# Neural Machine Translation Advised by Statistical Machine Translation

**Xing Wang**[†] **Zhengdong Lu**[‡] **Zhaopeng Tu**[‡] **Hang Li**[‡] **Deyi Xiong**[†*] **Min Zhang**[†]
[†]Soochow University, Suzhou
xingwsuda@gmail.com, {dyxiong, minzhang}@suda.edu.cn
[‡] Noah's Ark Lab, Huawei Technologies, Hong Kong
{lu.zhengdong, tu.zhaopeng, hangli.hl}@huawei.com

## Abstract

Neural Machine Translation (NMT) is a new approach to machine translation that has made great progress in recent years. However, recent studies show that NMT generally produces fluent but inadequate translations (Tu et al. 2016; He et al. 2016). This is in contrast to conventional Statistical Machine Translation (SMT), which usually yields adequate but non-fluent translations. It is natural, therefore, to leverage the advantages of both models for better translations, and in this work we propose to incorporate SMT model into NMT framework. More specifically, at each decoding step, SMT offers additional recommendations of generated words based on the decoding information from NMT (e.g., the generated partial translation and attention history). Then we employ an auxiliary classifier to score the SMT recommendations and a gating function to combine the SMT recommendations with NMT generations, both of which are jointly trained within the NMT architecture in an end-to-end manner. Experimental results on Chinese-English translation show that the proposed approach achieves significant and consistent improvements over state-of-the-art NMT and SMT systems on multiple NIST test sets.

## Introduction

Neural machine translation has been receiving considerable attention in recent years (Kalchbrenner and Blunsom 2013; Cho et al. 2014b; Sutskever, Vinyals, and Le 2014; Bahdanau, Cho, and Bengio 2015). Compared with the conventional SMT (Brown et al. 1993; Koehn, Och, and Marcu 2003; Chiang 2005), one great advantage of NMT is that the translation system can be completely constructed by learning from data without human involvement (cf., feature engineering in SMT). Another major advantage of NMT is that the gating (Hochreiter and Schmidhuber 1997) and attention (Bahdanau, Cho, and Bengio 2015) techniques prove to be effective in modeling long-distance dependencies and complicated alignment relations in the translation process, which poses a serious challenge for SMT.

Despite these benefits, recent studies show that NMT generally produces fluent yet sometimes inaccurate translations, mainly due to:

1. *Coverage problem* (Tu et al. 2016; Cohn et al. 2016): NMT lacks of a mechanism to record the source words that have been translated or need to be translated, resulting in either "over-translation" or "under-translation" (Tu et al. 2016).
2. *Imprecise translation problem* (Arthur, Neubig, and Nakamura 2016): NMT is prone to generate words that seem to be natural in the target sentence, but do not reflect the original meaning of the source sentence.
3. *UNK problem* (Jean et al. 2015; Luong et al. 2015): NMT uses a fixed modest-sized vocabulary to represent most frequent words and replaces other words with a UNK word. Experimental results show that translation quality degrades rapidly with the number of UNK words increasing (Cho et al. 2014a).

Instead of employing different models which individually focus on the above NMT problems, in this work, we try to address the problems together in a single approach. Our approach is based on the observation that SMT models have desirable properties that can properly deal with the above problems:

1. SMT has a mechanism to guarantee that every source word is translated.
2. SMT treats words as discrete symbols, which ensures that a source word will be translated into a target word which has been observed at least once in the training data.
3. SMT explicitly memorizes all the translations, including translations of rare words that are taken as UNK in NMT.

It is natural to leverage the advantages of the two sorts of models for better translations. Recently, several researchers proposed to improve NMT with SMT features or outputs. For example, He et al. (2016) integrated SMT features with the NMT model under the log-linear framework in the beam search on the development or test set. Stahlberg et al. (2016) extended the beam search decoding by expanding the search space of NMTs with translation hypotheses produced by a syntactic SMT model. In the above work, NMT was treated as a black-box and the combinations were carried out only in the testing phase.

In this work, we move a step forward along the same direction by incorporating the SMT model into the training phase of NMT. This enables NMT to effectively learn to incorporate SMT recommendations, rather than to heuristically combine two trained models. Specifically, SMT model

[*]Corresponding author

is firstly trained independently on a bilingual corpus using the conventional phrase-based SMT approach (Koehn, Och, and Marcu 2003). At each decoding step, in both training and testing phases, the trained SMT model provides translation recommendations based on the decoding information from NMT, including the generated partial translation and the attention history.

