USE OF LATENT WORDS LANGUAGE MODELS IN ASR: A SAMPLING-BASED IMPLEMENTATION

Ryo Masumura, Hirokazu Masataki, Takanobu Oba, Osamu Yoshioka, Satoshi Takahashi

NTT Media Intelligence Laboratories, NTT Corporation, Japan
{masumura.ryo, masataki.hirokazu, oba.takanobu, yoshioka.osamu, takahashi.satoshi}@lab.ntt.co.jp

ABSTRACT

This paper applies the latent words language model (LWLM) to automatic speech recognition (ASR). LWLMs are trained taking into account related words, i.e., grouping of similar words in terms of meaning and syntactic role. This means, for example, if a technical word and a general word play a similar syntactic role, they are given a similar probability. This is expected that the LWLM performs robustly over multiple domains. Furthermore, we can expect that the interpolation of the LWLM and a standard n-gram LM will be effective since each of the LMs have different learning criterion. In addition, this paper also describes an approximation method of the LWLM for ASR, in which words are randomly sampled on the LWLM and then a standard word n-gram language model is trained. This enables us one-pass decoding. Our experimental results show that the LWLM performs comparable to the hierarchical Pitman-Yor language model (HPYLM) in a target domain task, and more robustly performs in out-domain tasks. Moreover, an interpolation model with the HPYLM provides a lower word error rate in all the tasks.

Index Terms— Latent words language model, Hierarchical Pitman-Yor language model, Sampling-based implementation.

1. INTRODUCTION

Language models (LMs) are necessary to modern automatic speech recognition (ASR) systems. Word n-gram LMs are still widely used because of simplicity and compatibility to ASR [1]. For example, it is easy to express a word n-gram LM as a weighted finite state transducer (WFST). However, the word level modeling suffers from the data sparseness problem.

Smoothing is a fundamental technique to mitigate the data sparseness problem [2]. While various smoothing methods have been proposed, the Kneser-Ney smoothing is known to be one of the most accurate methods [3]. Its mechanism has been also revealed through the theory of the hierarchal Pitman-Yor process [4]. In fact, the hierarchal Pitman-Yor language models can achieve a slightly superior performance comparing to the Kneser-Ney method in the use for ASR [5].

An alternative approach to the data sparseness problem is clustering. It includes class n-gram LMs [6]. Similar ideas have been employed in decision tree LMs [7] and random forest LMs [8], in which context information is clustered into some groups. Neural network based LMs can also mitigate the data sparseness in another way, in which the dimensionality of word space is reduced [9, 10]. These methods are trained based on different criteria in learning. The combination of these methods is known to be effective in ASR [11].

The latent words language models (LWLM) was recently proposed in the machine learning area [12]. LWLMs are trained taking into account latent words and this is the approach of the LWLM to the data sparseness problem. Latent words are a class in fact. Similar words to a latent word have similar probabilities. For example, fruit related words are occurred with a high probability in the latent word (class) of 'orange'. If a technical word and a general word are similar in terms of syntactic role, the LWLM is trained so that these words have a similar probability. This means that an LWLM trained with an academic lecture corpus may accurately perform with other corpora. In short, the LWLM is expected to perform robustly over multiple domains. Furthermore, the criterion of training an LWLM much differs from that for a standard word n-grams LM. Therefore, the interpolation of the both LMs would be effective to degrade word error rate (WER) in ASR.

However, it is difficult to use an LWLM for one-pass decoding because of soft-decision clusters. Each word appears in all latent words. This seriously affects the search process. This is described in detail in 3.3 section. To overcome this problem, we propose a method based on sampling of words. First a text is made by sampling words in random on an LWLM. Then a standard word n-gram LM is trained from the generated text. In this paper, the HPYLM is used for this purpose. This model approaches the LWLM as increasing sampled words. In fact, a similar sampling approach has been succeed in neural network based LMs [13].

This paper is organized as follows. First, HPYLM is briefly described in section 2. Section 3 explains LWLM and the sampling based method. Section 4 describes our experimental results and then section 5 concludes this paper.

2. HIERARCHICAL PITMAN-YOR LANGUAGE MODEL

HPYLM is a theoretically elegant Bayesian language model based on the Pitman-Yor process, which was first proposed in the machine learning field [4].

