On Using Monolingual Corpora in Neural Machine Translation

Caglar Gulcehre[⋆] Université de Montréal
Orhan Firat[⋆,†] Middle East Technical University
Kelvin Xu Université de Montréal
Kyunghyun Cho Université de Montréal
Loïc Barrault Université du Maine
Huei-Chi Lin Université du Maine
Fethi Bougares Université du Maine
Holger Schwenk Université du Maine
Yoshua Bengio Université de Montréal, CIFAR Senior Fellow

Abstract

Recent work on end-to-end neural network-based architectures for machine translation has shown promising results for En-Fr and En-De translation. Arguably, one of the major factors behind this success has been the availability of high quality parallel corpora. In this work, we investigate how to leverage abundant monolingual corpora for neural machine translation. Compared to a phrase-based and hierarchical baseline, we obtain up to 1.96 BLEU improvement on the low-resource language pair Turkish-English, and 1.59 BLEU on the focused domain task of Chinese-English chat messages. While our method was initially targeted toward such tasks with less parallel data, we show that it also extends to high resource languages such as Cs-En and De-En where we obtain an improvement of 0.39 and 0.47 BLEU scores over the neural machine translation baselines, respectively.

1 Introduction

Neural machine translation (NMT) is a novel approach to machine translation that has shown promising results (Kalchbrenner and Blunsom, 2013; Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014). Until recently, the application of neural networks to machine translation was restricted to extending standard machine translation tools for rescoring translation hypotheses or re-ranking n-best lists (see, e.g., (Schwenk, 2012; Schwenk, 2007a). In contrast, it has been shown that, it is possible to build a competitive translation system for English-French and English-German using an end-to-end neural network architecture (Sutskever et al., 2014; Jean et al., 2014) (also see Sec. 2).

Arguably, a large part of the recent success of these methods has been due to the availability of large amounts of high quality, sentence aligned corpora. In the case of low resource language pairs or in a task with heavy domain restrictions, there can be a lack of such sentence aligned corpora. In contrast, monolingual corpora is almost always universally available. Despite being "unlabeled", monolingual corpora still exhibit rich linguistic structure that may be useful for translation tasks. This presents an opportunity to leverage such corpora to give hints to an NMT system.

In this work, we present a way to effectively integrate a language model (LM) trained only on monolingual data (target language) into an NMT system. We provide experimental results that incorporating monolingual corpora can improve a translation system on a low-resource language pair (Turkish-English) and a domain restricted translation problem (Chinese-English SMS chat). In addition, we show that these methods improve the performance on the relatively high-resource German-English (De-En) and Czech-English (Cs-En) translation tasks.

In the following section (Sec. 2), we review recent work in neural machine translation. We present our basic model architecture in Sec. 3 and describe our shallow and deep fusion approaches in Sec. 4. Next, we describe our datasets in Sec. 5. Finally, we describe our main experimental results in Sec. 6.

[⋆] Equal contribution. Order has been determined with a coin flip.

[†] Work done while author was at Université de Montréal

2 Background: Neural Machine Translation

Statistical machine translation (SMT) systems maximize the conditional probability $p(\mathbf{y} \mid \mathbf{x})$ of a correct target translation $\mathbf{y}$ given a source sentence $\mathbf{x}$. This is done by maximizing separately a language model $p(\mathbf{y})$ and the (inverse) translation model $p(\mathbf{x} \mid \mathbf{y})$ component by using Bayes' rule:

$$p(\mathbf{y} \mid \mathbf{x}) \propto p(\mathbf{x} \mid \mathbf{y})p(\mathbf{y}).$$

This decomposition into a language model and translation model is meant to make full use of available corpora: monolingual corpora for fitting the language model and parallel corpora for the translation model. In reality, however, SMT systems tend to model $\log p(\mathbf{y} \mid \mathbf{x})$ directly by linearly combining multiple features by using a so-called log-linear model:

$$\log p(\mathbf{y} \mid \mathbf{x}) = \sum_j f_j(\mathbf{x}, \mathbf{y}) + C, \qquad (1)$$

where f_j is the j-th feature based on both or either of the source and target sentences, and C is a normalization constant which is often ignored. These features include, for instance, pair-wise statistics between two sentences/phrases. The log-linear model is fitted to data, in most cases, by maximizing an automatic evaluation metric other than an actual conditional probability, such as BLEU.