We employ an auxiliary classifier to score the SMT recommendations, and use a gating function to linearly combine the two probabilities between the NMT generations and SMT recommendations. The gating function reads the current decoding information, and thus is able to dynamically assign weights to NMT and SMT probabilities at different decoding steps. Both the SMT classifier and gating function are jointly trained within the NMT architecture in an end-to-end manner. In addition, to better alleviate the UNK problem in the testing phase, we select a proper SMT recommendation to replace a target UNK word by jointly considering the attention probabilities from the NMT model and the coverage information from the SMT model. Experimental results show that the proposed approach can achieve significant improvements of 2.44 BLEU point over the NMT baseline and 3.21 BLEU point over the Moses (SMT) system on five Chinese-English NIST test sets.

## Background

In this section, we give a brief introduction to NMT and phrase-based SMT, the most popular SMT model.

### Neural Machine Translation

Given a source sentence $\mathbf{x} = x_1, x_2, ..., x_{T_x}$, attention-based NMT (Bahdanau, Cho, and Bengio 2015) encodes it into a sequence of vectors, then uses the sequence of vectors to generate a target sentence $\mathbf{y} = y_1, y_2, ..., y_{T_y}$.

At the encoding step, attention-based NMT uses a bidirectional RNN which consists of forward RNN and backward RNN (Schuster and Paliwal 1997) to encode the source sentence. The forward RNN reads the source sentence $\mathbf{x}$ in a forward direction, generating a sequence of forward hidden states $\overrightarrow{h} = \left[\overrightarrow{h_1}, \overrightarrow{h_2}, ..., \overrightarrow{h_{T_x}}\right]$. The backward RNN reads the source sentence $\mathbf{x}$ in a backward direction, generating a sequence of backward hidden states $\overleftarrow{h} = \left[\overleftarrow{h_1}, \overleftarrow{h_2}, ..., \overleftarrow{h_{T_x}}\right]$. The pair of hidden states at each position are concatenated to form the annotation of the word at the position, yielding the annotations of the entire source sentence $\mathbf{h} = [h_1, h_2, ..., h_{T_x}]$, where

$$h_j^\top = \left[\overrightarrow{h_j^\top}; \overleftarrow{h_j^\top}\right] \tag{1}$$

At the decoding step $t$, after outputting target sequence $\mathbf{y_{<t}} = y_1, y_2, ..., y_{t-1}$, the next word $y_t$ is generated with probability

$$p(y_t|\mathbf{y_{<t}}, \mathbf{x}) = softmax(f(s_t, y_{t-1}, c_t)) \tag{2}$$

where $f(\cdot)$ is a non-linear activation function and $s_t$ is the decoder's hidden state at step $t$:

$$s_t = g(s_{t-1}, y_{t-1}, c_t) \tag{3}$$

where $g(\cdot)$ is a non-linear activation function. Here we use Gated Recurrent Unit (Cho et al. 2014b) as the activation function for the encoder and decoder. $c_t$ is the context vector, computed as a weighted sum of the annotations of the source sentence:

$$c_t = \sum_{j=1}^{T_x} \alpha_{t,j} h_j \tag{4}$$

where $h_j$ is the annotation of source word $x_j$ and its weight $\alpha_{t,j}$ is computed by the attention model.

### Statistical Machine Translation

Most SMT models are defined with the log-linear framework (Och and Ney 2002).

$$p(\mathbf{y}|\mathbf{x}) = \frac{\exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{y}, \mathbf{x}))}{\sum_{\mathbf{y'}} \exp(\sum_{m=1}^{M} \lambda_m h_m(\mathbf{y'}, \mathbf{x}))} \tag{5}$$

where $h_m(\mathbf{y}, \mathbf{x})$ is a feature function and $\lambda_m$ is its weight. Standard SMT features include the bidirectional translation probabilities, the bidirectional lexical translation probabilities, the language model, the reordering model, the word penalty and the phrase penalty. The feature weights can be tuned by the minimum error rate training (MERT) algorithm (Och 2003).

During translation, the SMT decoder expands partial translation (called translation hypothesis in SMT) $\mathbf{y_{<t}} = y_1, y_2, ..., y_{t-1}$ by selecting a proper target word/phrase translation for an untranslated source span from a bilingual phrase table.

## NMT with SMT Recommendations

Different from attention-based NMT which predicts the next word based on vector representations, the proposed model makes the prediction also based on recommendations from an SMT model. The SMT model can be separately trained on a bilingual corpus using the conventional phrase-based SMT approach (Koehn, Och, and Marcu 2003). Given decoding information from NMT, the SMT model makes word recommendations[1] for the next word prediction with SMT features. To integrate the SMT recommendations into the proposed model, we employ a classifier to score the recommendations and a gate to combine SMT recommendations with NMT word prediction probabilities.

