with ambiguity (e.g., that caused by homophones), some special symbols are introduced in the building process. The operation π_ϵ removes these symbols once the graph has been built.

5.3 Similar word model with LM FST

The FST framework provides a simple way to implement the similar word approach. Looking back to the similar word model (6), for each target word $\tilde{w}$, the enhance process involves updating the weights according to the probability $P(\tilde{w}|C_k)$ for every context C_k. With the FST representation, a particular context C_k corresponds to a state in the LM FST, and therefore the enhancement process can be simply obtained by traversing the FST graph, and for each state, updating the weight of the transition whose input/output symbol is the target word. For a back-off n-gram LM, this transition always exists if the target word is in the LM lexicon; otherwise, the target word needs to be added into the lexicon at first, and a new transition from the state C_k to the back-off state for the null context is created. These two scenarios are illustrated in Fig. 2, where c is the target word and a is one of its similar words. The shaded node labelled by 'N' denotes the back-off node for words without any left context. Dash lines denote multi-hop transitions.

A potential problem of the above implementation is that only the left context is considered. This is because the context C_k in (6) is assumed to be a word history in the n-gram LM. This is clearly not optimal as the right context is equally important but not involved in C_k. To remedy this shortage, one may assume that for every similar word w, the target word $\tilde{w}$ and w share the same right context. This can be achieved by creating a new state $C'_k(w)$ that resembles the out-going state of

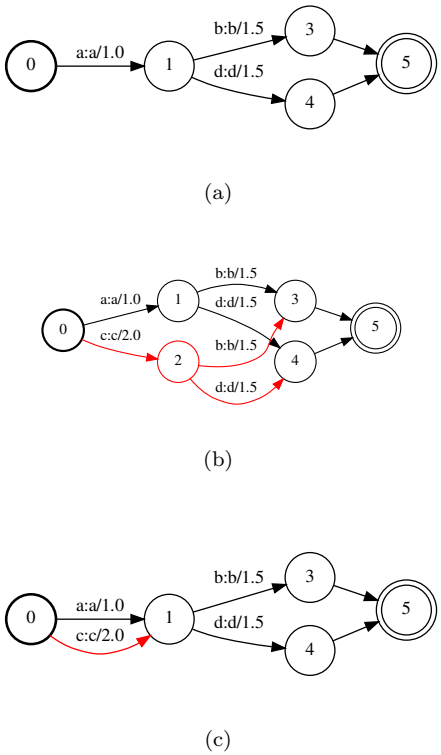

Figure 3: (a) LM FST before enhancement for c. (b) LM FST after enhancement with new states and transitions created for right-context sharing. (c) LM FST after enhancement by transition duplication.

the transition associated with the similar word w, and creating a new transition from C_k to $C'_k(w)$ with a weight obtained from the probability $P(w|C_k)P(\tilde{w}|w)$. Finally, we connect $C'_k(w)$ to all the sates that are involved in the right context of the similar word w. This is illustrated in Fig. 3, where (a) is the original LM FST, and (b) is the FST after enhancing word c. Note that this 'naive sharing' of the right context is not fully theoretically correct due to the nature of left-sharing of n-gram LMs. A possibly better way is to share states that are determined by some backward LMs [57], though it may complicate the FST manipulation. Nevertheless, this simple right-context sharing approach works pretty well in our study.

Interestingly, the right-context sharing approach leads to a highly efficient implementation. Referring to Fig. 3 (b), it can be found that the new created state can be simply merged to the existing state of the transition associated with the similar word a. The resultant FST is shown in Fig. 3 (c). This leads to a simple similar word enhancement algorithm which searches for all the transitions that are associated with a similar word, and then duplicates each transition with the input symbol changed to the target word and the weight updated according to $P(w|C_k)P(\tilde{w}|w)$. Note that adding multiple transitions with the same entry state and the same input symbol may result in an undeterminizable FST. We solve this problem by adding a disambiguity symbol for each duplication.

Considering that the FST is based on the log semiring, the weight for the duplicated transition for target word $\tilde{w}$ is given by:

$$\omega(C_k, \tilde{w}; w) = \omega(C_k, w) - lnP(\tilde{w}|w) - \theta, \tag{11}$$

where $\omega(C_k, \tilde{w}; w)$ represents the weight of the transition duplicated from similar word w, with C_k as its entry state and $\tilde{w}$ as its input symbol. The enhancement scale θ is used to control the strength of the enhancement: a large value leads to stronger enhancement.

The word pair probability $P(\tilde{w}|w)$ can be derived in various ways, though we focus on the word embedding approach given by (8) in Section 4. To simplify the computation, we approximate (8) by first selecting the similar word set $S(\tilde{w})$ following (9), and then computing $P(\tilde{w}|w)$ as follows:

$$P(\tilde{w}|w) = \frac{\exp(c_{\tilde{w}}^T c_w)}{\sum_{w' \in S(\tilde{w})} \exp(c_{\tilde{w}}^T c_{w'})}. \tag{12}$$

Note that for any word that is not in the similar word set, the word pair probability is zero and therefore the transitions labelled by this word do not need to be duplicated. It can be easily found that for any state C_k, after the transition duplication for all the similar words, the summation of the probabilities associated to all the duplicated transitions leads to exactly the same form given by (6).

It should be noted that the weight updating formulated by (6) is not well-defined since the values $P(\cdot|C_k)$ are not re-normalized after the update. This problem seems more evident with the empirical form (11) due to the introduction of the additional term θ. Re-normalization is not difficult; however, we note that the LM FST derived from a back-off n-gram LM is naturally unnormalized due to the back-off transitions. It is therefore not clear if the re-normalization is theoretically sound. In this study, we simply leave the weights unnormalized, which is easy and delivers reasonable performance anyway. The entire enhancement process is summarized in Algorithm 1, where the enhancement loops over all the transitions, with each transition represented by a tuple $(s, t, i : o/\omega)$, where s and t are respectively the entry and exiting states of the transition, and i, o, and ω represent the input symbol, output symbol and weight, respectively.

6 Experiment

We test the proposed similar word approach with a large-scale Chinese ASR system. In this section, the databases and configurations are first presented, and then the experimental results are reported.

6.1 Database

The experimental baseline is a practical Chinese ASR system built for transcribing conversations of a large call center. The domain is telecom service and the language is Chinese. The acoustic model (AM) was trained on $1,400$ hours of speech signals dumped from the call center, with the sampling rate set to be $8,000$ Hz and the

Algorithm 1 Similar Word Enhancement Algorithm

Require:
Input:
$\{\xi_j = (s_j, t_j, w_j : w_j/\omega_j) : j = 1, 2, ..., J\}$: transitions of LM FST derived from the original n-gram LM
$\{w^k : k = 1, 2, ..., K\}$: words to enhance
θ: enhancement scale
Ensure:
for $k = 1$ to K **do**
  Identify $S(w^k)$ according to (9)
  Compute $\{P(w^k|w); \quad \forall w \in S(w^k)\}$ according to (12)
  for $j = 1$ to J **do**
    if $w_j \in S(w^k)$ **then**
      Duplicate transition for w^k from ξ_j: $(s_j, t_j, w^k : w^k/\omega_j)$
      Update the weight of the transition to $\omega(s_j, w^k)$ according to (11)
    end if
  end for
end for

sample size set to be 16 bits. A general LM was trained on a large web database more than 2 terebytes, which was then interpolated with a domain-specific model trained with all the transcription of the training speech data and some posts dumped from web-based customer service. The lexicon of the LM involves $150,000$ words that were selected according to the unigram probability and the importance returned by a web search engine.

We compare the proposed similar word method with two implementations of the conventional class-based LM: one is based on automatically-derived classes (ADC) and the other is based on manually-defined classes (ADC). Since only part of the unfrequent words can be categorized to the manually defined classes, we use a smaller data set to test the MDC word-class LM. In summary, four test sets are used to in the evaluation, all of which are real-life recordings from the call center:

- CHINESE: Used to test monolingual word enhancement. It consists of $4,177$ utterances, each of which involves one or several unfrequent Chinese words. There are totally $1,161$ unfrequent Chinese words selected by hand.
- CHINESE-PART: A subset of CHINESE that consists of 418 utterances and covers 113 unfrequent words that can be categorized into two manually defined classes: 'device names' and 'actions'. This test set is mainly used to evaluate the MDC class-based LM approach.
- FOREIGN: Used to test multilingual word enhancement. It consists of 316 utterances and covers 32 unfrequent English words selected by hand.
- GENERAL: Used to test the general performance of the system. It involves $2,608$ utterances, each of which contains words in various frequencies, including 23 OOV words.

6.2 Word embedding

The word2vec toolkit provided by Google was used to train the word embedding model and infer the word vectors[2]. The training data was the same as that used to train the domain-specific LM. Note that all words in the text were embedded, except those occurring less than 5 times. Therefore the words not in the LM vocabulary may be covered by the embedding model. The training was based on the negative

[2]https://code.google.com/p/word2vec/

sampling algorithm [38], for which the context window was chosen to be 8, and the number of negative samples was set to 25. The dimension of the word vectors was set to 200.

6.3 Acoustic model training

The ASR system was based on the state-of-the-art HMM-DNN acoustic model, which represents dynamic properties of speech signals using the hidden Markov model (HMM), and represents static properties by the deep neural network (DNN) model. The feature was based on 40-dimensional FBanks. An 11-frame splice window was used to concatenate neighboring frames to capture long temporal dependence. The linear discriminative analysis (LDA) was applied to reduce the dimensionality of the concatenated feature vectors to 200.

The Kaldi toolkit [58] was used to train the HMM and DNN models. The training process largely followed the WSJ s5 GPU recipe published with Kaldi. Specifically, a pre-DNN system was firstly constructed based on the Gaussian mixture model (GMM), and this system was used to produce phone alignment for the DNN training.

The DNN model involved 4 hidden layers, each consisting of $1,200$ hidden units. The output layer consisted of $8,000$ output units, corresponding to the tided probability density functions (pdf) derived from context-dependent decision trees.

