A Hierarchical Word Sequence Language Model

Xiaoyi Wu
Nara Institute of Science and Technology
Computational Linguistics Laboratory
8916-5 Takayama, Ikoma, Nara Japan
xiaoyi-w@is.naist.jp

Yuji Matsumoto
Nara Institute of Science and Technology
Computational Linguistics Laboratory
8916-5 Takayama, Ikoma, Nara Japan
matsu@is.naist.jp

Abstract

Most language models used for natural language processing are continuous. However, the assumption of such kind of models is too simple to cope with data sparsity problem. Although many useful smoothing techniques are developed to estimate these unseen sequences, it is still important to make full use of contextual information in training data. In this paper, we propose a hierarchical word sequence language model to relieve the data sparsity problem. Experiments verified the effectiveness of our model.

1 Introduction

Most language models used for natural language processing, such as n-gram approach proposed by Shannon (1948), are continuous. However, the assumption that a word depends upon the preceding n-1 words is too simple to cope with data sparsity problem.

Thus, a number of useful smoothing techniques such as back-off (Katz,1987), Kneser-Ney (Kneser & Ney,1995), modified Kneser-Ney (Chen & Goodman,1999) have been developed to estimate the probabilities of unseen sequences. Yet even with 30 years worth of newswire text, more than one third of all trigrams are unseen (Allison et al., 2005). It is still important to make full use of contextual information hidden in training data.

D. Guthrie. et. al. (2006) proposed using skip-gram (Huang et. al., 1993) to overcome the data sparsity problem. The skip-gram model using discontinuous sequences to model languages has truly helped to decrease the unseen sequences, but we should not neglect the fact that it also brings the greatly increase of processing time and redundant contexts. D. Guthrie. et. al. (2006) examined the coverage of skip-gram, but didn't analyze the efficiency of them, which will be discussed in section 4.3 and section 4.4 in this paper.

Taking into account of the balance between coverage and usage, we present a hierarchical word sequence model to relieve the data sparsity problem. Differing from other hierarchical language models, such as hierarchical phrase-based model (Chiang, 2007) used in SMT systems, our model is essentially a n-gram language model whose modeling assumption is determined by tree structures.

We introduce our main idea in Section 2. In Section 3, we propose the hierarchical word sequence model. We show the effectiveness of this model by several experiments in Section 4 and conclude in Section 5.

2 Basic Ideas

Data sparsity is caused by the low frequency word combinations and unknown word combinations, which are inevitably increased by the assumption that a word depends upon the preceding n-1 words.

For instance, given two sentences A = 'I hit the tennis ball' and B = 'I hit the ball', suppose that A is in the training data and B is in the test data, then the bigram (the, ball) and trigram (hit, the, ball) will not be learned by normal n-gram models.

Skip-gram models relieve this problem by skipping some words so that bigram (the, ball) and trigram (hit, the, ball) can be learned. But suppose that

Models	Trained Bigrams	Tested Bigrams
bigram model	(the, ball), (a, naughty), (naughty, boy)	(the, tennis), (tennis, ball), (a, boy)
skip-bigram model	(the, ball), (a, naughty), (naughty, boy), **(a, boy)**	(the, tennis), (tennis, ball), **(a, boy)**
proposed model	**(the, ball)**, **(a, boy)**, (boy, naughty)	(ball, tennis), **(the, ball)**, **(a, boy)**

Table 1: An example of bigrams trained and tested by three different kinds of models. The bolded bigrams occur in both training data and test data.