As shown in Figure 1, the word generation process of the proposed model has three steps:

1. Inheriting from standard attention-based NMT, the NMT classifier (i.e., "classifier$_{\text{NMT}}$") estimates word prediction probabilities on the regular vocabulary $V^{nmt}$:

$$p_{nmt}(y_t|\mathbf{y_{<t}}, \mathbf{x}) = softmax(f(s_t, y_{t-1}, c_t)) \tag{6}$$

2. The SMT classifier (i.e., "classifier$_{\text{SMT}}$") computes probabilities of SMT recommendations, which are generated by the auxiliary SMT model.

[1] We have not adopted the recently proposed phraseNet which incorporates target phrases into the NMT decoder (Tang et al. 2016). We leave the investigation to future work.

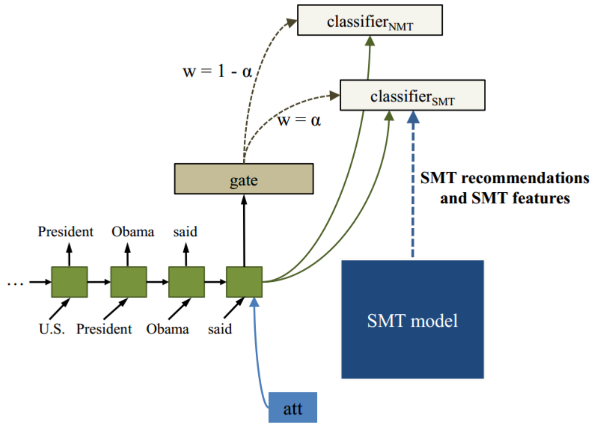

Figure 1: Architecture of decoder in NMT with SMT recommendations. The blue box with characters "att" is the context vector outputted by the encoder of NMT with attention mechanism.

3. The proposed model updates word prediction probabilities by using a gating mechanism (i.e., "gate" in brown box).

In the following subsections, we first elaborate on how the SMT model produces recommendations based on decoding information from the NMT model, which is the main feature of the proposed method. Then we illustrate how to integrate the SMT recommendations into NMT with an SMT classifier and a gating mechanism. Finally, we propose a novel approach to handle UNK in NMT with SMT recommendations.

## Generating SMT Recommendations

Given previously generated words $\mathbf{y}_{<\mathbf{t}} = y_1, y_2, ..., y_{t-1}$ from NMT, SMT generates the next word recommendations, and computes their scores by

$$SMT_{score}(y_t|\mathbf{y}_{<\mathbf{t}}, \mathbf{x}) = \sum_{m=1}^{M} \lambda_m h_m(y_t, x_t) \quad (7)$$

where $y_t$ is an SMT recommendation and $x_t$ is its corresponding source span. $h_m(y_t, x_t)$ is a feature function and $\lambda_m$ is its weight. The SMT model can generate proper word recommendations through expanding generated words (partial translation). However there are two problems with SMT recommendations due to the different word generation mechanisms between SMT and NMT:

**Coverage information** SMT lacks of coverage information about the source sentence. In conventional SMT, the decoder maintains a coverage vector to indicate whether a source word/phrase is translated or not, and generates a next target word/phrase from the untranslated part of the source sentence. This coverage information is however missing since our SMT decoder has only the partial translation produced by NMT, which often leads to inappropriate recommendations due to the lack of coverage information.

To overcome this problem, we introduce an SMT coverage vector $\mathbf{cv} = [cv_1, cv_2, ..., cv_{T_x}]$ for the SMT model. $cv_i = 1$ means that the source word $x_i$ has been translated and SMT will not recommend target words from translated source words. We initialize the SMT coverage vector with zeros. At each decoding step, if the output word which is generated by the proposed model is in the SMT recommendations, SMT coverage vector will be updated. Specifically, as SMT contains the positions of the recommendation words at the source sentence, we can update the SMT coverage vector by setting the corresponding vector element to 1.

**Alignment information** The SMT reordering model lacks of alignment information between source and target sentences. Due to the word order difference between languages, the reordering model plays an import role in existing SMT approaches. Typically, the SMT model computes reordering model features based on word alignments. For example, distance-based reordering model computes the reordering cost as follows (Koehn, Och, and Marcu 2003):

$$d(y_t) = -\left| sp_{y_t} - sp_{y_{t-1}} - 1 \right| \quad (8)$$

where $sp_y$ denotes the position of source word which is aligned to target word $y$. As mentioned in the section of Background, NMT utilizes attention mechanism to align words between the target and source sides. As shown in Equation (4), instead of aligning to a specific source word, each target word is aligned to all the source words with corresponding alignment weights. Therefore the SMT reordering model can not directly work with the attention mechanism of NMT.