HPYLM is a kind of n-gram LMs, which defines probability distribution over current word w_k given context $\boldsymbol{u}_k = w_{k-n+1}, ..., w_{k-1}$. The probability distribution is defined as Eq. (1).

$$P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{X}) = \sum_{\boldsymbol{S}} P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{S})P(\boldsymbol{S}|\boldsymbol{X}). \quad (1)$$

$\boldsymbol{X}$ is training text data, and $\boldsymbol{S}$ is a seating arrangement defined by the Chinese restaurant franchise representation of the Pitman-Yor process. In HPYLM, Bayesian inference is analytically intractable, so the Gibbs sampling technique is used as a feasible approximation by Eq. (2).

$$P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{X}) \approx \frac{1}{I}\sum_{i=1}^{I} P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{S}_i). \quad (2)$$

$P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{X})$ can be approximately obtained by collecting I samples of $\boldsymbol{S}$. Under a seating arrangement $\boldsymbol{S}$, $P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{S})$ is calculated by Eq. (3).

$$P_{\text{hpy}}(w_k|\boldsymbol{u}_k, \boldsymbol{S}) = \frac{c(w_k, \boldsymbol{u}_k) - d_{|\boldsymbol{u}_k|}t(w_k, \boldsymbol{u}_k)}{\theta_{|\boldsymbol{u}_k|} + c(\boldsymbol{u}_k)} + \frac{\theta + d_{|\boldsymbol{u}_k|}t(\boldsymbol{u}_k)}{\theta_{|\boldsymbol{u}_k|} + c(\boldsymbol{u}_k)} P_{\text{hpy}}(w_k|\pi(\boldsymbol{u}_k), \boldsymbol{S}). \quad (3)$$

$\pi(\boldsymbol{u}_k)$ is the shortened context obtained by removing the earliest word from $\boldsymbol{u}_k$. $c(w_k, \boldsymbol{u}_k)$ and $t(w_k, \boldsymbol{u}_k)$ are parameters based on the Chinese restaurant franchise representation. $d_{|\boldsymbol{u}_k|}$ and $\theta_{|\boldsymbol{u}_k|}$ are discount and strength parameters of the Pitman-Yor process, respectively.

3. LATENT WORDS LANGUAGE MODEL

3.1. Definition

LWLMs are generative models with a latent variable for every observed word in a text. The structure of LWLM is shown in Fig. 1. The latent variable, called latent word h_k, is generated by its context $\boldsymbol{l}_k = h_{k-n+1}, ..., h_{k-1}$, and observed word w_k is generated from latent word h_k, i.e.,

$$h_k \sim P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}), \quad (4)$$

$$w_k \sim P(w_k|h_k, \boldsymbol{\Theta}). \quad (5)$$

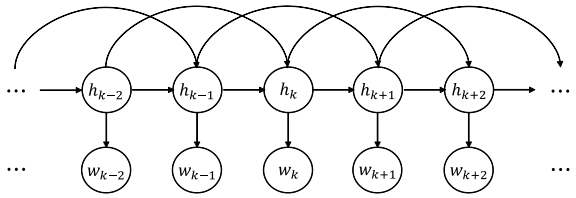

Fig. 1. Structure of LWLM.

$\boldsymbol{\Theta}$ is model parameters of LWLM. LWLM has a similar structure to the standard class n-gram LM, as noted in Eq. (4) and Eq. (5). The latent word corresponds approximately to the class of the stamdard class n-gram LM. LWLM has soft clustering structure which is different from a simple hard clustering structure. In the hard clustering structure, word only belongs to one class. But, in the soft clustering structure, word belongs to multi classes. In fact, word belongs to all the classes in LWLM. If word w_k is related to latent word h_k, it has high probability $P(w_k|h_k)$; conversely, its probability is low if w_k not similar to h_k.

