INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Comparative Study of Parametric and Representation Uncertainty Modeling for Recurrent Neural Network Language Models

Jianwei Yu[1*], *Max W. Y. Lam*[1*], *Shoukang Hu*[1], *Xixin Wu*[1], *Xu Li*[1], *Yuewen Cao*[1], *Xunying Liu*[1], *Helen Meng*[1]

[1]The Chinese University of Hong Kong, Hong Kong SAR, China

{jwyu, wylam, skhu, wuxx, xuli, ywcao, xyliu, hmmeng}@se.cuhk.edu.hk

Abstract

Recurrent neural network language models (RNNLMs) have shown superior performance across a range of tasks, including speech recognition. The hidden layer of RNNLMs plays a vital role in learning the suitable representation of contexts for word prediction. However, the deterministic model parameters and fixed hidden vectors in conventional RNNLMs have limited power in modeling the uncertainty over hidden representations. In order to address this issue, in this paper, a comparative study of parametric and hidden representation uncertainty modeling approaches based on Bayesian gates and variational RNNLMs respectively is investigated on long short-term memory (LSTM) and gated recurrent units (GRU) LMs. Experimental results are presented on two tasks: PennTreebank (PTB) corpus, Switchboard conversational telephone speech (SWBD). Consistent performance improvements were obtained over conventional RNNLMs in terms of both perplexity and word error rate.

Index Terms: Neural network language models, LSTM, GRU variational inference, speech recognition

1. Introduction

Language model (LM) is an essential component in many applications such as speech recognition. The task of LMs is to compute the joint probability of a given sentence $\boldsymbol{W} = (\boldsymbol{w_1}, \boldsymbol{w_2}, ..., \boldsymbol{w_n})$:

$$P(\boldsymbol{W}) = P(\boldsymbol{w_1}, \boldsymbol{w_2}, ..., \boldsymbol{w_n}) = \prod_{t=1}^{n} P(\boldsymbol{w_t}|\boldsymbol{w_{t-1}}, ...\boldsymbol{w_1}) \tag{1}$$

A variety of statistical language models have been proposed in the literature, such as n-gram LMs [1, 2] and neural network LMs (NNLMs) [3, 4, 5]. In recent years recurrent NNLMs (RNNLMs) [5, 6, 7] have been shown to yield state-of-the-art performance on a wide range of tasks.

One key problem in the statistical language models, including RNNLMs, is to learn the suitable representation for long-range context. The hidden layer outputs in RNNLMs are able to encode the complete preceding, and optionally future word contexts [8]. The resulting hidden vector representations and their similarity measures play a crucial role in determining the appropriate clustering of word contexts to be learned to allow a wider generalization over unseen sentences [9]. Depending on the nature of the underlying clustering and associated linguistic regularities [10, 11] being learned, for example, at morphological, syntactical or semantic level [12], uncertainty arises over multiple hidden representations of the same word context.

* Equal contribution

Hence, the use of deterministic hidden representations in standard RNNLMs may have limited power in modeling such uncertainty. In addition, the fixed model parameters in RNNLMs can also indirectly lead to the same issue, although this issue has been widely investigated in previous research [13, 14] in the context of sparsity issue due to limited and variable data.

Motivated by the close relationship between parametric and representational uncertainty issues in RNNLMs, this paper presents a comparative study of two modeling approached to address the potential issues for the state-of-the-art long short-term memory (LSTM) [15] and gated recurrent unit (GRU) [16] RNNLMs: **a)** explicitly modeling uncertainty at the hidden outputs level [17, 18, 19, 20] in the form of variational RNNLMs (VRNNLMS) by integrating over multiple forms of hidden vectors; **b)** replacing the conventional fixed parameters of hidden activations [21, 22] in the RNNLMs using Bayesian estimations marginalizing over multiple parameters estimation.

To the best of our knowledge, this paper is among the first attempt to comparatively study Bayesian and variational approaches for GRU and LSTM LMs to address the issue associated with hidden representation uncertainty. Experiments were conducted on two language modeling tasks: the Penn Treebank (PTB) corpus, and Switchboard conversational telephone speech (SWBD). Both the Bayesian and the variational RNNLMs show consistent improvements in terms of both perplexity and word error rate over the standard RNNLMs.

