Sequence to Sequence Mixture Model for Diverse Machine Translation

Xuanli He **Gholamreza Haffari**
Monash University, Australia
{xuanli.he1, gholamreza.haffari}@monash.edu

Mohammad Norouzi
Google Brain
mnorouzi@google.com

Abstract

Sequence to sequence (SEQ2SEQ) models often lack diversity in their generated translations. This can be attributed to the limitation of SEQ2SEQ models in capturing lexical and syntactic variations in a parallel corpus resulting from different styles, genres, topics, or ambiguity of the translation process. In this paper, we develop a novel sequence to sequence mixture (S2SMIX) model that improves both translation diversity and quality by adopting a committee of specialized translation models rather than a single translation model. Each mixture component selects its own training dataset via optimization of the marginal log-likelihood, which leads to a soft clustering of the parallel corpus. Experiments on four language pairs demonstrate the superiority of our mixture model compared to a SEQ2SEQ baseline with standard or diversity-boosted beam search. Our mixture model uses negligible additional parameters and incurs no extra computation cost during decoding.

1 Introduction

Neural sequence to sequence (SEQ2SEQ) models have been remarkably effective machine translation (MT) (Sutskever et al., 2014; Bahdanau et al., 2015). They have revolutionized MT by providing a unified end-to-end framework, as opposed to the traditional approaches requiring several submodels and long pipelines. The neural approach is superior or on-par with statistical MT in terms of translation quality on various MT tasks and domains e.g. (Wu et al., 2016; Hassan et al., 2018).

A well recognized issue with SEQ2SEQ models is the lack of diversity in the generated translations. This issue is mostly attributed to the decoding algorithm (Li et al., 2016), and recently to the model (Zhang et al., 2016; Schulz et al., 2018a). The former direction has attempted to design diversity encouraging decoding algorithm, particularly beam search, as it generates translations sharing the majority of their tokens except a few trailing ones. The latter direction has investigated modeling enhancements, particularly the introduction of continuous latent variables, in order to capture lexical and syntactic variations in training corpora, resulted from the inherent ambiguity of the human translation process.[1] However, improving the translation diversity and quality with SEQ2SEQ models is still an open problem, as the results of the aforementioned previous work are not fully satisfactory.

In this paper, we develop a novel sequence to sequence mixture (S2SMIX) model that improves both translation quality and diversity by adopting a committee of specialized translation models rather than a single translation model. Each mixture component selects its own training dataset via optimization of the marginal log-likelihood, which leads to a soft clustering of the parallel corpus. As such, our mixture model introduces a conditioning global discrete latent variable for each sentence, which leads to grouping together and capturing variations in the training corpus. We design the architecture of S2SMIX such that the mixture components share almost all of their parameters and computation.

We provide experiments on four translation tasks, translating from English to German/French/Vietnamese/Spanish. The experiments show that our S2SMIX model consistently outperforms strong baselines, including SEQ2SEQ model with the standard and diversity encouraged beam search, in terms of both translation diversity and quality. The benefits of our mixture model comes with negligible additional parameters and no extra computation at inference time, compared to the vanilla SEQ2SEQ model.

[1] For a given source sentence, usually there exist several valid translations.

Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018), pages 583–592
Brussels, Belgium, October 31 - November 1, 2018. ©2018 Association for Computational Linguistics

2 Attentional Sequence to Sequence

An attentional sequence to sequence (SEQ2SEQ) model (Sutskever et al., 2014; Bahdanau et al., 2015) aims to directly model the conditional distribution of an output sequence $\mathbf{y} \equiv (y_1, \ldots, y_T)$ given an input $\mathbf{x}$, denoted $P(\mathbf{y} \mid \mathbf{x})$. This family of autoregressive probabilistic models decomposes the output distribution in terms of a product of distributions over individual tokens, often ordered from left to right as,

$$P_\theta(\mathbf{y} \mid \mathbf{x}) \;=\; \prod\nolimits_{t=1}^{|\mathbf{y}|} P_\theta(y_t \mid \mathbf{y}_{<t}, \mathbf{x}) \;, \qquad (1)$$

where $\mathbf{y}_{<t} \equiv (y_1, \ldots, y_{t-1})$ denotes a prefix of the sequence $\mathbf{y}$, and θ denotes the tunable parameters of the model.