3.2. Inference

Inference in LWLM is to estimate latent words assignments $\boldsymbol{H}$ of all observed words $\boldsymbol{W}$ in the text data. In fact, $\boldsymbol{\Theta}$ means $\boldsymbol{W}$ and $\boldsymbol{H}$. We use the Gibbs sampling for inference. A probability distribution of possible values for latent word h_k is computed by Eq. (6).

$$P(h_k|\boldsymbol{\Theta}^{-k}) = \frac{P(w_k|h_k, \boldsymbol{\Theta}^{-k})\prod_{j=k}^{k+n-1} P(h_j|\boldsymbol{l}_j, \boldsymbol{\Theta}^{-k})}{\sum_{h_k \in V} P(w_k|h_k, \boldsymbol{\Theta}^{-k})\prod_{j=k}^{k+n-1} P(h_j|\boldsymbol{l}_j, \boldsymbol{\Theta}^{-k})}. \quad (6)$$

$\boldsymbol{\Theta}^{-k}$ denotes $\boldsymbol{W}$ and $\boldsymbol{H}^{-k}$ that is all latent words except for h_k. Gibbs sampling can be realized to sample a new value for the latent word according to this distribution and place it at position k.

Arbitrary smoothing methods can be applied to the probability distributions $P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta})$ and $P(w_k|h_k, \boldsymbol{\Theta})$. Although Kneser-Ney smoothing is used in [12], in this paper, we apply the Bayesian framework to the two probability distributions. We construct $P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta})$ as HPYLM, and apply the Dirichlet smoothing to $P(w_k|h_k, \boldsymbol{\Theta})$ [14], which are calculated as Eq. (7) and Eq. (8), respectively.

$$P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}) = P_{\text{hpy}}(h_k|\boldsymbol{l}_k, \boldsymbol{H}), \quad (7)$$

$$P(w_k|h_k, \boldsymbol{\Theta}) = \frac{c_0(w_k, h_k) + \alpha P_0(w_k)}{c_0(h_k) + \alpha}. \quad (8)$$

$P_0(w_k)$ is the ML estimation value of unigram probability in the training text data. $c_0(w_k, h_k)$ and $c_0(h_k)$ are counts calculated from $\boldsymbol{W}$ and $\boldsymbol{H}$. α is a hyper parameter used by Dirichlet smoothing.

3.3. Problems in direct use for ASR

If we directly implement LWLM to one-pass decoding, we have to calculate the probability distribution over current word w_k given context $\boldsymbol{u}_k$,

$$P_{\mathrm{lw}}(w_k|\boldsymbol{u}_k, \boldsymbol{X}) \approx \frac{1}{\mathrm{T}} \sum_{\tau=1}^{\mathrm{T}} \sum_{\boldsymbol{l}_k, h_k} P(w_k|h_k, \boldsymbol{\Theta}_\tau) P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}_\tau). \quad (9)$$

It is shown in Eq. (9) that we have to consider two kinds of summation. First, we have to consider all possible class assignment since LWLM has a soft clustering structure. In the case of a hard clustering structure such as standard class n-gram LM, class assignment can be identified uniquely, so we only have to calculate one $P(w_k|h_k, \boldsymbol{\Theta}_\tau)$ and one $P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}_\tau)$. In the case of the soft clustering structure, however, we have to calculate $P(w_k|h_k, \boldsymbol{\Theta}_\tau)$ and $P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}_\tau)$ with each combination of $\boldsymbol{l}_k$ with h_k. It is impractical to compute them for online decoding.

Second, LWLM uses T instances of $\boldsymbol{\Theta}$ for Bayesian inference. This means that we have to possess T class n-gram structures for decoding. It is hard because enormous memory is needed to handle each class n-gram LM.

3.4. Sampling based approximation

We propose a method that approximates an LWLM to use in ASR. As LWLM is a generative model, it is possible to generate latent words and observed words. After sampling words, we train a standard word based n-gram LM from the observed words generated in random based on **Algorithm 1**.

Algorithm 1 Random sampling on LWLM.

for $\kappa = 1$ to K **do**
  $\boldsymbol{\Theta}_\tau \sim P(\boldsymbol{\Theta}_\tau) = \frac{1}{\mathrm{T}}$
  $h_\kappa \sim P(h_\kappa|\boldsymbol{l}_\kappa, \boldsymbol{\Theta}_\tau)$
  $w_\kappa \sim P(w_\kappa|h_\kappa, \boldsymbol{\Theta}_\tau)$
end for

Through iterations, we can obtain a large number of sentences. By iterating K times, we can generate K latent words, and K observed words. We only use observed words for word n-gram LM estimation. It can be expected that the word n-gram LM approaches the LWLM as increasing the iterations. For the word n-gram LM, we used HPYLM.