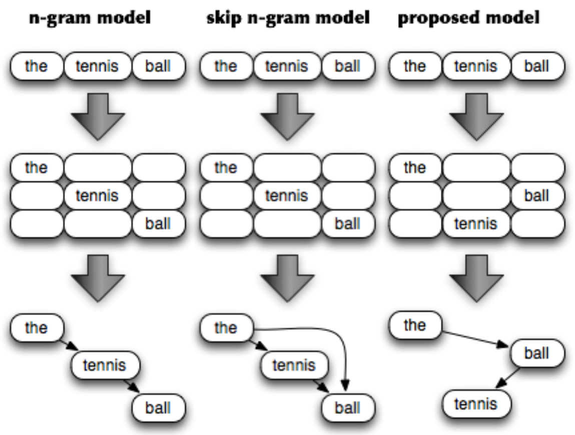

Figure 1: The assumptions of three different models

B is in the training data and A is in the test data, then the bigrams (the, tennis), (tennis, ball) and trigrams (hit, the, tennis), (the, tennis, ball) cannot be learned too. Besides, the skipping is actually a partial enumeration of word combinations, which come along with lots of modeling redundancy.

Since 'tennis ball' is a specification of pattern '... ball', it is more appropriate to consider that 'tennis' depends upon 'ball' rather than its preceding word 'the'. Similarly, 'the ball' can be considered as a specification of pattern 'the ...'. Based on such an assumption, we propose to reorder the dependent sequence as 'the→ball→tennis' instead of the original one ('the→tennis→ball'), and consequently, the bigrams are trained and tested as (the, ball), (ball, tennis), which is quite different from the traditional sequential way as shown in Figure 1.

To reveal the advantages of this idea, suppose we have {'the ball', 'a naughty boy'} in the training data and {'the tennis ball', 'a boy'} in the test data. Table 1 shows what we will have in the bigram model and the skip-bigram model, and what we hope to have in our proposed model. As shown in this table, our model learns pairs of words that hopefully have direct dependencies. Besides, without enumeration, proposed model can keep size of trained grams as small as normal n-gram model.

Although we also change the word sequence of test data in a different way, it is still appropriate to compare it with n-gram models for two reasons. First, the word sequence of training data and test data are reordered by the same assumption that a word depends upon its schematic pattern as we described above, just as the n-gram model assume that every word of test data depends upon its preceding words. Second, the number of total tested bigrams is still the same as that of n-gram models. For each word of test data, we only make a different assumption about what the dependent words should be.

Since these dependent words can be determined if we parse 'the tennis ball' into an intermediate structure as shown in Figure 1, the only remaining problem is how to achieve such kind of structure from any sequence. Although similar structures can be achieved by applying dependency parsers, the accuracy of word dependency parsing is highly language-dependent. It is expected for us to figure out a method that can be applied to any language as easily as normal n-gram models.

Intuitively, the more frequently a word is used, the more probable it becomes part of a useful pattern. We establish our method based on such a heuristic rule in the following section.

3 Method

As we discussed previously, we assume that a word depends upon its schematic pattern, and also assume that such a pattern consists of relatively high frequency words.

Based on these two assumptions, first, we calculate all the uni-grams of training data and sort them into a ranking list by frequency as shown in Table 2.

According to this ranking list, for each sentence in

Word	Frequecy
,	2501
the	2040
.	1950
of	1149
to	1087
a	1014
and	847
in	753
’s	465
...	...

Table 2: An example of frequency ranking list

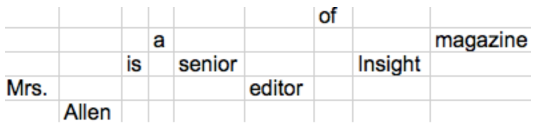

Figure 2: An example of divided sentence

training data, we find the most frequently used word [1] and use it to divide this sentence into 2 parts. For instance, in the sentence ’Mrs. Allen is a senior editor of insight magazine’, ’of’ is the most frequently used word in Table 2, then we use ’of’ to divide this sentence into ’Mrs. Allen is a senior editor’ and ’insight magazine’, recursively, for each part, we divide it into two shorter parts (or one if there are not remaining subsequences on both sides). Finally, the result is represented as a matrix in Figure 2.

Alternatively, this matrix is also represented in a binary tree as shown in Figure 3.

[1]If this word appears multiple times in this sentence, then select the first one.