Given a training dataset of input-output pairs, denoted by $\mathcal{D} \equiv \{(\mathbf{x}, \mathbf{y}^*)_d\}_{d=1}^{D}$, the *conditional log-likelihood* objective, predominantly used to train SEQ2SEQ models, is expressed as,

$$\ell_{\text{CLL}}(\theta) = \sum_{(\mathbf{x},\mathbf{y}^*)\in\mathcal{D}} \sum_{t=1}^{|\mathbf{y}^*|} \log P_\theta(y^*_t \mid \mathbf{y}^*_{<t}, \mathbf{x}) \,. \quad (2)$$

A standard implementation of the SEQ2SEQ model is composed of an encoder followed by a decoder. The encoder transforms a sequence of source tokens denoted $(x_1, \ldots, x_N)$, into a sequence of hidden states denoted $(\mathbf{h}_1, \ldots, \mathbf{h}_N)$ via a recurrent neural network (RNN). Attention provides an effective mechanism to represent a soft alignment between the tokens of the input and output sequences (Bahdanau et al., 2015), and more recently to model the dependency among the output variables (Vaswani et al., 2017).

In our model, we adopt a bidirectional RNN with LSTM units (Hochreiter and Schmidhuber, 1997). Each hidden state $\mathbf{h}_n$ is the concatenation of the states produced by the forward and backward RNNs, $\mathbf{h}_n = [\mathbf{h}_{\rightarrow n}, \mathbf{h}_{n\leftarrow}]$. Then, we use a two-layer RNN decoder to iteratively emit individual distributions over target tokens $(y_1, ..., y_T)$. At time step t, we compute the hidden representations of an output prefix $\mathbf{y}_{\leq t}$ denoted $\mathbf{s}^1_t$ and $\mathbf{s}^2_t$ based on an embedding of y_t denoted $\mathrm{M}[y_t]$, previous representations $\mathbf{s}^1_{t-1}$, $\mathbf{s}^2_{t-1}$, and a context vector $\mathbf{c}_t$ as,

$$\begin{aligned} \mathbf{s}^1_t &= \text{LSTM}(\mathbf{s}^1_{t-1}, \mathrm{M}[y_t]; \mathbf{c}_t) \;, && (3)\\ \mathbf{s}^2_t &= \text{LSTM}(\mathbf{s}^2_{t-1}, \mathbf{s}^1_t; \mathbf{c}_t) \;, && (4)\\ P_\theta(y_{t+1} \,|\, \mathbf{y}_{\leq t}, \mathbf{x}) &= \text{softmax}(W\mathbf{s}^2_t + W'\mathbf{c}_t) \;, && (5) \end{aligned}$$

where M is the embedding table, and W and W' are learnable parameters. The context vector $\mathbf{c}_t$ is computed based on the input and attention,

$$\begin{aligned} e_{t,n} &= \mathbf{v}^\top \tanh(W_h\mathbf{h}_n + W_s\mathbf{s}^1_{t-1} + \mathbf{b}_a) \;, && (6)\\ \mathbf{a}_t &= \text{softmax}(\mathbf{e}_t) \;, && (7)\\ \mathbf{c}_t &= \sum_n a_{t,n}\, \mathbf{h}_n \;, && (8) \end{aligned}$$

where W_h, W_s, $\mathbf{b}_a$, and $\mathbf{v}$ are learnable parameters, and $\mathbf{a}_t$ is the attention distribution over the input tokens at time step t. The decoder utilizes the attention information to decide which input tokens should influence the next output token y_{t+1}.

3 Sequence to Sequence Mixture Model

We develop a novel *sequence to sequence mixture* (S2SMIX) model that improves both translation quality and diversity by adopting a committee of specialized translation models rather than a single translation model. Each mixture component selects its own training dataset via optimization of the marginal log-likelihood, which leads to a soft clustering of the parallel corpus. We design the architecture of S2SMIX such that the mixture components share almost all of their parameters except a few conditioning parameters. This enables a direct comparison against a SEQ2SEQ baseline with the same number of parameters.

Improving translation diversity within SEQ2SEQ models has received considerable recent attention (*e.g.*, Vijayakumar et al. (2016); Li et al. (2016)). Given a source sentence, human translators are able to produce a set of diverse and reasonable translations. However, although beam search for SEQ2SEQ models is able to generate various candidates, the final candidates often share majority of their tokens, except a few trailing ones. The lack of diversity within beam search raises an issue for possible re-ranking systems and for scenarios where one is willing to show multiple translation candidates to the user. Prior work attempts to improve translation diversity by incorporating a diversity penalty during beam search (Vijayakumar et al., 2016; Li et al., 2016). By contrast, our S2SMIX model naturally incorporates diversity both during training and inference.

