Viterbi Decoding for Latent Words Language Models Using Gibbs Sampling

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

10.21437/Interspeech.2013-751

Abstract

This paper introduces a new approach that directly uses latent words language models (LWLMs) in automatic speech recognition (ASR). LWLMs are effective against data sparseness because of their soft-decision clustering structure and Bayesian modeling so it can be expected that LWLMs perform robustly in multiple ASR tasks. Unfortunately, implementing a LWLM to ASR is difficult because of its computation complexity. In our previous work, we implemented an approximate LWLM for ASR by sampling words according to a stochastic process and training a word n-gram LMs. However, the previous approach cannot take into account the latent variable sequence behind the recognition hypothesis. To solve this problem, we propose a method based on Viterbi decoding that simultaneously decodes the recognition hypothesis and its latent variable sequence. In the proposed method, we use Gibbs sampling for rapid decoding. Our experiments show the effectiveness of the proposed Viterbi decoding based on n-best rescoring. Moreover, we also investigate the effects on the combination of the previous approximate LWLM and the proposed Viterbi decoding.

Index Terms: Latent words language model, Viterbi decoding, Gibbs sampling, n-best rescoring.

1. Introduction

Language models (LMs) are necessary for several natural language processing tasks such as automatic speech recognition (ASR). One of the most common problems faced by LMs is data sparseness [1]. In ASR, LMs are often required to robustly predict the probability of unobserved linguistic phenomena even though the training data is limited. To mitigate the data sparseness problem, several techniques have been proposed. Various smoothing methods that improve LMs probability estimation have been studied for n-gram LMs [2, 3]. One candidate is based on dimensionality reduction. Class-based n-gram LMs [4] and decision tree LMs [5] are based on word classification, and Neural network based LMs are based on learning the distributed representation of words [6].

In addition, our interest has been piqued by the latent words language models (LWLMs) since they can flexibly realize smoothing and dimensionality reduction [7]. LWLMs have latent variables which are called latent words. LWLM training takes account of latent words so overcomes the data sparseness problem. Remarkably, LWLM has a soft clustering structure in common with Bayesian hidden Markov models (HMMs) [8, 9] and the Bayesian class-based LMs [10, 11]. In contrast to these models, LWLM has a vast latent variable space whose size is equivalent to the vocabulary size of the training data. These flexible attributes help us to efficiently realize the smoothing and the dimensionality reduction. Therefore, it can be expected that LWLM robustly covers multiple domains in ASR. However, some approximation is inevitable for in ASR implementation because these attributes seriously increase computation complexity.

In our previous work, we proposed a method that approximates LWLM as a structure suitable for ASR [12]. We randomly generate text data according to a stochastic process and train a word-based n-gram LM (hierarchical Pitman-Yor LM [3]) from the generated data. The approximate LWLM provides results comparable to the standard n-gram LM if the test speech has the same domain as the training data, and performs robustly over multiple domains. However, the approximate LWLM cannot take into account the latent words behind the recognition hypothesis because the LWLM approximation is based on simple n-gram LM. In fact, the concept of LWLM is that every word in a text has a latent word. If we can directly take into account the latent words, further ASR improvements seem likely.

Accordingly, this paper describes a new approach based on Viterbi decoding for LWLMs that can directly take account of the latent words. Viterbi decoding simultaneously decodes a recognition hypothesis and its latent variable sequence using the joint probability between the two sequences. This technique is usually used in HMMs and it is known that the decoding performance is comparable to taking account of all possible latent variable assignments [13]. We implement the Viterbi decoding proposal as a two-pass process because there are innumerable combinations of the recognition hypothesis and its latent word assignment in LWLM. We preliminarily decode the several recognition hypotheses in a first pass, and then decode each latent word assignment in a second pass. If the recognition hypotheses are given, we only have to find each optimal latent word assignment.

Based on dynamic programming methods, the Viterbi algorithm is a formal technique for Viterbi decoding [14]. The computation complexity to determine best latent words via the Viterbi algorithm is, however, $O(|V|^n)$; where $|V|$ is the size of the latent variable space and n is the n-gram order for the latent variable. It is difficult to apply the Viterbi algorithm to LWLM because the latent variable space of LWLM is vast and n-gram order is usually over two.

