An Algorithm for Fast Calculation of Back-off N-gram Probabilities with Unigram Rescaling

Masaharu Kato*, Tetsuo Kosaka†, Akinori Ito‡and Shozo Makino§

Abstract— **Topic-based stochastic models such as the probabilistic latent semantic analysis (PLSA) are good tools for adapting a language model into a specific domain using a constraint of global context. A probability given by a topic model is combined with an n-gram probability using the unigram rescaling scheme. One practical problem to apply PLSA to speech recognition is that calculation of probabilities using PLSA is computationally expensive, that prevents the topic-based language model from incorporating that model into decoding process. In this paper, we proposed an algorithm to calculate a back-off n-gram probability with unigram rescaling quickly, without any approximation. This algorithm reduces the calculation of a normalizing factor drastically, which only requires calculation of probabilities of words that appears in the current context. The experimental result showed that the proposed algorithm was more than 6000 times faster than the naive calculation method.**

Keywords: probabilistic latent semantic analysis, unigram rescaling, n-gram, back-off smoothing

1 Introduction

A language model is an important component of a speech recognition system [1]. Especially an n-gram model is widely used for speech recognition purpose. Although an n-gram is a simple and powerful model, it has a drawback that it is strongly affected by a domain of the training data, and thus the performance is greatly deteriorated when applied to speech from other domain.

To overcome this problem, language model adaptation techniques have been developed so far [2]. There are various types of language model adaptation methods; among them, adaptation methods based on topic models are promising because they exploit global constraint (or "topic") of a speech into calculation of word occurrence probabilities.

There are many topic models, including the latent semantic analysis (LSA) [3], the probabilistic latent semantic analysis (PLSA) [4], and the latent Dirichlet allocation (LDA) [5]. They use a vector representation of a word or a document. In these models, a word or a document is regarded as a point in a vector space or a continuous probability space. Then a probability of a word is calculated by estimating a point in the space under a constraint of the adaptation data.

These topic models only estimate unigram probabilities, where the occurrence probability of a word is independent of the previous or following words. However, bigram or trigram model is used as a language model of speech recognition, where a word occurrence probability is determined based on one or two preceding words. Therefore, we need to combine the unigram probability obtained from the topic model with the bigram or trigram probability. Gildea and Hofmann proposed the unigram rescaling for combining these probabilities [6]. The unigram rescaling is a method to adjust an n-gram probability according to the ratio between the unigram probability from the topic model and the unigram model without the topic model. The unigram rescaling can be applied any topic model [7, 6], and is proved to be effective compared with other combination methods such as linear combination [8]. The probability combination of a topic model and an n-gram model is widely used in many speech recognition researches [9, 10, 11].

A problem of the unigram rescaling is its computational complexity. The unigram rescaling needs a normalization of probability, and calculation of the normalizing factor needs calculation of probabilities that is proportion to the vocabulary size. Furthermore, we need to compute the normalizing factor for all word history independently, the number of which are proportion to the vocabulary size for bigram, and to the square of the vocabulary size for trigram. When the adaptation data are fixed, the calculation of the normalizing factor is not a big problem because we need to calculate them only once when the adaptation data are given. However, the calculation of the normalizing factor can be a matter when we adapt

*Graduate School of Science and Technology, Yamagata University, Jonan 4-3-16, Yonezawa, Yamagata, 992-8510 Japan. Tel/Fax: +81 238 22 4814 Email: katoh@yz.yamagata-u.ac.jp

†Graduate School of Science and Technology, Yamagata University, Jonan 4-3-16, Yonezawa, Yamagata, 992-8510 Japan. Tel/Fax: +81 238 22 4814 Email: tkosaka@yz.yamagata-u.ac.jp

‡Graduate School of Engineering, Tohoku University, Aramaki aza Aoba 6-6-5, Sendai, Miyagi, 980-8579 Japan. Tel/Fax: +81 22 795 7084 Email:aito@makino.ecei.tohoku.ac.jp

§Graduate School of Engineering, Tohoku University, Aramaki aza Aoba 6-6-5, Sendai, Miyagi, 980-8579 Japan. Tel/Fax: +81 22 795 7084 Email: makino@makino.ecei.tohoku.ac.jp

the language model dynamically (for example, word-by-word) [6]. In this case, the calculation of word occurrence probability becomes several thousand times slower.

In this paper, we propose an algorithm to expedite the calculation of unigram-rescaled n-gram probability. The basic idea of the proposed algorithm is to combine the calculation of unigram rescaling with calculation of the back-off n-gram. Using the proposed algorithm, we can calculate the rescaled probability by only considering those words actually appear in the linguistic context.