The rest of this paper is organized as follows. Section 2 gives a brief review of RNNLMs. Section 3 describes the proposed Bayesian LSTM and Bayesian GRU LMs, followed by the description of the VRNNLMs in Section 4. Experiment results are presented in section 5. Section 6 is the conclusion.

2. Recurrent Neural Network Language Models

In recurrent neural network language models (RNNLMs), the word probability is written as:

$$P(\boldsymbol{w}_t|\boldsymbol{w}_{t-1}, ..., \boldsymbol{w_1}) \approx P(\boldsymbol{w}_t|\boldsymbol{w}_{t-1}, \boldsymbol{h_{t-1}}) = P(\boldsymbol{w}_t|\boldsymbol{h_t}), \tag{2}$$

where $\boldsymbol{h_{t-1}} \in \mathbb{R}^H$ is the hidden vector that represents the previous history $(\boldsymbol{w_{t-1}}, ..., \boldsymbol{w_1})$. The RNNLM can be generally divided into three parts: the embedding layer, the recurrent layer and the output layer. The embedding layer projects the one-hot word vector $\boldsymbol{w_t} \in R^N$ into a continuous space $\boldsymbol{x_t} \in \mathbb{R}^M$, where N is vocabulary size and usually $M \ll N$. Followed by the embedding layer, the recurrent layer computes the hidden vector by recursively applying a gated unit $\mathbf{U}$:

$$\boldsymbol{h_t} = \mathbf{U}(\mathbf{w_{t-1}}, \mathbf{h_{t-1}}), \tag{3}$$

http://dx.doi.org/10.21437/Interspeech.2019-1927

which is normally based on sigmoid activations in standard RNNLMs. Finally, the output layer uses the hidden vector h_{t-1} to compute the word probabilities via a softmax function:

$$P(\boldsymbol{w}_t|\boldsymbol{h}_t) = \frac{\exp(\mathbf{v}_{\mathbf{w}_t}\boldsymbol{h}_t)}{\sum_{w\in\mathcal{V}}\exp(\mathbf{v}_{\mathbf{w}}\boldsymbol{h}_t)}, \tag{4}$$

where $\mathbf{v}_{\mathbf{w}}$ is the weight vector for word w in the output layer, and $\mathcal{V}$ denotes the vocabulary.

In practice, it turns out that a simple architecture for RNNLMs (e.g. using a single sigmoid activation) does not perform well in learning long-term dependencies due to the issue of vanishing gradients and significant improvements can be achieved by using two more specific architectures called long short-term memory (LSTM) [15] and gated recurrent unit (GRU) [16], respectively.

2.1. Long Short-Term Memory

In the LSTM architecture, the problem of vanishing gradients is tackled by introducing another recursively computed vector c_t, namely *memory cell*, which aims to preserve the information over longer periods of time. Analogous to the logic gates in an electronic circuit as shown in Fig.1 (a), at time t four gates are computed – the forget gate $\boldsymbol{f_t}$, the input gate $\boldsymbol{i_t}$, the cell gate $\boldsymbol{g_t}$ and the output gate $\boldsymbol{o_t}$:

$$\begin{aligned}
\boldsymbol{f_t} &= \boldsymbol{\sigma}(\boldsymbol{\Theta_f}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T) \\
\boldsymbol{i_t} &= \boldsymbol{\sigma}(\boldsymbol{\Theta_i}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T) \\
\boldsymbol{g_t} &= \tanh(\boldsymbol{\Theta_c}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T) \\
\boldsymbol{o_t} &= \boldsymbol{\sigma}(\boldsymbol{\Theta_o}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T).
\end{aligned} \tag{5}$$

Give the four gating outputs, we update

$$\begin{aligned}
\boldsymbol{c_t} &= \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \boldsymbol{g_t} \\
\boldsymbol{h_t} &= \boldsymbol{o_t} \odot \tanh(\boldsymbol{c_t}),
\end{aligned} \tag{6}$$

where $\odot$ is the element-wise product, $\boldsymbol{\Theta_{(*)}}$ denotes the model parameters, $\boldsymbol{\sigma}(\cdot)$ is sigmoid activation function.