To overcome this problem, we use Gibbs sampling to find the approximately best latent words [15, 16]. Gibbs sampling technique is a simple and widely used method for generating random samples from a joint distribution over several variables. We sample several latent word assignments based on Gibbs sampling and find the best of the samples. Gibbs sampling can reduce the computation complexity from $O(|V|^n)$ to $O(|V|)$ so we can conduct Viterbi decoding more rapidly than is possible with the formal Viterbi algorithm.

The organization of this paper is as follows. We detail LWLM based on Bayesian inference in Section 2. Our Viterbi decoding proposal and the Gibbs sampling based method are addressed in Section 3. Section 4 describes our experiments and Section 5 concludes this paper.

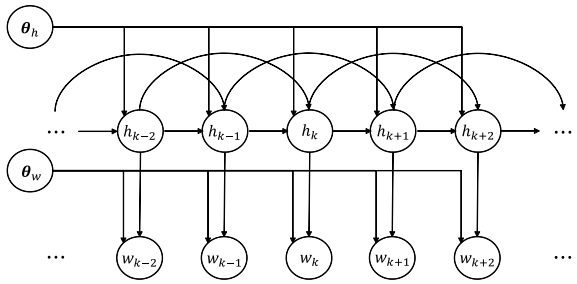

Figure 1: *Structure of latent words language model.*

2. Latent words language model

2.1. Definition

LWLMs are generative models with a latent variable for every observed word. The structure of LWLM is shown in Figure 1. The latent variable, called latent word h_k, is generated from a transition probability distribution given its context $\boldsymbol{l}_k = h_{k-n+1}, ..., h_{k-1}$. Observed word w_k is generated from an emission probability distribution given latent word h_k, i.e.,

$$h_k \sim P(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h), \tag{1}$$

$$w_k \sim P(w_k|h_k, \boldsymbol{\theta}_w). \tag{2}$$

$\boldsymbol{\theta}_h$ are model parameters of the transition probability distribution and $\boldsymbol{\theta}_w$ are model parameters of the emission probability distribution. $P(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h)$ is expressed as an n-gram for latent words, and $P(w_k|h_k, \boldsymbol{\theta}_w)$ models the dependency between the observed word and the latent word. If word w_k is related to latent word h_k, $P(w_k|h_k, \boldsymbol{\theta}_m)$ has a high probability. Conversely, this probability is low if w_k is not related to h_k.

LWLM has a soft clustering structure that differs from a simple hard clustering structure. In the hard clustering structure, one word belongs to only one class. In the soft clustering structure, on the other hand, one word belongs to multiple classes. In fact, each word belongs to all classes in LWLM. In addition, a latent word is expressed as a specific word that can be selected from entire vocabulary V. Thus, the number of the latent words and the number of the observed words are $|V|$.

2.2. Training based on Bayesian inference

Bayesian inference is suitable for training LWLM. Bayesian inference of LWLM produces the following predictive distribution of observed words $\boldsymbol{w}$ given training data $\boldsymbol{W}$:

$$\begin{aligned} P(\boldsymbol{w}|\boldsymbol{W}) &= \sum_{\boldsymbol{h}} \int P(\boldsymbol{w}|\boldsymbol{h}, \boldsymbol{\Theta}) P(\boldsymbol{h}|\boldsymbol{\Theta}) P(\boldsymbol{\Theta}|\boldsymbol{W}) d\boldsymbol{\Theta}, \\ &= \sum_{\boldsymbol{h}} \int \prod_{k=1}^{L} \left[P(w_k|h_k, \boldsymbol{\Theta}) P(h_k|\boldsymbol{l}_k, \boldsymbol{\Theta}) \right] P(\boldsymbol{\Theta}|\boldsymbol{W}) d\boldsymbol{\Theta}, \end{aligned} \tag{3}$$

where $\boldsymbol{\Theta}$ denotes $\{\boldsymbol{\theta}_h, \boldsymbol{\theta}_w\}$, $\boldsymbol{h}$ is a latent word assignment, and L is observed word length. Bayesian inference of LWLM takes account of all possible model parameters. As the integral with respect to $\boldsymbol{\Theta}$ is analytically intractable, a sampling technique is used as a feasible approximation. Eq. (3) is approximated as,

$$P(\boldsymbol{w}|\boldsymbol{W}) \simeq \frac{1}{T} \sum_{\boldsymbol{h}} \sum_{\tau=1}^{T} P(\boldsymbol{w}|\boldsymbol{h}, \boldsymbol{\Theta}_\tau) P(\boldsymbol{h}|\boldsymbol{\Theta}_\tau). \tag{4}$$