The key difference between the SEQ2SEQ and S2SMIX models lies in the formulation of the conditional probability of an output sequence $\mathbf{y}$ given an input $\mathbf{x}$. The S2SMIX model represents $P_\theta(\mathbf{y} \mid$

$\mathbf{x}$) by marginalizing out a discrete latent variable $z \in \{1, \dots, K\}$, which indicates the selection of the mixture component, *i.e.,*

$$P_\theta(\mathbf{y} \,|\, \mathbf{x}) = \sum_{z=1}^{K} P_\theta(\mathbf{y} \,|\, \mathbf{x}, z)\, P(z \,|\, \mathbf{x})\,, \qquad (9)$$

where K is the number of mixture components. For simplicity and to promote diversity, we assume that the mixing coefficients follow a uniform distribution such that for all $z \in \{1, \dots, K\}$,

$$P(z \mid \mathbf{x}) \;=\; 1/K\,. \qquad (10)$$

For the family of S2SMIX models with uniform mixing coefficients (10), the conditional log-likelihood objective (2) can be re-expressed as:

$$\ell_{\text{CLL}}(\theta) \;=\; \text{constant} + \sum_{(\mathbf{x},\mathbf{y}^*)\in\mathcal{D}} \log \sum_{z=1}^{K} \underbrace{\exp \sum_{t=1}^{|\mathbf{y}^*|} \log P_\theta(y_t^* \mid \mathbf{y}_{<t}^*, \mathbf{x}, z)}_{P_\theta(\mathbf{y}|\mathbf{x},z)}\;, \qquad (11)$$

where $\log(1/K)$ terms were excluded because they offset the objective by a constant value. Such a constant has no impact on learning the parameters θ. One can easily implement the objective in (11) using automatic differentiation software such as tensorflow (Abadi et al., 2016), by adopting a LogSumExp operator to aggregate the loss of the individual mixture components. When the number of components K is large, computing the terms $P_\theta(y_t^* \mid \mathbf{y}_{<t}^*, \mathbf{x}, z)$ for all values of $z \in \{1, \dots, K\}$ can require a lot of GPU memory. To mitigate this issue, we will propose a memory efficient formulation in Section 3.3 inspired by the EM algorithm.

3.1 S2SMIX Architecture

We design the architecture of the S2SMIX model such that individual mixture components can share as many parameters and as much computation as possible. Accordingly, all of the mixture components share the same encoder, which requires processing the input sentence only once. We consider different ways of injecting the conditioning signal into the decoder. As depicted in Figure 1, we consider different ways of injecting the conditioning on z into our two-layer decoder. These different variants require additional lookup tables denoted $\mathrm{M}_1, \mathrm{M}_2$, or M_b.

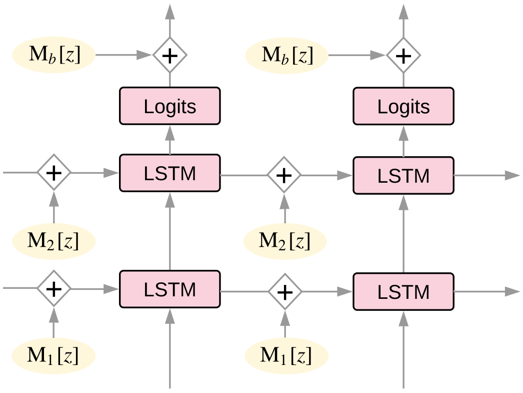

Figure 1: An illustration of a two-layer LSTM decoder with different ways of injecting the conditioning signal.

When we incorporate the conditioning on z into the LSTM layers, each lookup table (*e.g.,* M_1 and M_2) has K rows and D_{LSTM} columns, where D_{LSTM} denotes the number of dimensions of the LSTM states (512 in our case). We combine the state of the LSTM with the conditioning signal via simple addition. Then the LSTM update equations take the form,

$$\mathbf{s}_t^i \;=\; \text{LSTM}(\mathbf{s}_{t-1}^i + \mathrm{M}_i[z],\ \text{input})\,, \qquad (12)$$

for $i \in \{1, 2\}$. We refer to the addition of the conditioning signal to the bottom and top LSTM layers of the decoder as *bt* and *tp* respectively. Note that in the bt configuration, the attention mask depends on the indicator variable z, whereas in the tp configuration that attention mask is shared across different mixture components.

We also consider incorporating the conditioning signal into the softmax layer to bias the selection of individual words in each mixture component. Accordingly, the embedding table M_b has K rows and D_{vocab} entries, and the logits from (5) are added to the corresponding row of M_b as,

$$P_\theta(y_{t+1} \,|\, \mathbf{y}_{\leq t}, \mathbf{x}, z) \;=\; \text{softmax}(\text{logits}{+}\mathrm{M}_b[z])\,. \qquad (13)$$

We refer to this configuration as *sf* and to the configuration that includes all of the conditioning signals as *all*.