Neural machine translation, on the other hand, aims at directly optimizing $\log p(\mathbf{y} \mid \mathbf{x})$ including the feature extraction as well as the normalization constant by a single neural network. This is typically done under the encoder-decoder framework (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Sutskever et al., 2014) consisting of neural networks. The first network encodes the source sentence $\mathbf{x}$ into a continuous-space representation from which the decoder produces the target translation sentence. By using RNN architectures equipped to learn long term dependencies such as Gated Recurrent Units (GRU) or Long Short-Term Memory (LSTM), the whole system can be trained end-to-end (Cho et al., 2014; Sutskever et al., 2014).

Once the model learns the conditional distribution or translation model, given a source sentence we can find a translation that approximately maximizes the conditional probability using, for instance, a beam search algorithm.

3 Model Description

We use the model recently proposed by (Bahdanau et al., 2014) that learns to jointly (soft-)align and translate as the baseline neural machine translation system in this paper. Here we describe in detail this model to which we refer as "NMT".

The encoder of the NMT is a bidirectional RNN which consists of forward and backward RNNs (Schuster and Paliwal, 1997). The forward RNN reads the input sequence/sentence $\mathbf{x} = (x_1, \ldots, x_T)$ in a forward direction, resulting in a sequence of hidden states $(\overrightarrow{\mathbf{h}}_1, \ldots, \overrightarrow{\mathbf{h}}_T)$. The backward RNN reads $\mathbf{x}$ in an opposite direction and outputs $(\overleftarrow{\mathbf{h}}_1, \ldots, \overleftarrow{\mathbf{h}}_T)$. We concatenate a pair of hidden states at each time step to build a sequence of *annotation* vectors $(\mathbf{h}_1, \ldots, \mathbf{h}_T)$, where

$$\mathbf{h}_j^\top = \left[\overleftarrow{\mathbf{h}}_j^\top; \overrightarrow{\mathbf{h}}_j^\top\right].$$

Each annotation vector $\mathbf{h}_j$ encodes information about the j-th word with respect to all the other surrounding words in the sentence.

In our decoder, which we construct with a single layer RNN, at each timestep t a soft-alignment mechanism first decides on which annotation vectors are most relevant. The relevance weight α_{tj} of the j-th annotation vector for the t-th target word is computed by a feedforward neural network f that takes as input $\mathbf{h}_j$, the previous decoder's hidden state $\mathbf{s}_{t-1}$ and the previous output $\mathbf{y}_{t-1}$:

$$e_{tj} = \mathrm{f}(\mathbf{s}_{t-1}, \mathbf{h}_j, \mathbf{y}_{t-1}).$$

The outputs e_{tj} are normalized over the sequence of the annotation vectors so that the they sum to 1:

$$\alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^{T} \exp(e_{tk})}, \qquad (2)$$

and we call α_{tj} a relevance score, or an alignment weight, of the j-th annotation vector.

The relevance scores are used to get the *context vector* $\mathbf{c}_t$ of the t-th word in the translation:

$$\mathbf{c}_t = \sum_{j=1}^{T} \alpha_{tj}\mathbf{h}_j \ ,$$

Then, the decoder's hidden state $\mathbf{s}_t^{\mathrm{TM}}$ at time t is computed based on the previous hidden state $\mathbf{s}_{t-1}^{\mathrm{TM}}$, the context vector $\mathbf{c}_t$ and the previously translated word $\mathbf{y}_{t-1}$:

$$\mathbf{s}_t^{\mathrm{TM}} = \mathrm{f}_r(\mathbf{s}_{t-1}^{\mathrm{TM}}, \mathbf{y}_{t-1}, \mathbf{c}_t), \qquad (3)$$

where f_r is the gated recurrent unit (Cho et al., 2014).