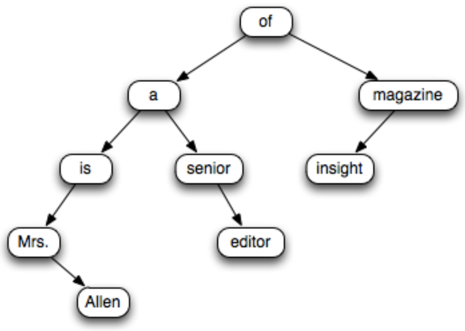

Figure 3: An example of binary tree

In this binary tree, each node (word) is generated from its parent nodes, which can be considered as a schematic pattern of this node. For instance, in Figure 3, the node ’Mrs.’ is generated from the path ’of→a→is’, which means that the word ’Mrs.’ is generated from the pattern ’... is a ... of ...’ in the original sentence.

Assuming that each node in this tree depends on the preceding n-1 parent nodes, then a special n-gram model can be trained. We define this kind of model as a **hierarchical word sequence n-gram language model** (abbreviated as **hws-n-gram model**). For instance, the hws-2-grams of Figure 3 are {(\$[2], of), (of, a), (a, is), (is, Mrs.), (Mrs., Allen), (a, senior), (senior, editor), (of, magazine), (magazine, insight)}, while the hws-3-grams are {(\$, \$, of), (\$, of, a), (of, a, is), (a, is, Mrs.), (is, Mrs. Allen), (of, a, senior), (a, senior, editor), (\$, of, magazine), (of, magazine, insight)}.

4 Experiments

4.1 Setting

To test the performance on out-of-domain data, we use two different corpora **British National Corpus** and **English Gigaword Corpus** to perform experiments.

British National Corpus is a balanced synchronic text corpus consisting of English sentences with 100 million word tokens of written and spoken language from a wide range of sources. We use the entire BNC corpus as training data.

English Gigaword Corpus consists of over 1.7 billion words of English newswire from four distinct international sources. We use 100,000 words of wpb_eng file (Washington Post/Bloomberg Newswire Service) as test data.

As preprocessing of training and test data, all words were converted to lowercase and all numbers were replaced with a special tag, <NUM>.

4.2 Perplexity

This experiment evaluates the performance of the proposed model based on perplexity.

We compared our model with normal n-gram models and skip-n-gram models by applying additive smoothing (as Equation 1), Kneser-Ney (Kneser

[2]’\$’ represents the beginning of a sentence.

Model+Smoothing	Perplexity
2gram+ADD	1634.22
2gram+KN	1034.543
2gram+MKN	999.213
1skip-2gram+ADD	1096.116
2skip-2gram+ADD	972.283
3skip-2gram+ADD	884.781
hws-2gram+ADD	**865.795**
3gram+ADD	9556.827
3gram+KN	973.766
3gram+MKN	**912.076**
1skip-3gram+ADD	6568.764
2skip-3gram+ADD	4444.833
3skip-3gram+ADD	3460.362
hws-3gram+ADD	1284.708

Table 3: Perplexity values of normal n-gram models, skip-n-gram models and proposed model by applying different smoothing methods

& Ney,1995) and modified Kneser-Ney (Chen & Goodman,1999) as smoothing method separately.

$$P(w_i|w_{i-n+1}^{i-1}) = \frac{C(w_{i-n+1}^{i}) + \alpha}{C(w_{i-n+1}^{i-1}) + \alpha V}^{3} \quad (1)$$

The results are shown in Table 3, since the grams of our model is trained in a special way, it's not appropriate to directly incorporate lower order models to higher ones, and consequently, we cannot directly apply Kneser-ney Smoothing on our model[4]. Yet even though with additive smoothing, our bigram model outperforms normal bigram model with Modified Kneser-Ney Smoothing. Thus, if we can figure out an appropriate way to incorporate it to our trigram model, it is highly possible that ours outperforms normal trigram models as well.

4.3 Coverage and Usage

This experiment illustrates the coverage and usage of our model compared to those of normal n-gram model and skip-n-gram model. We trained on the entire BNC corpus (100 million words) and measured the coverage on 100,000 words of newswire from the Gigaword corpus.

