Generating Diverse Translation from Model Distribution with Dropout

Xuanfu Wu[1,2], **Yang Feng**[1,2][*], **Chenze Shao**[1,2]

[1] Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences (ICT/CAS)
[2] University of Chinese Academy of Sciences, Beijing, China
`{wuxuanfu20s, fengyang, shaochenze18z}@ict.ac.cn`

Abstract

Despite the improvement of translation quality, neural machine translation (NMT) often suffers from the lack of diversity in its generation. In this paper, we propose to generate diverse translations by deriving a large number of possible models with Bayesian modelling and sampling models from them for inference. The possible models are obtained by applying concrete dropout to the NMT model and each of them has specific confidence for its prediction, which corresponds to a posterior model distribution under specific training data in the principle of Bayesian modeling. With variational inference, the posterior model distribution can be approximated with a variational distribution, from which the final models for inference are sampled. We conducted experiments on Chinese-English and English-German translation tasks and the results shows that our method makes a better trade-off between diversity and accuracy.

1 Introduction

In the past several years, neural machine translation (NMT) (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Gehring et al., 2017; Vaswani et al., 2017; Zhang et al., 2019) based on the end-to-end model has achieved impressive progress in improving the accuracy of translation. Despite its remarkable success, NMT still faces problems in diversity. In natural language, due to lexical, syntactic and synonymous factors, there are usually multiple proper translations for a sentence. However, existing NMT models mostly implement one-to-one mapping between natural languages, that is, one source language sentence corresponds to one target language sentence. Although beam search, a widely used decoding algorithm, can generate a group of translations, its search space is too narrow to extract diverse translations.

There are some researches working at enhancing translation diversity in recent years. Li et al. (2016) and Vijayakumar et al. (2016) proposed to add regularization terms to the beam search algorithm so that it can possess greater diversity. He et al. (2018) and Shen et al. (2019) introduced latent variables into the NMT model, thus the model can generate diverse outputs using different latent variables. Moreover, Sun et al. (2019) proposed to combine the structural characteristics of Transformer and use the different weights between each head in the multi-head attention mechanism to obtain diverse results. In spite of improvement in balancing accuracy and diversity, these methods do not represent the diversity in the NMT model directly.

In this paper, we take a different approach to generate diverse translation by explicitly maintaining different models based on the principle of Bayesian Neural Networks (BNNs). These models are derived by applying concrete dropout (Gal et al., 2017) to the original NMT model and each of them is given a probability to show its confidence in generation. According to Bayesian theorem, the probabilities over all the possible models under a specific training dataset forms a posterior model distribution which should be involved at inference. To make the posterior model distribution obtainable at inference, we further employ variational inference (Hinton and Van Camp, 1993; Neal, 1995; Graves, 2011; Blundell et al., 2015) to infer a variational distribution to approximate it, then at inference we can sample a specific model based on the variational distribution for generation.

We conducted experiments on the NIST Chinese-English and the WMT'14 English-German translation tasks and compared our method with different strong baseline approaches. The experiment results show that our method can get a good trade-off in translation diversity and accuracy with little train-

---
[*]Corresponding author: Yang Feng

ing cost.

Our contributions in this paper are as follows:

- We introduce Bayesian neural networks with variational inference to NMT tasks to explicitly maintain different models for diverse generation.

- We apply concrete dropout to the NMT model to derive the possible models which only demands a small cost in computation.

2 Background

Assume a source sentence with n words $\boldsymbol{x} = x_1, x_2, ..., x_n$, and its corresponding target sentence with m words $\boldsymbol{y} = y_1, y_2, ..., y_m$, NMT models the probability of generating $\boldsymbol{y}$ with $\boldsymbol{x}$ as the input. Based on the encoder-decoder framework, NMT model Θ encodes source sentence into hidden states by its encoder, and uses its decoder to find the probability of t-th word $\boldsymbol{y}_t$, which depends on the hidden states and the first $t-1$ words of target sentence $\boldsymbol{y}$. The translation probability from sentence $\boldsymbol{x}$ and $\boldsymbol{y}$ can be expressed as:

$$P(\boldsymbol{y}|\boldsymbol{x}) = \prod_{t=1}^{m} P(y_t|y_{<t}, \boldsymbol{x}; \Theta) \quad (1)$$

Given a training dataset with source-target sentence pairs $\mathcal{D} = \{(\boldsymbol{x_1}, \boldsymbol{y_1^*}), ..., (\boldsymbol{x_D}, \boldsymbol{y_D^*})\}$, the loss function we want to minimize in the training is the sum of negative log-likelihood of Equation 1:

$$L = - \sum_{(\boldsymbol{x_i}, \boldsymbol{y_i^*}) \in \mathcal{D}} \log P(\boldsymbol{y} = \boldsymbol{y_i^*}|\boldsymbol{x} = \boldsymbol{x_i}; \Theta) \quad (2)$$

In practice, by properly designing neural network structures and training strategies, we can get the specific model parameters that minimize Equation 2 and obtain translation results through the model with beam search.