Table 1. Experimental data set.

	Domain	# of words
Training	Lecture	7,317,392
Development	Lecture	28,046
Test A	Lecture	27,907
Test B	Contact center	24,665
Test C	Voice mail	21,044

4. EXPERIMENTS

4.1. Experimental conditions

Our Experiments employed the Corpus of Spontaneous Japanese (CSJ) [15]. We divided the CSJ into training set, development set, and test set. In addition, we used a contact center task and a voice mail task for evaluation in out-of-domain environments. Table 1 shows detail.

We used triphone HMM acoustic models for each domain. The speech recognition decoder is VoiceRex, a WFST-based decoder [16, 17]. JTAG was used as the morpheme analyzer to split sentence into words [18].

In this paper, we trained trigram LM and count cutoff pruning was not used. Vocabulary size of the training data was 83,536. We compared four methods:

1. MKNLM: Word N-gram LM with Modified Kneser-Ney constructed from the training set.
2. HPYLM: Hierarchical Pitman-Yor LM constructed from the training set.
3. LWLM: LWLM based on sampling-based approximation.
4. LWLM+HPYLM: Mixed model which combined both HPYLM and LWLM by linear interpolation.

We used 200 iterations for burn-in, and collected 10 samples to train HPYLM. And we used 500 iterations for burn-in, and collected 10 samples to train LWLM. The interpolation weights and hyper parameters were optimized for the development set.

4.2. Experimental results

We investigated the relation between data size generated by random sampling and perplexity (PPL) reduction. We constructed LWLM and LWLM+HPYLM by varying the generated data size and computed the corresponding PPL. We plot the results in Fig. 2, where the horizontal axis is in log-scale.

Fig. 2 shows that PPL by LWLM was reduced as the generated data size increased. When 10^3 M words were generated, PPL by LWLM was comparable to that by HPYLM,

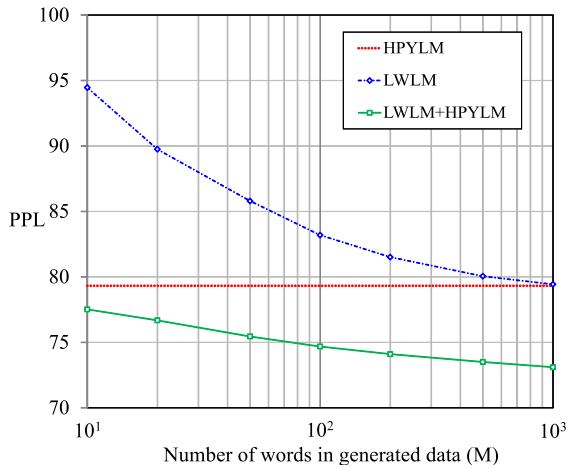

Fig. 2. Relations between data size generated by random sampling and perplexity

Table 2. PPL results.

Setup	Test A	Test B	Test C
MKNLM	79.32	164.07	189.91
HPYLM	67.50	158.13	175.62
LWLM	66.93	141.34	147.87
LWLM+HPYLM	**62.05**	**134.65**	**141.23**

Table 3. WER results (%).

Setup	Test A	Test B	Test C
MKNLM	28.80	49.32	40.78
HPYLM	27.94	48.72	40.68
LWLM	27.85	46.86	38.71
LWLM+HPYLM	**26.42**	**46.19**	**37.92**

and PPL reduction approached convergence. This result shows that we can construct LWLM comparable to HPYLM if we generate sufficient text data. Moreover, highest performance was achieved with LWLM+HPYLM. This results shows that LWLM possesses properties different from those of the HPYLM, and further improvement is achieved if they are combined.

In the evaluation for each test set, we used generated data size: 10^3 M. PPL results are shown in Table 2, and WER results are shown in Table 3.

With regard to Test A, same domain as training and development set, the PPL results are similar to those from the evaluation of development set. LWLM is comparable to HPYLM, and LWLM+HPYLM achieved highest performance. In the WER result, we obtained WER reduction by LWLM+HPYLM compared to HPYLM.