To address this issue, reordering cost is computed between a target word and the corresponding source words by using the alignment weights. Similar to the estimation of bidirectional word translation probabilities in (He et al. 2016), the reordering cost is computed as follows:

$$d(y_t) = -\sum_{j=1}^{T_x} \alpha_{t-1,j} \left| sp_{y_t} - j - 1 \right| \quad (9)$$

where $\alpha_{t-1,j}$ is alignment weight produced by NMT. The lexical reordering model can work in a similar way.

The last question is: what kind of words does SMT recommend? Considering the second limitation of NMT described in the section of Introduction, we keep content words and filter out stop words from SMT recommendations. Now at each decoding step, with the SMT coverage vector and NMT attention mechanism, SMT can use SMT features[2] to produce word recommendations which convey the meaning of the untranslated source word/phrase.

## Integrating SMT Recommendations into NMT

**SMT Classifier** After SMT generates recommendations, a classifier scores the recommendations and generates probability estimates on them. To ensure the quality of SMT

[2] In training, as words out of NMT vocabulary $V^{nmt}$ may also appear in previously generated words, we should train a new language model by replacing the words with UNK and use this language model to score a target sequence.

recommendations, we adopt two strategies to filter low-quality recommendations: 1) only top $N_{tm}$ translations for each source word are retained according to their translation scores, each of which is computed as weighted sum of translation probabilities[3]. 2) top $N_{rec}$ recommendations with highest SMT scores are selected, each of which is computed as weighted sum of SMT features.

At the decoding step $t$, the SMT classifier takes current hidden state $s_t$, previous word $y_{t-1}$, context vector $c_t$ and SMT recommendation $y_t$ as input to estimate word probability of recommendation on vocabulary $V_t^{smt}$ . We denote the word estimated probability as:

$$p_{smt}(y_t|\mathbf{y_{<t}}, \mathbf{x}) = softmax(score_{smt}(y_t|\mathbf{y_{<t}}, \mathbf{x})) \quad (10)$$

where $score_{smt}(y_t|\mathbf{y_{<t}}, \mathbf{x})$ is the scoring function for SMT recommendation defined as follows:

$$score_{smt}(y_t|\mathbf{y_{<t}}, \mathbf{x}) = g_{smt}(s_t, y_{t-1}, y_t, c_t) \qquad (11)$$

where $g_{smt}(\cdot)$ is an activation function which is either an identity or a non-linear function.

As we do not want to introduce extra embedding parameters, we let SMT recommendations share the same target word embedding matrix with the NMT model. In this case, SMT word recommendation which is out of regular vocabulary $V^{nmt}$ is replaced by UNK word in the SMT classifier. Since a UNK word does not retain the meaning of the original words, SMT will not record the source coverage information if a UNK word is generated.

**Gating Mechanism** At last, a gate is introduced to update word prediction probabilities for the proposed model. It is computed as follows:

$$\alpha_t = g_{gate}(s_t, y_{t-1}, c_t) \qquad (12)$$

where $g_{gate}(\cdot)$ is $sigmoid$ function.

With the help of the gate, we update word prediction probabilities on regular vocabulary $V^{nmt}$ by combining the two probabilities through linear interpolation between the NMT generations and SMT recommendations:

$$\begin{aligned} p(y_t|\mathbf{y_{<t}}, \mathbf{x}) = (1 - \alpha_t)p_{nmt}(y_t|\mathbf{y_{<t}}, \mathbf{x}) \\ +\alpha_t p_{smt}(y_t|\mathbf{y_{<t}}, \mathbf{x}) \end{aligned} \qquad (13)$$

Note that $p_{smt}(y_t|\mathbf{y_{<t}}, \mathbf{x}) = 0 \; for \; y_t \notin V_t^{smt}$.

In Equation (13), when the gate is close to 0, the proposed model will ignore the SMT recommendations and only focus on the NMT word prediction (a stop word may also be generated). This effectively allows our model to drop SMT recommendations that are irrelevant.