We use a deep output layer (Pascanu et al., 2014) to compute the conditional distribution over words:

$$p(\mathbf{y}_t|\mathbf{y}_{<t},\mathbf{x}) \propto \exp(\mathbf{y}_t^\top(\mathbf{W}_o \mathrm{f}_o(\mathbf{s}_t^{\text{TM}},\mathbf{y}_{t-1},\mathbf{c}_t)+\mathbf{b}_o)), \quad (4)$$

where $\mathbf{y}_t$ is a one-hot encoded vector indicating one of the words in the target vocabulary. $\mathbf{W}_o$ is a learned weight matrix and $\mathbf{b}_o$ is a bias. f_o is a single-layer feedforward neural network with a two-way maxout non-linearity (Goodfellow et al., 2013).

The whole model, including both the encoder and decoder, is jointly trained to maximize the (conditional) log-likelihood of the bilingual training corpus:

$$\max_{\boldsymbol{\theta}} \frac{1}{N}\sum_{n=1}^{N} \log p_\theta(\mathbf{y}^{(n)}|\mathbf{x}^{(n)}),$$

where the training corpus is a set of $(\mathbf{x}^{(n)},\mathbf{y}^{(n)})$'s, and $\boldsymbol{\theta}$ denotes a set of all the tunable parameters.

4 Integrating Language Model into the Decoder

In this paper, we propose two alternatives to integrating a language model into a neural machine translation system which we refer as *shallow fusion* (Sec. 4.1) and *deep fusion* (Sec. 4.2). Without loss of generality, we use a language model based on recurrent neural networks (RNNLM, (Mikolov et al., 2011)) which is equivalent to the decoder described in the previous section except that it is not biased by a context vector (i.e., $\mathbf{c}_t = 0$ in Eqs. (3)–(4)).

In the sections that follow, we assume that both an NMT model (on parallel corpora) as well as a recurrent neural network language model (RNNLM, on larger monolingual corpora) have been pre-trained separately before being integrated. We denoted the hidden state at time t of the RNNLM with $\mathbf{s}_t^{\text{LM}}$.

4.1 Shallow Fusion

Shallow fusion is analogous to how language models are used in the decoder of a usual SMT system (Koehn, 2010). At each time step, the translation model proposes a set of candidate words. The candidates are then scored according to the weighted sum of the scores given by the translation model and the language model.

More specifically, at each time step t, the translation model (in this case, the NMT) computes the score of every possible next word for each hypothesis all of hypotheses $\left\{\mathbf{y}_{\leq t-1}^{(i)}\right\}$. Each score is the summation of the score of the hypothesis and the score given by the NMT to the next word. All these new hypotheses (a hypothesis from the previous timestep with a next word appended at the end) are then sorted according to their respective scores, and the top K ones are selected as candidates $\left\{\hat{\mathbf{y}}_{\leq t}^{(i)}\right\}_{i=1,\dots,K}$.

We then rescore these hypotheses with the weighted sum of the scores by the NMT and RNNLM, where we only need to recompute the score of the "new word" at the end of each candidate hypothesis. The score of the new word is computed by

$$\log p(\mathbf{y}_t = k) = \log p_{\text{TM}}(\mathbf{y}_t = k) + \beta \log p_{\text{LM}}(\mathbf{y}_t = k), \quad (5)$$

where β is a hyper-parameter that needs to be tuned to maximize the translation performance on a development set.

See Fig. 1 (a) for illustration.