6.4 Language model training

As mentioned already, the LM for the baseline system was constructed by interpolating a general large model and a domain-specific model. These two models are all based on word 3-grams and were trained with the SRILM toolkit[3], where the modified Kneser-Ney discounting was employed for smoothing. The model was pruned by a threshold 1e-7 on the n-gram probability. The Kaldi toolkit was used to convert the 3-gram LM to an LM FST, and the openFST toolkit[4] was used to optimize the LM FST and compose it with other low-level FSTs.

Two class-based LMs were constructed. The MDC class-based LM relies on manually defined classes. We defined two classes: device names and actions, which cover 46 and 67 unfrequent words respectively. For the two classes, 182 and 198 class-representative words were selected respectively. With the original LM in the form of FST, the class-based LM is derived by duplicating the transitions of these representative words and labeling the new transitions by the tags of the corresponding classes. This is roughly equivalent to duplicating and relabeling the training text (changing the unfrequent words to appropriate class tags) and then rebuild the LM.

The ADC class-based LM employs a k-mean algorithm to cluster all the words (both INV and OOV) into 100 classes, based on cosine distance of word vectors. Among the 100 classes, 45 classes involve unfrequent words. For each of them, moderate-frequency words were selected as class-representative words and were used to derive the class-based LM. In our experiments, we found words whose unigram PPL between 300 and 400 (corresponding to the unigram probability from 0.0025 to 0.0033) are good class representatives.

[3] http://www.speech.sri.com/projects/srilm/
[4] http://www.openfst.org

6.5 Monolingual test

In this section, we study enhancement for Chinese unfrequent words. Since Chinese is a character-based language, the ASR performance is evaluated in terms of character error rate (CER). An obvious problem of CER is that unfrequent words take only a small proportion in the test data, and so CER is not sensitive to the performance of unfrequent words. To have a better evaluation for effectiveness of the enhancement methods, we define the name entity error rate (NEER): the proportion of the unfrequent words that are correctly recognized. In contrast to CER that measures the accuracy on all words, NEER evaluates the accuracy on focused words, i.e., the unfrequent words need to enhance.

6.5.1 Experiments on CHINESE

The first experiment is conducted on the CHINESE test set. The baseline performance without any enhancement on this set is CER=32.65% and NEER=70.03%. With the similar word enhancement applied, the CER and NEER results are presented in Table 1 and Table 2. For a more clear representation, the same information is presented in Fig. 4 and Fig. 5, where we experimented with various settings of the number of similar words, denoted by $SimNum$, as well as the enhancement scale θ. For a particular $SimNum$, the required number of similar words are selected according to the cosine distance between the vectors of the target and the candidate words.

From Fig. 4 and Fig. 5, it can be seen that with an increasing θ, the performance is improved in both CER and NEER. If θ continues increasing, NEER keeps decreased but CER starts to increase. This is understandable as a large θ tends to recognize any words as unfrequent words incorrectly. Additionally, it can be seen that considering more similar words improves both CER and NEER. This is also not surprising, as more similar words provide more rich context information for unfrequent words. As discussed in Section 3, the similar word set can be as large as the entire lexicon, however in practice, too many similar words may cause bias towards unfrequent words particularly if θ is large, as demonstrated in Fig. 4. Roughly, a reasonable configuration is $\theta = 5$ and $SimNum = 7$. A MAPSSWE test by the NIST sclite tool[5] shows that with this configuration, the improvement with the similar word approach is statistically significant on both CER and NEER. Most strikingly, with a cost of 3% CER degradation, NEER can be reduced to 0. This demonstrates that the similar word approach is highly effective.

A careful analysis shows that the enhancement contributes differently for different words. More worse an unfrequent word is recognized with the original LM, more improvement is obtained by the similar word enhancement. This nice property can be attribute to the characteristics of Chinese: in Chinese any word can be split into a character sequence, and therefore some unfrequent words may find their character sequences in the original training data. These words tend to be recognized well with the original LM and so the enhancement does not contribute much. In contrast, for the words that are truly unfrequent, i.e., their character sequences are not well represented by the original LM, the enhancement can provide valuable context information and so contributes more significantly.

[5]http://www1.icsi.berkeley.edu/Speech/docs/sctk-1.2/sclite.htm

	CER%						
θ \ $SimNum$	1	2	3	4	5	6	7
$-\infty$ (Baseline)	26.69						
-3	26.69	26.68	26.67	25.83	25.83	25.83	25.8
-2	26.67	26.67	26.67	25.57	25.56	25.56	25.51
-1	26.67	26.65	26.52	25.42	25.42	25.41	25.37
0	26.46	26.46	26.44	25.24	25.23	25.23	25.19
1	26.4	26.42	26.39	25.1	25.09	25.08	25.03
2	26.33	26.35	26.28	24.93	24.95	24.96	24.93
3	26.21	26.24	26.15	24.7	24.73	24.75	24.72
4	26.11	26.19	26.03	24.51	24.53	24.53	24.51
5	26	26.18	26.18	**24.34**	24.36	24.4	24.4
6	25.43	25.59	25.65	25.76	25.85	25.88	25.88
7	25.42	26.02	26.07	26.09	26.17	26.21	26.35
8	25.52	26.42	26.93	26.98	27.12	27.24	27.32
9	26.05	27.07	28.25	29.33	29.63	29.86	29.97

Table 1: CER results on the CHINESE test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

	NEER%						
θ \ $SimNum$	1	2	3	4	5	6	7
$-\infty$ (Baseline)	70.03						
-3	69.88	69.88	69.81	62.33	62.33	62.33	61.87
-2	69.78	69.72	69.67	60.35	60.38	60.33	59.89
-1	69.61	69.54	68.49	58.8	58.78	58.73	58.25
0	68.01	67.85	67.48	57.24	57.18	57.13	56.69
1	66.91	66.6	65.97	55.18	55.09	55	54.48
2	65.31	64.94	64.04	51.94	51.85	51.61	51.21
3	63.16	62.7	60.77	47.57	47.46	47.11	46.76
4	59.83	59.61	57.18	42.95	42.68	42.3	42.11
5	55.84	56.6	55.44	38.06	37.84	37.22	37.33
6	49.15	49.15	48.25	37.86	37.57	36.84	36.71
7	45.97	45.41	45.36	24.9	23.74	23.69	23.69
8	41.87	41.61	36.87	9.38	8.92	7.64	6.4
9	36.96	25.67	16.15	**0**	0	0	0

Table 2: NEER results on the CHINESE test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

Next we compare the similar word approach with the ADC class-based LM approach, where the classes are generated automatically by word vector clustering. As θ in the similar word approach, an enhancement scale β is introduced in the ADC class-based LM to control the prior of entering the class tags. The CER and NEER results with this enhancement are presented in Table 3 and Table 4 respectively. The same information is presented as figures in Fig. 6 and Fig. 7.

β	CER%
$-\infty$ (Baseline)	26.69
-8	**26.69**
-6	26.69
-4	26.74
-2	26.78
0	26.96
2	27.43
4	28.9

Table 3: CER results on the CHINESE test set with the clustering class-based method. β is the enhancement scale.

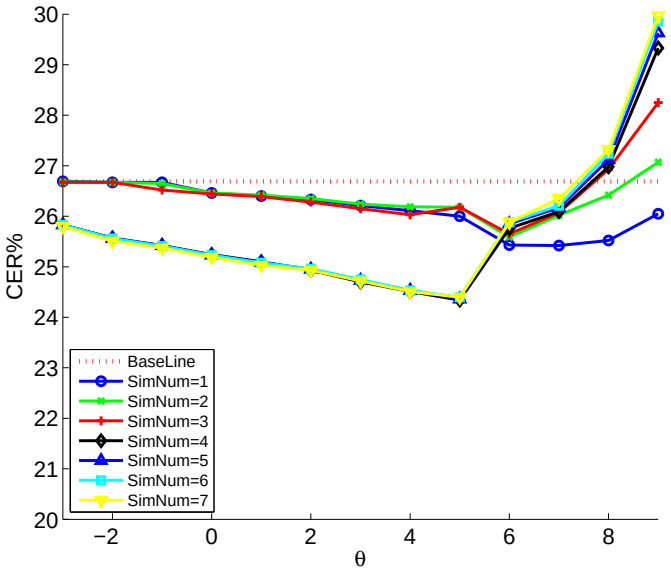

Figure 4: CER results on unfrequent words for the CHINESE test set with the similar word enhancement.

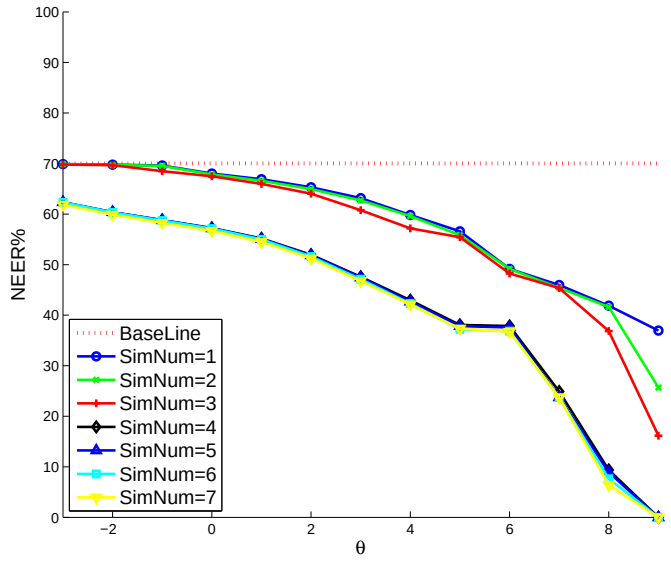

Figure 5: NEER results on unfrequent words for the CHINESE test set with the similar word enhancement.

For a clear comparision, we set $simNum$=7 in the similar word approach, and plot the CER and NEER curves with the two approaches in the same picture, by shifting θ in the similar word approach by 3 units to match β in the ADC class-based LM approach. The results are shown in Fig. 8. It is clear that with the same

β	NEER%
$-\infty$ (Baseline)	70.03
-8	70.03
-6	70
-4	69.83
-2	69.39
0	68.34
2	65.4
4	**55.38**

Table 4: NEER results on the CHINESE test set with the artificial class-based method. β is the enhancement scale.