One of the most popular model in NMT is Transformer, which was proposed by Vaswani et al. (2017). Without recurrent and convolutional networks, Transformer constructs its encoder and decoder by stacking self-attention and fully-connected network layers. Self attention is operated with three inputs: query(Q), key(K) and value(V) as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}}V) \quad (3)$$

where the dimension of key is d_k.

Note that Transformer implements the multi-head attention mechanism, projecting inputs into h group inputs to generate h different outputs in Equation 3, and these outputs are concatenated and projected into final outputs:

$$head_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \quad (4)$$
$$\text{Output} = \text{Concat}(head_1, ..., head_h)W^O \quad (5)$$

where W_i^Q, W_i^K, W_i^V and W^O are the projection matrices. The output of Equation 5 is then fed into the fully-connected layer named feed-forward network. The feed-forward network uses two linear networks and a ReLU activation function:

$$\text{FFN}(x) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \quad (6)$$

We only give a brief description of Transformer above. Please refer to Vaswani et al. (2017) for more details.

3 Uncertainty Modeling

3.1 Bayesian Neural Networks with Variational Inference

For most of machine learning tasks based on neural networks, a model with specific parameters is trained to explain the observed training data. However, there are usually a large number of possible models that can fit the training data well, which leads to model uncertainty. Model uncertainty may result from noisy data, uncertainty in model parameters or structure uncertainty, and it is represented as the confidence which model to choose to predict with. In order to express model uncertainty, we consider all possible models with parameters $\boldsymbol{\omega}$ and define a prior distribution $P(\boldsymbol{\omega})$ over the model (i.e., the space of the parameters) to denote model uncertainty. Then given a training data set $(\boldsymbol{X}, \boldsymbol{Y})$, the predicted distribution for $\boldsymbol{Y}$ can be denoted as $P(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\omega})$.

Following Gal et al. (2017), we employ Bayesian neural networks (BNNs) to represent the $P(\boldsymbol{\omega}|\boldsymbol{X}, \boldsymbol{Y})$, which is the posterior distribution of the models under $(\boldsymbol{X}, \boldsymbol{Y})$. BNNs offer a probabilistic interpretation of deep learning models by inferring distributions over the models' parameters which are trained using Bayesian inference. The posterior distribution can be got by invoking Bayes'

theorem as:

$$P(\boldsymbol{\omega}|\boldsymbol{X},\boldsymbol{Y}) = \frac{P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\omega})P(\boldsymbol{\omega})}{P(\boldsymbol{Y}|\boldsymbol{X})}$$
$$= \frac{P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\omega})P(\boldsymbol{\omega})}{\mathbb{E}_{\boldsymbol{\omega}}[P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\omega})]}. \tag{7}$$

Then according to BNNs, given a new test data $\boldsymbol{x}'$, the predictive distribution of the output $\boldsymbol{y}'$ is:

$$P(\boldsymbol{y}'|\boldsymbol{x}',\boldsymbol{X},\boldsymbol{Y}) = \mathbb{E}_{\boldsymbol{\omega}\sim P(\boldsymbol{\omega}|\boldsymbol{X},\boldsymbol{Y})}[P(\boldsymbol{y}'|\boldsymbol{x}',\boldsymbol{\omega})] \tag{8}$$

The expectations in Equation 7 and 8 are integrated over model distribution $\boldsymbol{\omega}$, and the huge space of $\boldsymbol{\omega}$ makes it intractable to obtain the results. Therefore, inspired by Hinton and Van Camp (1993), Graves (2011) proposes a variational approximation method, using a variational distribution $Q(\boldsymbol{\omega}|\theta)$ with the parameters θ to approximate the posterior distribution $P(\boldsymbol{\omega}|\boldsymbol{X},\boldsymbol{Y})$. To this end, the training objective is to minimize the Kullback-Leibler (KL) divergence between the model distribution and the posterior distribution $\mathrm{KL}(Q(\boldsymbol{\omega}|\theta)||P(\boldsymbol{\omega}|\boldsymbol{X},\boldsymbol{Y}))$. With variational inference, the objective is equivalent to maximizing its evidence lower bound (ELBO), so we get

$$\theta^* = \arg\max_\theta \mathbb{E}_{Q(\boldsymbol{\omega}|\theta)} \log P(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{\omega})$$
$$- \mathrm{KL}((Q(\boldsymbol{\omega}|\theta)||P(\boldsymbol{\omega})) \tag{9}$$

As we can see in Equation 9, the first term on the right side is the expectation of the predicted probability over model distribution on the training set, which can be unbiased estimated with the Monte-Carlo method. And the second term is the KL divergence between the approximate model distribution and the prior distribution. From the perspective of Hinton and Van Camp (1993) and Graves (2011), with the above objective, we can express model uncertainty under the training data and meanwhile regularize model parameters and avoid over-fitting. Therefore, at inference, we can use the distribution $Q(\boldsymbol{\omega}|\theta)$ instead of $P(\boldsymbol{\omega}|\boldsymbol{X},\boldsymbol{Y})$ to evaluate model confidence (i.e., model uncertainty).