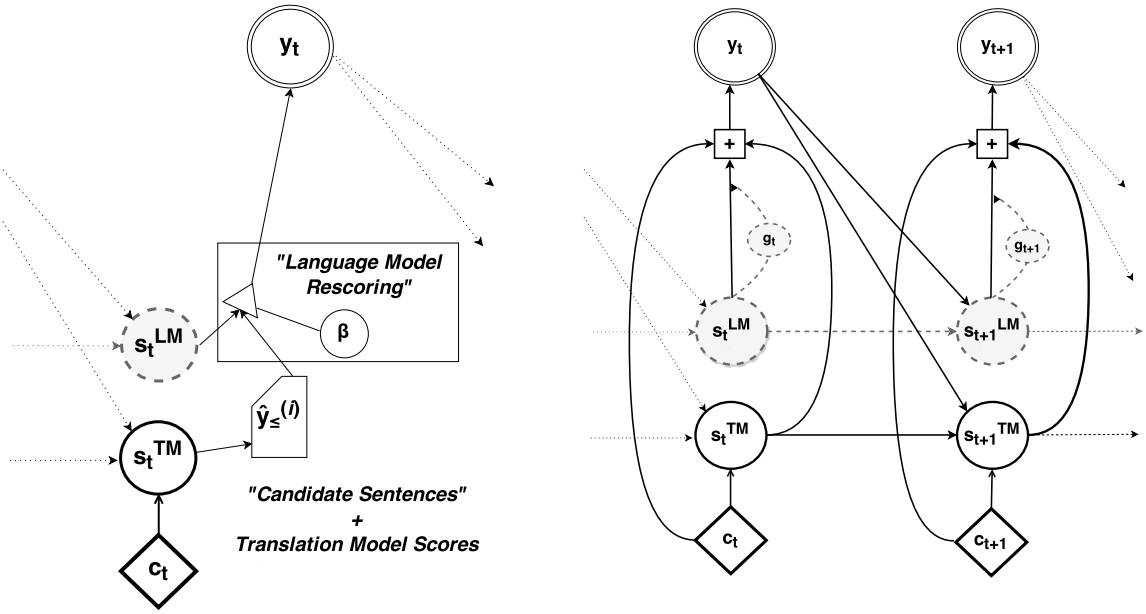
4.2 Deep Fusion

In deep fusion, we integrate the RNNLM and the decoder of the NMT by concatenating their hidden states next to each other (see Fig. 1 (b)). The model is then finetuned to use the hidden states from both of these models when computing the output probability of the next word (see Eq. (4)). Unlike the vanilla NMT (without any language model component), the hidden layer of the deep output takes as input the hidden state of the RNNLM in addition to that of the NMT, the previous word and the context such that

$$p(\mathbf{y}_t|\mathbf{y}_{<t},\mathbf{x}) \propto \exp(\mathbf{y}_t^\top(\mathbf{W}_o \mathrm{f}_o(\mathbf{s}_t^{\text{LM}},\mathbf{s}_t^{\text{TM}},\mathbf{y}_{t-1},\mathbf{c}_t)+\mathbf{b}_o)), \quad (6)$$

where again we use the superscripts $^{\text{LM}}$ and $^{\text{TM}}$ to denote the hidden states of the RNNLM and NMT respectively.

During the finetuning of the model, we tune only the parameters that were used to parameterize the output (6). This is to ensure that the structure



(a) Shallow Fusion (Sec. 4.1)

(b) Deep Fusion (Sec. 4.2)

Figure 1: Graphical illustrations of the proposed fusion methods.

learned by the LM from monolingual corpora is not overwritten. It is possible to use monolingual corpora as well while finetuning all the parameters, but in this paper, we alter only the output parameters in the stage of finetuning.

4.2.1 Balancing the LM and TM

In order for the decoder to flexibly balance the input from the LM and TM, we augment the decoder with a "controller" mechanism. The need to flexibly balance the signals arises depending on the work being translated. For instance, in the case of Zh-En, there are no Chinese words that correspond to articles in English, in which case the LM may be more informative. On the other hand, if a noun is to be translated, it may be better to ignore any signal from the LM, as it may prevent the decoder from choosing the correct translation. Intuitively, this mechanism helps the model dynamically weight the different models depending on the word being translated.

The controller mechanism is implemented as a function taking the hidden state of the LM as input and computing

$$g_t = \sigma\left(\mathbf{v}_g^\top \mathbf{s}_t^{\text{LM}} + b_g\right), \tag{7}$$

where σ is a logistic sigmoid function. $\mathbf{v}_g$ and b_g are learned parameters.

The output of the controller is then multiplied with the hidden state of the LM. This lets the decoder use the signal from the TM fully, while the controller controls the magnitude of the LM signal.

In our experiments, we empirically found that it was better to initialize the bias $\mathbf{b}_g$ to a small, negative number. This allows the decoder to decide the importance of the LM only when it is deemed necessary.