The posterior predictive distribution can be approximated using T instances of $\boldsymbol{\Theta}$. Although the point estimation, which uses one instance of $\boldsymbol{\Theta}$, is available, we conduct ensemble modeling. In fact, the ensemble of several models is effective for LMs such as random class-based LMs [17] and random forest LMs [18].

An arbitrary prior distribution can be applied for each model parameter. In this paper, we use a hierarchical Pitman-Yor prior for $\boldsymbol{\theta}_h$. $P(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h)$ is given as,

$$P(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h) = \frac{1}{I} \sum_{i=1}^{I} P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k, \boldsymbol{S}_i). \tag{5}$$

where $\boldsymbol{S}$ is a seating arrangement defined by the Chinese restaurant franchise representation of the Pitman-Yor process [19]. $P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k, \boldsymbol{\theta}_h)$ can be approximately obtained by collecting I samples of $\boldsymbol{S}$. Under seating arrangement $\boldsymbol{S}$, $P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k, \boldsymbol{S})$ is obtained as,

$$\begin{aligned} P_{\mathrm{hpy}}(h_k|\boldsymbol{l}_k, \boldsymbol{S}) = &\frac{c(h_k, \boldsymbol{l}_k) - d_{|\boldsymbol{u}_k|} t(h_k, \boldsymbol{l}_k)}{\theta_{|\boldsymbol{l}_k|} + c(\boldsymbol{l}_k)} \\ &+ \frac{\theta + d_{|\boldsymbol{l}_k|} t(\boldsymbol{l}_k)}{\theta_{|\boldsymbol{l}_k|} + c(\boldsymbol{l}_k)} P_{\mathrm{hpy}}(h_k|\pi(\boldsymbol{l}_k), \boldsymbol{S}), \end{aligned} \tag{6}$$

where $\pi(\boldsymbol{l}_k)$ is the shortened context obtained by removing the earliest word from $\boldsymbol{l}_k$. $c(h_k, \boldsymbol{l}_k)$ and $t(h_k, \boldsymbol{l}_k)$ 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. Moreover, we use a Dirichlet prior for $\boldsymbol{\theta}_w$. $P(w_k|h_k, \boldsymbol{\theta}_w)$ is given as,

$$P(w_k|h_k, \boldsymbol{\theta}_w) = \frac{c_0(w_k, h_k) + \alpha P_0(w_k)}{c_0(h_k) + \alpha}, \tag{7}$$

where $P_0(w_k)$ is the ML estimation value of unigram probability in the training data $\boldsymbol{W}$. $c_0(w_k, h_k)$ and $c_0(h_k)$ are counts calculated from $\boldsymbol{W}$ and its latent word assignment $\boldsymbol{H}$. α is a hyper parameter used by Dirichlet smoothing [20].

For training LWLM, i.e. for calculating Eqs (5) to (7), we have to inference a latent word assignment $\boldsymbol{H}$ behind training data $\boldsymbol{W}$. To this end, Gibbs sampling is suitable. A conditional probability distribution of possible values for latent word h_k is obtained as,

$$\begin{aligned} &P(h_k|\boldsymbol{W}, \boldsymbol{H}^{-k}, \boldsymbol{\Theta}^{-k}) \\ &\qquad \propto P(w_k|h_k, \boldsymbol{\Theta}^{-k}) \prod_{j=k}^{k+n-1} P(h_j|\boldsymbol{l}_j, \boldsymbol{\Theta}^{-k}), \end{aligned} \tag{8}$$

where $\boldsymbol{H}^{-k}$ represents all latent words except for h_k, and $\boldsymbol{\Theta}^{-k}$ denotes $\{\boldsymbol{\theta}_h^{-k}, \boldsymbol{\theta}_w^{-k}\}$. Gibbs sampling samples a new value for the latent word according to its distribution and places it at position k in $\boldsymbol{H}$. Along with $\boldsymbol{H}$, the model parameters are updated by recalculating Eqs (5) to (7). This procedure is iterated until convergence is achieved.