[3]V stands for vocabulary size, and smoothing parameter α (0.0001$\leq\alpha\leq$1.0) is determined by golden section search (Kiefer,1953).

[4]Neither can skip-n-gram model.

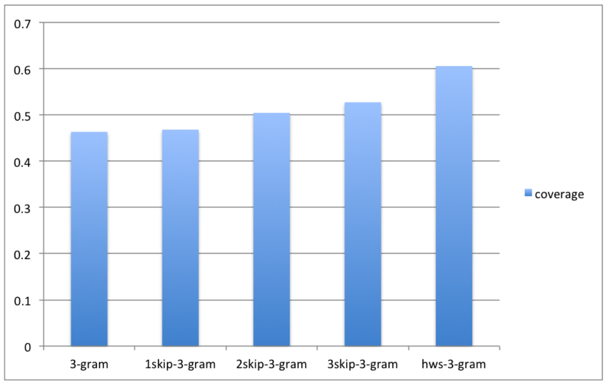

Figure 4: The coverage of 3-grams

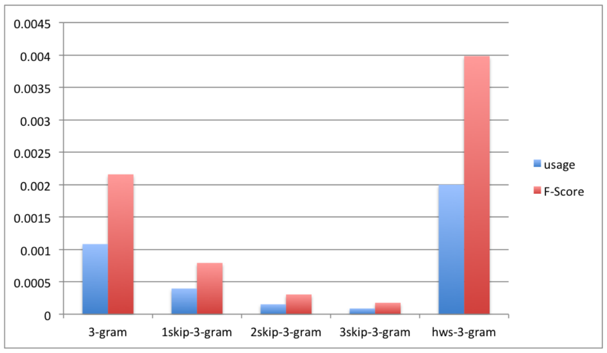

Figure 5: The usage and F-Score of 3-grams

We list all trigrams of test data to examine how many of them actually occurred in trained model and how many trigrams of trained model actually are used in test data.

We define the grams of training data as TR, and unique grams of test data as TE, then we calculate coverage by Equation 2.

$$coverage = \frac{|TR \bigcap TE|}{|TE|} \quad (2)$$

We also use Equation 3 to estimate how much redundancy contained in a model and Equation 4 as a balanced measure.

$$usage = \frac{|TR \bigcap TE|}{|TR|} \quad (3)$$

$$FScore = \frac{2 \times coverage \times usage}{coverage + usage} \quad (4)$$

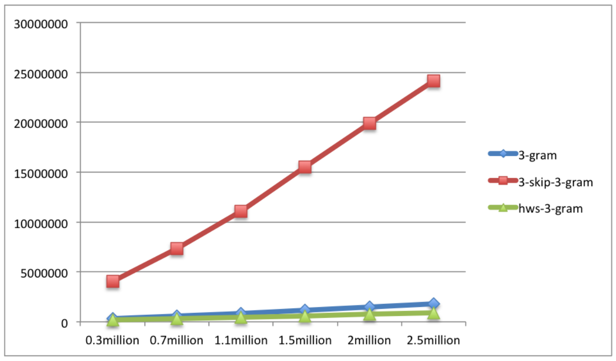

Figure 6: The growth of trained grams with the addition of training data size

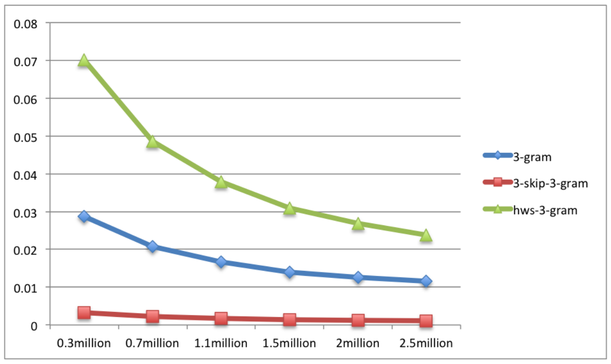

Figure 7: The decreasing of usage with the addition of training data size

The results of coverage are shown in Figure 4. Even though skip-gram model use a partial enumeration of word combinations to expand trained grams, proposed model still outperforms 3skip-3-gram model by 7.3 percent.