2.2. Gated Recurrent Unit

Similarly to the LSTM unit, the GRU has gated cells that modulate the flow of information inside the unit to adaptively capture dependencies of different time scales, however, without having a separate memory cell. Thus, as shown in Fig.1 (b) the GRU has just three gates called reset $\boldsymbol{r}$, update $\boldsymbol{z}$, and new $\boldsymbol{n}$ gates respectively:

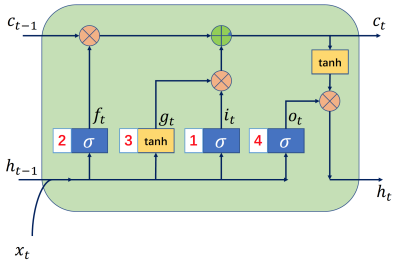
$$\begin{aligned}
\boldsymbol{r_t} &= \boldsymbol{\sigma}(\boldsymbol{\Theta_r}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T) \\
\boldsymbol{z_t} &= \boldsymbol{\sigma}(\boldsymbol{\Theta_z}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T) \\
\boldsymbol{n_t} &= \tanh(\boldsymbol{\Theta_{n,1}}[\boldsymbol{x_{t-1}}, \mathbf{1}] + \boldsymbol{r_t} \odot (\boldsymbol{\Theta_{n,2}}[\boldsymbol{h_{t-1}}, \mathbf{1}])).
\end{aligned} \tag{7}$$

Given the outputs of each gate, the hidden state at time t is computed by

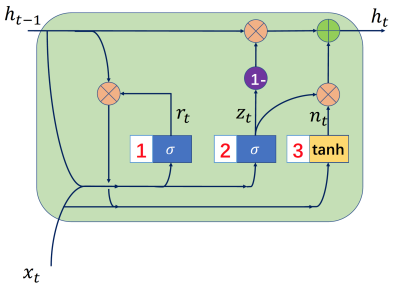
$$\boldsymbol{h_t} = (\mathbf{1} - \boldsymbol{z_t}) \odot \boldsymbol{h_{t-1}} + \boldsymbol{z_t} \odot \boldsymbol{n_t}. \tag{8}$$

3. Bayesian RNNLM

In this section, we first introduce the formulation of Bayesian gate and then present an efficient training algorithm [23, 24, 22, 21] based on variational inference for the Bayesian RNNLM.



(a) LSTM



(b) GRU

Figure 1: *An illustration of LSTM and GRU units. The red numbers (e.g. 1, 2, 3) at each gate denotes the position where Bayesian gate may apply.*

3.1. Bayesian Gate

The hidden representation uncertainty can be addressed by modeling the uncertainty of parameters in RNNLMs. In this way, the model parameters $\boldsymbol{\Theta}$ can be treated as a random variable drawn from a probability distribution $p(\boldsymbol{\Theta})$. As a first milestone, we narrow down the scope of parameters uncertainty modeling to handle only a single gate within a GRU or LSTM unit. In this sense, we investigate how the uncertainty of each gate affects the performance of our model. Note that although modeling all gates as Bayesian gates is theoretically feasible, it is practically expensive to conduct inference and difficult for the investigation of the uncertainty nature of each gate. By this way, Eqn.(2) can be rewritten as:

$$\begin{aligned}
&P(\boldsymbol{w_t}|\boldsymbol{h_{t-1}}, \boldsymbol{w_{t-1}}) = \\
&\int P(\boldsymbol{w_t}|\boldsymbol{h_{t-1}}, \boldsymbol{w_{t-1}}, \boldsymbol{\Theta}^{(p)}) p(\boldsymbol{\Theta}^{(p)}) d\boldsymbol{\Theta}^{(p)},
\end{aligned} \tag{9}$$

where the superscript (p) denotes the position to apply Bayesian gate in the LSTM and GRU units, shown as the red number in Fig. 1. The Bayesian gate is defined as:

$$\boldsymbol{g_t} = \boldsymbol{a}(\boldsymbol{\Theta}^{(p)}[\boldsymbol{x_{t-1}}, \boldsymbol{h_{t-1}}, \mathbf{1}]^T), \ \boldsymbol{\Theta}^{(p)} \sim p(\boldsymbol{\Theta}^{(p)}), \tag{10}$$

where $\boldsymbol{a}(\cdot)$ is the activation function of a specific gate, e.g. for LSTM input gate, $\boldsymbol{a}(\cdot)$ is $\boldsymbol{\sigma}(\cdot)$.