2.3. Approximate LWLM

In our previous work, we approximated LWLM as a structure suitable to implement LWLM to ASR because it is impractical to rigorously compute the posterior probability $P(\boldsymbol{w}|\boldsymbol{W})$ [12]. If we compute the posterior probability, we must consider all possible latent word assignments. In fact, the computation complexity based on the forward algorithm is $O(|V|^n)$; where $|V|$ is the size of latent variable space and n is order of n-gram

about latent variable [13]. In LWLM, $|V|$ is equivalent to the vocabulary size of the training data and often reaches into the tens of thousands, and n is often over two. This computation complexity is much larger than that in standard HMMs.

The approximate LWLM is constructed by randomly generating text data according to the stochastic process of Eqs (1) and (2), and training word based n-gram LM from the generated data. In addition, we mixed the standard n-gram LM and the approximate LWLM. The previous method was effective for ASR but cannot guarantee an enough approximation. LWLM has the concept that the observed words are actually generated from single latent words. Therefore, we should consider the single latent word assignment behind the observed words.

3. Viterbi decoding for LWLM

3.1. Two-pass Viterbi decoding

We propose here a method based on Viterbi decoding for LWLM in order to directly take into account the latent words. The Viterbi decoding simultaneously decodes the recognition hypothesis $\boldsymbol{w}$ and its latent word assignment $\boldsymbol{h}$ using the joint probability $P(\boldsymbol{w}, \boldsymbol{h}|\boldsymbol{W})$. The joint probability is defined as,

$$P(\boldsymbol{w}, \boldsymbol{h}|\boldsymbol{W}) = \frac{1}{T}\sum_{\tau=1}^{T} P(\boldsymbol{w}|\boldsymbol{h}, \boldsymbol{\Theta}_\tau)P(\boldsymbol{h}|\boldsymbol{\Theta}_\tau). \quad (9)$$

In LWLM, Viterbi decoding is impractical to implement as a one-pass process because we have to consider innumerable combinations of the recognition hypothesis and its latent word assignment. Therefore, we conduct the two-pass Viterbi decoding. In a first pass, we preliminarily decode the several recognition hypotheses using the standard n-gram LM to narrow down the search space, and then, estimate each optimal latent word assignment in a second pass. This computation cost is much smaller than intuitive one-pass Viterbi decoding.

Given recognition hypothesis $\boldsymbol{w}$, we only have to find the optimal latent word assignment $\hat{\boldsymbol{h}}$. The optimal latent word assignment $\hat{\boldsymbol{h}}$ are ,with regard to $\boldsymbol{w}$, obtained as,

$$\begin{aligned}\hat{\boldsymbol{h}} &= \arg\max_{\boldsymbol{h}} P(\boldsymbol{h}|\boldsymbol{w}, \boldsymbol{W}) \\ &= \arg\max_{\boldsymbol{h}} \frac{1}{T}\sum_{\tau=1}^{T} P(\boldsymbol{w}|\boldsymbol{h}, \boldsymbol{\Theta}_\tau)P(\boldsymbol{h}|\boldsymbol{\Theta}_\tau). \quad (10)\end{aligned}$$

The joint probability $P(\boldsymbol{w}, \hat{\boldsymbol{h}}|\boldsymbol{W})$ is used as a language model score for ASR decoding. We can also use $P(\boldsymbol{w}, \hat{\boldsymbol{h}}|\boldsymbol{W})$ in conjunction with the score calculated by the standard n-gram LM.