### Handling UNK with SMT recommendations

To address the UNK word problem, we further directly utilize SMT recommendations to replace UNK words in testing phase. For each UNK word, we choose the recommendation with the highest SMT score as the final replacement.[4]

Our method is similar to the method described in (He et al. 2016), both of which can make use of rich contextual information on both the source and target sides to handle UNK. The difference is that our method takes into account the reordering information and the SMT coverage vector to generate more reliable recommendations.

## Model Training

We train the proposed model by minimizing the negative log-likelihood on a set of training data $\{(\mathbf{x}^n, \mathbf{y}^n)\}_{n=1}^N$:

$$C(\theta) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_y}\log p(y_t^n|\mathbf{y^n_{<t}}, \mathbf{x^n}) \qquad (14)$$

The cost function of our proposed model is the same as a conventional attention-based NMT model, except that we introduce extra parameters for the SMT classifier and the gate. We optimize our model by minimizing the cost function. In practice, we use a simple pre-training strategy to train our model. We first train a regular attention-based NMT model. Then we use this model to initialize the parameters of encoder and decoder of the proposed model and use random initialization to initialize the parameters of the SMT classifier and the gate. At last, we train all parameters of our model to minimize the cost function.

We adopt this pre-training strategy for two reasons: 1) The training of the proposed model may take a longer time with computation of SMT recommendation features and ranking of SMT recommendations, both of which are time-consuming. We treat the pre-trained model as an anchor point, the peak of the prior distribution in model space, which allows us to shorten the training time. 2) The quality of automatically learned attention weights is crucial for computing SMT reordering cost (see Equation (9)). Low quality of attention weights obtained without pre-training may produce unreliable SMT recommendations and consequently negatively affect the training of the proposed model.

## Experiments

In this section, we evaluate our approach on Chinese-English machine translation.

### Setup

The training set is a parallel corpus from LDC, containing 1.25M sentence pairs with 27.9M Chinese words and 34.5M English words[5]. We use NIST 2006 dataset as development set, and NIST 2002, 2003, 2004, 2005 and 2008 datasets as test sets. Case-insensitive BLEU score[6] is adopted as evaluation metric. We compare our proposed model with two state-of-the-art systems:

* Moses: a state-of-the-art phrase-based SMT system with its default setting.

[3] Bidirectional translation probabilities and bidirectional lexical translation probabilities.

[4] There is no UNK word in the previously generated words as our model generates the target sentence from left to right. Here we can use the original language model to score the target sequence. Language model which is trained on large monolingual corpora help make accurate replacements.

[5] The corpus includes LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

[6] ftp://jaguar.ncsl.nist.gov/mt/resources/mteval-v11b.pl

| SYSTEM | NIST06 | NIST02 | NIST03 | NIST04 | NIST05 | NIST08 | Avg |
|---|---|---|---|---|---|---|---|
| Moses | 32.43 | 34.08 | 34.16 | 34.74 | 31.99 | 23.69 | 31.73 |
| Groundhog | 30.23 | 34.79 | 32.21 | 34.02 | 30.56 | 23.37 | 30.99 |
| RNNSearch$^*$ | 33.53 | 35.59 | 32.55 | 36.52 | 33.05 | 24.79 | 32.50 |
| +SMT rec | 34.48$^{\dagger}$ | 36.40$^{\dagger}$ | 34.27$^{\ddagger}$ | 37.54$^{\ddagger}$ | 34.25$^{\ddagger}$ | 26.04$^{\ddagger}$ | 33.70 |
| +SMT rec, UNK Replace | **34.90**$^{\ddagger}$ | **38.02**$^{\ddagger}$ | **36.04**$^{\ddagger}$ | **38.81**$^{\ddagger}$ | **35.64**$^{\ddagger}$ | **26.19**$^{\ddagger}$ | **34.94** |

Table 1: Experiment results on the NIST Chinese-English translation task. For RNNSearch, we adopt the open source system Groundhog as our baseline. The strong baseline, denoted RNNSearch$^*$, is our in-house NMT system. [+SMT rec, UNK Replace] is the proposed model and [+SMT rec] is the proposed model without replacing UNK words. The BLEU scores are case-insensitive. "†": significantly better than RNNSearch$^*$ ($p < 0.05$); "‡": significantly better than RNNSearch$^*$ ($p < 0.01$).

* RNNSearch: an attention-based NMT system with its default setting. We use the open source system GroundHog[7] as the NMT baseline.

For Moses, we use the full training data (parallel corpus) to train the model and the target portion of the parallel corpus to train a 4-gram language model using the KenLM[8] (Heafield 2011). We use the default system setting for both training and testing.