3.2 Separate Beam Search per Component

At the inference stage, we conduct a separate beam search per mixture component. Performing beam search independently for each component encourages diversity among the translation candidates as different mixture components often prefer certain phrases and linguistic structures over each other. Let $\hat{\mathbf{y}}_z$ denote the result of the beam search for a mixture component z. The final output of our model, denoted $\hat{\mathbf{y}}$ is computed by selecting the translation candidate with the highest probability under the corresponding mixture component, *i.e.,*

$$\hat{\mathbf{y}} \;=\; \operatorname*{argmax}_{1\leq z\leq K} \; \log P_\theta(\hat{\mathbf{y}}_z \mid \mathbf{x}, z)\;. \qquad (14)$$

In order to accurately estimate the conditional probability of each translation candidate based on (9), one needs to evaluate each candidate using all of the mixture components. However, this process considerably increases the inference time and latency. Instead, we approximate the probability of each candidate by only considering the mixture component based on which the candidate translation has been decoded, as outlined in (14). This approximation also encourages the diversity as we emphasized in this work.

Note that we have K mixture components and a beam search of b per component. Overall, this requires processing $K \times b$ candidates. Accordingly, we compare our model with a SEQ2SEQ model using the same beam size of $K \times b$.

3.3 Memory Efficient Formulation

In this paper, we adopt a relatively small number of mixture components (up to 16), but to encompass various clusters of linguistic content and style, one may benefit from a large number of components. Based on our experiments, the memory footprint of a S2SMIX with K components increases by about K folds, partly because the softmax layers take a big fraction of the memory. To reduce the memory requirement for training our model, inspired by prior work on EM algorithm (Neal and Hinton, 1998), we re-express the gradient of the conditional log-likelihood objective in (11) *exactly* as,

$$\frac{d}{d\theta}\ell_{\mathrm{CLL}}(\theta) \;=\; \sum_{(\mathbf{x},\mathbf{y}^*)\in\mathcal{D}} \sum_{z=1}^{K} P(z \mid \mathbf{x}, \mathbf{y}^*) \frac{d}{d\theta} \log P_\theta(\mathbf{y}^* \mid \mathbf{x}, z)\;, \qquad (15)$$

where with uniform mixing coefficients, the posterior distribution $P(z \mid \mathbf{x}, \mathbf{y}^*)$ takes the form,

$$P(z \mid \mathbf{x}, \mathbf{y}) \;=\; \frac{\exp \ell_z(\mathbf{y} \,|\, \mathbf{x})}{\sum_k \exp \ell_k(\mathbf{y} \,|\, \mathbf{x})}\;, \qquad (16)$$

where $\ell_z(\mathbf{y} \,|\, \mathbf{x}) = \log P_\theta(\mathbf{y} \,|\, \mathbf{x}, z)$.

Based on this formulation, one can compute the posterior distribution in a few forward passes, which require much less memory. Then, one can draw one or a few Monte Carlo (MC) samples from the posterior to obtain an unbiased estimate of the gradient in (15). As shown in algorithm 1, the training procedure is divided into two parts. For each minibatch we compute the component-specific log-loss for different mixture components in the first stage. Then, we exponentiate and normalize the losses as in (16) to obtain the posterior distribution. Finally, we draw one sample from the posterior distribution per input-output example, and optimize the parameters according to the loss of such a component. These two stages are alternated until the model converges. We note that this algorithm follows an *unbiased* stochastic gradient of the marginal log likelihood.

Algorithm 1 Memory efficient S2SMIX

```
Initialize a computational graph: cg
Initialize a optimizer: opt
repeat
    draw a random minibatch of the data
    empty list Γ
    for $z = 1$ to $K$ do
        $\ell_z$ := cg.forward(minibatch, $z$)
        Γ := add exp($\ell_z$) to Γ
    end for
    Γ := normalize(Γ)
    $\tilde{z}$ := sample(Γ)
    $\ell$ := cg.forward(minibatch, $\tilde{z}$)
    opt.gradient_descent($\ell$)
until converge
```

4 Experiments

Dataset. To assess the effectiveness of the S2SMIX model, we conduct a set of translation experiments on TEDtalks on four language pairs: English→French (en-fr), English→German (en-de), English→Vietnamese (en-vi), and English→Spanish (en-es).

We use IWSLT14 dataset[2] for en-es, IWSLT15

[2] https://sites.google.com/site/iwsltevaluation2014/home

Data	en-fr	en-de	en-vi	en-es
Train	208,719	189,600	133,317	173,601
Dev	5,685	6,775	1,553	5,401
Test	2,762	2,762	1,268	2,504

Table 1: Statistics of all language pairs for IWSLT data after preprocessing

dataset for en-vi, and IWSLT16 dataset[3] for en-fr and en-de. We pre-process the corpora by Moses tokenizer[4], and preserve the true case of the text. For en-vi, we use the pre-processed corpus distributed by Luong and Manning (2015)[5]. For training and dev sets, we discard all of the sentence pairs where the length of either side exceeds 50 tokens. The number of sentence pairs of different language pairs after preprocessing are shown in Table 1. We apply byte pair encoding (BPE) (Sennrich et al., 2016) to handle rare words on en-fr, en-de and en-es, and share the BPE vocabularies between the encoder and decoder for each language pair.