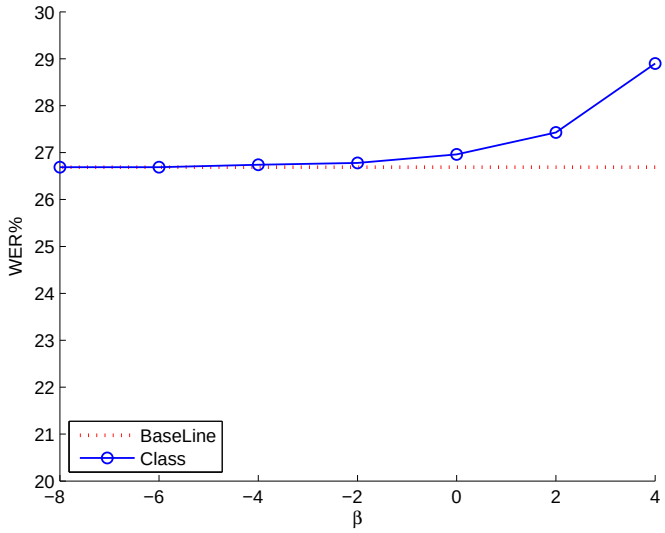

Figure 6: CER results on unfrequent words for the CHINESE test set with the clustering class-based method. β is the enhancement scale.

cost in CER, the NEER reduction with the similar word approach is much more remarkable than with the ADC class-based LM approach. Particularly, with the similar word approach, CER and NEER can be improved simultaneously if θ is small, which is impossible with the ADC class-based LM approach. A MAPSSWE test shows that with all the settings of θ/β shown in Fig. 8, the advantage on NEER with the similar word approach compared to the ADC class-based LM approach is statistically significant.

Finally, we examine the impact of the two enhancement approaches to the general performance of our ASR system. The test is conducted on the GENERAL test set for which the baseline CER is 32.65%. The CER results with the similar word approach is presented in Table 5 and Fig. 10; the results with the ADC class-based LM approaches are presented in Table 6 and Fig. 10.

For a better comparison, the CERs after the two enhancement approaches applied are presented in the same picture as shown in Fig. 11. For a more informative comparison, the NEER results on the CHINESE test set are also presented. It can be seen that with a small enhancement scale, the CER performance is nearly not impacted; however if the enhancement scale is large, the CER may be increased.

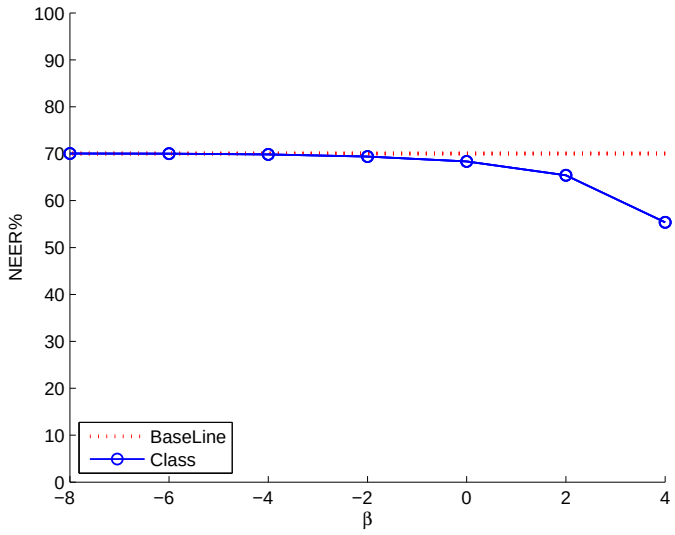

Figure 7: NEER results on unfrequent words for the CHINESE test set with the artificial class-based method. β is the enhancement scale.

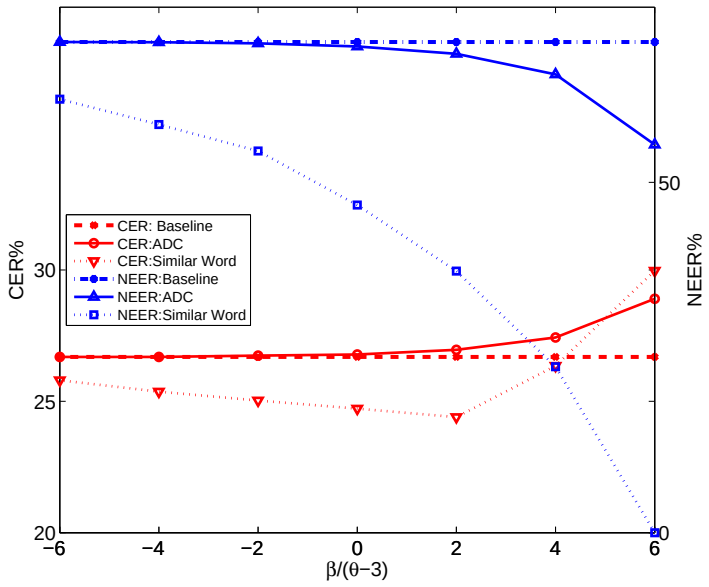

Figure 8: CER and NEER results on the CHINESE test set with the similar word approach and the ADC class-based LM approach.

For example with $\beta = 2$ or $\theta = 5$, the CER is increased by 0.6% with both the two enhancement methods. Nevertheless, the similar word approach is clearly more effective: it is possible to reduce NEER to 46% without any evident CER increase, which is not possible with the ADC class-based LM approach. Additionally, if we

θ \ $SimNum$	CER%						
	1	2	3	4	5	6	7
$-\infty$ (Baseline)	32.65						
-3	32.66	32.66	32.66	32.65	**32.64**	32.64	32.64
-2	32.65	32.66	32.65	32.64	32.64	32.65	32.65
-1	32.67	32.67	32.68	32.65	32.66	32.67	32.67
0	32.68	32.68	32.68	32.66	32.66	32.68	32.68
1	32.69	32.71	32.69	32.66	32.67	32.69	32.69
2	32.72	32.74	32.73	32.73	32.74	32.72	32.7
3	32.76	32.79	32.79	32.78	32.82	32.83	32.83
4	32.85	32.91	32.93	32.94	32.96	32.99	33.01
5	32.98	33.13	33.14	33.16	33.16	33.19	33.27
6	32.86	33.04	33.16	33.32	33.36	33.39	33.42
7	33.01	33.3	33.61	34.01	34.01	34.04	34.09
8	33.5	33.9	34.4	35.12	35.28	35.38	35.52
9	34.27	35.27	36.03	37.62	37.74	37.87	38.04

Table 5: CER results on the GENERAL test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

β	CER%
$-\infty$ (Baseline)	32.65
-8	32.65
-6	**32.64**
-4	32.7
-2	32.76
0	33.05
2	33.78
4	35.42

Table 6: CER results on the GENERAL test set with the ADC class-based LM approach. β is the enhancement scale.

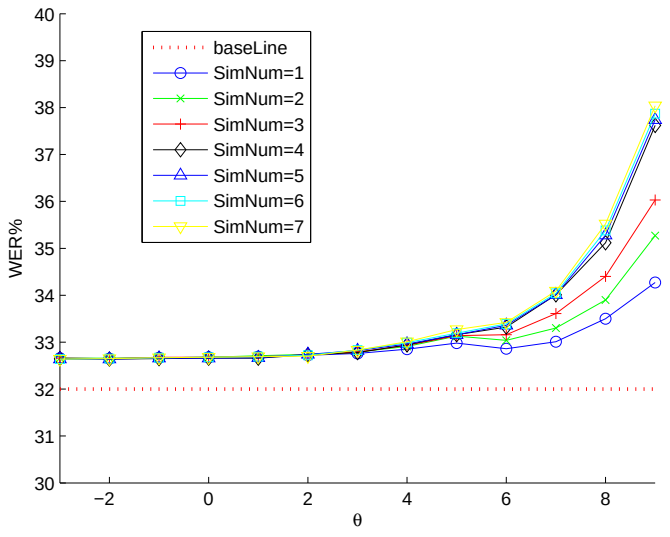

Figure 9: CER results on unfrequent words for the GENERAL test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words for each target word.

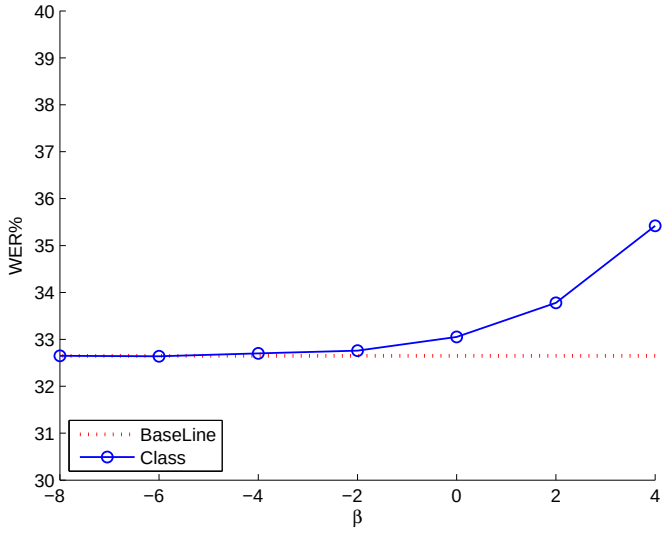

Figure 10: CER results on unfrequent words for the GENERAL test set with the clustering class-based method. β is the enhancement scale.

pay a cost of 0.6% CER increase, the NEER is reduced to 37% from 70% with the similar word approach; with the ADC class-based LM approach, the reduction is just from 70% to 68% .