For RNNSearch, we use the parallel corpus to train the attention-based NMT model. We limit the source and target vocabularies to the most frequent 30K words in Chinese and English, covering approximately 97.7% and 99.3% of the data in the two languages respectively. All other words are mapped to a special token UNK. We train the model with the sentences of length up to 50 words in training data and keep the test data at the original length. The word embedding dimension of both sides is 620 and the size of hidden layer is 1000. All the other settings are the same as in (Bahdanau, Cho, and Bengio 2015). We also use our implementation of RNNSearch which adopts feedback attention and dropout as NMT baseline system. Dropout is applied only on the output layer and the dropout rate is set to 0.5.

For the proposed model, we put it on the top of encoder same as in (Bahdanau, Cho, and Bengio 2015). As for pretraining, we train the regular attention-based NMT model using our implementation of RNNSearch and use its parameters to initialize the NMT part of our proposed model.

We use a minibatch stochastic gradient descent (SGD) algorithm together with Adadelta (Zeiler 2012) to train the NMT models and the decay rates $\rho$ and $\epsilon$ are set as 0.95 and $10^{-6}$. Each SGD update direction is computed using a minibatch of 80 sentences.

For SMT recommendation, we re-implement an SMT decoder which only adopts 6 features. The adopted features are bidirectional translation features, bidirectional lexical translation features, language model feature and distance-based reordering feature. Lexical reordering features are abandoned in order to speed up the SMT recommendation process. As for word-level recommendation, word penalty feature and phrase penalty feature are also abandoned as they cannot provide useful information for calculating recommendation scores. The feature weights are obtained from Moses[9]. We add English punctuations into the stop word list and remove numerals from the list. Finally the stop list contains 572 symbols. To ensure the quality of SMT recommendations, we set $N_{tm}$ to 5 and $N_{rec}$ to 25. We adopt a forward neural network with two hidden layers for the SMT classifier (Equation (11)) and gating function (Equation (12)). The numbers of units in the hidden layers are 200 and 50 respectively.

## Results

Table 1 reports the experiment results measured in terms of BLEU score. We find that our implementation RNNSearch$^*$ using feedback attention and dropout outperforms Groundhog and Moses by 1.51 BLEU point and 0.77 BLEU point. RNNSearch$^*_{\texttt{+SMT rec}}$ using SMT recommendations but keeps target UNK words achieves a gain of up to 1.20 BLEU point over RNNSearch$^*$. Surprisingly, RNNSearch$^*_{\texttt{+SMT rec, UNK Replace}}$ which further replaces UNK word with SMT recommendation during translation, achieves further improvement over RNNSearch$^*_{\texttt{+SMT rec}}$ by 1.24 BLEU point. It outperforms RNNSearch$^*$ and Moses by 2.44 BLEU point and 3.21 BLEU point respectively.

We also conduct additional experiments to validate the effectiveness of the key components of our proposed model, namely SMT recommendations and gating mechanism. More specifically, we adopt the following three tests: (1) we set a fixed gate value 0 for RNNSearch$^*_{\texttt{+SMT rec}}$, to block SMT recommendations ($+\alpha = 0$ in Table 2); (2) we set a fixed gate value 0.20 for RNNSearch$^*_{\texttt{+SMT rec}}$, to change the gating mechanism to a fixed mixture ($+\alpha = 0.20$ in Table 2); (3) we randomly generate some high frequency target words and submit the pseudo recommendations to RNNSearch$^*_{\texttt{+SMT rec}}$ during translation, to deliberately confuse RNNSearch$^*_{\texttt{+SMT rec}}$ (+pseudo recs in Table 2). From Table 2, we can observe that:

1. Without SMT recommendations, the proposed model suffers from degraded performance (-1.55 BLUE point). This indicates that SMT recommendations are essential for RNNSearch$^*_{\texttt{+SMT rec}}$.
2. Without a flexible gating mechanism, the proposed model performances on all test sets deteriorate considerably (-

[7] https://github.com/lisa-groundhog/GroundHog

[8] https://kheafield.com/code/kenlm/

[9] Compared with other weights, the weight of distance-based reordering feature is too small, therefore we manually increase it by 10 times.

| SYSTEM | NIST06 | NIST02 | NIST03 | NIST04 | NIST05 | NIST08 | Avg |
|---|---|---|---|---|---|---|---|
| RNNSearch$^*$ | 33.53 | 35.59 | 32.55 | 36.52 | 33.05 | 24.79 | 32.50 |
| +SMT rec | 34.48 | 36.40 | 34.27 | 37.54 | 34.25 | 26.04 | 33.70 |
| $+\alpha = 0$ | 32.64 | 34.93 | 32.39 | 35.94 | 32.14 | 25.33 | 32.15 |
| $+\alpha = 0.20$ | 31.24 | 34.40 | 31.84 | 36.06 | 31.61 | 24.16 | 31.61 |
| +pseudo recs | 32.62 | 35.00 | 32.57 | 35.90 | 32.15 | 25.20 | 32.16 |