3.2. Variational Training for Bayesian RNNLMs

To estimate the posterior distribution of model parameters $p(\boldsymbol{\Theta}^{(p)}|\mathcal{D})$ the usual approach in Bayesian learning is to maximize the marginal probability in Eqn.(9), where $\mathcal{D}$ is the training set. However, computing this marginal is intractable under the RNNLM framework. Thus, the following variational lower

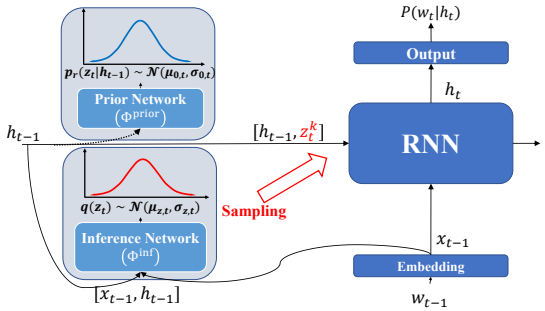


Figure 2: *An illustration of the variation recurrent neural network language model*

bound of log likelihood is often used as an approximation [17]:

$$\log P(\boldsymbol{W}) = \log \int P(\boldsymbol{W}|\boldsymbol{\Theta}^{(p)}) p_r(\boldsymbol{\Theta}^{(p)}) d\boldsymbol{\Theta}^{(p)}$$
$$\geq \underbrace{-\mathrm{KL}(q(\boldsymbol{\Theta}^{(p)})||p_r(\boldsymbol{\Theta}^{(p)}))}_{\mathcal{L}_1} + \underbrace{\int q(\boldsymbol{\Theta}^{(p)}) \log P(\boldsymbol{W}|\boldsymbol{\Theta}^{(p)}) d\boldsymbol{\Theta}^{(p)}}_{\mathcal{L}_2}, \tag{11}$$

where $q(\boldsymbol{\Theta}^{(p)})$ is the variational approximation of the posterior distribution $p(\boldsymbol{\Theta}^{(p)}|\mathcal{D})$, $\mathrm{p_r}(\boldsymbol{\Theta})$ is the prior distribution of $\boldsymbol{\Theta}^{(p)}$ and $\mathrm{KL}(q(\boldsymbol{\Theta}^{(p)})||p_r(\boldsymbol{\Theta}^{(p)}))$ denotes the Kullback-Leiber (KL) divergence between distributions $q(\boldsymbol{\Theta}^{(p)})$ and $p_r(\boldsymbol{\Theta}^{(p)})$.

For simplicity, both $q(\boldsymbol{\Theta}^{(p)})$ and $p_r(\boldsymbol{\Theta}^{(p)})$ are assumed to be normal distributions, namely $q(\Theta_{i,j}^{(p)}) = \mathcal{N}(\mu_{i,j}, \sigma_{i,j})$ and $p_r(\Theta_{i,j}^{(p)}) = \mathcal{N}(\mu'_{i,j}, \sigma'_{i,j})$. Then, the KL divergence can be computed as:

$$\mathrm{KL}(q(\boldsymbol{\Theta}^{(p)})||p_r(\boldsymbol{\Theta}^{(p)})) =$$
$$\frac{1}{2}\sum_{i,j}\left\{\frac{(\mu_{i,j}-\mu'_{i,j})^2+\sigma_{i,j}^2}{(\sigma'_{i,j})^2} - \log\frac{\sigma_{\mathrm{i,j}}^2}{(\sigma'_{\mathrm{i,j}})^2} - 1\right\}. \tag{12}$$

The Bayesian gate posterior distribution can be parameterized by;

$$q(\boldsymbol{\Theta}^{(p)}) = \{\boldsymbol{\mu}, \boldsymbol{\gamma}\}, \tag{13}$$

where $\boldsymbol{\mu}$ is the mean of $q(\boldsymbol{\Theta}^{(p)})$ and $\boldsymbol{\gamma}$ denotes the log-scale standard deviation parameters which satisfies $\boldsymbol{\sigma} = \exp(\boldsymbol{\gamma})$.