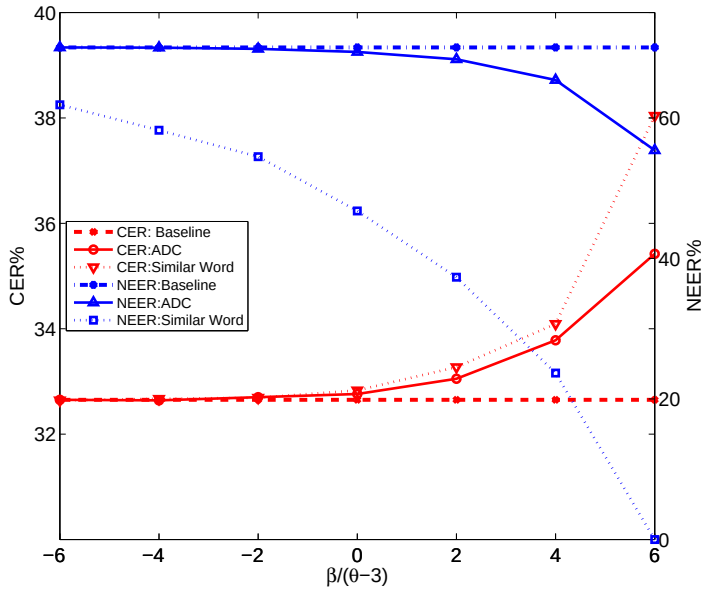

Figure 11: CER results on the GENERAL test set and NEER results on the CHINESE test set.

6.5.2 Experiments on CHINESE-PART

In the second experiment we compare the similar word approach with the MDC class-based LM approach, where the classes are manually defined. The experiment is conducted on the CHINESE-PART test. The same experimental scheme used in the previous experiment is followed, except that the test set is smaller and covers only those unfrequent words that can be categorized into the manually defined classes.

The base results without any enhancement methods are CER=23.34% and NEER=45.73%. The results in terms of CER and NEER with the similar word approach are presented in Table 7 and Table 8 respectively. The same information are also shown Fig. 12 and Fig. 13 as pictures. We found the same trend as in the previous experiment.

	CER%						
θ \ $SimNum$	1	2	3	4	5	6	7
$-\infty$ (Baseline)	23.34						
-3	23.34	23.34	23.31	23.31	23.31	23.31	23.31
-2	23.32	23.32	23.3	23.3	23.3	23.3	23.3
-1	23.32	23.32	23.28	23.28	23.28	23.31	23.31
0	23.27	23.27	23.24	23.24	23.24	23.26	23.26
1	23.24	23.24	23.13	23.13	23.13	23.16	23.16
2	23.23	23.22	23.12	23.12	23.12	23.13	23.13
3	23.2	23.2	23.16	23.16	23.16	23.16	23.18
4	23.15	23.15	23.07	23.07	23.07	23.07	23.05
5	23.15	23.13	23.08	**23.03**	23.09	23.12	23.13
6	23.18	23.26	23.2	23.2	23.2	23.18	23.23
7	23.18	23.15	23.28	23.32	23.39	23.38	23.5
8	23.39	23.39	23.75	23.73	23.75	23.73	23.82
9	23.2	23.4	24.16	24.26	24.3	24.52	24.79

Table 7: CER results on the CHINESE-PART test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

	NEER%						
θ \ $SimNum$	1	2	3	4	5	6	7
$-\infty$ (Baseline)	45.73						
-3	45.49	45.49	45.26	45.26	45.26	45.26	45.26
-2	45.26	45.26	45.02	45.02	45.02	44.78	44.78
-1	45.02	45.02	45.02	45.02	45.02	44.78	44.78
0	44.78	44.78	44.54	44.54	44.54	44.54	44.54
1	44.31	44.31	43.6	43.6	43.6	43.83	43.83
2	43.6	43.36	42.65	42.65	42.65	42.65	42.65
3	43.12	43.12	42.41	42.41	42.41	42.41	42.4
4	42.18	42.18	41.23	41.23	41.23	41.23	41.23
5	41.46	41.46	39.81	39.57	39.57	39.09	39.09
6	40.75	40.52	37.44	36.96	36.96	36.49	36.72
7	39.09	38.86	35.78	35.07	35.3	34.59	34.59
8	38.38	36.96	33.17	32.93	33.17	31.75	30.56
9	35.3	34.36	25.35	24.88	24.64	22.74	**20.85**

Table 8: NEER results on the CHINESE-PART test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

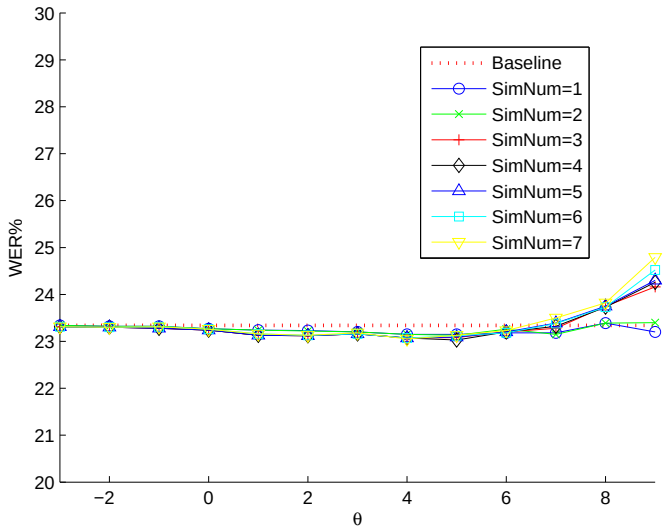

Figure 12: CER results on unfrequent words for the CHINESE-PART test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words for each target word.

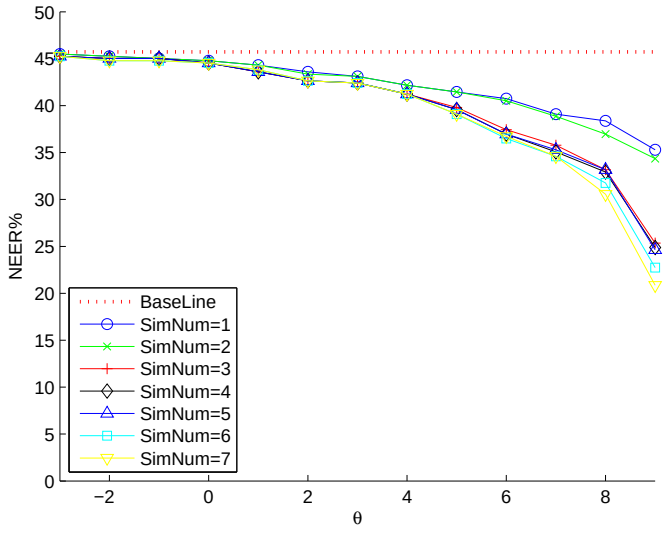

Figure 13: NEER results on unfrequent words for the CHINESE-PART test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

The CER and NEER results with the MDC class-based LM approach are presented in Table 9 and Table 10 respectively, and the corresponding figures are in Fig 14 and Fig. 15.

β	CER%
$-\infty$ (Baseline)	23.34
-6	**23.31**
-4	23.31
-2	23.31
0	23.36
2	23.61
4	24.05
6	24.96

Table 9: CER results on the CHINESE-PART test set with the MDC class-based LM method. β is the enhancement scale.

β	NEER%
$-\infty$ (Baseline)	45.73
-6	45.73
-4	45.73
-2	45.73
0	45.73
2	45.4
4	43.6
6	**33.64**

Table 10: NEER results on the CHINESE-PART test set with the MDC class-based LM method. β is the enhancement scale.

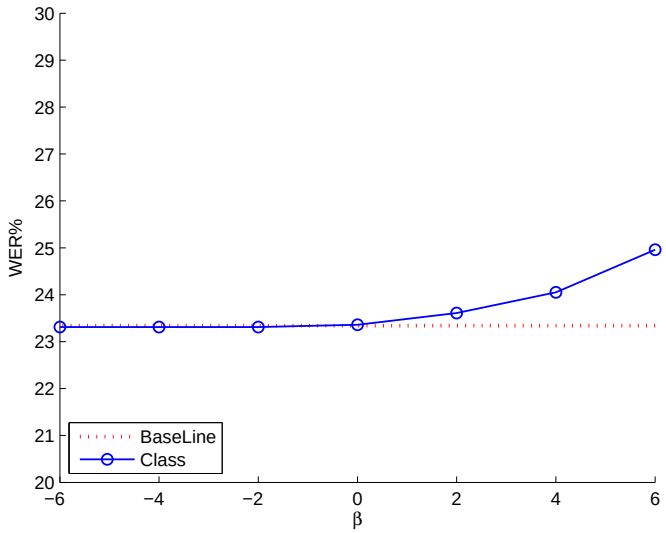

Figure 14: CER results on unfrequent words for the CHINESE-PART test set with the MDC class-based method. β is the enhancement scale.

For a more clear comparison, the CER and NEER results with the two enhancement approaches are presented in the same picture as shown in Fig. 19, where we have set $SimNum = 7$ for the similar word approach. It can be seen that with the MDC class-based LM approach, the improvement on NEER is very marginal unless $\beta \geq 4$. However the cost for this NEER improvement is a clear CER increase (23.31% to 24.05% for $\beta = 4$). With the same cost, the similar word approach has

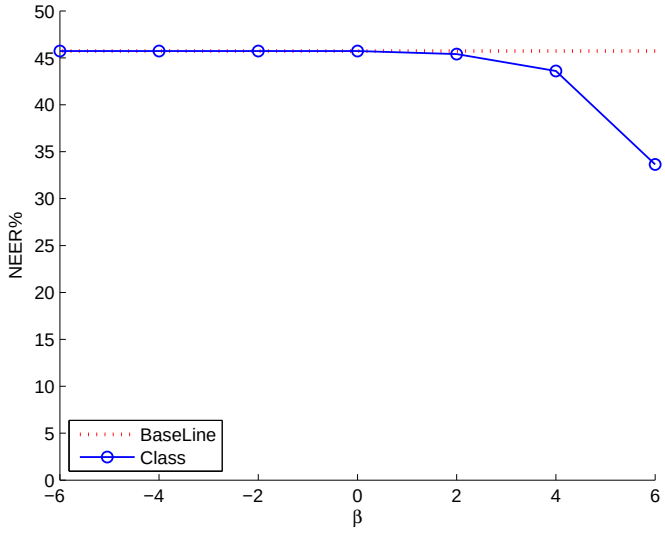

Figure 15: NEER results on unfrequent words for the CHINESE-PART test set with the MDC class-based method. β is the enhancement scale.

reduced NEER nearly to 30%. This demonstrates that the similar word approach is much more effective than the MDC class-based LM.