Table 2: Effect of SMT recommendations and gating mechanism. BLEU scores in the table are case insensitive. [+SMT rec] is the proposed model without handling UNK words. $+\alpha = 0$ is [+SMT rec] with fixed gate value 0, which ignores SMT recommendations during translation. $+\alpha = 0.20$ is [+SMT rec] with a fixed gate value 0.20, which ignores flexible gating mechanism. +pseudo recs is [+SMT rec] with pseudo SMT recommendations.

2.09 BLEU point). This shows that gating mechanism plays an important role in RNNSearch$^*_{\texttt{+SMT rec}}$.

3. The experiment results in Table 2 empirically show that the proposed model makes wrong translations and has a significant decrease in performance (- 1.54 BLEU point), which demonstrates that the quality of SMT recommendations is also very important for RNNSearch$^*_{\texttt{+SMT rec}}$.

## Related Work

In this section we briefly review previous studies that are related to our work. Here we divide previous work into three categories:

**Combination of SMT and NMT:** Stahlberg et al. (2016) extended the beam search decoding by expanding the search space of NMT with translation hypotheses produced by a syntactic SMT model. He et al. (2016) enhanced NMT system with effective SMT features. They integrated three SMT features, namely translation model, word reward feature and language model, with the NMT model under the log-linear framework in the beam search. Arthur et al. (2016) proposed to incorporate discrete translation lexicons into NMT model. They calculated lexical predictive probability and integrated the probability with the NMT model probability to predict the next word. Wuebker et al. (2016) applied phrase-based and neural models to complete partial translations in interactive machine translation and find the models can improve next-word suggestion. The significant difference between our work and these studies is that NMT is treated as a black-box in the previous work, while in our work the NMT and SMT models are tightly integrated with the execution of the former being advised by the latter in the training and testing phase.

**Coverage problem in NMT:** Tu et al. (2016) proposed a coverage mechanism for NMT to alleviate the “over-translation” and “under-translation” problems. They introduced a coverage vector for the attention model, to make the NMT model consider more about untranslated source words in the target word generation. Cohn et al. (2016) enhanced the attention model with structural biases from word based alignment models, including positional bias, Markov conditioning, fertility and agreement over translation directions. Feng et al. (2016) proposed to add implicit distortion and fertility models to attention model. These studies tackle the coverage problem by enhancing the encoder of NMT. As we incorporate SMT model into the decoder part of NMT, our work is complementary to the above studies.

**UNK word problem in NMT:** Luong et al. (2015) proposed several approaches to track the source word of a UNK word in the target sentence. They first attached aligned information for each target UNK word in training data and trained a model on the data, and then they used the model to generate translation with UNK alignment information. Jean et al. (2015) proposed an efficient approximation based on importance sampling on the softmax function which allows to train NTM with very large vocabulary. On the other hand, instead of modeling word unit, some work focused on smaller unit modeling (Chitnis and DeNero 2015; Sennrich, Haddow, and Birch 2016), especially on character modeling (Ling et al. 2015; Costa-Jussà and Fonollosa 2016; Luong and Manning 2016; Chung, Cho, and Bengio 2016). Our work is also motivated by Copynet (Gu et al. 2016), which incorporates copying mechanism in sequence-to-sequence learning.

## Conclusion

In this paper, we have presented a novel approach that incorporates SMT model into NMT with attention mechanism. Our proposed model remains the power of end-to-end NMT while alleviating its limitations by utilizing recommendation from SMT for better generation in NMT. Different from prior work which usually used a separately trained NMT model as an additional feature, our proposed model containing NMT and SMT is trained in an end-to-end manner. Experiment results on Chinese-English translation have demonstrated the efficacy of the proposed model.

## References

[Arthur, Neubig, and Nakamura 2016] Arthur, P.; Neubig, G.; and Nakamura, S. 2016. Incorporating discrete translation lexicons into neural machine translation. In *EMNLP 2016*.

[Bahdanau, Cho, and Bengio 2015] Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR 2015*.

[Brown et al. 1993] Brown, P. F.; Pietra, V. J. D.; Pietra, S. A. D.; and Mercer, R. L. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*.