2 Unigram rescaling

A PLSA or other topic-based model gives an occurrence probability of a word w in a document d as

$$P(w|d) = \sum_t P(w|t)P(t|d) \tag{1}$$

where t is a latent topic. To combine a document-dependent word occurrence probability with an n-gram probability, the unigram rescaling is used:

$$P(w|h,d) = \frac{1}{Z(h,d)} \frac{P(w|d)}{P(w)} P(w|h) \tag{2}$$

Here, h is a word history where a word w occurs. $P(w)$ is a unigram probability and $Z(h,d)$ is a normalizing factor calculated as follows.

$$Z(h,d) = \sum_{w \in \mathcal{V}} \frac{P(w|d)}{P(w)} P(w|h) \tag{3}$$

where $\mathcal{V}$ is a vocabulary.

According to the Eq. (3) naively, calculation time of the normalizing factor $Z(h,d)$ is in proportion to the vocabulary size, which means the calculation of a unigram-rescaled probability takes several thousands times slower than calculation time of a simple n-gram probability. This is a problem when using unigram-rescaled probabilities in a decoding process.

3 Back-off n-gram

To calculate an n-gram probability, back-off smoothing[12] is often used. Consider calculating a trigram probability $P(w_i|w_{i-2}^{i-1})$, where w_j^k denotes a word sequence $w_j \dots w_k$. The trigram probability is calculated based on a trigram count $N(w_{i-2}^i)$, which is the number of occurrence of word sequence $w_{i-2}w_{i-1}w_i$ in a training corpus. Based on maximum likelihood estimation, the trigram probability is estimated as

$$P_M(w_i|w_{i-2}^{i-1}) = \frac{N(w_{i-2}^i)}{N(w_{i-2}^{i-1})}. \tag{4}$$

However, the probability P_M becomes zero when the word sequence w_{i-2}^i does not occur in the training corpus. To avoid the "zero-frequency" problem, probabilities of unseen trigrams are estimated from bigram frequencies.

$$P(w_i|w_{i-2}^{i-1}) = \begin{cases} \beta(w_{i-2}^i)P_M(w_i|w_{i-2}^{i-1}) & \text{if } N(w_{i-2}^i) > 0 \\ \alpha(w_{i-2}^{i-1})P(w_i|w_{i-1}) & \text{if } N(w_{i-2}^{i-1}) > 0 \\ P(w_i|w_{i-1}) & \text{otherwise} \end{cases} \tag{5}$$

In this equation, $0 < \beta(w_{i-2}^i) < 1$ denotes a discount coefficient, which is used to reserve a certain amount of probability mass for unseen n-grams. Various methods for determining the discount coefficient β have been proposed, including the Good-Turing discounting [12], Witten-Bell discounting [13] and Kneser-Ney discounting [14]. $\alpha(w_{i-2}^{i-1})$ is a normalizing factor of n-gram calculation, calculated as follows.

$$\alpha(w_{i-2}^{i-1}) = \frac{1 - \sum_{w \in \mathcal{V}_1(w_{i-2}^{i-1})} \beta(w_{i-2}^i) P_M(w|w_{i-2}^{i-1})}{\sum_{w \in \mathcal{V}_0(w_{i-2}^{i-1})} P(w|w_{i-1})} \tag{6}$$

Here, $\mathcal{V}_0(w_{i-2}^{i-1})$ denotes a set defined as

$$\mathcal{V}_0(w_{i-2}^{i-1}) = \{w | w \in \mathcal{V} \text{ and } N(w_{i-2}w_{i-1}w) = 0\}, \tag{7}$$

and $\mathcal{V}_1(w_{i-2}^{i-1})$ is a complementary set of $\mathcal{V}_0$, i.e.

$$\mathcal{V}_1(w_{i-2}^{i-1}) = \{w | w \in \mathcal{V} \text{ and } N(w_{i-2}w_{i-1}w) > 0\}. \tag{8}$$

The bigram probability $P(w_i|w_{i-1})$ in Eq. (5) is calculated using the back-off smoothing recursively. Note that the coefficients α and β are calculated when the language model is generated.

