A Generalized Framework for Hierarchical Word Sequence Language Model

Xiaoyi Wu, Kevin Duh, Yuji matsumoto
Nara Institute of Science and Technology
Computational Linguistics Laboratory
8916-5 Takayama, Ikoma, Nara Japan
{xiaoyi-w,kevinduh,matsu}@is.naist.jp

Abstract

Language modeling is a fundamental research problem that has wide application for many NLP tasks. For estimating probabilities of natural language sentences, most research on language modeling use n-gram based approaches to factor sentence probabilities. However, the assumption under n-gram models is not robust enough to cope with the data sparseness problem, which affects the final performance of language models.

At the point, Hierarchical Word Sequence (abbreviated as HWS) language models can be viewed as an effective alternative to normal n-gram method. In this paper, we generalize HWS models into a framework, where different assumptions can be adopted to rearrange word sequences in a totally unsupervised fashion, which greatly increases the expandability of HWS models.

For evaluation, we compare our rearranged word sequences to conventional n-gram word sequences. Both intrinsic and extrinsic experiments verify that our framework can achieve better performance, proving that our method can be considered as a better alternative for n-gram language models.

1 Introduction

Probabilistic Language Modeling is a fundamental research direction of Natural Language Processing. It is widely used in various application such as machine translation (Brown et al., 1990), spelling correction (Mays et al., 1990), speech recognition (Rabiner and Juang, 1993), word prediction (Bickel et al., 2005) and so on.

Most research about Probabilistic Language Modeling, such as Katz back-off (Katz, 1987), Kneser-Ney (Kneser and Ney, 1995), and modified Kneser-Ney (Chen and Goodman, 1999), only focus on smoothing methods because they all take the n-gram approach (Shannon, 1948) as a default setting for modeling word sequences in a sentence. Yet even with 30 years worth of newswire text, more than one third of all trigrams are still unseen (Allison et al., 2005), which cannot be distinguished accurately even using a high-performance smoothing method such as modified Kneser-Ney (abbreviated as MKN).

An alternative solution is to factor the language model probabilities such that the number of unseen sequences are reduced. It is necessary to extract them in another way, instead of only using the information of left-to-right continuous word order.

In (Guthrie et al., 2006), skip-gram (Huang et al., 1993)[1] is proposed to overcome the data sparseness problem. For each n-gram word sequence, the skip-gram model enumerates all possible word combinations to increase valid sequences. This has truly helped to decrease the unseen sequences, but we should not neglect the fact that it also brings a greatly increase of processing time and redundant contexts.

In (Wu and Matsumoto, 2014), a heuristic approach is proposed to convert any raw sentence into a hierarchical word sequence (abbreviated as

[1]The k-skip-n-grams for a sentence $w_1, ...w_m$ is defined as the set $\{w_{i_1}, w_{i_2}, ...w_{i_n} | \Sigma_{j=1}^{n} i_j - i_{j-1} < k\}$.

HWS) structure, by which much more valid word sequences can be modeled while remaining the model size as small as that of n-gram. In (Wu and Matsumoto, 2015) (Wu et al., 2015), instead of only using the information of word frequency, the information of direction and word association are also used to construct higher quality HWS structures. However, they are all specific methods based on certain heuristic assumptions. For the purpose of further improvements, it is also necessary to generalize those models into one unified structure.

This paper is organized as follows. In Section 2, we review the HWS language model. Then we present a generalized hierarchical word sequence structure (GHWSS) in Section 3. In Section 4, we present two strategies for rearranging word sequences under the framework of GHWSS. In Sections 5 and 6, we show the effectiveness of our model by both intrinsic experiments and extrinsic experiments. Finally, we summarize our findings in Section 7.

2 Review of HWS Language Model

In (Wu and Matsumoto, 2014), the *HWS* structure is constructed from training data in an unsupervised way as follows:

Suppose that we have a frequency-sorted vocabulary list $V = \{v_1, v_2, ..., v_m\}$, where $C(v_1) \geq C(v_2) \geq ... \geq C(v_m)$[2].