3.2 Model distribution with Dropout

To derive the BNN, we need to first decide how to explore for the possible models and then decide the prior distribution and the variational distribution for the models. As in Gal et al., 2017, we can define a simple model with parameters $\boldsymbol{\omega}_{\boldsymbol{W}}$ ($\boldsymbol{W} \in \mathbb{R}_{m\times n}$) and then drop out some column of $\boldsymbol{\omega}_{\boldsymbol{w}}$ to get the possible models. We use matrix Gaussian distribution as the prior model distribution and Bernoulli distribution as the posterior model distribution.

Using $\boldsymbol{W}.j$ to denote the j-th column of $\boldsymbol{W}$, we draw a matrix Gaussian distribution as the probability distribution of dropping out the j-th column as

$$P(\boldsymbol{\omega}_{\boldsymbol{W}.j}) \sim \mathcal{MN}(\boldsymbol{\omega}_{\boldsymbol{W}.j};0,I/l,I/l) \tag{10}$$

where l is the hyper-parameter.

The above matrix Gaussian distribution is used as the prior distribution of the models got by dropping out the j-th column. Then we introduce $\boldsymbol{p}$ ($\boldsymbol{p} \in \mathbb{R}_{1\times n}$) as the probability vector of dropping out the columns of $\boldsymbol{\omega}_{\boldsymbol{W}}$, which means dropping out the j-th column with the probability of $\boldsymbol{p}_j$, and keeping the j-th column unchanged with the probability of $1 - \boldsymbol{p}_j$. Therefore the posterior model distribution of dropping out the j-th column is defined as

$$Q(\boldsymbol{\omega}_{\boldsymbol{W}.j}|\theta) = \begin{cases} 1 - \boldsymbol{p}_j & \boldsymbol{\omega}_{\boldsymbol{W}.j} = \boldsymbol{W}.j \\ \boldsymbol{p}_j & \boldsymbol{\omega}_{\boldsymbol{W}.j} = \boldsymbol{0} \end{cases} \tag{11}$$

where $\boldsymbol{W} \in \theta$ and $\boldsymbol{p} \in \theta$ are trainable parameters.

With Equation 10 as prior, the KL divergence for the j-th column of the matrix can be represented as:

$$\mathrm{KL}(Q(\omega_{\boldsymbol{W}.j}|\theta)||P(\omega_{\boldsymbol{W}.j}))$$
$$= \mathcal{R}(\boldsymbol{p}_{\boldsymbol{W}.j}, \boldsymbol{W}.j, l) - \mathcal{H}(\boldsymbol{p}_{\boldsymbol{W}.j}) \tag{12}$$

where

$$\mathcal{R}(\boldsymbol{p}_{\boldsymbol{W}.j}, \boldsymbol{W}.j, l) = \frac{(1 - \boldsymbol{p}_j)l^2}{2} \sum_{i=1}^{m} \boldsymbol{W}_{ij}^2 \tag{13}$$

and

$$\mathcal{H}(\boldsymbol{p}_{\boldsymbol{W}.j}) = -[\boldsymbol{p}_j \log(\boldsymbol{p}_j) + (1 - \boldsymbol{p}_j) \log(1 - \boldsymbol{p}_j)] \tag{14}$$

Since the probability distribution among different neural networks and different columns of neural network are independent. For a complex multi-layer neural network θ, the KL divergence between model distribution $Q(\omega|\theta)$ and prior distribution $P(\omega)$ is

$$\mathrm{KL}(Q(\omega|\theta)||P(\omega))$$
$$= \sum_{\boldsymbol{W}_{m\times n}, \boldsymbol{p}\in\theta} \sum_{j=1}^{n} \mathcal{R}(\boldsymbol{p}_{\boldsymbol{W}.j}, \boldsymbol{W}.j, l) - \mathcal{H}(\boldsymbol{p}_{\boldsymbol{W}.j}) \tag{15}$$

4 Application to Transformer

Previous sections show how to use concrete dropout to realize variational approximation of the posterior model distribution. In this section we will introduce the implementation in representing model distribution for Transformer with aforementioned methods.