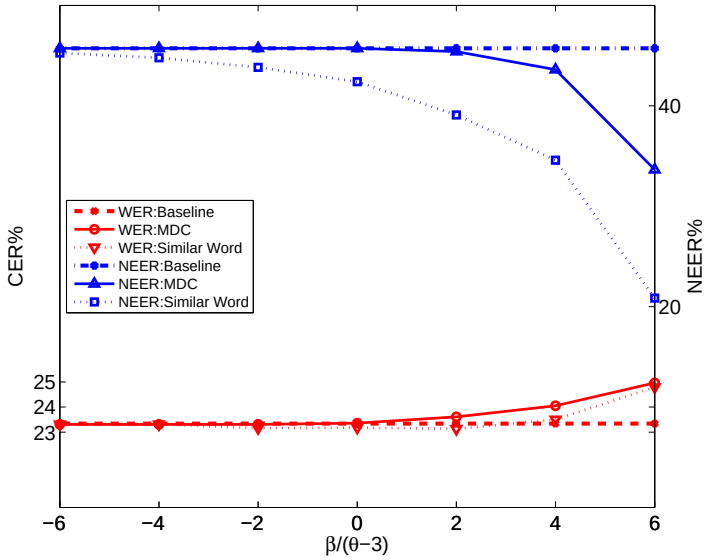

Figure 16: CER and NEER results on the CHINESE-PART test set with the similar word approach and the MDC class-based LM approach.

Finally, we examine the performance on the GENERAL test set after the two enhancement approaches are applied to the LM. The baseline CER is 32.65%. The

results with the similar word approach and the MDC class-based LM approach are reported in Table 11 and Table 12, as well as in Fig. 17 and Fig. 18.

	CER%						
θ \ $SimNum$	1	2	3	4	5	6	7
$-\infty$ (Baseline)	32.65						
-3	32.64	32.65	32.65	32.65	32.65	32.66	32.66
-2	32.65	32.65	32.64	32.64	32.64	32.64	32.64
-1	32.65	32.65	32.65	32.65	32.65	32.64	32.64
0	32.64	32.66	32.64	32.64	**32.63**	32.64	32.64
1	32.65	32.66	32.64	32.66	32.66	32.66	32.66
2	32.66	32.66	32.63	32.64	32.63	32.63	32.63
3	32.66	32.67	32.65	32.66	32.66	32.67	32.68
4	32.67	32.69	32.68	32.7	32.69	32.68	32.7
5	32.67	33.67	32.67	32.7	32.72	32.74	32.78
6	32.72	32.73	32.76	32.79	32.8	32.81	32.89
7	32.83	32.87	32.92	32.98	32.99	33.1	33.19
8	33.02	33	33.27	33.36	33.4	33.53	33.71
9	33.36	33.48	33.92	34.08	34.17	34.29	34.54

Table 11: CER results on the GENERAL test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

β	CER%
$-\infty$ (Baseline)	32.65
-6	**32.65**
-4	32.65
-2	32.66
0	32.65
2	32.71
4	33.01
6	33.74

Table 12: CER results on the GENERAL test set with the MDC class-based LM method. β is the enhancement scale.

For a clear representation, we put the results of the two approaches in the same picture, by selecting $SimNum = 7$ in the similar word approach. To demonstrate the contribution of the enhancement approaches, the NEER results on the CHINESE-PART are also presented. Fig. 19 presents the CER and NEER results.

It can be seen that with a small enhancement scale ($\theta \leq 5$ for the similar word approach and $\beta \leq 2$ for the MDC class-based LM), the performance reduction is as small as 0.1%. This can be regarded as fairly acceptable. In this setting, the NEER reduction with the similar word approach is much more significant than the MDC class-based LM (5% vs. 0.5%). This again confirms that the superiority of the similar word approach.

6.6 Multilingual test

6.6.1 Top-1 translation

In this experiment, the similar word approach is employed to enhance foreign words. Different from the case of monolingual enhancement, foreign words are totally absent in the original LM and cannot be recognized at all. In our study, the baseline system is in Chinese, and the foreign words are in English. We first add English words into the system lexicon and spell the pronunciation in Chinese phones, so that these words can be recognized. Then for each English word, it is translated

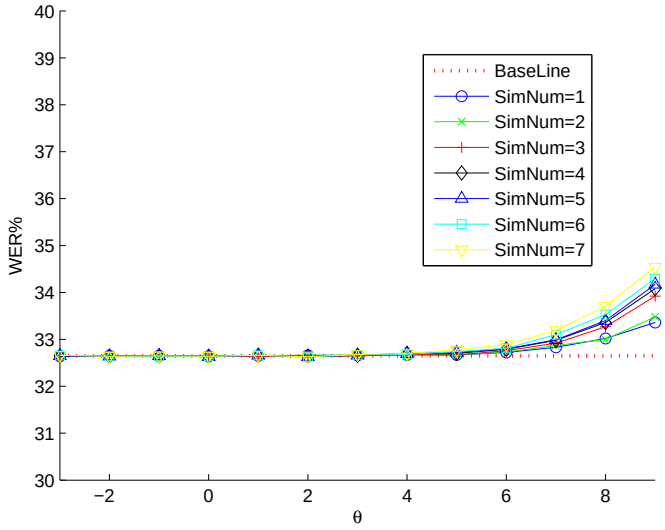

Figure 17: CER results on unfrequent words for the GENERAL test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words for each target word.

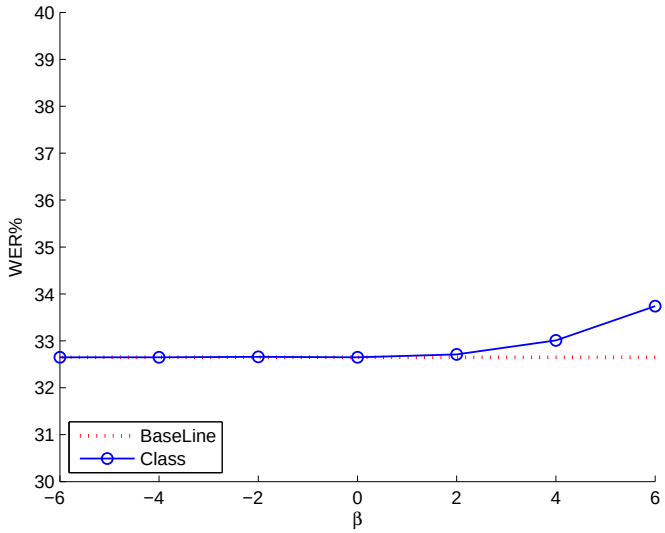

Figure 18: CER results on unfrequent words for the GENERAL test set with the MDC class-based method. β is the enhancement scale.

to a Chinese words by Google translator. With this translation, similar words are selected as in monolingual enhancement. Note that it is not easy to cluster these English words into classes, we don't consider the class-based LM approach here.

The CER and NEER results on the FOREIGN test set are shown in Table 13 and Table 14, as well as in Fig. 20 and Fig. 21.

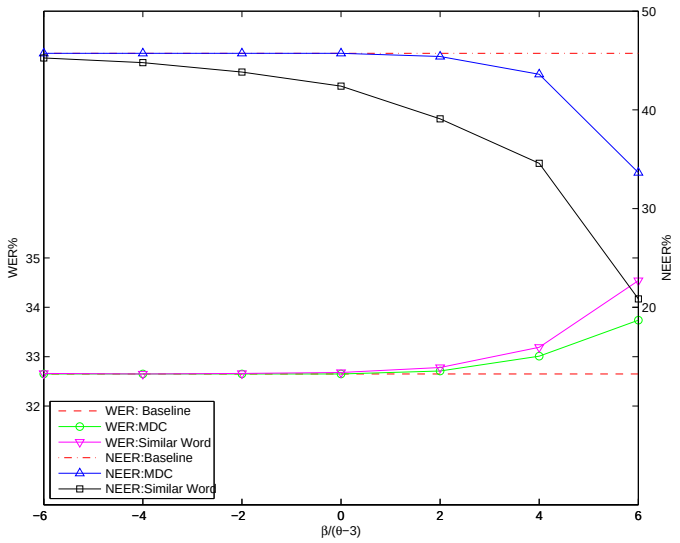

Figure 19: CER results the GENERAL test set and NEERs on the CHINESE-PART test set. β and θ are the enhancement scales for the MDC class-based LM method and the similar word method respectively.

	CER%						
$SimNum$ / θ	1	2	3	4	5	6	7
$-\infty$ (Baseline)	45.23						
0	45.19	45.01	45.01	45.01	45.01	45.01	45.01
1	45.04	44.92	44.92	44.92	44.92	45.04	45.01
2	44.92	44.92	44.92	44.92	44.83	44.83	44.83
3	44.98	44.92	44.92	44.83	44.83	44.83	**44.79**
4	45.01	44.93	44.99	45.01	45.03	45.03	45.05
5	45.07	44.93	45.01	45.09	45.09	45.15	45.15

Table 13: CER results on the FOREIGN test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

	NEER%						
$SimNum$ / θ	1	2	3	4	5	6	7
$-\infty$ (Baseline)	99.58						
0	94.83	92.75	92.33	92.33	92.33	91.91	91.91
1	93.58	90.25	90.25	90.25	89.83	89.42	89.42
2	92.75	86.91	86.5	86.5	86.5	86.5	85.25
3	90.25	82.75	82.33	82.33	82.33	82.33	81.92
4	87.33	76.5	76.08	76.08	75.67	75.25	75.25
5	84	74.17	69.13	67.33	67.33	67.33	**66.92**

Table 14: NEER results on the FOREIGN test set with numbers of similar words for each English word. θ is the enhancement scale, $SimNum$ is the amount of similar words.