According to V, given any sentence $S = w_1, w_2, ..., w_n$, the most frequently used word $w_i \in S (1 \leq i \leq n)$ can be selected[3] for splitting S into two substrings $S_L = w_1, ..., w_{i-1}$ and $S_R = w_{i+1}, ..., w_n$. Similarly, for S_L and S_R, $w_j \in S_L (1 \leq j \leq i-1)$ and $w_k \in S_R (i+1 \leq k \leq n)$ can also be selected, by which S_L and S_R can be splitted into two smaller substrings separately. Executing this process recursively until all the substrings become empty strings, then a tree $T = (\{w_i, w_j, w_k, ...\}, \{(w_i, w_j), (w_i, w_k), ...\})$ can be generated, which is defined as an *HWS structure* (Figure 1).

In an HWS structure T, assuming that each node depends on its preceding n-1 parent nodes, then special n-grams can be trained. Such kind of n-grams are defined as *HWS-n-grams*.

[2] $C(v)$ represents the frequency of v in a certain corpus.

[3] If w_i appears multiple times in S, then select the first one.

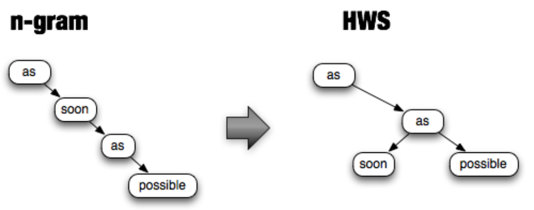

Figure 1: A comparison of structures between HWS and n-gram

The advantage of HWS models can be considered as *discontinuity*. Taking Figure 1 as an example, since n-gram model is a continuous language model, in its structure, the second ‘as’ depends on ‘soon’, while in the HWS structure, the second ‘as’ depends on the first ‘as’, forming a discontinuous pattern to generate the word ‘soon’, which is closer to our linguistic intuition. Rather than ‘as soon ...’, taking ‘as ... as’ as a pattern is more reasonable because ‘soon’ is quite easy to be replaced by other words, such as ‘fast’, ‘high’, ‘much’ and so on. Consequently, even using 4-gram or 5-gram, sequences consisting of ‘soon’ and its nearby words tend to be low-frequency because the connection of ‘as...as’ is still interrupted. On the contrary, the HWS model extracts sequences in a discontinuous way, even ‘soon’ is replaced by another word, the expression ‘as...as’ won’t be affected. This is how the HWS models relieve the data sparseness problem.

The HWS model is essentially an n-gram language model based on a different assumption that a word depends upon its nearby high-frequency words instead of its preceding words. Different from other special n-gram language models, such as class-based language model (Brown et al., 1992), factored language model(FLM) (Bilmes and Kirchhoff, 2003), HWS language model doesn’t use any specific linguistic knowledge or any abstracted categories. Also, differs from dependency tree language models (Shen et al., 2008) (Chen et al., 2012), HWS language model constructs a tree structure in an unsupervised fashion.

In HWS structure, word sequences are adjusted so that irrelevant words can be filtered out from contexts and long distance information can be used

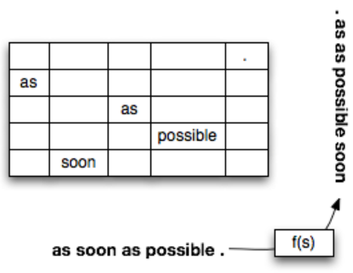

Figure 2: An Example of Generative Hierarchical Word Sequence Structure

for predicting the next word, which make it more effective and flexible in relieving the data sparseness problem. On this point, it has something in common with structured language model (Chelba, 1997), which firstly introduced parsing into language modeling. The significant difference is, structured language model is based on CFG parsing structures, while HWS model is based on pattern-oriented structures.