4.1 Dropout in Transformer

Stated in detail in Vaswani et al. (2017), in Transformer, dropout is commonly used to the output of modules, including the output of embedding, attention layer and feed-forward layer. Also, from Equation 13, we find it's important to find the network W corresponding to the dropout module. The correspondences in Transformer are as follows:

Embedding module Embedding module works for mapping the words into embedding vectors. The embedding module contains a matrix $W_E \in \mathbb{R}_{l_d \times d}$, where l_d is the length of dictionary and d is the dimension of embedding vector. For the i-th word in the dictionary, its embedding vector is the i-th column of W_E. Since dropout the j-th dimension of word embedding is equivalent to dropping out the j-th row of W_E, we utilize W_E^T and its corresponding dropout in Equation 13.

Attention module For attention modules, as we can see in 5, their outputs are generated by concatenating the output of different heads and projecting by matrix W^O. Since dropout is used in the output of attention module, we take W^O and its corresponding dropout in calculating Equation 13.

Feed-forward module As shown in Equation 6, the output is generated through W_2 with bias b_2. As we can see, for network $y = xW + b$, we can find that

$$y = \text{Concat}(x, 1) \cdot \text{Concat}(W^T, b^T)^T \quad (16)$$

as we can see, dropout to the output of the feed-forward module can be regraded as dropping out W_2 and b_2. So, during training, we use $\text{Concat}(W^T, b^T)^T$ to calculate Equation 13.

4.2 Training and Inference

Although dropout is frequently utilized in Transformer, there are some networks in Transformer like W_i^Q, W_i^K, W_i^V in Equation 4, and their output is not masked by dropout. So, in our implementation, we obtain the model distribution by fine-tuning the pre-trained model, freezing their

parameters and only updating dropout probabilities. Moreover, we choose different trained modules to train their output dropout probability, and in calculating Equation 15, we only take those trained dropout probabilities into consideration. By allowing dropout probabilities to change, our method can better represent model distributions under the training dataset than the fixed dropout probabilities. The mini-batch training algorithm is expressed in Equation 1. It is worth to mention that since we train the model distribution with batches of data, we scale the KL divergence with the proposition of the batch in the entire training dataset.

Algorithm 1 Mini-batch Training of Bayesian NN using Variational Inference with Dropout in NMT

Input: Training dataset $\mathcal{D} = (X, Y)$ with size N, pre-trained model parameter θ, learning rate η, learning epoch E
Output: model parameter θ
1: Initial θ
2: Split $\mathcal{D}$ into $(X_1, Y_1), ..., (X_n, Y_n)$ with size $M_1, ..., M_n$
3: $i = 0$
4: **while** $i < E$ **do**
5:    **for** $j = 1$ to n **do**
6:       sample ω' from $Q(\omega|\theta)$
7:       $L = \sum\limits_{(x,y) \in (X_j, Y_j)} \sum\limits_{k} \log P(y_k | y_{<k}, x; \omega')$
8:       $L = L + \frac{M_j}{N}[\sum\limits_{W_{m \times n} \in \theta} \sum\limits_{j=1}^{n} -\mathcal{H}(p_{W_{\cdot j}}) + \mathcal{R}(p_{W_{\cdot j}}, W_{\cdot j}, l)]$
9:       $\theta \leftarrow \theta + \eta \frac{\partial}{\partial \theta} L$
10:    **end for**
11: **end while**

During updating the dropout probability, due to the discrete characteristics of the Bernoulli distribution, we cannot directly calculate the gradient of the first term in Equation 9 to the dropout probability. So, we adopt concrete dropout, which is used in Gal et al. (2017). As a continuous relaxation of dropout, for its input $\boldsymbol{y}$, the output can be expressed as $\boldsymbol{y'} = \boldsymbol{y} \odot \boldsymbol{z}$, and vector $\boldsymbol{z}$ satisfies:

$$\boldsymbol{z} = \text{sigmoid}(\frac{1}{t}(\log(p) - \log(1-p) + \log(u) - \log(1-u)))$$

$$(17)$$

where $u \sim \mathcal{U}(0, 1)$, p is dropout probability.

In the inference stage, we just randomly mask model parameters with trained dropout probabil-

ities, with different random seeds, NMT models with different parameters are sampled. Since diverse translations are demanded, we performed several forward passes through different sampled NMT models, and different translations are generated with different model outputs and beam search.

5 Experiment Setup

Dataset In our experiment, we select datasets in the following translation tasks:

- NIST Chinese-to-English (NIST Zh-En). Its dataset is based on LDC news corpus and contains about 1.34 million sentence pairs. It also includes 6 relatively small datasets, MT02, MT03, MT04, MT05, MT06, and MT08. In our experiments, we use MT02 as the development set, and the rest work as the test sets. Without special explanation, we use average result of test sets as final results.
- WMT'14 English-to-German (WMT'14 En-De). Its dataset comes from the WMT'14 news translation task, which contains about 4.5 million sentence pairs. In our experiment, we use newstest2013 as the development set and newstest2014 as the test set.