On the other hand, in Test B and C, out-of-domains, LWLM achieved remarkably high performance compared to HPYLM in terms of PPL and WER results. This result shows that LWLM robustly handles speech domains different from that of the training data. It seems that the learning criteria, which identify related words, are effective in expanding the versatility of LMs.

5. CONCLUSIONS

In this paper, we applied LWLM to ASR. LWLM can consider the related words in the given context, and our expectation was that LWLM could robustly handle various domain, and that its combination with standard word n-gram LM would be effective.

To implement one pass decoding, we proposed a method which approximates LWLM as a structure suitable for ASR. We randomly generate text data according to the stochastic process in LWLM, and train standard word based n-gram LM from the generated text data.

Experiments showed that LWLM provided comparable to HPYLM if the speech has the same domain as the training set, and performed robustly over multiple domains. Moreover, we could achieve the highest performance by combining HPYLM with LWLM.

6. REFERENCES

[1] Joshua T. Goodman, "A bit of progress in language modeling," *Computer Speech & Language*, vol. 15, pp. 403–434, 2001.

[2] Stanley F. Chen and Joshua Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, pp. 359–383, 1999.

[3] Reinhard Kneser and Hermann Ney, "Improved backing-off for m-gram language modeling," *In Proc. ICASSP*, vol. 1, pp. 181–184, 1995.

[4] Yee Whye Teh, "A hierarchical bayesian language model based on pitman-yor processes," *In Proc. COLING/ACL 2006*, pp. 985–992, 2006.

[5] Songfang Huang and Marc Yor, "Hierarchical pitman-yor language models for asr in meetings," *In Proc ASRU 2007*, pp. 124–129, 2007.

[6] Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18, pp. 467–479, 1992.

[7] Gerasimos Potamianos and Frederick Jelinek, "A study of n-gram and decision tree letter language modeling methods," *Speech Communication*, vol. 24, no. 3, pp. 171–192, 1998.

[8] Peng Xu and Frederick Jelinek, "Random forests in language modeling," *In Proc. EMNLP 2004*, pp. 325–332, 2004.

[9] Yoshua Bengio, Rejean Ducharme, Pascal Vincent, and Christian Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[10] Stefan Kombrink, Tomas Mikolov, Martin Karafiat, and Lukas Burget, "Recurrent neural network based language modeling in meeting recognition," *In Proc. Interspeech 2011*, pp. 2877–2880, 2011.

[11] Mikolov Tomas, Deoras Anoop, Kombrink Stefan, Burget Lukas, and Cernocky Jan, "Empirical evaluation and combination of advanced language modeling techniques," *In Proc. Interspeech 2011*, vol. 605-608, 2011.

[12] Koen Deschacht, Jan De Belder, and Marie-Francine Moens, "The latent words language model," *Computer Speech & Language*, vol. 26, pp. 384–409, 2012.

[13] Anoop Deoras, Tomas Mikolov, Stefan Kombrink, Martin Karafiat, and Sanjeev Khudanpur, "Variational approximation of long-span language models in lvcsr," *In Proc. ICASSP 2011*, pp. 5532–5535, 2011.

[14] David J. C. MacKay and Linda C. Peto, "A hierarchical dirichlet language model," *Natural language engineering*, vol. 1, pp. 289–308, 1994.

[15] Kikuo Maekawa, Hanae Koiso, Sadaoki Furui, and Hitoshi Isahara, "Spontaneous speech corpus of japanese," *In proc.LREC*, pp. 947–952, 2000.

[16] Takaaki Hori, Chiori Hori, Yasuhiro Minami, and Atsushi Nakamura, "Efficient wfst-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition," *IEEE transactions on Audio, Speech and Language Processing*, vol. 15, no. 4, pp. 1352–1365, 2007.

[17] Hirokazu Masataki, Daisuke Shibata, Yuichi Nakazawa, Satoshi Kobashikawa, Atsunori Ogawa, and Katsutoshi Ohtsuki, "Voicerex spontaneous speech recognition technology for contact-center conversations," *NTT Tech. Rev.*, vol. 5, no. 1, pp. 22–27, 2007.

[18] Takeshi Fuchi and Shinichiro Takagi, "Japanese morphological analyzer using word co-occurence-jtag," *In Proc. COLING-ACL*, pp. 409–413, 1998.