3.2. Viterbi algorithm for LWLM

The Viterbi algorithm concurrently solves the problems of finding the optimal latent word assignment and computing the joint probability based on dynamic programming methods. The joint probability $P(\boldsymbol{w}, \hat{\boldsymbol{h}}|\boldsymbol{W})$ can, based on the Viterbi algorithm, be obtained as,

$$P(\boldsymbol{w}, \hat{\boldsymbol{h}}|\boldsymbol{W}) = \max_{\boldsymbol{l}_{L+1}} \sum_{\tau=1}^{T} \delta(\boldsymbol{l}_{L+1}, \boldsymbol{\Theta}_\tau). \quad (11)$$

$\delta(\boldsymbol{l}_{i+1}, \boldsymbol{\Theta}_\tau)$ is the joint probability of observing $w_1, \cdots, w_i$ together with sequence $\boldsymbol{l}_{i+1} = h_{i-n+2}, \cdots, h_i$. It is defined recursively as,

$$\delta(\boldsymbol{l}_{i+1}, \boldsymbol{\Theta}_\tau) = P(w_i|h_i, \boldsymbol{\Theta}_\tau)\delta(\boldsymbol{l}_i^*, \boldsymbol{\Theta}_\tau)P(h_i|\boldsymbol{l}_i^*, \boldsymbol{\Theta}_\tau), \quad (12)$$

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

where $\boldsymbol{l}_i^* = h_{i-n+1}^*, h_{i-n+2}, \cdots, h_{i-1}$. h_{i-n+1}^* is the optimal latent word through $\boldsymbol{l}_{i+1}$. If we do not conduct ensemble modeling, it is easy to determine h_{i-n+1}^*. But in ensemble modeling, h_{n-i+1}^* is determined according to Eq. (13).

$$\begin{aligned}&h_{i-n+1}^* = \\ &\arg\max_{h_{i-n+1}} \sum_{\tau=1}^{T} P(w_k|h_k, \boldsymbol{\Theta}_\tau)\delta(\boldsymbol{l}_i, \boldsymbol{\Theta}_\tau)P(h_i|\boldsymbol{l}_i, \boldsymbol{\Theta}_\tau). \quad (13)\end{aligned}$$

The computation complexity via Viterbi algorithm is equal to that of the forward algorithm, that is $O(|V|^n)$. As noted previously, it is difficult to use the Viterbi algorithm directly in LWLM.

3.3. Viterbi decoding based on Gibbs sampling

We use the Gibbs sampling technique in LWLM in order to find the approximately optimal latent word assignment more rapidly. First, we sample several latent words assignments based on Gibbs sampling. A conditional probability distribution of the possible values for latent word h_k is defined as,

$$\begin{aligned}&P(h_k|\boldsymbol{w}, \boldsymbol{h}^{-k}, \boldsymbol{W}) \\ &\quad \propto \sum_{\tau=1}^{T} \big[P(w_k|h_k, \boldsymbol{\Theta}_\tau) \prod_{j=k}^{k+n-1} P(h_j|\boldsymbol{l}_j, \boldsymbol{\Theta}_\tau)\big]. \quad (14)\end{aligned}$$

This equation is similar to Eq. (8). We obtain M samples of latent words assignments $\boldsymbol{h}_1, \cdots, \boldsymbol{h}_M$ and find the optimal one by Eq. (15).

$$P(\boldsymbol{w}, \hat{\boldsymbol{h}}|\boldsymbol{W}) \simeq \max_{\boldsymbol{h}\in\{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_M\}} \frac{1}{T}\sum_{\tau=1}^{T} P(\boldsymbol{w}|\boldsymbol{h}, \boldsymbol{\Theta}_\tau)P(\boldsymbol{h}|\boldsymbol{\Theta}_\tau). \quad (15)$$

The computation complexity based on Gibbs sampling is $O(|V|)$ so we can find the approximately optimal latent words assignment much more rapidly than is possible with the formal Viterbi algorithm.

4. Experiments

4.1. Experimental conditions

Our experiments employed the Corpus of Spontaneous Japanese (CSJ) [21]. We divided the CSJ into training set, development set, and test set also in the previous work [12]. In addition, we used a contact center task and a voice mail task for evaluation in out-of-domain environments; details are shown in Table 1.

We used triphone HMM acoustic models for each domain. The speech recognition decoder is VoiceRex, a WFST-based decoder [22, 23]. JTAG was used as the morpheme analyzer to split sentence into words [24].