5 Datasets

We evaluate the proposed approaches on four diverse tasks: Chinese to English (Zh-En), Turkish to English (Tr-En), German to English (De-En) and Czech to English (Cs-En). We describe each of these datasets in more detail below.

5.1 Parallel Corpora

5.1.1 Zh-En: OpenMT'15

We use the parallel corpora made available as a part of the NIST OpenMT'15 Challenge. Sentence-aligned pairs from three domains are combined to form a training set: (1) SMS/CHAT and (2) conversational telephone speech (CTS) from DARPA BOLT Project, and (3) newsgroups/weblogs from DARPA GALE Project. In total, the training set consists of 430K sentence pairs (see Table 1 for the detailed statistics). We train

In all our experiments, we set $\mathbf{b}_g = -1$ to ensure that $\mathbf{g}_t$ is initially 0.26 on average.

models with this training set and the development set (the concatenation of the provided development and tune sets from the challenge), and evaluate them on the test set. The domain of the development and test sets is restricted to CTS.

Preprocessing Importantly, we did "not segment" the Chinese sentences and considered each character as a symbol, unlike other approaches which use a separate segmentation tool to segment the Chinese characters into words (Devlin et al., 2014). Any consecutive non-Chinese characters such as Latin alphabets were, however, considered as an individual word. Lastly, we removed any HTML/XML tags from the corpus, chose only the intended meaning word if both intended and literal translations are available, and ignored any indicator of, e.g., typos. The only preprocessing we did on the English side of the corpus was a simple tokenization using the tokenizer from Moses. .

5.1.2 Tr-En: IWSLT'14

We used the WIT parallel corpus (Cettolo et al., 2012) and SETimes parallel corpus made available as a part of IWSLT'14 (machine translation track). The corpus consists of the sentence-aligned subtitles of TED and TEDx talks, and we concatenated dev2010 and tst2010 to form a development set, and tst2011, tst2012, tst2013 and tst2014 to form a test set. See Table 1 for the detailed statistics of the parallel corpora.

Preprocessing As done with the case of Zh-En, initially we removed all special symbols from the corpora and tokenized the Turkish side with the tokenizer provided by Moses. To overcome the exploding vocabulary due to the rich inflections and derivations in Turkish, we segmented each Turkish sentence into a sequence of sub-word units using Zemberek followed by morphological disambiguation on the morphological analysis (Sak et al., 2007). We removed any non-surface morphemes corresponding to, for instance, part-of-speech tags.

5.2 Cs-En and De-En: WMT'15

For the training of our models, we used all the available training data provided for Cs-En and De-En in the WMT'15 competition. We used newstest2013 as a development set and newstest2014 for a test set. The detailed statistics of the parallel corpora is provided in Table 1.

Preprocessing We tokenized the datasets with Moses tokenizer first. Sentences longer than eighty words and those that have large mismatch between lengths of the source and target sentences were removed from the training set. Then, we filtered the training data by removing sentence pairs in which one sentence (or both) was written in the wrong language by using a language detection toolkit (Shuyo, 2010), unless the sentence had 5 words or less. For De-En, we also split the compounds in the German side by using Moses. Finally we shuffled the training corpora seven times and concatenated its outputs.

5.3 Monolingual Corpora

The English Gigaword corpus by the Linguistic Data Consortium (LDC), which mainly consists of newswire documents, was allowed in both OpenMT'15 and IWSLT-15 challenges for language modelling. We used the tokenized Gigaword corpus without any other preprocessing step to train three different RNNLM's to fuse into NMT for Zh-En, Tr-En and the WMT'15 translation tasks (De-En and Cs-En.)

6 Settings

6.1 Training Procedure

6.1.1 Neural Machine Translation

The input and output of the network were sequences of one-hot vectors whose dimensionality correspond to the sizes of the source and target vocabularies, respectively. We constructed the vocabularies with the most common words in the parallel corpora. The sizes of the vocabularies for Chinese, Turkish and English were 10K, 30K and 40K, respectively, for the Tr-En and Zh-En tasks. Each word was projected into the continuous space of 620-dimensional Euclidean space first to reduce the dimensionality, on both the encoder and the decoder. We chose the size of the recurrent units for Zh-En and Tr-En to be $1,200$ and $1,000$ respectively.