For above two datasets, We adopt Moses tokenizer (Koehn et al., 2007) in English and German corpus. We also use the byte pair encoding (BPE) algorithm (Sennrich et al., 2015), and limit the size of the vocabulary $K = 32000$. And we train a joint dictionary for WMT'14 En-De. For NIST, we use THULAC toolkit (Sun et al., 2016) to segment Chinese sentence into words. In addition, we remove the examples in datasets from the above two tasks where length of the source language sentence or target language sentence exceed 100 words.

Model Architecture In our experiments, we all adopt the Transformer Base model in Vaswani et al. (2017). Transformer base model has 6 layers in encoder and decoder, and it has hidden units with 512 dimension, except for the feed-forward network, where the inner-layer output dimension is 2048. The number of heads in Transformer base model is 8 and the default dropout probability is 0.1. And our model is implemented in python3 with the Fairseq-py (Ott et al., 2019) toolkit.

Experimental Setting During training, in order to improve the accuracy, we use the label smoothing (Szegedy et al., 2016) with $\epsilon = 0.1$. In terms of optimizer, we adopt the Adam optimizer (Kingma and Ba, 2014), the main parameters of the opti-

mizer is $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-9}$. As for the learning rate, we adopt the dynamic learning rate method in Vaswani et al. (2017) with $warmup\_steps = 4000$. Also, we use mini-batch training with $max\_token = 4096$.

Metrics In terms of evaluation metrics, referring to Shen et al. (2019), we adopt the BLEU and Pairwise-BLEU to evaluate translation quality and diversity. Both two metrics are calculated with case-insensitive BLEU algorithm in Papineni et al. (2002). In our experiments, the BLEU is to measure the average similarity between the output translations and the standard translation. The higher the BLEU value, the better the accuracy of translation. And the Pairwise-BLEU reflects the average similarity between the output translations of different groups. The lower the Pairwise-BLEU value, the lower the similarities, and the more diverse the translations. In our experiment, we use the NLTK toolkit to calculate the two metrics.

6 Experiment Results

6.1 Analysis of Training Modules and Hyper-parameter

In this experiment, we train models with different training modules and hyper-parameter l with NIST dataset to evaluate their effects, and some results are shown in Table 1.

As we can see in Table 1, for those training modules, when l is small, with the same hyper-parameter l, choosing smaller part of training modules will lead to lower BLEU and Pairwise-BLEU, showing that accuracy of the generate translations increases while diversity decreases. We also find that in the same training modules, with the increase of l, the Pairwise-BLEU decreases steadliy, and then increases when the BLEU is close to zero; and the BLEU has similar trends with Pairwise-BLEU, however, when l is relatively low, the BLEU tends to stablize.

For the above-mentioned experimental results, we can interpret as follows: in training modules, since Equation 15 is the sum of the training modules' KL divergence, with the training modules increase, the KL divergence accordingly increases, pushing the dropout probability higher and making translations diverse. In terms of hyper-parameter l, as we can see in Equation 10, when l increases, the prior distribution is squeezed to zero matrix; thus during training, the dropout probabilities will get higher to make the model distribution close to

l^2		10^1	10^2	10^3	10^4	10^5	10^6	10^7	10^8
Decoder's 1-3 Layer	BLEU	42.48	42.47	42.56	41.33	26.83	8.68	5.14	2.90
	Pairwise-BLEU	80.97	79.66	72.12	62.16	43.11	33.24	25.43	35.18
Decoder	BLEU	42.34	42.30	42.25	38.17	7.79	12.22	0.59	0.26
	Pairwise-BLEU	75.89	74.52	67.26	49.67	16.86	20.47	56.99	58.92
Encoder+Decoder	BLEU	42.20	42.12	41.69	32.70	2.43	0.83	0.37	0.68
	Pairwise-BLEU	69.75	68.08	58.99	38.02	9.85	79.80	56.76	58.54

Table 1: Results with 3 different training modules and hyper-parameter l. From the table we can see the BLEU and Pairwise-BLEU change with training modules. Also, the BLEU decreases with the increase of l, and the Pairwise-BLEU decreases steadily and then increases when the BLEU value is close to its minimum.

prior distribution. However, when the l is too high to make most of dropout probabilities close to 1, uncertainty of model parameters decreases, making Pairwise-BLEU increases.