To enable updating of $q(\boldsymbol{\Theta}^{(p)})$, $\mathcal{L}_2$ can be approximated by Monte-Carlo sampling,

$$\int q(\boldsymbol{\Theta}^{(p)}) \log P(\boldsymbol{W}|\boldsymbol{\Theta}^{(p)}) d\boldsymbol{\Theta}^{(p)} \approx \frac{1}{K}\sum_k \log P(\boldsymbol{W}|\boldsymbol{\Theta}^{(p,k)}), \tag{14}$$

where K is the number of samples. It is worth mentioning that directly sampling $\Theta^{(p,k)}$ using mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\gamma}$ is unstable with high variance. To address this issue, the reparameterization trick [25] is used to sample $\boldsymbol{\Theta}^{(p,k)} = \boldsymbol{\mu} + \boldsymbol{\gamma} \odot \boldsymbol{\epsilon}^k$ with each $\epsilon_{i,j}^k$ drawn from $\mathcal{N}(0,1)$. In this case, the standard back-propagation algorithm is applicable to update the model parameters. Moreover, when minibatch training is employed, it suffices in practice to take only one sample ($K = 1$) per batch so that no additional time cost is introduced to the typical training process of RNNLMs. In this paper zero mean and unit variance is used for each entry of the model parameters weights.

4. Variational RNNLM

Inspired by the previous work on variational auto encoder (VAE) [17] in text modeling [18, 19] and the variational recurrent neural network (VRNN) [20, 26] in speech processing , this paper proposed a VRNN language model (VRNNLM) to model the uncertainty in the recurrent hidden representation.

4.1. Model Construction

In VRNNLM, the temporal and stochastic information in the recurrent hidden representation is captured by latent variable $\boldsymbol{z_t}$ from the input $\boldsymbol{w_{t-1}}$ at each time t and hidden feature $\boldsymbol{h_{t-1}}$ at previous time $t-1$. Given the distribution of latent variables $p(\boldsymbol{z_t}|\boldsymbol{h_{t-1}}, \boldsymbol{w_{t-1}})$, the language model outputs are estimated by:

$$P(\boldsymbol{W}) = \prod_{t=1}^{T} \int P(\boldsymbol{w_t}|\boldsymbol{w_{t-1}}, \boldsymbol{z_t}, \boldsymbol{h_{t-1}}) p(\boldsymbol{z_t}|\boldsymbol{h_{t-1}}, \boldsymbol{w_{t-1}}) d\boldsymbol{z_t}. \tag{15}$$

4.2. Model Inference

Similar to the inference procedure of Bayesian RNNLM, in VRNNLM the variational lower bound is again used as an approximation of Eqn.(15):

$$\mathcal{L} = \sum_{t=1}^{T}\Big\{\int \log P(\boldsymbol{w_t}|\boldsymbol{w_{t-1}}, \boldsymbol{z_t}, \boldsymbol{h_{t-1}}) q(\boldsymbol{z_t}) d\boldsymbol{z_t}$$
$$- KL(q(\boldsymbol{z_t})||p_r(\boldsymbol{z_t}|\boldsymbol{h_{t-1}}))\Big\}, \tag{16}$$

where $q(\boldsymbol{z_t})$ is the approximation of the posterior distribution $p(\boldsymbol{z_t}|\boldsymbol{h_{t-1}}, \boldsymbol{w_{t-1}})$, and $p_r(\boldsymbol{z_t}|\boldsymbol{h_{t-1}})$ is the prior distribution of latent variable $\boldsymbol{z_t}$. As the VRNNLM architecture shown in Fig.2, $p_r(\boldsymbol{z_t}|\boldsymbol{h_{t-1}})$ is assumed to be a Gaussian $\mathcal{N}(\boldsymbol{\mu_{0,t}}, \mathrm{diag}(\boldsymbol{\sigma_{0,t}}))$ where the mean and variance are calculated by a prior network $[\boldsymbol{\mu_{0,t}}, \boldsymbol{\sigma_{0,t}}] = \Phi^{\mathrm{prior}}(\boldsymbol{h_{t-1}})$. The posterior approximation $q(\boldsymbol{z_t})$ is calculated at each time step by using another Gaussian $\mathcal{N}(\boldsymbol{\mu_{z,t}}, \mathrm{diag}(\boldsymbol{\sigma_{z,t}}))$ with the mean and variance calculated by an inference network: $[\boldsymbol{\mu_{z,t}}, \boldsymbol{\sigma_{z,t}}] = \Phi^{\mathrm{inf}}(\boldsymbol{h_{t-1}}, \boldsymbol{x_{t-1}})$, where $\boldsymbol{x_{t-1}}$ is the word embedding of word $\mathbf{w_{t-1}}$. In particular, there are three types of parameters to be updated in VRNNLMs: (1) all parameters originally defined in the RNNLMs, (2) the parameters in the prior network and (3) the parameters in the inference network. Following the inference approach in Bayesian RNNLMs, we applied Monte-Carlo sampling in the first term of the Eqn.(14):