Figure 5 shows the results of usage and F-score. Apparently, there is much less modeling redundancy in our model, and as a result, ours keeps better balance between coverage and usage than the other ones.

4.4 Length of Trained Grams and Training Data Size

This experiment examines the relation between length of trained grams and training data size. We use exactly the same test data (100,000 words of Gigaword corpus) as above. But for training data, we use different portions of different sizes of BNC corpus. We gradually increase the amount of training data to examine how it affects these trained grams.

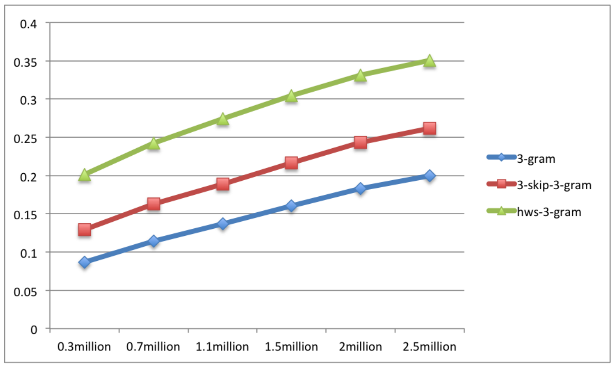

Figure 8: The increasing of coverage with the addition of training data size

Intuitively, the length of trained grams will be increased with the addition of corpus size. As shown in Figure 6, in comparison with normal 3-gram model and hws-3-gram model, the grams learned by 3-skip-3-gram grow very fast, which means the cost of producing and storing them is quite considerable.

Consequently, the growth of grams comes along with modeling redundancy, appearing as the decreasing of usage. As shown in Figure 7, though hws-3-gram is decreasing, it is still more efficient (with higher usage) than the other two models.

Of course, the inefficiency of 3-skip-3-gram would be worth if it resulted in higher coverage. But as shown in Figure 8, all the three kinds of models increase at almost the same speed, and proposed model still hold the lead.

5 Conclusions and Future Work

In this paper, we proposed a hierarchical word sequence language model to make full use of contextual information and relieve the data sparsity problem.

Proposed model has a good performance on decreasing perplexity, which also keeps better balance between coverage and usage than normal n-gram model and skip-n-gram model. Besides, the cost of storing our model is more economical than other models.

In this paper, we only used additive smoothing as the smoothing method for our model, the performance can be further improved if we incorporate lower order models to higher ones. Besides that, if we use certain criteria to filter schematic patterns trained by our model, some useful sentence patterns can be extracted, which is also a promising future study.

References

B. Allison, D. Guthrie, et. al. 2005. *Quantifying the Likelihood of Unseen Events: A further look at the data Sparsity problem.* Awaiting publication.

C. E. Shannon. 1948. *A Mathematical Theory of Communication. The Bell System Technical Journal*, 27: 379-423.

D. Chiang. 2007. *Hierarchical Phrase-based Translation. Computational Linguistics*, Vol33, No.2, 201–228.

D. Guthrie, B. Alliso, et. al. 2006. *A Closer Look at Skip-gram Modelling. Proceedings of the 5th international Conference on Language Resources and Evaluation*, 2006: 1-4.

J. Kiefer. 1953. *Sequential minimax search for a maximum. Proceedings of the American Mathematical Society*, 1953, 4(3): 502-506.

R. Kneser, H. Ney. 1995. *Improved backing-off for m-gram language modeling. Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on. IEEE*, 1995, 1: 181-184.

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

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

X. Huang, F. Alleva, H. W. Hon, et al. 1993. *The SPHINX-II speech recognition system: an overview. Computer Speech & Language*, 1993, 7(2): 137-148.