For a more clear representation, the CER and NEER results on the FOREIGN test set are shown together in Fig 22. Note that almost all the English words to enhance are absent in the original LM, so the NEER before the enhancement is

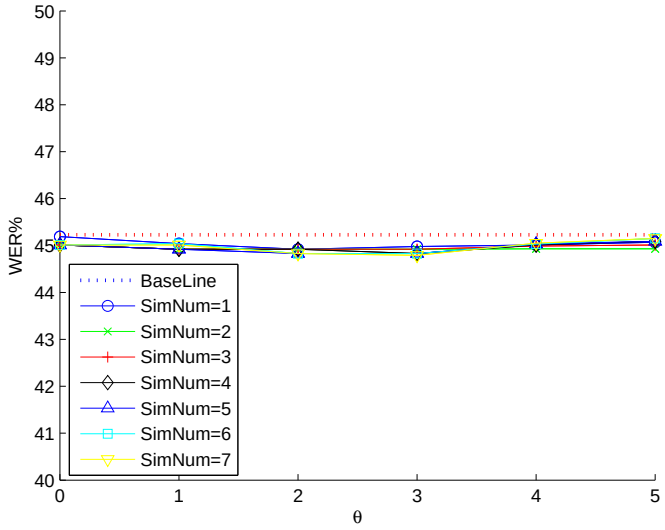

Figure 20: CER results on the FOREIGN test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

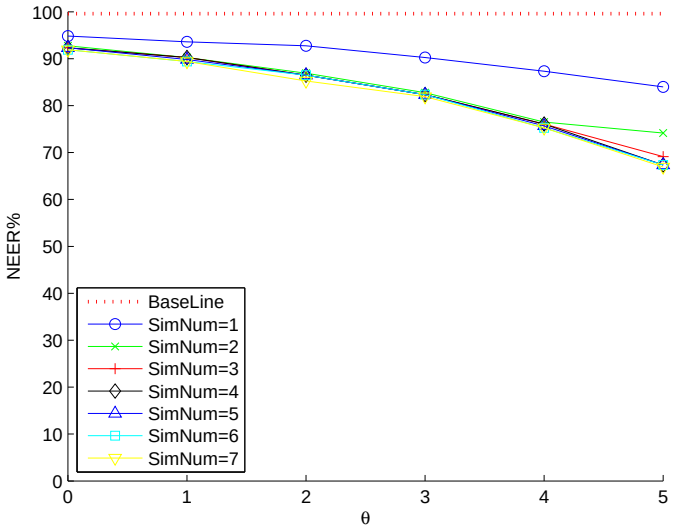

Figure 21: NEER results on the FOREIGN test set with numbers of similar words for each English word. θ is the enhancement scale, $SimNum$ is the amount of similar words.

nearly 100%. It can be seen that the similar word approach is highly effective to enhance foreign words: the NEER can be reduced by 30% without any impact on CER. Compared to the monolingual scenario, the enhancement for foreign words seems more important since it is nearly impossible to recognize them without the enhancement.

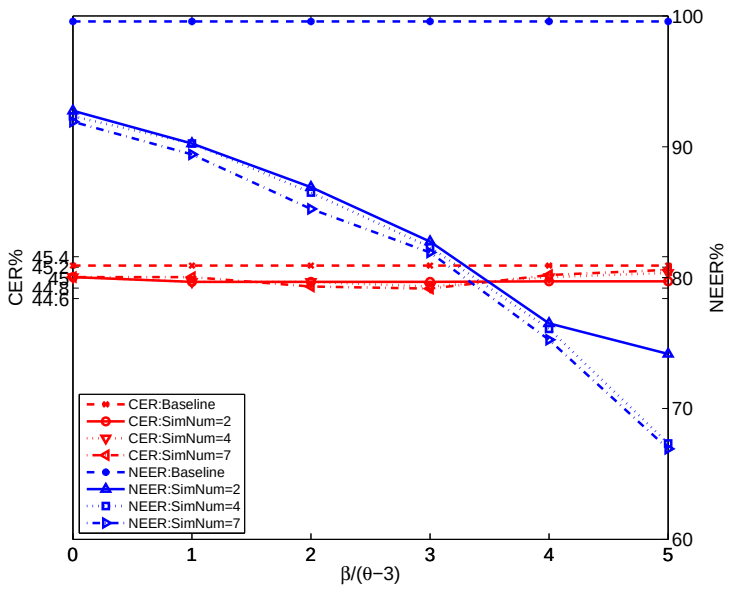

Figure 22: CER and NEER results on the FOREIGN test set with the similar word approach.

We also examine the CER performance on the GENERAL test set to make sure that the enhancement for foreign words doesn't impact much on the general performance of the system. The results are shown in Table 15 as well as Fig. 23. The results demonstrated that the unfrequent word enahancement leads to nearly no impact on general performance of the ASR system.

	CER%						
θ \ $SimNum$	1	2	3	4	5	6	7
$-\infty$ (Baseline)	32.65						
0	32.67	32.66	32.67	32.66	**32.65**	32.66	32.66
1	32.69	32.67	32.66	32.66	32.66	32.66	32.67
2	32.68	32.68	32.67	32.69	32.69	32.71	32.7
3	32.68	32.71	32.71	32.73	32.73	32.73	32.75
4	32.7	32.71	32.71	32.73	32.75	32.75	32.77
5	32.73	32.73	32.78	32.78	32.78	32.81	32.82

Table 15: CER results on the GENERAL test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is the number of similar words.

6.6.2 N-best translation

We also evaluated the performance with n-best translations for each English words. For simplicity, we fix $theta = 3$ that is safe for most of $SimNum$ choices. The CER results and NEER results are presented in Table 16 and Table 17. It can be seen that with reasonably multiple translations, we indeed obtain better performance. This can be attributed that English and Chinese may have multiple-to-multiple correspondence in translation, and so involving multiple translations may enrich the context for the English words in the Chinese LM.

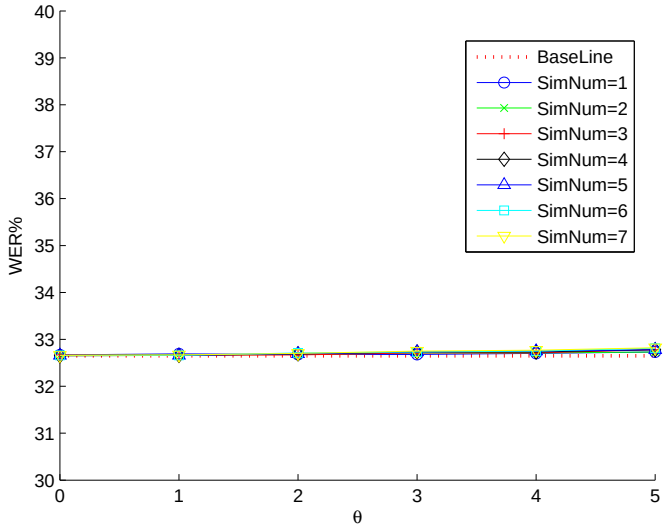

Figure 23: CER results on the GENERAL test set with the similar word enhancement. θ is the enhancement scale, and $SimNum$ is he number of similar words.

	CER%						
$SimNum$ / $N-best$	1	2	3	4	5	6	7
1-best	44.98	44.92	44.92	44.83	44.83	44.83	**44.79**
2-best	44.92	44.91	44.82	44.82	44.8	**44.76**	44.81
3-best	44.91	44.8	44.79	**44.75**	44.82	44.85	44.89
4-best	44.8	**44.75**	44.86	44.89	44.94	44.99	45.04

Table 16: CER results on the FOREIGN test set with the N-best similar word enhancement. $\theta = 3$ and $SimNum$ is the number of similar words.

	NEER%						
$SimNum$ / $N-best$	1	2	3	4	5	6	7
1-best	90.25	82.75	82.33	82.33	82.33	82.33	**81.92**
2-best	87.33	82.75	82.75	82.33	82.33	80.75	**78.33**
3-best	82.33	82.25	80.25	78.33	78.33	**76.67**	76.67
4-best	82.23	80.35	78.33	76.35	76.35	**75.35**	75.35

Table 17: NEER results on the FOREIGN test set with the N-best similar word enhancement. $\theta = 3$ and $SimNum$ is the number of similar words.

Finally, we examine the performance on the GENERAL test set with the n-best foreign similar word enhancement. The results are shown in Table 18 as well as Fig. 26. It can be seen that the enhancement still has a very limited impact on the general performance, though the trend is that involving more translations causes more performance reduction, as expected.

6.7 Similar word model with Euclidean similarity

An interesting question is can we use other distance measures in the similar pair model? As have been discussed in Section 4.3, the word2vec word vectors are opti-

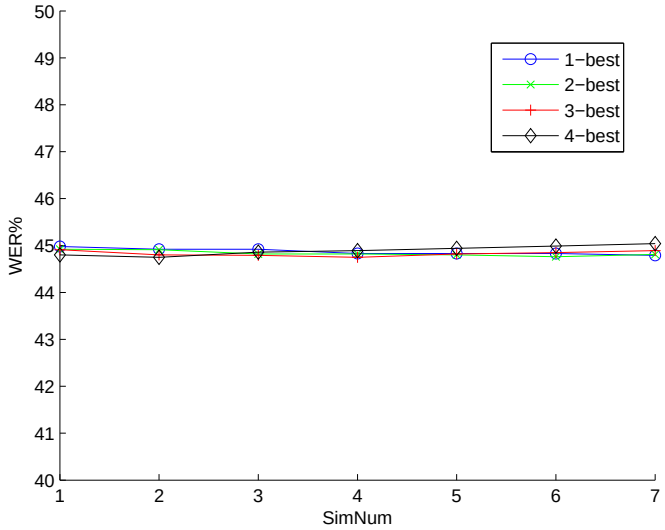

Figure 24: CER results on the FOREIGN test set with the N-best Chinese word enhancement. $\theta = 3$ and $SimNum$ is the number of similar words.