$$\mathcal{L} \approx \sum_{t=1}^{T}\Big\{\frac{1}{K}\sum_{k=1}^{K}\log P(\boldsymbol{w_t}|\boldsymbol{w_{t-1}}, \boldsymbol{z_t^k}, \boldsymbol{h_{t-1}})$$
$$- KL(q(\boldsymbol{z_t})||p_r(\boldsymbol{z_t}|\boldsymbol{h_{t-1}}))\Big\}, \tag{17}$$

where K is the number of samples. Similarly, we employ reparameterization trick to sample $\boldsymbol{z_t^k} = \boldsymbol{\mu_{z,t}} + \boldsymbol{\epsilon}^k \odot \boldsymbol{\sigma_{z,t}}$ with $\boldsymbol{\epsilon}^k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Same as the widely used variational auto-encoder (VAE) and other types of VRNN [17, 26], the stochastic training procedure of standard recurrent neural networks can be directly applied to this VRNNLM framework.

5. Experiment

In this section, we evaluate the performance of Bayesian RNNLMs and VRNNLMs with GRU and LSTM units using the perplexity (PPL) and the word error rate (WER) obtained in automatic speech recognition (ASR) tasks. All the models were

Table 1: Perplexities (PPL) obtained on PTB test set by applying Bayesian RNN and VRNNLM to LSTM LMs. "B-" and "V-" denotes Bayesian and variational RNNLMs respectively. As shown in Fig.1, the number in the B-LSTM denotes the position to apply Bayesian gate.

Language Model	PPL	PPL(+4g)	PPL(+4g+LSTM)
4gram	141.7	-	-
LSTM	114.4	99.7	-
B-LSTM 1	109.8	97.4	90.3
B-LSTM 2	112.1	97.8	90.4
B-LSTM 3	**109.5**	**96.9**	**90.2**
B-LSTM 4	111.7	97.5	90.7
V-LSTM	111.2	97.6	**90.2**

Table 2: Perplexities (PPL) obtained on PTB test set by applying Bayesian RNN and VRNNLM to GRU.

Language Model	PPL	PPL(+4g)	PPL(+4g+GRU)
4gram	141.7	-	-
GRU	116.2	100.2	-
B-GRU 1	**115.4**	**99.7**	92.8
B-GRU 2	115.5	99.8	**92.5**
B-GRU 3	116.7	100.3	92.7
V-GRU	115.9	100.0	92.8

implemented using PyTorch [27]. For all RNNLMs, the recurrent layer is set to be a 200 nodes single layer LSTM or GRU. The dimension of word embedding is set to 200. For Bayesian RNNLMs, we model the mean $\boldsymbol{\mu}$ and log-scale standard deviation $\boldsymbol{\gamma}$ of the Bayesian gate parameters $\boldsymbol{\Theta}^{(p)}$ by two 400×200 weight matrices respectively. All the gates in GRU and LSTM are investigated. For VRNNLM, the prior network contains one 200×200 linear layer followed by a ReLU [28, 29] activation function to compute the mean $\boldsymbol{\mu}_{0,t}$ and one 200×200 linear layer followed by a Softplus [29] activation function to compute the variance $\boldsymbol{\sigma}_{0,t}$. In addition, the inference network contains one 400×200 linear layer followed by a ReLU activation function to compute the mean $\boldsymbol{\mu}_{z,t}$ and one 400×200 linear layer followed by a Softplus activation function to compute the variance $\boldsymbol{\sigma}_{z,t}$. The number of samples K is set to 1 in both Bayesian RNNLMs and VRNNLMs in training. When evaluating, only the mean of hidden representation is used.