3 Generalized Hierarchical Word Sequence Structure

Suppose we are given a sentence $s = w_1, w_2, ..., w_n$ and a permutation function $f : s \rightarrow s^{'}$, where $s^{'} = w^{'}_1, w^{'}_2, ..., w^{'}_n$ is a permutation of s. For each word index $i(1 \leq i \leq n, w_i \in s)$, there is a corresponding reordered index $j(1 \leq j \leq n, w^{'}_j \in s^{'}, w^{'}_j = w_i)$.

Then we create an $n \times n$ matrix A. For each row j, we fill cell $A_{j,i}$ with w_i. We define the matrix A as the **generalized hierarchical word sequence structure** (abbreviated as **GHWSS**) of the sentence s. An example is shown in Figure 2.

In a GHWSS, given any word $w \in \{A_{j,i} | w^{'}_j = w_i\}$, the words in its higher rows are $X = \{A_{k,m} | k < j, 1 \leq m \leq n, w^{'}_k = w_m\}$, in which the nearest two neighbors of w are $\hat{l} = A_{k_l,\hat{m}_l}(k_l < j, \hat{m}_l = \underset{1 \leq m < i}{\operatorname{argmin}}(i - m))$ and $\hat{r} = A_{k_r,\hat{m}_r}(k_r < j, \hat{m}_r = \underset{i < m \leq n}{\operatorname{argmin}}(m - i))$ respectively[4]. Then we assume that w depends on $\hat{w} = \hat{l}$ if $k_l > k_r$ or $\hat{w} = \hat{r}$ if $k_l < k_r$. For example, in Figure 2, given the word 'soon', its higher rows $X = \{as, as, possible, .\}$, in which the nearest neighbors of 'soon' are $\hat{l} = as$ and $\hat{r} = as$, since the second 'as' is closer to 'soon' vertically, we assume 'soon' depends the second 'as' in this GHWSS.

[4]There is no $\hat{l}$ when $i = 1$, while no $\hat{r}$ when $i = n$.

Further, for the word $A_{1,i}$, we define that it depends on symbol '$\langle s \rangle$'. We also use the symbol '$\langle /s \rangle$' to represent the end of generation.

For each word $w = A_{j,i}$, if we assume that it only depends on its previous few words in its dependency chain, then we can achieve special n-grams under the GHWSS. Taking Figure 2 as the example, we can train 3-grams like $\{$($\langle s \rangle$, $\langle s \rangle$, .), ($\langle s \rangle$, ., as), (., as, as), (as, as, possible), (as, possible, $\langle /s \rangle$), (as, as, soon), (as, soon, $\langle /s \rangle$)$\}$.

In (Wu and Matsumoto, 2015), it is verified that the performance of HWS model can be further improved by using directional information. Thus, in this paper, we defaultly use directional information to model word sequences. Then the above 3-grams should be $\{$($\langle s \rangle$, $\langle s \rangle$, .), ($\langle s \rangle$, .-R, $\langle /s \rangle$), ($\langle s \rangle$, .-L, as), (.-L, as-L, $\langle /s \rangle$), (.-L, as-R, as), (as-R, as-L, soon), (as-L, soon-L, $\langle /s \rangle$), (as-L, soon-R, $\langle /s \rangle$), (as-R, as-R, possible), (as-R, possible-L, $\langle /s \rangle$), (as-R, possible-R, $\langle /s \rangle$) and the probability of the whole sentence 'as soon as possible .' can be estimated by the product of conditional probabilities of all these word sequences.

4 Two Strategies for constructing GHWSS

Once a permutation function f is implemented, the GHWSS of any sentence can be constructed. Thus, the performance of GHWSS is totally determined by how to implement the function f for rearranging word sequences.

Since n-gram models assume that a word depends on its previous n-1 words, the function f of n-gram methods can be considered as the identity permutation. For each word w_i, we fill cell $A_{i,i}$ with w_i, then the n-gram method is a special case of GHWSS.

In this section, we propose two kinds of methods for implementing function f under GHWSS.