Implementation details. All of the models use a one-layer bidirectional LSTM encoder and a two-layer LSTM decoder. Each LSTM layer in the encoder and decoder has a 512 dimensional hidden state. Each input word embeddings is 512 dimensional as well. We adopt the Adam optimizer (Kingma and Ba, 2014). We adopt dropout with a 0.2 dropout rate. The minibatch size is 64 sentence pairs. We train each model 15 epochs, and select the best model in terms of the perplexity on the dev set.

Diversity metrics. Having more diversity in the candidate translations is one of the major advantages of the S2SMix model. To quantify diversity within a set $\{\hat{\mathbf{y}}_m\}_{m=1}^M$ of translation candidates, we propose to evaluate average pairwise BLEU between pairs of sentences according to

$$\text{div\_bleu} \equiv 100 - \frac{\sum_{i=1}^{M}\sum_{j=i+1}^{M} \text{BLEU}(\hat{\mathbf{y}}_i, \hat{\mathbf{y}}_j)}{M(M-1)/2} \quad (17)$$

As an alternative metric of diversity within a set $\{\hat{\mathbf{y}}_m\}_{m=1}^M$ of translations, we propose a metric based on the fraction of the n-grams that are not shared among the translations, *i.e.*,

$$\text{div\_ngram} \equiv 1 - \frac{\left|\cap_{m=1}^{M} \text{ngrams}(\hat{\mathbf{y}}_m)\right|}{\left|\cup_{m=1}^{M} \text{ngrams}(\hat{\mathbf{y}}_m)\right|} \quad (18)$$

where ngram$(\mathbf{y})$ returns the set of unique n-grams in a sequence $\mathbf{y}$. We report average div_bleu and average div_ngram across the test set for the translation candidates found by beam search. We measure and report bigram diversity in the paper and report unigram diversity in the supplementary material.

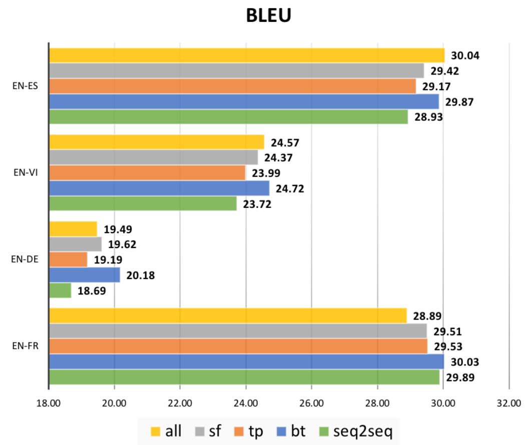

Figure 2: BLEU scores of the different variants of S2SMix model and Seq2Seq model.

4.1 Experimental results

S2SMix configuration. We start by investigating which of the ways of injecting the conditioning signal into the S2SMix model is most effective. As seen in Section 3, the mixture components can be built by adding component-specific vectors to the logits (sf), the top LSTM layer (tp) or the bottom LSTM layer (bt) in the decoder, or all of them (all). Figure 2 shows the BLEU score of these variants on the translation tasks across four different language pairs. We observe that adding a component-specific vector to the recurrent cells in the bottom layer of the decoder is the most effective, and results in BLEU scores superior or on-par with the other variants across the four language pairs.

Therefore, we use this model variant in all experiments for the rest of the paper.

Furthermore, Table 2 shows the number of parameters in each of the variants as well as the base Seq2Seq model. We confirm that the mixture model variants introduce negligible number

[3] https://sites.google.com/site/iwsltevaluation2016

[4] https://github.com/moses-smt/mosesdecoder

[5] https://nlp.stanford.edu/projects/nmt

	en-fr	en-de	en-vi	en-es
SEQ2SEQ	173.22	173.78	112.76	173.21
S2SMIX-4				
bt	173.23	173.79	112.77	173.22
tp	173.23	173.79	112.77	173.22
sf	173.70	174.27	112.88	173.70
all	173.72	174.29	112.90	173.72

Table 2: Size of the parameters (MB) for the base SEQ2SEQ model and the variants of S2SMIX with four mixtures.

	beam	en-fr	en-de	en-vi	en-es
SEQ2SEQ	4	30.26	19.52	24.82	29.40
	8	30.18	19.77	23.55	29.76
	16	27.63	19.13	19.05	28.19
S2SMIX-4	1	30.61	20.18	25.16	31.17
	2	31.22	20.71	25.28	31.47
	4	31.97	21.08	25.36	31.21

Table 3: BLEU scores of different systems over different search space.

of new parameters compared to the base SEQ2SEQ model. Specifically, only up to 0.002% increase in the parameter size are introduced, across all of the language pairs and mixture model variants.