4 Fast calculation of unigram rescaling

4.1 The bigram case

Let us consider calculating a bigram probability with unigram rescaling. By rewriting Eq. (2), we obtain

$$P(w_i|w_{i-1},d) = \frac{1}{Z(w_{i-1},d)} \frac{P(w_i|d)}{P(w_i)} P(w_i|w_{i-1}). \tag{9}$$

When $w_i \in \mathcal{V}_1(w_{i-1})$, we calculate Eq. (9) as

$$P(w_i|w_{i-1},d) = \frac{\beta(w_{i-1}^i)}{Z(w_{i-1},d)} \frac{P(w_i|d)}{P(w_i)} P_M(w_i|w_{i-1}), \tag{10}$$

and when $w_i \in \mathcal{V}_0(w_{i-1})$, assuming $P(w_i) = P_M(w_i)$, we obtain

$$\begin{aligned} P(w_i|w_{i-1},d) &= \frac{\alpha(w_{i-1})}{Z(w_{i-1},d)} \frac{P(w_i|d)}{P(w_i)} P_M(w_i) \\ &= \frac{\alpha(w_{i-1})}{Z(w_{i-1},d)} P(w_i|d) \end{aligned} \tag{11}$$

Therefore the normalizing factor $Z(w_{i-1},d)$ is calculated as

$$Z(w_{i-1},d) = \sum_{w \in \mathcal{V}} \frac{P(w|d)}{P(w)} P(w|w_{i-1}) \tag{12}$$

$$= \sum_{w \in \mathcal{V}_1(w_{i-1})} \frac{P(w|d)}{P(w)} \beta(w_{i-1}w) P_M(w|w_{i-1}) + \quad (13)$$
$$\sum_{w \in \mathcal{V}_0(w_{i-1})} \alpha(w_{i-1}) P(w|d)$$
$$= \sum_{w \in \mathcal{V}_1(w_{i-1})} \frac{P(w|d)}{P(w)} \beta(w_{i-1}w) P_M(w|w_{i-1}) + \quad (14)$$
$$\alpha(w_{i-1}) \left(1 - \sum_{w \in \mathcal{V}_1(w_{i-1})} P(w|d) \right)$$

Equation (14) indicates that the normalizing factor $Z(w_{i-1}, d)$ can be calculated by summing up all words in $\mathcal{V}_1(w_{i-1})$, which is a set of words that appears just after w_{i-1}. As we can assume $|\mathcal{V}_1(w_{i-1})| \ll |\mathcal{V}_0(w_{i-1})|$, we can expect reduction of calculation time.

4.2 The trigram case

The reduction of calculation in the previous section relies on the fact that the back-off bigram probability can be reduced to a unigram probability when the word is contained in $\mathcal{V}_0(w_{i-1})$ and the unigram probability cancels out the denominator of unigram rescaling factor. This does not hold for the trigram case. However, we can reduce calculation time of a trigram probability with unigram rescaling by considering the bigram normalizing factor $Z(w_{i-1}, d)$.

The normalizing factor for a trigram probability is calculated as

$$Z(w_{i-2}^{i-1}, d) = \sum_{w \in \mathcal{V}} \frac{P(w|d)}{P(w)} P(w|w_{i-2}^{i-1}) \quad (15)$$
$$= \sum_{w \in \mathcal{V}_1(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-2}^{i-1}) + \quad (16)$$
$$\sum_{w \in \mathcal{V}_0(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-2}^{i-1}).$$

Let the second term of Eq. (16) be

$$Z_0(w_{i-2}^{i-1}, d) = \sum_{w \in \mathcal{V}_0(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-2}^{i-1}). \quad (17)$$

Then $Z_0(w_{i-2}^{i-1}, d)$ can be rewritten as

$$Z_0(w_{i-2}^{i-1}, d) = \sum_{w \in \mathcal{V}_0(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} \alpha(w_{i-2}^{i-1}) P(w|w_{i-1}) \quad (18)$$
$$= \alpha(w_{i-2}^{i-1}) \sum_{w \in \mathcal{V}_0(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-1}) \quad (19)$$

Here, from equation (9) we obtain

$$Z(w_{i-1}, d) P(w_i|w_{i-1}, d) = \frac{P(w_i|d)}{P(w_i)} P(w_i|w_{i-1}). \quad (20)$$

Then Eq. (19) can be rewritten as

$$Z_0(w_{i-2}^{i-1}, d)$$
$$= \alpha(w_{i-2}^{i-1}) \sum_{w \in \mathcal{V}_0(w_{i-2}^{i-1})} Z(w_{i-1}, d) P(w|w_{i-1}, d) \quad (21)$$
$$= \alpha(w_{i-2}^{i-1}) Z(w_{i-1}, d) \times \quad (22)$$
$$\left(1 - \sum_{w \in \mathcal{V}_1(w_{i-2}^{i-1})} P(w|w_{i-1}, d) \right)$$
$$= \alpha(w_{i-2}^{i-1}) \times \quad (23)$$
$$\left(Z(w_{i-1}, d) - \sum_{w \in \mathcal{V}_1(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-1}) \right)$$