6.2 Results in Diverse Translation

From the previous section, we can see that by selecting different training modules and hyper-parameter, translations with different accuracy and diversity can be obtained. Then we conduct experiments to generate 5 groups of different translations on NIST Zh-En dataset and WMT'14 En-De dataset, and compare the diversity and accuracy of the translations generated by our method and the following baseline approaches:

- Beam Search: we choose the optimal 5 results directly generated by beam search in this paper.
- Diverse Beam Search (DBS) (Vijayakumar et al., 2016): it works by grouping the outputs and adding regularization terms in beam search to encourage diversity. In our experiment, the number of output translations of groups are all 5.
- HardMoE (Shen et al., 2019): it trains model with different hidden states and obtains different translations by controlling hidden state. In our experiment, we set the number of hidden states is 5.
- Head Sampling (Sun et al., 2019): it generate different translations by sampling different encoder-decoder attention heads according to their attention weight, and copying the samples to other heads in some conditions. Here, we set the parameter K = 3.

The results are shown in Figure 1. In Figure 1 we plot the BLEU versus Pairwise-BLEU, the scattered points show the results of baseline approaches, and the points on the curves are results in the same training modules with different hyper-
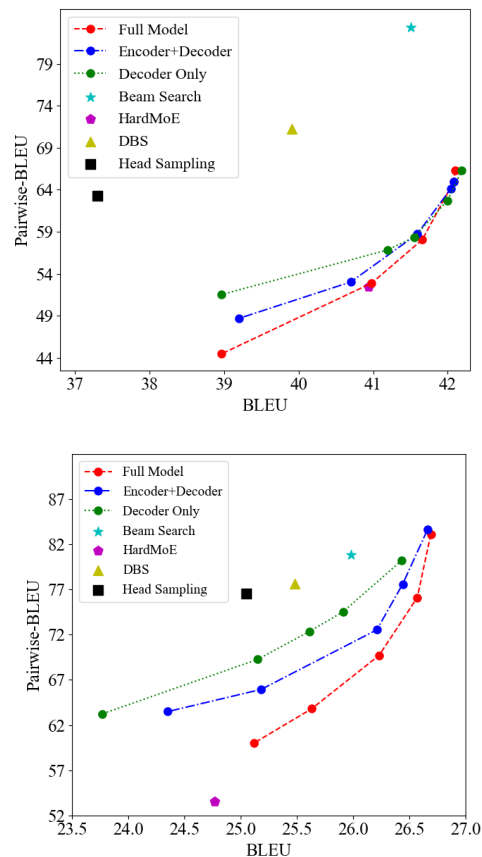

Figure 1: Experiment result in NIST Zh-En (upper one) and WMT'14 En-De (lower one). The X axis and Y axis represents BLEU and Pairwise-BLEU value. The first three groups in legend (connected with curves) are results with our methods under specific training modules with different prior parameters l, and the latter four groups (scattered points) are results from baseline methods.

Source	此次会议的一个重要议题是跨大西洋关系。
Reference	One of the important topics for discussion in this meeting is the cross atlantic relation.
	One of the top agendas of the meeting is to discuss the cross-atlantic relations.
	An important item on the agenda of the meeting is the trans-atlantic relations.
	One of the major topics for the conference this time is transatlantic relations.
Beam search	An important item on the agenda of this meeting is transatlantic relations.
	An important topic of this conference is transatlantic relations.
	An important topic of this meeting is transatlantic relations.
	An important topic of this conference is transatlantic ties.
	An important topic of the meeting is transatlantic relations.
Our Work	One of the important topics of this conference is transatlantic relations.
	An important item on the agenda of this meeting is the transatlantic relationship.
	An important item on the agenda of this conference is the transatlantic relationship.
	An important topic for discussion at this conference is cross-atlantic relations.
	One of the important topics of this conference is the transatlantic relationship.

Table 2: Translation examples in NIST Zh-En Task. Results of our work is generated by training dropout in decoder with $l^2 = 1000$. The result shows that by adjusting model parameters, our method can generate translations with higher diversity while maintaining accuracy.

parameter l. From Figure 1, firstly, we can verity that choosing different training modules can lead to different balance of translation diversity and accuracy, for NIST Zh-En and WMT'14 En-De, training dropout probabilities in the full model can get better translations.

Also, we suggest that in NIST Zh-En task, by adjusting training modules and hyper-parameter l, our results which has higher BLEU and lower Pairwise-BLEU values than baselines without HardMoE, even for HardMoE itself, our method is comparable with proper l while training the whole model. In WMT'14 En-De, we also find that our method exceeds the baseline approach except HardMoE. For the gap in performace with HardMoE, we interpreted that since our models are randomly sampled from the model distribution, it could be hard for our models to represent such distinguishable characteristics like HardMoE, which trains multiple different latent variables.