S2SMIX vs. SEQ2SEQ. We compare our mixture model against a vanilla SEQ2SEQ model both in terms of translation quality and diversity. To be fair, we compare models with the same number of beams during inference, *e.g.,* we compare vanilla SEQ2SEQ using a beam size of 8 with S2SMIX-4 with 4 component and a beam size of 2 per component.

As an effective regularization strategy, we adopt label smoothing to strengthen generalisation performance (Szegedy et al., 2016; Pereyra et al., 2017; Edunov et al., 2018). Unlike conventional cross-entropy loss, where the probability mass for the ground truth word y is set to 1 and $q(y') = 0$ for $y' \neq y$, we smooth this distribution as:

$$q(y) = 1 - \epsilon, \quad (19)$$

$$q(y') = \frac{\epsilon}{V-1} \quad (20)$$

where ϵ is a smoothing parameter, and V is the vocabulary size. In our experiments, ϵ is set to 0.1.

Table 3 shows the results across four language pairs. Each row in the top part should be compared with the corresponding row in the bottom part for a fair comparison in terms of the effective beam size. Firstly, we observe that increasing the beam size deteriorates the BLEU score for the SEQ2SEQ model. Similar observations have been made in the previous work (Tu et al., 2017). This behavior is in contrast to our S2SMIX model where increasing the beam size improves the BLEU score, except en-es, which demonstrates a decreasing trend when beam size increases from 2 to 4. Secondly, our S2SMIX models outperform their SEQ2SEQ counterparts in all settings with the same number of bins.

	en-fr	en-de	en-vi	en-es
BLEU				
SEQ2SEQ-d	29.85	19.18	24.62	29.72
S2SMIX-4	30.61	20.18	25.16	31.17
DIV_BLEU				
SEQ2SEQ-d	20.43	22.66	14.51	18.83
S2SMIX-4	34.85	47.85	37.40	38.31

Table 4: S2SMIX with 4 components vs SEQ2SEQ endowed with the beam-diverse decoder (Li et al., 2016) with the beam size of 4.

Figure 3 shows the diversity comparison between the S2SMIX model and the vanilla SEQ2SEQ model where the number of decoding beams is 8. The diversity metrics are bigram and BLEU diversity as defined earlier in the section. Our S2SMIX models significantly dominate the SEQ2SEQ model across language pairs in terms of the diversity metrics, while keeping the translation quality high (c.f. the BLEU scores in Table 3).

We further compare against the SEQ2SEQ model endowed with the beam-diverse decoder (Li et al., 2016). This decoder penalizes sibling hypotheses generated from the same parent in the search tree, according to their ranks in each decoding step. Hence, it tends to rank high those hypotheses from different parents, hence encouraging diversity in the beam.

Table 4 presents the BLEU scores as well as the diversity measures. As seen, the mixture model significantly outperforms the SEQ2SEQ endowed with the beam-diverse decoder, in terms of the diversity in the generated translations. Furthermore, the mixture model achieves up to 1.7 BLEU score improvements across three language pairs.

	en-fr		en-de		en-vi		en-es	
	BLEU	time	BLEU	time	BLEU	time	BLEU	time
S2SMix-4	30.61	1.25	20.18	1.33	25.16	1.14	31.17	1.30
MC sampling:								
S2SMix-4	30.43	1.67	19.74	1.67	24.93	1.58	31.27	1.67
S2SMix-8	30.66	2.08	20.41	2.05	24.86	2.00	31.44	2.06
S2SMix-16	30.74	3.10	20.43	2.88	24.90	2.83	30.82	3.02

Table 5: BLEU scores using greedy decoding and training time based on the original log-likelihood objective and online EM coupled with gradient estimation based on a single MC sample. The training time is reported by taking the average running time of one minibatch update across a full epoch.

Large mixture models. Memory limitations of the GPU may make it difficult to increase the number of mixture components beyond a certain amount. One approach is to decrease the number of sentence pairs in a minibatch, however, this results in a substantial increase in the training time. Another approach is to resort to MC gradient estimation as discussed in Section 3.3.

The top-part of Table 5 compares the models trained by online EM vs the original log-likelihood objective, in terms of the BLEU score and the training time. As seen, the BLEU score of the EM-trained models are on-par with those trained on the log-likelihood objective. However, online EM leads to up to 35% increase in the training time for S2SMix-4 across four different language pairs, as we first need to do a forward pass on the minibatch in order to form the lower bound on the log-likelihood training objective.