Finally, we obtain

$$Z(w_{i-2}^{i-1}, d) =$$
$$\sum_{w \in \mathcal{V}_1(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-2}^{i-1}) + \quad (24)$$
$$\alpha(w_{i-2}^{i-1}) \left(Z(w_{i-1}, d) - \sum_{w \in \mathcal{V}_1(w_{i-2}^{i-1})} \frac{P(w|d)}{P(w)} P(w|w_{i-1}) \right).$$

This equation contains only summation over $\mathcal{V}_1(w_{i-2}^{i-1})$, which is a set of words that appear just after w_{i-2}^{i-1}. Note that calculation of $Z(w_{i-1}, d)$ requires summation over $\mathcal{V}_1(w_{i-1})$, which is larger than $\mathcal{V}_1(w_{i-2}^{i-1})$. However, as we have only $|\mathcal{V}|$ kinds of $Z(w_{i-1}, d)$, we can cache the calculated value of $Z(w_{i-1}, d)$ for a specific d. Therefore, on calculating $Z(w_{i-2}^{i-1}, d)$, if $Z(w_{i-1}, d)$ has not been calculated yet, we first calculate $Z(w_{i-1}, d)$ and then calculate $Z(w_{i-2}^{i-1}, d)$. On the other hand, if $Z(w_{i-1}, d)$ has been calculated, we just use the cached value of $Z(w_{i-1}, d)$ to calculate $Z(w_{i-2}^{i-1}, d)$.

4.3 Estimation of calculation time

Let us estimate how fast the proposed calculation algorithm is, based on a simple assumption. When calculating probabilities of word occurrences in an evaluation corpus based on Eq. (2), we have to calculate normalizing factors for all word w in the vocabulary and all history h in the evaluation corpus. Let K_n be the number of distinct n-grams that appears in the training corpus, and H_n be that in the evaluation corpus.

Considering the bigram case, the number of calculation of probabilities of all distinct bigrams in the evaluation corpus can be estimated as $O(H_1 K_1)$, because we have to calculate probabilities of all words in the vocabulary for each of word history that appears in the evaluation corpus. On the other hand, the calculation based on Eq.

(14) only requires probability calculation for those words that appears just after the current context. Because

$$E_w[|\mathcal{V}_1(w)|] = \frac{1}{|\mathcal{V}|}\sum_w |\mathcal{V}_1(w)| \quad (25)$$

$$= \frac{K_2}{|\mathcal{V}|} = \frac{K_2}{K_1}, \quad (26)$$

the computational complexity of the proposed mathod is proportion to $O(H_1K_2/K_1)$.

As for the trigram case, the naive calculation needs $O(H_2K_1)$. According to Eq. (24), we need to calculate all $Z(w_{i-1}, d)$ to calculate the normalizing factors for trigram probabilities, which means the calculation time is in proportion to $O(H_1K_2/K_1 + H_2K_3/K_2)$.

5 Evaluation by an experiment

5.1 Experimental conditions

We carried out an experiment to actually calculate probabilities of trigrams in a corpus using the naive calculation method and the proposed algorithm. In this experiment, we calculated probabilities of all distinct trigrams in the corpus, that means the calculation of a specific trigram is conducted only once.

We used the Corpus of Spontaneous Japanese (CSJ) [15] for both training and testing purpose. The text is extracted from the XML representation of the corpus, and we employed the short-unit word (SUW) as a definition of words. All inter-pausal units were enclosed by the beginning-of-sentence and end-of-sentence symbols.

We used 2668 lectures drawn from academic presentations and public speaking, which contain 833140 sentences (word sequences between pauses) and 8486301 words. The vocabulary size was 47309, which includes words that appeared more than once in the training corpus. We trained a trigram model from the training corpus. The cut-off frequency was set to 1 for the bigram and 3 for the trigram, which means that the bigrams that occurred only once were excluded from the n-gram calculation, and the trigrams that occurred less than four times were also excluded from the n-gram calculation . We also trained a PLSA model from the training corpus using a parallel training algorithm of PLSA [16]. We used 10 lectures (test-set 1) as an evaluation corpus, which contained 1217 sentences and 30837 words. Numbers of distinct n-grams in the training and evaluation corpora are shown in Table 1. In this table, K_n shows the number of n-grams in the generated n-gram model. Therefore, the number of distinct trigrams is smaller than that of bigrams because the trigram counts with small frequency were cut off while constructing the language model.