In Cs-En and De-En experiments, we were able to use larger vocabularies. We trained our NMT model for Cs-En and De-En with large vocabularies using the importance sampling based technique introduced in (Jean et al., 2014) and with this tech-

https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl

https://github.com/ahmetaa/zemberek-nlp

	Chinese	English
# of Sentences	436K	
# of Unique Words	21K	150K
# of Total Words	8.4M	5.9M
Avg. Length	19.3	13.5

(a) Zh-En

	Turkish	English
# of Sentences	160K	
# of Unique Words	96K*	95K
# of Total Words	11.4M*	8.1M
Avg. Length	31.6	22.6

(b) Tr-En

	Czech	English
# of Sentences	12.1M	
# of Unique Words	1.5M	911K
# of Total Words	151M	172M
Avg. Length	12.5	14.2

(c) Cs-En

	German	English
# of Sentences	4.1M	
# of Unique Words	1.16M†	742K
# of Total Words	11.4M†	8.1M
Avg. Length	24.2	25.1

(d) De-En

Table 1: Statistics of the Parallel Corpora. $\star$: After segmentation, †: After compound splitting.

nique we were able to use large vocabulary of size $200k$.

Each model was optimized using Adadelta (Zeiler, 2012) with minibatches of 80 sentence pairs. At each update, we normalized the gradient such that if the L_2 norm of the gradient exceeds 5, gradient is renormalized back to 5 (Pascanu et al., 2013). For the non-recurrent layers (see Eq. (4)), we used dropout (Hinton et al., 2012) and additive Gaussian noise (mean 0 and std. dev. 0.001) on each parameter to prevent overfitting (Graves, 2011). Training was early-stopped to maximize the performance on the development set measured by BLEU. We initialized all recurrent weight matrices as random orthonormal matrices.

6.1.2 Language Model

We trained three RNNLM's with $2,400$ long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997) units on English Gigaword Corpus using respectively the vocabularies constructed separately from the English sides of Zh-En and Tr-En corpora. The third language model was trained using $2,000$ LSTM units on the English Gigaword Corpus again but with a vocabulary constructed from the intersection the English sides of Cs-En and De-En. The parameters of the former two language models were optimized using RMSProp (Tieleman and Hinton, 2012), and Adam optimizer (Kingma and Ba, 2014) was used for the latter one. Any sentence with more than ten percent of its words out of vocabulary was discarded from the training set. We did early-stopping using the perplexity of development set.

We compute the BLEU score using the multi-blue.perl script from Moses on tokenized sentence pairs.

6.2 Shallow and Deep Fusion

6.2.1 Shallow Fusion

The hyperparameter β in Eq. (5) was selected to maximize the translation performance on the development set, from the range 0.001 and 0.1. In preliminary experiments, we found it important to renormalize the softmax of the LM without the end-of-sequence and out-of-vocabulary symbols ($\beta = 0$ in Eq. (5)). This may be due to the difference in the domains of TM and LM.

6.2.2 Deep Fusion

We finetuned the parameters of the deep output layer (Eq. (6)) as well as the controller (see Eq. (7) using the Adam optimizer for Zh-En, and RMSProp with momentum for Tr-En. During the finetuning, the dropout probability and the standard deviation of the weight noise were set to 0.56 and 0.005, respectively. Based on our preliminary experiments, we reduced the level of regularization after the first $10K$ updates. In Cs-En and De-En tasks with large vocabularies, the model parameters were finetuned using Adadelta while scaling down the magnitude of the update steps by 0.01.

6.2.3 Handling Rare Words

On the De-En and Cs-En translation tasks, we replaced the unknown words generated by the NMT with the words the NMT assigned to which the highest score in the source sentence (Eq. (2)). We copied the selected source word in the place of the corresponding unknown token in the target sentence. This method is similar to the technique proposed by (Luong et al., 2014) for addressing

rare words. But instead of relying on an external alignment tool, we used the attention mechanism of the NMT model to extract alignments. This method consistently improved the results by approximately 1.0 BLEU score.