The bottom-part of Table 5 shows the effect of online EM coupled with sampling only one mixture component to form a stochastic approximation to the log-likelihood lower bound. For each minibatch, we first run a forward pass to compute the probability of each mixture component given each sentence pair in the minibatch. We then sample a mixture component for each sentence-pair to form the approximation of the log-likelihood lower bound for the minibatch, which is then optimized using back-propagation. As we increase the number of mixture components from 4 to 8, we see about 0.7 BLEU score increase for en-de; while there is no significant change in the BLEU score for en-fr, en-vi and en-es.

Increasing the number of mixture components further to 16 does not produce gains on these datasets. Time-wise, training with online EM coupled with 1-candidate sampling should be significantly faster that the vanilla online EM and the original likelihood objective in principle, as we need to perform the backpropagation only for the selected mixture component (as opposed to all mixture components). Nonetheless, the additional computation due to increasing the number of mixtures from 4 to 8 is about 26%, which increases to about 55% when going from 8 to 16 mixture components.

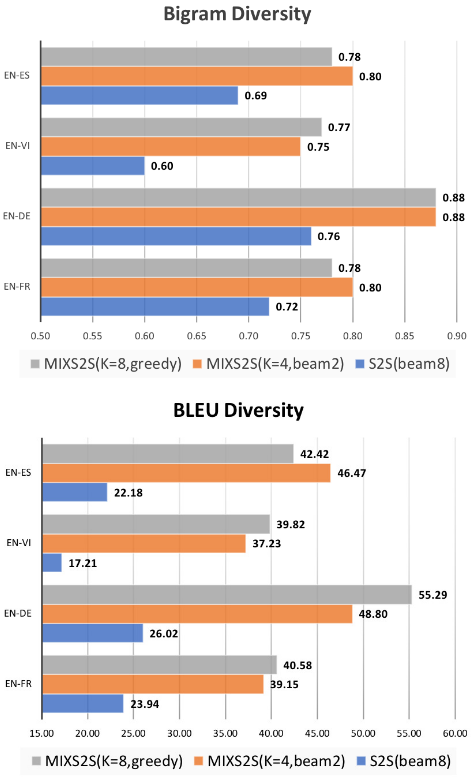

Figure 3: Diversity bigram (top) and BLEU (bottom) for the Seq2Seq model vs S2SMix models, with the number of decoding beams set to 8.

4.2 Qualitative Analysis

Finally, we would like to demonstrate that our S2SMix does indeed encourage diversity and im-

Source	And this information is stored for at least six months in Europe , up to two years .
Reference	Và những thông tin này được lưu trữ trong ít nhất sáu tháng ở châu Âu , cho tới tận hai năm .
SEQ2SEQ	
	Và thông tin này được lưu trữ trong ít sáu tháng ở Châu Âu , hai năm tới .
	Và thông tin này được lưu trữ trong ít sáu tháng ở Châu Âu , trong hai năm tới .
	Và thông tin này được lưu trữ trong ít sáu tháng ở Châu Âu , hai năm tới
	Và thông tin này được lưu trữ trong ít sáu tháng ở Châu Âu , trong hai năm tới
S2SMIX	
	Và thông tin này được lưu trữ trong ít nhất 6 tháng ở châu Âu , đến 2 năm .
	Và thông tin này được lưu trữ trong ít nhất 6 tháng ở châu Âu , lên tới hai năm .
	Và thông tin này được lưu trữ trong ít nhất 6 tháng ở châu Âu , trong vòng hai năm .
	Và thông tin này được lưu trữ trong ít nhất 6 tháng ở châu Âu , lên tới hai năm

Table 6: Words indicate diversity compared with the references, while red words denote translation improvement.

prove the translation quality. As shown in Table 6, compared with SEQ2SEQ, which mistranslates the second clause, our S2SMIX is not only capable of generating a group of correct translation, but also emitting synonyms for different mixture components. We provide more examples in the supplementary material.

5 Related Work

Obviously, different domains aim at different readers, thus they exhibit distinctive genres compared to other domains. A well-tuned MT system cannot directly apply to new domains; otherwise, translation quality will degrade. Based on this factor, out-domain adaptation has been widely studied for MT, ranging from data selection (Li et al., 2010; Wang et al., 2017), tuning (Luong and Manning, 2015; Farajian et al., 2017) to domain tags (Chu et al., 2017). Similarly, in-domain adaptation is also a compelling direction. Normally, to train an universal MT system, the training data consist of gigantic corpora covering numerous and various domains.This training data is naturally so diverse that Mima et al. (1997) incorporated extra-linguistic information to enhance translation quality. Michel and Neubig (2018) argue even without explicit signals (gender, politeness etc.), they can handle domain-specific information via annotation of speakers, and easily gain quality improvement from a larger number of domains. Our approach is considerably different from the previous work. We remove any extra annotation, and treat domain-related information as latent variables, which are learned from corpus.