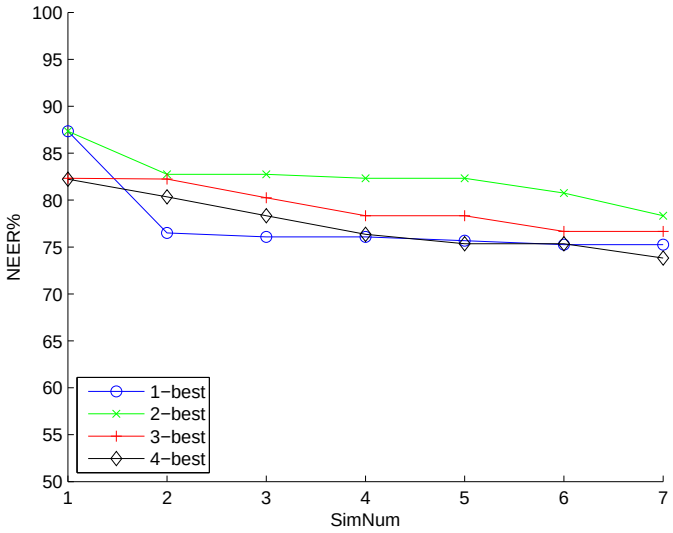

Figure 25: NEER results on the FOREIGN test set with the N-best Chinese word enhancement. $\theta = 3$ and $SimNum$ is the number of similar words.

mized based on cosine distance, and so cosine distance is optimal when computing similarities. We present the experimental evidence in this section.

Basically, we simply replace the cosine distance with the Euclidean distance in similar word selection and the transition weight computation. The experiments are conducted on the CHINESE test set. The CER and NEER results are shown in Table 19 and Table 20 respectively. As before, the same information is shown in

$N-best$ \ $SimNum$	CER%						
	1	2	3	4	5	6	7
1-best	**32.68**	32.71	32.71	32.73	32.73	32.73	32.75
2-best	**32.71**	32.73	32.73	32.75	32.77	32.78	32.82
3-best	**32.71**	32.73	32.75	32.77	32.79	32.81	32.85
4-best	**32.73**	32.78	32.81	32.88	32.92	32.98	33.01

Table 18: CER results on the GENERAL test set with the N-best similar word enhancement. $\theta = 3$ and $SimNum$ is the number of similar words.

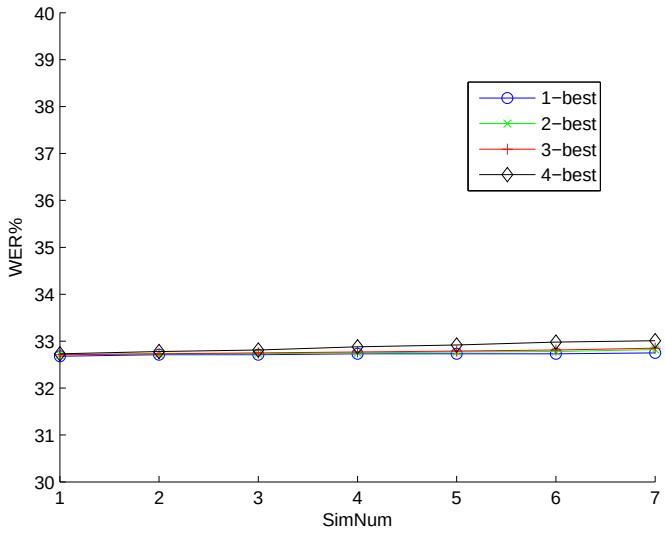

Figure 26: CER results on the GENERAL test set with the N-best similar word enhancement. $\theta = 3$ and $SimNum$ is the number of similar words.

Fig. 27 and Fig. 28. We see very similar patterns as in the case of cosine distance shown in Fig. 4 and Fig. 5.

θ \ $SimNum$	CER%						
	1	2	3	4	5	6	7
$-\infty$ (Baseline)	26.69						
-3	26.62	26.61	26.58	26.33	26.33	26.33	26.33
-2	26.58	26.57	26.41	26.2	26.2	26.2	26.2
-1	26.42	26.41	26.37	26.05	26.05	26.05	26.05
0	26.36	26.32	26.27	25.94	25.94	25.93	25.94
1	26.29	26.28	26.11	25.69	25.59	25.7	25.7
2	26.17	26.17	25.92	25.65	25.66	25.67	25.67
3	25.96	25.93	25.69	25.56	25.56	25.57	25.58
4	25.76	25.79	25.62	25.76	25.77	25.81	25.84
5	**25.54**	25.63	25.58	25.72	25.78	25.79	25.81

Table 19: CER results on the CHINESE test set with the Euler's similar word method. θ is 1.

For a clear comparison, the NEER results with the two distance metric, cosine distance and Euclidean distance, are shown in Fig. 29, where $SimNum$ has been set to be 7. It can be seen that the two metrics result in comparable results, and

θ \ $SimNum$	NEER%						
	1	2	3	4	5	6	7
$-\infty$ (Baseline)	70.03						
-3	69.34	69.28	69.12	65.24	65.18	65.16	65.16
-2	68.95	68.8	67.52	62.94	62.88	62.88	62.88
-1	67.55	67.35	66.3	59.92	59.87	59.83	59.83
0	66.58	66.36	65.27	57.07	57	57	57
1	65.07	64.78	63.01	53.36	53.12	53.16	53.1
2	63.34	62.92	59.92	51.32	51.17	51.15	51.17
3	59.83	59.15	55.42	48.84	48.67	48.6	48.58
4	56.87	56.28	52.97	48.71	48.45	48.49	48.6
5	52.55	51.98	50.69	44.46	44.13	44	**43.91**

Table 20: NEER results on the CHINESE test set with the Euler's similar word method. θ is 1.

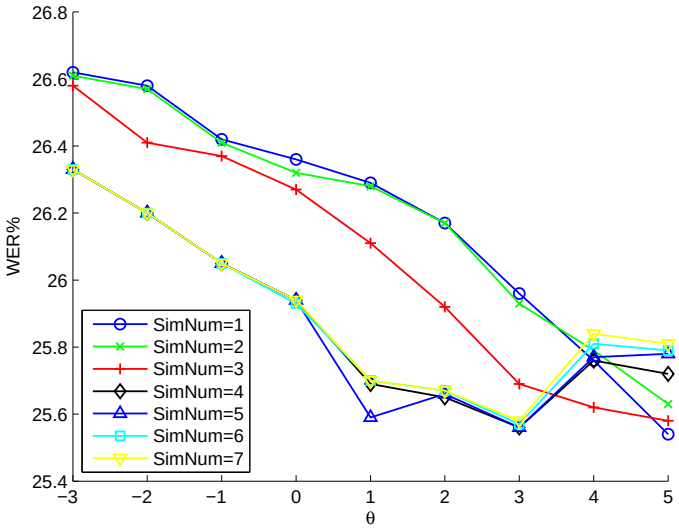

Figure 27: CER results on unfrequent words for the CHINESE test set with the Euler's similar word method. θ is the enhancement scale, and $SimNum$ is the number of similar words for each target word.

cosine distance performs slightly better. This demonstrates that cosine distance is a good similarity metric in the similar word approach.

7 Conclusion

We proposed a simple similar word model to enhance unfrequent words for speech recognition. The new approach constructs fine-grained and word-specific context sharing, which is more flexible and accurate than conventional methods such as class-based LMs. Experiments with a practical large-scale Chinese ASR system demonstrated that the proposed method can effectively improve performance on unfrequent words, and it outperforms the conventional class-based LM approach in a significant way.

Future work involves studying more powerful word pair models, particularly vector models derived from knowledge graphs. Another direction is to apply the similar

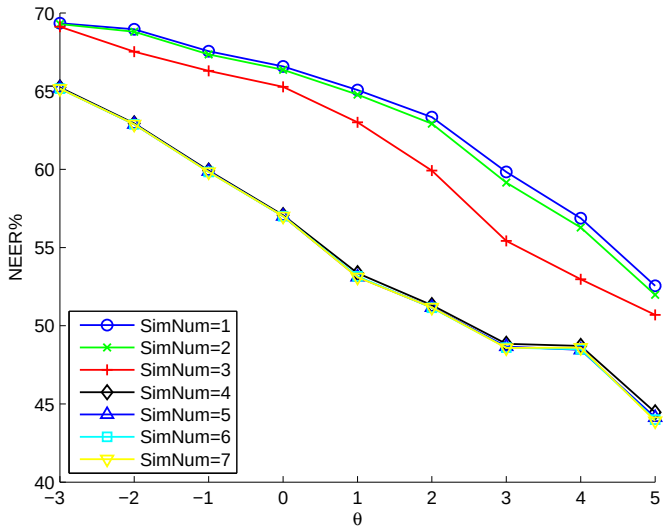

Figure 28: NEERs on unfrequent words for the CHINESE test set with the Euler's similar word method. θ is the enhancement scale, and $SimNum$ is the number of similar words.

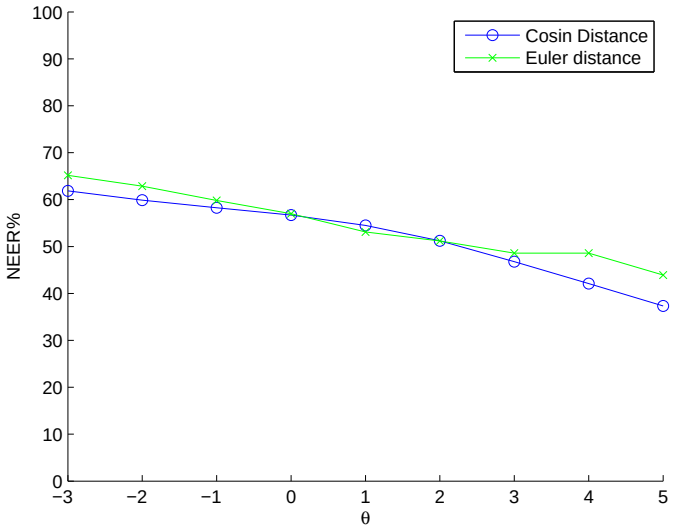

Figure 29: NEERs on unfrequent words for the CHINESE test set with the Cosine similar word method and the Euler's similar word method. θ is the enhancement scale, and $SimNum = 7$.

word approach to other types of LMs, particularly neural LMs based on recurrent neural networks (RNN-LM) [59]. RNN-LMs have exhibited clear advantage compared to conventional n-gram LMs, at least in moderate scale tasks [60]. However, training RNN-LMs with a large vocabulary is still very slow, and it is not easy

to add new words into an already-trained RNN-LM. The similar word approach may provide an interesting solution to these problems, particularly with the similar pair model based on word embedding since it holds intrinsic connections to neural LMs [30]. More investigation is certainly required.