Table 3: *Word error rate (WER) results.*

	1-pass	2-pass (1000-best rescoring)	Dev. (%)	Test A (%)	Test B (%)	Test C (%)
1.	HPYLM	-	26.43	27.94	48.72	40.68
1-a.		LWLM(Viterbi)	25.96	27.42	47.93	39.97
1-b.		+LWLM(Viterbi)	25.67	27.15	47.56	39.64
2.	LWLM∗	-	25.85	27.85	46.86	38.71
2-a.		LWLM(Viterbi)	25.61	27.52	46.69	38.35
2-b.		+LWLM(Viterbi)	25.35	27.40	46.55	38.16
3.	LWLM∗+HPYLM	-	24.93	26.42	46.19	37.92
3-a.		LWLM(Viterbi)	25.35	26.85	46.46	37.96
3-b.		+LWLM(Viterbi)	**24.48**	**25.94**	**45.96**	**37.58**

Table 2: *Perplexity (PPL) results.*

Setup	Dev.	Test A	Test B	Test C
HPYLM	79.85	67.50	158.13	175.62
LWLM∗	79.64	66.93	141.34	147.87
LWLM∗+HPYLM	73.85	62.05	134.65	141.34
LWLM(Viterbi)	154.38	129.94	273.64	289.87

We compared the first pass result with the second pass result, which was applied the LWLM using the proposed Viterbi decoding, in terms of perplexity (PPL) and word error rate (WER). The n-best hypotheses generated using the standard n-gram LM were used for Viterbi decoding as a first pass. We used three n-gram LMs for a first pass. Each n-gram LM was trigram LM and count cutoff pruning was not used. Vocabulary size of the training data was 83,536.

1. HPYLM: Hierarchical Pitman-Yor LM constructed from the training set.
2. LWLM∗: LWLM based on sampling-based approximation [12]; we generated one Giga words using LWLM and approximated them as a hierarchical Pitman-Yor LM.
3. LWLM∗+HPYLM: Mixed model which combined both LWLM∗ and HPYLM by linear interpolation.

We used 200 iterations for burn-in, and collected 10 samples to train HPYLM. For standard LWLM training, we used 500 iterations for burn-in, and collected 10 samples.

For 2-pass Viterbi decoding, we generated 1000-best hypotheses using each n-gram LM. The Viterbi decoding was implemented using two methods.

a. LWLM(Viterbi): Replaced the LM score of the first LM into the score of the LWLM decoded using the proposed method.

b. +LWLM(Viterbi): Rescored by the combination of the LM score of the first LM and the score of the LWLM decoded using the proposed method.

For Gibbs sampling in the Viterbi decoding, we used 100 samples of latent word assignment and determined the best assignment. Several parameters such as hyper parameters, the interpolation weights and LM score factors were optimized for the development data.

4.2. Experimental Results

Table 2 shows PPL results on each condition. PPL results shows that LWLM∗ achieved high performance compared to HPYLM. Moreover, LWLM∗+HPYLM provided highest performance. The PPL results of the LWLM(Viterbi) is relatively not so good compared with the other condition. It is because the joint probability between the target data and its optimal latent word assignment was used for computing PPL.

Table 3 shows WER results. In using HPYLM for the first pass, we could achieve further improvement in proposed Viterbi decoding (1-a) compared with the first pass result (1). Moreover, the combination of the first pass score and the Viterbi decoding score (1-b) provided higher performance than each decoding in isolation. It seems that taking account of the latent word assignment of the recognition hypothesis yields characteristics different from only using the standard n-gram LM.

Next, we compared the proposed Viterbi decoding based on LWLM (2-a) with the previous approximate LWLM (2). The Viterbi decoding was more effective than the approximate LWLM, and we obtained further improvement with their combination (2-b). This result shows that the Viterbi decoding based on LWLM possessed properties different from the approximate LWLM. It confirms that we should simultaneously use both implementations.

In addition, we used all possible score (3-b), based on the hierarchical Pitman-Yor LMs, the approximate LWLM, and the joint probability based on Viterbi decoding. This condition achieved the highest performance of all conditions. This performance might be attributed to the efficiency of the proposed Viterbi decoding that directly takes account of the latent words.

5. Conclusions

In this paper, we proposed a LWLM-based Viterbi decoding approach that directly takes account of the latent words. Viterbi decoding well reflects the concept of LWLM in that every word in a text has a latent word. We implement the Viterbi decoding proposal as a two-pass process in which several recognition hypotheses are initially created based on standard decoding using the standard n-gram LM; these hypotheses are then rescored using the joint probability between the recognition hypothesis and the optimal latent word assignment.