Prior to our work, diverse generation has been studied in image captioning, as some of the training set are comprised of images paired with multiple reference captions. Some work puts their efforts on decoding stages, and form a group of beam search to encourage diversity (Vijayakumar et al., 2016), while others pay more attention to adversarial training (Shetty et al., 2017; Li et al., 2018). Within translation, our method is similar to Schulz et al. (2018b), where they propose a MT system armed with variational inference to account for translation variations. Like us, their diversified generation is driven by latent variables. Albeit the simplicity of our model, it is effective and able to accommodate variation or diversity. Meanwhile, we propose several diversity metrics to perform quantitative analysis.

Finally, Yang et al. (2018) proposes a mixture of softmaxes to enhance the expressiveness of language model, which demonstrate the effectiveness of our S2SMIX model under the matrix factorization framework.

6 Conclusions and Future Work

In this paper, we propose a sequence to sequence mixture (S2SMIX) model to improve translation diversity within neural machine translation via incorporating a set of discrete latent variables. We propose a model architecture that requires negligible additional parameters and no extra computation at inference time. In order to address prohibitive memory requirement associated with large mixture models, we augment the training procedure by computing the posterior distribution followed by Monte Carlo sampling to estimate the gradients. We observe significant gains both in terms of BLEU scores and translation diversity with a mixture of 4 components. In the future, we intend to replace the uniform mixing coefficients with learnable parameters, since different components should not necessarily make an equal contribution to a given sentence pair. Moreover, we will consider applying our S2SMIX model to other NLP problems in which diversity plays an important role.

7 Acknowledgements

We would like to thank Quan Tran, Trang Vu and three anonymous reviewers for their valuable comments and suggestions. This work was supported by the Multi-modal Australian ScienceS Imaging and Visualisation Environment (MASSIVE)[6], and in part by an Australian Research Council grant (DP160102686).

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian J. Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Józefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Gordon Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul A. Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda B. Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv:1603.04467*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *ICLR*.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of simple domain adaptation methods for neural machine translation. *CoRR*, abs/1701.03214.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. *In Proceedings of NAACL-HLT 2018*.

M Amin Farajian, Marco Turchi, Matteo Negri, and Marcello Federico. 2017. Multi-domain neural machine translation through unsupervised adaptation. In *Proceedings of the Second Conference on Machine Translation*, pages 127–137.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, et al. 2018. Achieving human parity on automatic chinese to english news translation. *arXiv:1803.05567*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Dianqi Li, Xiaodong He, Qiuyuan Huang, Ming-Ting Sun, and Lei Zhang. 2018. Generating diverse and accurate visual captions by comparative adversarial learning. *arXiv preprint arXiv:1804.00861*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. A simple, fast diverse decoding algorithm for neural generation. *CoRR*, abs/1611.08562.

Mu Li, Yinggong Zhao, Dongdong Zhang, and Ming Zhou. 2010. Adaptive development data selection for log-linear model in statistical machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 662–670. Association for Computational Linguistics.

Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *IWSLT*.

Paul Michel and Graham Neubig. 2018. Extreme adaptation for personalized neural machine translation. In *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

Hideki Mima, Osamu Furuse, and Hitoshi Iida. 1997. Improving performance of transfer-driven machine translation with extra-linguistic informatioon from context, situation and environment. In *IJCAI (2)*, pages 983–989.

Radford M Neal and Geoffrey E Hinton. 1998. A view of the em algorithm that justifies incremental, sparse, and other variants. *Learning in graphical models*.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*.

Philip Schulz, Wilker Aziz, and Trevor Cohn. 2018a. A stochastic decoder for neural machine translation. *CoRR*, arXiv:1805.10844.

Philip Schulz, Wilker Aziz, and Trevor Cohn. 2018b. A stochastic decoder for neural machine translation. In *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

[6] https://www.massive.org.au/

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1715–1725.

Rakshith Shetty, Marcus Rohrbach, Lisa Anne Hendricks, Mario Fritz, and Bernt Schiele. 2017. Speaking the same language: Matching machine to human captions by adversarial training. In *In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *NIPS*.

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.

Zhaopeng Tu, Yang Liu, Lifeng Shang, Xiaohua Liu, and Hang Li. 2017. Neural machine translation with reconstruction. *AAAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NIPS*.

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*.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv:1609.08144*.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2018. Breaking the softmax bottleneck: A high-rank RNN language model. In *International Conference on Learning Representations*.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. *CoRR*, arXiv:1605.07869.