4.1 Word Frequency Based Method

Step 1. Calculate word frequencies from training data and sort all words by their frequency. Assume we get a frequency-sorted list $V = \{v_1, v_2, ..., v_m\}$, where $C(v_j) > C(v_{j+1}), 1 \leq j \leq m - 1$. [5]

[5]$C(v_j)$ represents the frequency of v_j.

Step 2. According to V, for each sentence $s = w_1, w_2, ..., w_n$, we permute it into $s^{'} = w_1^{'}, w_2^{'}, ..., w_n^{'}(w_k^{'} = v_x, w_{k+1}^{'} = v_y, 1 \leq k \leq n-1, 1 \leq x \leq y \leq m)$.

Then the GHWSS constructed by the permutation $s\prime$ is equivalent to that of frequency-based HWS method.

4.2 Word Association Based Method

Step 1. For each sentence s in corpus D, we convert it into $s^{'}$, in which each word only appear once.

Step 2. For each word w_i in the corpus $D' = \{s_i^{'} | 1 \leq i \leq |D|\}$, we count its frequency $C(w_i)$ and its cooccurrence with another word $C(w_i, w_j)$.

Step 3. For each original sentence $s \in D$, we initiate an empty list X and set the beginning symbol '$\langle s \rangle$' as the initial context c [6].

Step 4. For each word $w \in s$, we calculate its word association score with context c. In this paper, we use T-score[7] as the word association measure.

$$T(c, w) = (C(c, w) - \frac{C(c) \times C(w)}{V}) \div \sqrt{C(c, w)} \tag{1}$$

Then we add the i-th word $\hat{w}$ with the maximum score to list X[8] and use it to split s into two substrings $s_l = w_1, ..., w_{i-1}$ and $s_r = w_{i+1}, ..., w_n$.

Step 5. We set $\hat{w}$ as the new context $c^{'}$. For each word in s_l, we calculate its word association score with $c^{'}$ and add the word with the maximum score to list X[9] and use it to divide s_l into two smaller substrings. Then we apply the same process to the substring s_r.

Execute Step4 and Step5 recursively until anymore substrings cannot be divided, then the original sentence s is permuted as list X, by which GHWSS of s can be constructed.

5 Intrinsic Evaluation

We use two different corpus: **British National Corpus** and **English Gigaword Corpus**.

[6]Since $\langle s \rangle$ appears only once in each sentence, we set $C(\langle s \rangle)$ as the size of corpus.

[7]V stands for the total number of words in corpus.

[8]If $\hat{w}$ appears multiple times in s, then select the first one.

[9]If the context word $c^{'}$ also appears in s_l, then we regard it as the word with the maximum score and add it to X directly.

British National Corpus (BNC) [10] is a 100 million word collection of samples of written and spoken English from a wide range of sources. We use all the 6,052,202 sentences (100 million words) for the training data.

English Gigaword Corpus [11] consists of over 1.7 billion words of English newswire from 4 distinct international sources. We choose the wpb_eng part (162,099 sentences, 20 million words) for the test data.

As preprocessing of the training data and the test data, we use the tokenizer of NLTK (Natural Language Toolkit) [12] to split raw English sentences into words. We also converted all words to lowercase.

To ensure the openness of our research, the source code used in the following experiments is available on the internet.[13]

As intrinsic evaluation of language modeling, perplexity (Manning and Schütze, 1999) is the most common metric used for measuring the usefulness of a language model. However, since we unsupervisedly 'parse' the test sentence s into a GHWSS structure before we estimate its probability, its conditional entropy is actually $H(s|T(s))$, where $T(s)$ represents the GHWSS assigned to the test sentence s. Consequently, our method has much lower perplexity. It's not appropriate to directly compare the perplexity of GHWSS-based models to that of n-gram models.

Also, perplexity is not necessarily a reliable way of determining the usefulness of a language model since a language models with low perplexity may not work well in a real world application. Thus, for intrinsic evaluation, we evaluate models only based on how much they can *actually relieve the data sparseness problem* (reduce the unseen sequences).