[Chiang 2005] Chiang, D. 2005. A hierarchical phrase-based model for statistical machine translation. In *ACL 2015*.

[Chitnis and DeNero 2015] Chitnis, R., and DeNero, J. 2015. Variable-length word encodings for neural translation models. In *EMNLP 2015*.

[Cho et al. 2014a] Cho, K.; van Merriënboer, B.; Bahdanau, D.; and Bengio, Y. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint*.

[Cho et al. 2014b] Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014b. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP 2014*.

[Chung, Cho, and Bengio 2016] Chung, J.; Cho, K.; and Bengio, Y. 2016. A character-level decoder without explicit segmentation for neural machine translation. In *ACL 2016*.

[Cohn et al. 2016] Cohn, T.; Hoang, C. D. V.; Vymolova, E.; Yao, K.; Dyer, C.; and Haffari, G. 2016. Incorporating structural alignment biases into an attentional neural translation model. In *NAACL 2016*.

[Costa-Jussà and Fonollosa 2016] Costa-Jussà, M. R., and Fonollosa, J. A. 2016. Character-based neural machine translation. In *ACL 2016*.

[Feng et al. 2016] Feng, S.; Liu, S.; Li, M.; and Zhou, M. 2016. Implicit distortion and fertility models for attention-based encoder-decoder nmt model. *arXiv preprint*.

[Gu et al. 2016] Gu, J.; Lu, Z.; Li, H.; and Li, V. O. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL 2016*.

[He et al. 2016] He, W.; He, Z.; Wu, H.; and Wang, H. 2016. Improved neural machine translation with smt features. In *AAAI 2016*.

[Heafield 2011] Heafield, K. 2011. Kenlm: Faster and smaller language model queries. In *Workshop on SMT*.

[Hochreiter and Schmidhuber 1997] Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

[Jean et al. 2015] Jean, S.; Cho, K.; Memisevic, R.; and Bengio, Y. 2015. On using very large target vocabulary for neural machine translation. In *ACL 2015*.

[Kalchbrenner and Blunsom 2013] Kalchbrenner, N., and Blunsom, P. 2013. Recurrent continuous translation models. In *EMNLP 2013*.

[Koehn, Och, and Marcu 2003] Koehn, P.; Och, F. J.; and Marcu, D. 2003. Statistical phrase-based translation. In *NAACL 2003*.

[Ling et al. 2015] Ling, W.; Trancoso, I.; Dyer, C.; and Black, A. W. 2015. Character-based neural machine translation. *arXiv preprint*.

[Luong and Manning 2016] Luong, M.-T., and Manning, C. D. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *ACL 2016*.

[Luong et al. 2015] Luong, M.-T.; Sutskever, I.; Le, Q. V.; Vinyals, O.; and Zaremba, W. 2015. Addressing the rare word problem in neural machine translation. 11–19.

[Meng et al. 2015] Meng, F.; Lu, Z.; Tu, Z.; Li, H.; and Liu, Q. 2015. A deep memory-based architecture for sequence-to-sequence learning. *arXiv preprint arXiv:1506.06442*.

[Och and Ney 2002] Och, F. J., and Ney, H. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *40th ACL*, 295–302.

[Och 2003] Och, F. J. 2003. Minimum error rate training in statistical machine translation. In *41st ACL-Volume 1*, 160–167.

[Schuster and Paliwal 1997] Schuster, M., and Paliwal, K. K. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on* 45(11):2673–2681.

[Sennrich, Haddow, and Birch 2016] Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural machine translation of rare words with subword units. In *54th ACL*, 1715–1725.

[Stahlberg et al. 2016] Stahlberg, F.; Hasler, E.; Waite, A.; and Byrne, B. 2016. Syntactically guided neural machine translation. *arXiv preprint arXiv:1605.04569*.

[Sutskever, Vinyals, and Le 2014] Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, 3104–3112.

[Tang et al. 2016] Tang, Y.; Meng, F.; Lu, Z.; Li, H.; and Yu, P. L. 2016. Neural machine translation with external phrase memory. *arXiv preprint arXiv:1606.01792*.

[Tu et al. 2016] Tu, Z.; Lu, Z.; Liu, Y.; Liu, X.; and Li, H. 2016. Modeling coverage for neural machine translation. In *ACL 2016*.

[Wuebker et al. 2016] Wuebker, J.; Green, S.; DeNero, J.; Hasan, S.; and Luong, M.-T. 2016. Models and inference for prefix-constrained machine translation. In *54th ACL-Volume 1*, 66–75.

[Zeiler 2012] Zeiler, M. D. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.