In the training procedure, model parameters were updated in mini-batch optimization (10 sentences per batch) using the typical stochastic gradient descent (SGD) with an initial learning rate 4. In our experiments, all RNNLMs were interpolated with n-gram LMs [30, 31] to complement with each other as in many state-of-the art systems. The weight of n-gram is determined using the EM algorithm on a validation set.

5.1. Experiments on Penn Treebank Corpus

We first analyze the performance of placing Bayesian gates at different positions in LSTM and GRU cells and the proposed VRNNLMs on the Penn Treebank (PTB) corpus, which consists of 10K vocabulary, 930K words for training, 74K words for development, and 82K words for testing. The PPL results are shown in Table 1 and Table 2. We observed that the VRNNLMs and most of the Bayesian RNNLMs outperform the standard GRU and LSTM LMs, the position of the Bayesian gate causes no significant difference on the performance of the Bayesian RNNLMs. In addition, the proposed system can provide PPL reduction when interpolated with standard RNNLMs and 4-grams.

5.2. Experiments on Conversational Telephone Speech

To evaluate the performance of proposed Bayesian RNNLMs and VRNNLMs in speech recognition, we used RNNLMs to rescore the N-best list generated by the acoustic model trained on Switchboard (swbd) English corpus. The SWBD system has 300 hour of conversational telephone speech from Switchboard I for acoustic modeling and 3.6M words of acoustic transcription with 30k words lexicon for language modeling. The acoustic model is a minimum phone error (MPE) trained stacked hybrid DNN-HMM acoustic model [32]. The PPL and the WER results on Switchboard (swbd) and CallHome (callhm) test data can be found in Table 3 and Table 4 respectively. Consistent with the result on PTB data set, the Bayesian RNNLMs and VRNNLMs both yield better results on PPL and WER on swbd and callhm test data than standard LSTM and GRU LMs. The comparable results between Bayesian and variational approaches indicates that the uncertainty of hidden representation can be both effectively modelled on the hidden vector or on the model parameters.

Table 3: PPL and WER results on SWBD test set on various LMs with LSTM recurrent unit. "(*)" denotes the results of interpolation of Bayesian/VRNN, 4-gram and LSTM LMs.

LMs	PPL	WER swbd/callhm	PPL (+4g)	WER(+4g) swbd/callhm	PPL (*)	WER(*) swbd/callhm
4gram	80.6	12.1/23.9	-	-/-	-	-/-
LSTM	90.9	11.4/23.9	71.7	11.3/23.2	-	-/-
B-LSTM 1	86.5	11.2/**23.4**	68.2	11.1/23.1	66.6	11.0/23.0
B-LSTM 2	**86.3**	**11.1**/23.5	**67.8**	11.1/**23.0**	**66.4**	10.9/22.9
B-LSTM 3	**86.3**	11.2/23.6	68.0	11.1/**23.0**	66.5	**10.9**/23.0
B-LSTM 4	86.7	**11.1/23.6**	68.1	**11.0**/23.1	66.5	**10.9/22.8**
V-LSTM	87.6	**11.1/23.5**	68.6	**11.0**/23.1	66.7	**10.9**/23.0

Table 4: PPL and WER results on SWBD test set on various LMs with GRU recurrent unit. "(*)" denotes the results of interpolation of Bayesian/VRNN, 4-gram and GRU LMs.

LMs	PPL	WER swbd/callhm	PPL (+4g)	WER(+4g) swbd/callhm	PPL (*)	WER(*) swbd/callhm
4gram	80.6	12.1/23.9	-	-/-	-	-/-
GRU	97.9	11.8/24.2	71.6	11.5/23.7	-	-/-
B-GRU 1	97.2	11.7/24.0	71.2	11.3/23.4	70.0	11.2/**23.2**
B-GRU 2	**96.5**	**11.6/23.7**	**71.0**	**11.3/23.3**	**69.8**	11.2/**23.2**
B-GRU 3	99.5	11.9/24.4	72.1	11.4/23.6	70.5	11.2/23.4
V-GRU	98.1	11.7/23.9	71.5	11.3/23.5	70.2	**11.1**/23.3

6. Conclusions

In most previous work, a deterministic hidden representation was used to model history context. However, it has limited power of modeling the underlying multilevel information in word context. In this paper, two approaches, Bayesian RNNLMs and variational RNNLMs, were proposed to model the uncertainty of hidden representations. The experiments reveals that significant and consistent performance improvements in both perplexity and WER can be obtained by taking this uncertainty into consideration.