In (Wu and Matsumoto, 2014), **coverage score** are used to perform this kind of evaluation. The word sequences modeled from training data are defined as TR, while that of test data as TE, then the coverage score is calculated by Equation (2). Obviously, the higher coverage score a language model can achieve, the more it can relieve the data sparseness problem (reduce the unseen sequences).

[10]http://www.natcorp.ox.ac.uk

[11]https://catalog.ldc.upenn.edu/LDC2011T07

[12]http://www.nltk.org

[13]https://github.com/aisophie/HWS

Table 1: Performance of Various Word Sequences

Models	Coverage		Usage		F-score	
	Unique	Total	Unique	Total	Unique	Total
bi-gram	**46.471**	83.121	12.015	76.336	**19.093**	79.584
frequency-based-bi	46.066	89.730	**12.019**	86.937	19.064	88.312
tscore-based-bi	45.709	**89.949**	11.872	**87.252**	18.848	**88.580**
tri-gram	27.164	51.151	5.626	40.191	9.321	45.013
frequency-based-tri	**36.512**	72.432	**8.546**	67.221	**13.850**	69.729
tscore-based-tri	36.473	**72.926**	8.501	**67.382**	13.788	**70.045**

$$score_{coverage} = \frac{|TR \bigcap TE|}{|TE|} \tag{2}$$

If all possible word combinations are enumerated as word sequences, then considerable coverage score can be achieved. However, the processing efficiency of a model become extremely low. Thus, **usage score** (Equation (3)) is also necessary to estimate how much redundancy is contained in a model.

$$score_{usage} = \frac{|TR \bigcap TE|}{|TR|} \tag{3}$$

A balanced measure between coverage and usage is calculated by Equation (4).

$$F\text{-}Score = \frac{2 \times coverage \times usage}{coverage + usage} \tag{4}$$

In this paper, we use the same metric to compare word sequences modeled under GHWSS framework with normal n-gram sequences.

The result is shown in Table 1 [14]. According to the results, for total word sequences, which actually affect the final performance of language models, GHWSS-based methods have obvious advantage over the normal bi-gram model. As for trigrams, the GHWSS-based methods can even improve around 25%.

6 Extrinsic Evaluation

For the purpose of examining how our models work in the real world application, we also performed extrinsic experiments to evaluate our method. In this paper, we use the reranking of n-best translation candidates to examining how language models work in a statistical machine translation task.

We use the French-English part of TED talk parallel corpus for the experiment dataset. The training data contains 139,761 sentence pairs, while the test data contains 1,617 sentence pairs. For training language models, we set English as the target language.

As for statistical machine translation toolkit, we use **Moses system**[15] to train the translation model and output 50-best translation candidates for each French sentence of the test data. Then we use 139,761 English sentences to train language models. With these models, 50-best translation candidates are reranked. According to these reranking results, the performance of machine translation system is evaluated, which also means, the language models are evaluated indirectly. In this paper, we use the following measures for evaluating reranking results[16].

BLEU (Papineni et al., 2002): BLEU score measures how many words overlap in a given candidate translation when compared to a reference translation, which provides some insight into how good the fluency of the output from an engine will be.

METEOR (Banerjee and Lavie, 2005): METEOR score computes a one-to-one alignment between matching words in a candidate translation and a reference.

TER (Snover et al., 2006): TER score measures the number of edits required to change a system output into one of the references, which gives an indication as to how much post-editing will be required

[14]"Unique" means counting each word sequence only once in spite of the amount of times it really occurs.

[15]http://www.statmt.org/moses/

[16]We use open source tool **multeval** (https://github.com/jhclark/multeval) to perform the evaluation.

Table 2: Performance on French-English SMT Task Using Various Word Arranging Assumptions

Models	**BLEU**	**METEOR**	**TER**
tri-gram	31.3	33.5	49.0
frequency-based-tri	31.5	**33.6**	48.6
tscore-based-tri	**31.7**	**33.6**	**48.5**

on the translated output of an engine.