Also, to intuitively display the improvement of diversity in our translations, we choose a case from NIST Zh-En task, the results are shown in Table 2. The case shows that compared with beam search, which only varies in few words, our method can obtain more diverse translations while ensuring the accuracy of translation, and diversities are not only shown in words, but also reflected in lexical characteristic.

6.3 Analyzing Module Importance with Dropout Probability

Some researches (Voita et al., 2019; Michel et al., 2019; Fan et al., 2019) found that a well-trained Transformer model is over-parameterized. Useful information gathers in some parameters and some modules and layers can be pruned to improve the efficiency during test time. Since dropout can play the role of regularization and there are differences in the trained dropout probabilities of different neuron, we conjecture that the trained dropout probability and the importance of each module are correlated. To investigate this, we choose the model in which dropout probabilities of the full model is trained with $l^2 = 400$ in NIST Zh-En task, and separately calculate the average dropout probability $\bar{p}_{dropout}$ of different attention modules. Also, we manually pruned the corresponding modules of the model, obtained translations and calculated its BLEU. the more the BLEU drops, the more important the module is to translation. To quantify their relevance, we calculate the Pearson correlation coefficient (PCC) ρ in different kinds of training modules, and highlights the highest and lowest results.

Results of our experiment are shown in Table 3. Firstly, we can see that the average dropout probabilities and BLEU are not fully positively correlated, which might be explained by the contingency of sampling from model distribution during training. But from the maximum and minimum of

| | Encoder | | Decoder | | | |
| | Self-attention | | Self-attention | | E-D Attention | |
Layer	$\bar{p}_{dropout}$	BLEU	$\bar{p}_{dropout}$	BLEU	$\bar{p}_{dropout}$	BLEU
1	**0.0400**	**32.65**	0.0484	40.20	**0.0915**	**42.15**
2	0.0858	40.97	**0.0793**	**41.67**	0.0798	41.03
3	**0.0863**	**41.70**	0.0670	40.83	0.0620	35.65
4	0.0763	39.87	**0.0460**	39.56	0.0556	37.29
5	0.0632	40.17	0.0476	**37.93**	0.0394	32.18
6	0.0769	39.15	0.0490	40.88	**0.0335**	**18.48**
ρ	0.919		0.689		0.858	

Table 3: Average dropout probabilities of each module and BLEU of translations generated by model where the module is pruned. From the maximum and minimum of $\bar{p}_{dropout}$ and BLEU, and correlation coefficient ρ in different modules, it is obvious that dropout probabilities of module and its importance is correlated.

$\bar{p}_{dropout}$ and BLEU, we can find that the dropout probabilities $\bar{p}_{dropout}$ and the BLEU of translations show some similar information in module importance. Also, we quantify the correlation between the $\bar{p}_{dropout}$ and BLEU, finding that it is highly correlated in self-attention module in encoder and in E-D attention in decoder, since its correlation coefficient $\rho > 0.8$, and the $\bar{p}_{dropout}$ and BLEU is also correlated in self-attention in decoder.

7 Related Work

Researches in Bayesian Neural Network have a long history, Hinton and Van Camp (1993) firstly proposes a variational inference approximation methods to BNN to minimize the minimum description length (MDL), then Neal (1995) approximate BNN by Hamiltonian Monte Carlo methods. In recent years, Graves (2011) introduces the concept of variational inference, by approximating posterior distribution with model distribution, the model minimizes its MDL and reduces the model weight; and Blundell et al. (2015) proposes an algorithm similar to Graves (2011), however, it uses mixture of Gaussian densities as prior and achieved comparable performance with dropout in regularization.

Introduced by Hinton et al. (2012), dropout, which is easy to implement, works as a stochastic regularization to avoid over-fitting. And there are several theoretical explanations such as getting sufficient model combinations (Hinton et al., 2012; Srivastava et al., 2014) to train and augumenting training data (Bouthillier et al., 2015). Gal and Ghahramani (2016) proposes that dropout can be understood as a bayesian inferences algorithm, and Gal et al. (2017) uses concrete dropout in updating dropout probabilities. Also, the author implements the dropout methods to represent uncertainty in different kinds of deep learning tasks in Gal (2016).

In neural machine translation task, lack of diversity is a widely acknowledged problem, some researches like Ott et al. (2018) investigate the cause of uncertainty in NMT, and some provide metrics to evaluate the translation uncertainty like Galley et al. (2015); Dreyer and Marcu (2012). There are also other researches that put forward methods to obtain diverse translation. Li et al. (2016); Vijayakumar et al. (2016) adjust decoding algorithms, adding different kinds of diversity regularization terms to encourage generating diverse outputs. He et al. (2018); Shen et al. (2019) utilize mixture of experts (MoE) method, using differentiated latent variables to control generation of translation. Sun et al. (2019) generates diverse translation by sampling heads in encoder-decoder attention module in Transformer model, since different heads may present different target-source alignment. Shu et al. (2019) uses sentence codes to condition translation generation and obtain diverse translations. Shao et al. (2018) propose a new probabilistic ngram-based loss to conduct sequence-level training for generating diverse translation.Feng et al. (2020) propose to employ future information to evaluate fluency and faithfulness to encourage diverse translation.