To determine the optimal latent word assignment, we used Gibbs sampling which can rapidly finds the optimal latent words. This technique runs far more rapidly than the formal Viterbi algorithm.

Experiments showed that the Viterbi decoding proposal is more effective than the first pass result with standard n-gram LM and the previous approximate LWLM. Moreover, we could achieve the highest performance by combining the hierarchical Pitman-Yor LMs with the approximate modeling of LWLM and the Viterbi decoding of LWLM.

6. References

[1] Joshua T. Goodman, "A bit of progress in language modeling," *Computer Speech & Language*, vol.15, no.4, pp.403-434, 2001.

[2] Stanley F. Chen and Joshua T. Goodman, "An Empirical Study of Smoothing techniques for language modeling," *Computer Speech & Language*, vol.13, no.4, pp.359-383, 1999.

[3] Yhee Whye Teh, "A Hierarchical Bayesian Language Model based on Pitman-Yor Processes," *In Proc. COLING/ACL 2006*, pp.985-992, 2006.

[4] 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, no.4, pp.467-479, 1992.

[5] 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.

[6] 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.

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

[8] Sharon Goldwater and Tom Griffiths, "A fully bayesian approach to unsupervised part-of-speech tagging," *In Proc. ACL 2007*, pp.744-751, 2007.

[9] Phil Blunsom and Trevor Cohn, "A Hierarchical Pitman-Yor Process HMM for Unsupervised Part of Speech Induction," *In Proc. ACL 2011*, pp.865-874, 2011.

[10] Yi Su, "Bayesian class-based language models," *In Proc. ICASSP 2011*, pp.5564-5567, 2011.

[11] Jen-Tzung Chien, Chuang-Hua Chueh, "Dirichlet class language models for speech recognition," *IEEE transactions on Audio, Speech and Language Processing*, vol.19, no.3, pp.1352-1365, 2011.

[12] Ryo Masumura, Hirokazu Masataki, Takanobu Oba, Osamu Yoshioka and Satoshi Takahashi, "Use of latent words language modes in ASR: a sampling-based implementation," *In Proc. ICASSP 2013*, pp.8445-8449, 2013.

[13] Lawrence R. Rabiner, "A tutorial on hidden markov models and selected application in speech recognition," *Proceedings of the IEEE*, vol.77, no.2, pp.257-286, 1989.

[14] G. David Forney, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol.61 no.3, pp.268-278, 1973.

[15] Christian P. Robert, Gilles Celeux and Jean Diebolt "Bayesian Estimation of Hidden Markov Chains: A Stochastic Implementation," *Statistics & Probability Letters*, vol.16, pp.77-83, 1993.

[16] Steven L. Scott, "Bayesian methods for hidden markov models: Recursive computing in the 21st century," *Journal of the American Statistical Association*, vol.97, pp.337-351, 2002.

[17] Ahmad Emami and Frederick Jelinek, "Random clusterings for language modeling," *In Proc. ICASSP 2005*, vol.1, pp.581-584, 2005.

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

[19] Yee Whye Teh, Michael. I. Jordan, atthewM. J. Beal and David. M. Blei, "Hierarchical dirichlet processes," *Journal of the American Statistical Association*, vol.101, pp.1566-1581, 2006.

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

[21] Kikuo Maekawa, Hanae Koiso, Sadaoki Furui and Hitoshi Isahara, "Spontaneous speech corpus of Japanese," *In Proc. LREC*, pp.947-952, 2000.

[22] 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.

[23] Hirokazu Masataki, Daisuke Shibata, Yuichi Nakazawa, Satoshi Kobashikawa, Atsunori Ogawa and KatsutoshiOhtsuki, "VoiceRex Spontaneous speech recognition technology for contact-center conversations," *NTT Tech. Rev.*, vol.5, no.1, pp.22-27, 2007.

[24] Takeshi Fuchi and Shinichiro Takagi, "Japanese Morphological Analyzer using Word Co-occurence-JTAG," *In Proc. COLING-ACL*, pp.409-413, 1998.