We use GHWSS word rearranging strategies to perform experiments and compared them to the normal n-gram strategy. For estimating the probabilities of translation candidates, we use the modified Kneser-Ney smoothing (MKN) as the smoothing method of all strategies. As shown in Table 2, GHWSS based strategies outperform that of n-gram on each score.

7 Conclusion

In this paper, we proposed a generalized hierarchical word sequence framework for language modeling. Under this framework, we presented two different unsupervised strategies for rearranging word sequences, where the conventional n-gram strategy as one special case of this structure.

For evaluation, we compared our rearranged word sequences to conventional n-gram word sequences and performed intrinsic and extrinsic experiments. The intrinsic experiment proved that our methods can greatly relieve the data sparseness problem, while the extrinsic experiments proved that SMT tasks can benefit from our strategies. Both verified that language modeling can achieve better performance by using our word sequences rearranging strategies, which also proves that our strategies can be used as better alternatives for n-gram language models.

Further, instead of conventional n-gram word sequences, our rearranged word sequences can also be used as the features of various kinds of machine learning approaches, which is an interesting future study.

References

B. Allison, D. Guthrie, L. Guthrie, W. Liu, and Y Wilks. 2005. *Quantifying the Likelihood of Unseen Events: A further look at the data Sparsity problem*. Awaiting publication.

S. Banerjee and A. Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

S. Bickel, P. Haider, and T. Scheffer. 2005. Predicting sentences using n-gram language models. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 193–200.

J. A. Bilmes and K. Kirchhoff. 2003. Factored language models and generalized parallel backoff. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, volume 2, pages 4–6.

P.F Brown, J Cocke, S.A Pietra, V.J Pietra, F Jelinek, J.D Lafferty, R.L Mercer, and P.S Roossin. 1990. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, and J. C. La. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

C. Chelba. 1997. A structured language model. In *Proceedings of ACL-EACL*, pages 498–500.

S. F. Chen and J. Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–393.

W. Chen, M. Zhang, and H Li. 2012. Utilizing dependency language models for graph-based dependency parsing models. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 213–222.

D. Guthrie, B. Allison, W. Liu, and L. Guthrie. 2006. A closer look at skip-gram modeling. In *Proceedings of the 5th international Conference on Language Resources and Evaluation*, pages 1–4.

X. Huang, F. Alleva, H.W. Hon, M.Y. Hwang, and K. F. Lee. 1993. The sphinx-ii speech recognition system: an overview. 7(2):137–148.

S. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *Acoustics, Speech and Signal Processing*, 35(3):400–401.

R. Kneser and H. Ney. 1995. Improved backing-off for m-gram language modeling. *Acoustics, Speech, and Signal Processing*, 1:181–184.

C. D. Manning and H. Schütze. 1999. *Foundations of statistical natural language processing*. MIT Press.

E. Mays, F. J. Damerau, and R. L. Mercer. 1990. Context based spelling correction. *Information Processing and Management*, 27(5):517–522.

K. Papineni, S. Roukos, T. Ward, and W.J. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318.

L. Rabiner and B.H. Juang. 1993. *Fundamentals of Speech Recognition*. Prentice Hall.

C. E. Shannon. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423.

L. Shen, J. Xu, and R.M. Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *ACL*, pages 577–585.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–231.

X. Wu and Y. Matsumoto. 2014. A hierarchical word sequence language model. In *Proceedings of The 28th Pacific Asia Conference on Language*, pages 489–494.

X. Wu and Y. Matsumoto. 2015. An improved hierarchical word sequence language model using directional information. In *Proceedings of The 29th Pacific Asia Conference on Language*, pages 453–458.

X. Wu, Y. Matsumoto, K. Duh, and S. Hiroyuki. 2015. An improved hierarchical word sequence language model using word association. In *Statistical Language and Speech Processing*, pages 275–287.