There are also a few papers in interpreting Transformer model, Voita et al. (2019) suggests that some heads play a consistent role in machine translation, and their roles can be interpreted linguistically; also, they implement L_0 penalty to prune heads. Michel et al. (2019) shows that huge amounts of heads in Transformer can be pruned, and the importance of head is cross-domain. Also,

Fan et al. (2019) shows that the layers in Transformer are also able to be pruned: similar to our work, during training, they drop the whole layer with dropout and trained their probability; however, variational inference strategy is not used in their paper, and they take different kinds of inference strategies to balance performance and efficiency rather than sampling.

8 Conclusion

In this paper, we propose to utilize variational inference in diverse machine translation tasks. We represent the Transformer model distribution with dropout, and train the model distributions to minimize its distance to the posterior distribution under specific training dataset. Then we generate diverse translations with the models sampled from the trained model distribution. We further analyze the correlations between module importance and trained dropout probabilities. Experiment results in Chinese-English and English-German translation tasks suggest that by properly adjusting trained modules and prior parameters, we can generate translations which balance accuracy and diversity well.

In future work, firstly, since our model is randomly sampled from model distribution to generate diverse translation, it is meaningful to explore better algorithms and training strategies to represent model distribution and search for the most distinguishable results in model distribution. Also, we'll try to extend our methods in a wider range of NLP tasks.

Acknowledgments

References

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.

Xavier Bouthillier, Kishore Konda, Pascal Vincent, and Roland Memisevic. 2015. Dropout as data augmentation. *arXiv preprint arXiv:1506.08700*.

Markus Dreyer and Daniel Marcu. 2012. Hyter: Meaning-equivalent semantics for translation evaluation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 162–171. Association for Computational Linguistics.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Yang Feng, Wanying Xie, Shuhao Gu, Chenze Shao, Wen Zhang, Zhengxin Yang, and Dong Yu. 2020. Modeling fluency and faithfulness for diverse neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 59–66.

Yarin Gal. 2016. Uncertainty in deep learning. *University of Cambridge*, 1:3.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

Yarin Gal, Jiri Hron, and Alex Kendall. 2017. Concrete dropout. In *Advances in neural information processing systems*, pages 3581–3590.

Michel Galley, Chris Brockett, Alessandro Sordoni, Yangfeng Ji, Michael Auli, Chris Quirk, Margaret Mitchell, Jianfeng Gao, and Bill Dolan. 2015. deltableu: A discriminative metric for generation tasks with intrinsically diverse targets. *arXiv preprint arXiv:1506.06863*.

Jonas Gehring, Michael Auli, David Grangier, and Yann Dauphin. 2017. A convolutional encoder model for neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 123–135, Vancouver, Canada. Association for Computational Linguistics.

Alex Graves. 2011. Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356.

Xuanli He, Gholamreza Haffari, and Mohammad Norouzi. 2018. Sequence to sequence mixture model for diverse machine translation. *arXiv preprint arXiv:1810.07391*.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Geoffrey E Hinton and Drew Van Camp. 1993. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *arXiv preprint arXiv:1611.08562*.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, pages 14014–14024.

Radford M Neal. 1995. *Bayesian learning for neural networks*. Ph.D. thesis, University of Toronto.

Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. *arXiv preprint arXiv:1803.00047*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Chenze Shao, Xilin Chen, and Yang Feng. 2018. Greedy search with probabilistic n-gram matching for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4778–4784.

Tianxiao Shen, Myle Ott, Michael Auli, and Marc'Aurelio Ranzato. 2019. Mixture models for diverse machine translation: Tricks of the trade. *arXiv preprint arXiv:1902.07816*.

Raphael Shu, Hideki Nakayama, and Kyunghyun Cho. 2019. Generating diverse translations with sentence codes. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Maosong Sun, Xinxiong Chen, Kaixu Zhang, Zhipeng Guo, and Zhiyuan Liu. 2016. Thulac: An efficient lexical analyzer for chinese.

Zewei Sun, Shujian Huang, Hao-Ran Wei, Xin-yu Dai, and Jiajun Chen. 2019. Generating diverse translation by manipulating multi-head attention. *arXiv preprint arXiv:1911.09333*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2016. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*.

Wen Zhang, Yang Feng, Fandong Meng, Di You, and Qun Liu. 2019. Bridging the gap between training and inference for neural machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4334–4343.