7. Acknowledgements

The authors would like to thank Dr.Xie Chen for insightful discussion leading to this research. This research is supported by Hong Kong Research Grants Council General Research Fund No.14200218 and Shun Hing Institute of Advanced Engineering Project No.MMT-p1-19.

8. References

[1] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *1995 International Conference on Acoustics, Speech, and Signal Processing*, vol. 1. IEEE, 1995, pp. 181–184.

[2] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech & Language*, vol. 13, no. 4, pp. 359–394, 1999.

[3] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[4] H. Schwenk, "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[5] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *Eleventh annual conference of the international speech communication association*, 2010.

[6] T. Mikolov, S. Kombrink, L. Burget, J. Černockỳ, and S. Khudanpur, "Extensions of recurrent neural network language model," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2011, pp. 5528–5531.

[7] M. Sundermeyer, I. Oparin, J.-L. Gauvain, B. Freiberg, R. Schlüter, and H. Ney, "Comparison of feedforward and recurrent neural network language models," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 8430–8434.

[8] X. Chen, X. Liu, A. Ragni, Y. Wang, and M. J. Gales, "Future word contexts in neural network language models," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 97–103.

[9] X. Liu, X. Chen, Y. Wang, M. J. Gales, and P. C. Woodland, "Two efficient lattice rescoring methods using recurrent neural network language models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1438–1449, 2016.

[10] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 746–751.

[11] A. Deoras, T. Mikolov, S. Kombrink, and K. Church, "Approximate inference: A sampling based modeling technique to capture complex dependencies in a language model," *Speech Communication*, vol. 55, no. 1, pp. 162–177, 2013.

[12] X. Liu, X. Chen, M. J. Gales, and P. C. Woodland, "Paraphrastic recurrent neural network language models," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5406–5410.

[13] J.-T. Chien and Y.-C. Ku, "Bayesian recurrent neural network for language modeling," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 2, pp. 361–374, 2016.

[14] M. W. Y. Lam, X. Chen, S. Hu, J. Yu, X. Liu, and H. Meng, "Gaussian process lstm recurrent neural network language models for speech recognition," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[18] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," *arXiv preprint arXiv:1511.06349*, 2015.

[19] Z. Yang, Z. Hu, R. Salakhutdinov, and T. Berg-Kirkpatrick, "Improved variational autoencoders for text modeling using dilated convolutions," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3881–3890.

[20] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in neural information processing systems*, 2015, pp. 2980–2988.

[21] S. Hu, M. W. Lam, X. Xie, S. Liu, J. Yu, X. Wu, X. Liu, and H. Meng, "Bayesian and gaussian process neural networks for large vocabulary continuous speech recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6555–6559.

[22] M. W. Lam, S. Hu, X. Xie, S. Liu, J. Yu, R. Su, X. Liu, and H. Meng, "Gaussian process neural networks for speech recognition," *Proc. Interspeech 2018*, pp. 1778–1782, 2018.

[23] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

[24] D. Barber and C. M. Bishop, "Ensemble learning in bayesian neural networks," *Nato ASI Series F Computer and Systems Sciences*, vol. 168, pp. 215–238, 1998.

[25] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Advances in Neural Information Processing Systems*, 2015, pp. 2575–2583.

[26] J.-T. Chien and K.-T. Kuo, "Variational recurrent neural networks for speech separation." in *INTERSPEECH*, 2017, pp. 1193–1197.

[27] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[28] K. Jarrett, K. Kavukcuoglu, Y. LeCun *et al.*, "What is the best multi-stage architecture for object recognition?" in *2009 IEEE 12th International Conference on Computer Vision (ICCV)*. IEEE, 2009, pp. 2146–2153.

[29] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[30] J. Park, X. Liu, M. J. Gales, and P. C. Woodland, "Improved neural network based language modelling and adaptation," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[31] H.-S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Structured output layer neural network language models for speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 1, pp. 197–206, 2013.

[32] X. Liu, S. Liu, J. Sha, J. Yu, Z. Xu, X. Chen, and H. Meng, "Limited-memory bfgs optimization of recurrent neural network language models for speech recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6114–6118.