Acknowledgement

This research was supported by the National Science Foundation of China (NSFC) under the project No. 61371136, and the MESTDC PhD Foundation Project No. 20130002120011. It was also supported by Sinovoice and Huilan Ltd.

Author details

[1]Center for Speech and Language Technology, Research Institute of Information Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [2]Center for Speech and Language Technologies, Division of Technical Innovation and Development, Tsinghua National Laboratory for Information Science and Technology, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [3]Department of Computer Science and Technology, Tsinghua University, ROOM 1-303, BLDG FIT, 100084 Beijing, China. [4]University of Alcalá, Madrid, Spain.

References

1. X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.
2. D. Jurafsky and J. H. Martin, *Speech and language processing*. Pearson Prentice Hall, 2000.
3. C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT press, 1999.
4. R. Rosenfield, "Two decades of statistical language modeling: Where do we go from here?" *Proceedings of the IEEE*, 2000.
5. J. R. Bellegarda, "Statistical language model adaptation: review and perspectives," *Speech communication*, vol. 42, no. 1, pp. 93–108, 2004.
6. A. Stolcke *et al.*, "SRILM-an extensible language modeling toolkit." in *Proceedings of ICSLP'02*, 2002, pp. 257–286.
7. B.-J. Hsu and J. Glass, "Iterative language model estimation: efficient data structure & algorithms," in *Proceedings of Interspeech'08*, vol. 8, 2008, pp. 1–4.
8. M. Federico, N. Bertoldi, and M. Cettolo, "IRSTLM: an open source toolkit for handling large scale language models." in *Proceedings of Interspeech'08*, 2008, pp. 1618–1621.
9. S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. A. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, "HTK book 3.4," Cambridge Universeity, 2009.
10. S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.
11. C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to information retrieval," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 2, pp. 179–214, 2004.
12. S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 35, no. 3, pp. 400–401, 1987.
13. R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Procedings of ICASSP'95*, vol. 1. IEEE, 1995, pp. 181–184.
14. Y. W. Teh, "A hierarchical Bayesian language model based on Pitman-Yor processes," in *Proceedings of ACL'06*, 2006, pp. 985–992.
15. ——, "A Bayesian interpretation of interpolated Kneser-Ney," NUS, Tech. Rep. TRA2/06, 2006.
16. P. F. Brown, P. V. de Souza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18, no. 4, pp. 467–479, 1992.
17. F. Pereira, N. Tishby, and L. Lee, "Distributional clustering of English words," in *Proceedings of ACL'93*, 1993, pp. 183–190.
18. H. Ney and R. Kneser., "Improved clustering techniques for class-based statistical language modelling." in *Proceedings of EuroSpeech'93*, 1993, pp. 973–976.
19. W. Ward and S. Issar, "A class based language model for speech recognition," in *Proceedings of ICASSP'96*, vol. 1. IEEE, 1996, pp. 416–418.
20. T. Niesler, E. Whittaker, and P. Woodland., "Comparison of part-of-speech and automatically derived category-based language models for speech recognition." in *Proceedings of ICASSP'98*. IEEE, 1998, pp. 177–180.
21. C. Samuelsson and W. Reichl, "A class-based language model for large-vocabulary speech recognition extracted from part-of-speech statistics," in *Proceedings of ICASSP'99*, vol. 1. IEEE, 1999, pp. 537–540.
22. H. C. Dixon Paul Richard and K. Hideki, "A specialized WFST approach for class models and dynamic vocabulary," in *Proceedings of Interspeech'12*, 2012, pp. 1075–1078.
23. W. Naptali, M. Tsuchiya, and S. Nakagawa, "Topic-dependent-class-based-gram language model," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1513–1525, 2012.
24. I. Bazzi, "Modelling out-of-vocabulary words for robust speech recognition," Ph.D. dissertation, Massachusetts Institute of Technology, 2002.
25. T. Mikolov, I. Sutskever, A. Deoras, H.-S. Le, S. Kombrink, and J. Cernocky, "Subword language modeling with neural networks," *preprint (http://www. fit. vutbr. cz/imikolov/rnnlm/char. pdf)*, 2012.
26. X. Liu, J. L. Hieronymus, M. J. Gales, and P. C. Woodland, "Syllable language models for mandarin speech recognition: Exploiting character language models," *The Journal of the Acoustical Society of America*, vol. 133, no. 1, pp. 519–528, 2013.
27. K. Kirchhoff, D. Vergyri, J. Bilmes, K. Duh, and A. Stolcke, "Morphology-based language modeling for conversational Arabic speech recognition," *Computer Speech & Language*, vol. 20, no. 4, pp. 589 – 608, 2006.
28. H. Sak, M. Saraclar, and T. Gungor, "Morpholexical and discriminative language models for Turkish automatic speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 8, pp. 2341–2351, Oct 2012.
29. M. Ablimit, T. Kawahara, and A. Hamdulla, "Lexicon optimization based on discriminative learning for automatic speech recognition of agglutinative language," *Speech Communication*, vol. 60, pp. 78–87, 2014.

30. Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain, "Neural probabilistic language models," in *Innovations in Machine Learning*. Springer, 2006, pp. 137–186.
31. T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *Proceedings of Interspeech'10*, 2010, pp. 1045–1048.
32. N. Chomsky, *The Architecture of Language*. Oxford University Press, 2000.
33. X. Ma, X. Wang, and D. Wang, "Low-frequency word enhancement with similar pairs in speech recognition," in *Proceedings of ChinaSIP'15*, 2015, pp. 343–347.
34. X. Ma, X. Wang, D. Wang, and Z. Zhang, "Recognize foreign low-frequency words with similar pairs," in *Proceedings of Interspeech'15*, 2015.
35. J. Schalkwyk, L. Hetherington, and E. Story, "Speech recognition with dynamic grammars using finite-state transducers," in *Proceedings of Interspeech'03*, 2003, pp. 1969–1972.
36. K. S. Georges Munir and K. Dietrich, "Transducer-based speech recognition with dynamic language models." in *Proceedings of Interspeech'13*. Citeseer, 2013, pp. 642–646.
37. B. Yuan, X. Wang, D. Wang, Z. Zhang, X. Ma, and B. Xiao, "Semi-dynamic graph embedding for large scale language model adaptation," CSLT, Tsinghua University, 2015. [Online]. Available: http://cslt.riit.tsinghua.edu.cn/mediawiki/images/7/7d/Taglm.pdf
38. T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013. [Online]. Available: http://arxiv.org/abs/1301.3781
39. A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unit hypersphere using von mises-fisher distributions," in *Journal of Machine Learning Research*, 2005, pp. 1345–1382.
40. A. Mnih and G. Hinton, "Three new graphical models for statistical language modelling," in *Proceedings of ICML'07*, 2007, pp. 641–648.
41. T. Mikolov, Q. V. Le, and I. Sutskever, "Exploiting similarities among languages for machine translation," *arXiv preprint arXiv:1309.4168*, 2013.
42. B. Tang, H. Cao, X. Wang, Q. Chen, and H. Xu, "Evaluating word representation features in biomedical named entity recognition tasks," *BioMed research international*, vol. 2014, 2014.
43. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of NIPS'13*, 2013, pp. 3111–3119.
44. A. Bordes, J. Weston, R. Collobert, and Y. Bengio, "Learning structured embeddings of knowledge bases," in *Proceedings of Conference on Artificial Intelligence*, 2011.
45. Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of AAAI'14*, 2014, pp. 1112–1119.
46. M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs: From multi-relational link prediction to automated knowledge graph construction," *arXiv preprint arXiv:1503.00759*, 2015.
47. C. Xu, Y. Bai, J. Bian, B. Gao, G. Wang, X. Liu, and T.-Y. Liu, "RC-NET: A general framework for incorporating knowledge into word representations," in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, 2014, pp. 1219–1228.
48. M. Yu and M. Dredze, "Improving lexical embeddings with semantic knowledge," in *Proceedings of the ACL'14*, vol. 2, 2014, pp. 545–550.
49. Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph and text jointly embedding," in *Proceedings of EMNLP'14*, 2014.
50. D. Zhang, D. Wang, and R. Liu, "Joint semantic relevance learning with text data and graph knowledge," in *Proceedings of ACL'15, Workshop CVSC*, 2015.
51. C. Xing, D. Wang, C. Liu, and Y. Lin, "Normalized word embedding and orthogonal transform for bilingual word translation," in *Proceedings of NAACL'15*, 2015.
52. J. Berstel, *Transductions and Context-Free Languages*. Vieweg+Teubner Verlag, 1979.
53. W. Kuich and A. Salomaa, "Semirings, automata, languages," in *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1986.
54. M. Mohri, "Finite-state transducers in language and speech processing," *Computational linguistics*, vol. 23, no. 2, pp. 269–311, 1997.
55. F. P. Mehryar Mohri and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.
56. M. Mohri, "Weighted automata algorithms," in *Handbook of weighted automata*. Springer, 2009, pp. 213–254.
57. D. Xiong, M. Zhang, and H. Li, "Enhancing language models in statistical machine translation with backward n-grams and mutual information triggers," in *Proceeding of HLT'11*, vol. 1, 2011, pp. 1288–1297.
58. D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *Proceedings of ASRU*, 2011, pp. 1–4.
59. T. Mikolov, "Statistical language models based on neural networks," Ph.D. dissertation, Ph. D. thesis, Brno University of Technology, 2012.
60. M. Sundermeyer, I. Oparin, J.-L. Gauvain, B. Freiberg, R. Schluter, and H. Ney, "Comparison of feedforward and recurrent neural network language models," in *Proceedings of ICASSP'13*, 2013, pp. 8430–8434.