Calculation of the PLSA was conducted using a computer equipped with Intel Celeron 2.80GHz and 1GB memory.

Table 1: Number of distinct n-grams

n-gram	K_n	H_n
unigram	47100	665
bigram	327604	34296
trigram	208234	72618

Table 2: CPU time for evaluation

Calculation	CPU time [sec]
Naive	5683.00
Proposed	0.85

Table 2 shows the evaluation result. Note that this result does not include the calculation time for PLSA adaptation. This result shows that the proposed algorithm was about 6700 times faster than the naive method that calculates the normalizing factors for all words in the vocabulary.

6 Conclusion

In this paper, we proposed an algorithm that calculates probabilities that are calculated by back-off n-gram and unigram rescaling. This algorithm drastically reduces calculation of a normalizing factor of unigram-rescaled probability. From the experimental result, the proposed algorithm could calculate the unigram-rescaled probabilities 6700 times faster.

We used the proposed algorithm for combining n-gram and PLSA probabilities, but the proposed algorithm can be used for combining any language models by unigram rescaling. Using this calculation algorithm, it becomes realistic to incorporate a unigram-rescaled language model into the decoder.

References

[1] Rosenfeld, R., "Two decades of statistical language modeling: where do we go from here." *Proceedings of the IEEE*, V88, N8, pp. 1270–1278, 2000.

[2] Bellegarda, J.R., "Statistical language model adaptation: review and perspectives," *Speech Communication,* V42, N1, pp. 93–109, 2004.

[3] Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R., "Indexing by latent semantic analysis," *J. American Society for Information Science*, V41, pp. 391–407, 1990.

[4] Hoffmann, T., "Unsupervised Learning by Probabilistic Latent Semantic Analysis," *Machine Learning,* V42, N1–2, pp. 177–196, 1999.

[5] Blei, D.M., Ng, A.Y., Jordan, M.I., Lafferty, J., "Latent dirichlet allocation," *J. Machine Learning Research,* V3, pp. 993–1022, 2003.

[6] Gildea, D., Hofmann, T., "Topic-based Language Models using EM," *Proc. Eurospeech*, Budapest, Hungary, pp. 2167–2170, 9/99.

[7] Bellegarda, J.R., "Exploiting Latent Semantic Information in Statistical Language Modeling," *Proceedings of the IEEE,* V88, pp. 1279–1296, 2000.

[8] Broman, S., Kurimo, M., "Methods for Combining Language Models in Speech Recognition," *Proc. Interspeech,* Lisbon, Portugal, pp. 1317–1320, 9/05.

[9] Akita, Y., Kawahara, T., "Language Model Adaptation based on PLSA of Topics and Speakers," *Proc. Interspeech,* Jeju, South Korea, pp. 1045–1048, 10/04.

[10] Chiu, H.S., Chen, B., "Word Topical Mixture Models for Dynamic Language Model Adaptation," *Proc. Int. Conf. on Acoustics, Speech and Signal Processing,* Honolulu, U.S.A, pp. IV-169 – IV-172, 4/07.

[11] Hsieh, Y.C., Huang, Y.T., Wang, C.C., Lee., L.S., "Improved Spoken Document Retrieval With Dynamic Key Term Lexicon and Probabilistic Latent Semantic Analysis (PLSA)," *Proc. Int. Conf. on Acoustics, Speech and Signal Processing,* Toulouse, France, pp. I-961 – I-964, 5/06.

[12] Katz, S., "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer," *IEEE Trans. Acoustics, Speech & Signal Processing,* V35, N3, pp.400–401, 1987.

[13] Witten, I.H., Bell, T.C., "The Zero-frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression," *IEEE Trans. Information Theory,* V37, N4, pp. 1085–1094, 1991.

[14] Ney, H., Essen, U., Kneser, R., "On Structuring Probabilistic Dependences in Stochastic Language Modelling," *Computer Speech and Language,* V8, N1, pp. 1–38, 1994.

[15] Maekawa, K., Koiso, H., Furui, S., Isahara, H., "Spontaneous speech corpus of Japanese," *Proc. Second International Conference on Language Resources and Evaluation (LREC),* Athens, Greece, pp. 947–952, 6/00.

[16] Kato, M., Kosaka, T., Ito, A., Makino, S., "Fast and Robust Training of a Probabilistic Latent Semantic Analysis Model by the Parallel Learning and Data Segmentation," *J. Communication and Computer,* V6, N5, pp. 28-35, 2009.