7 Results and Analysis

7.1 Zh-En: OpenMT'15

In addition to NMT-based systems, we also trained a phrase-based as well as hierarchical phrase-based SMT systems (Koehn et al., 2003; Chiang, 2005) with/without re-scoring by an external neural language model (CSLM) (Schwenk, 2007b). We present the results in Table 2.

We observed that integrating an additional LM by deep fusion (see Sec. 4.2) helped the models achieving better performance in general, except in the case of the CTS task. We noticed that the NMT-based models, regardless of whether the LM was integrated or not, outperformed the more traditional phrase-based SMT systems.

	SMS/CHAT		CTS	
	Dev	Test	Dev	Test
PB	15.5	14.73	21.94	21.68
+ CSLM	16.02	15.25	23.05	22.79
HPB	15.33	14.71	21.45	21.43
+ CSLM	15.93	15.8	22.61	22.17
NMT	17.32	17.36	23.4	**23.59**
Shallow	16.59	16.42	22.7	22.83
Deep	**17.58**	**17.64**	**23.78**	23.5

Table 2: Results on the task of Zh-En. PB and HPB stand for the phrase-based and hierarchical phrase-based SMT systems, respectively.

7.2 Tr-En: IWSLT'14

In Table 3, we present our results on Tr-En. Compared to Zh-En, we saw a greater performance improvement up to +1.19 BLEU points from the basic NMT to the NMT integrated with the LM under the proposed method of deep fusion. Furthermore, by incorporating the LM using deep fusion, the NMT systems were able to outperform the best previously reported result (Yılmaz et al., 2013) by up to $+1.96$ BLEU points on all of the separate test sets.

7.3 Cs-En and De-En: WMT-15

We provide the results of Cs-En and De-En on Table 4. Shallow fusion achieved 0.09 and 0.29 BLEU score improvements respectively on De-En and Cs-En over the baseline NMT model. With deep fusion the improvements of 0.39 and 0.47 BLEU score were observed again over the NMT baseline.

	De-En		Cs-En	
	Dev	Test	Dev	Test
NMT Baseline	25.51	23.61	21.47	21.89
Shallow Fusion	25.53	23.69	21.95	22.18
Deep Fusion	**25.88**	**24.00**	**22.49**	**22.36**

Table 4: Results for De-En and Cs-En translation tasks on WMT'15 dataset.

7.4 Analysis: Effect of Language Model

The performance improvements we report in this paper reflect a heavy dependency on the degree of similarity between the domain of monolingual corpora and the target domain of translation.

In the case of Zh-En, intuitively, we can tell that the style of writing in both SMS/CHAT as well as the conversational speech will be different from that of news articles (which constitutes the majority of the English Gigaword corpus). Empirically, this is supported by the high perplexity on the development set with our LM (see the column Zh-En of Table 5). This explains the marginal improvement we observed in Sec. 7.1.

On the other hand, in the case of Tr-En, the similarity between the domains of the monolingual corpus and parallel corpora is higher (see the column Tr-En of Table 5). This led to a significantly larger improvement in translation performance by integrating the external language model than the case of Zh-En. Similarly, we observed the improvement by both shallow and deep fusion in the case of De-En and Cs-En, where the perplexity on the development set was much lower.

Unlike shallow fusion, deep fusion allows a model to selectively incorporate the information from the additional LM by the controller mechanism from Sec. 4.2.1. Although this controller mechanism works on per-word basis, it can be expected that if the additional LM models the target domain better, the controller mechanism will be more frequently active on average, i.e., $\mathbb{E}[g_t] \gg 0$. From Table 5, we can see that, on average, the controller mechanism is most active with De-En and Cs-En, where the additional LM was able to model the target sentences best. This effectively means

	Development Set		Test Set			
	dev2010	**tst2010**	**tst2011**	**tst2012**	**tst2013**	**Test 2014**
Previous Best (Single)	15.33	17.14	18.77	18.62	18.88	-
Previous Best (Combination)	-	17.34	18.83	18.93	18.70	-
NMT	14.50	18.01	18.40	18.77	19.86	18.64
NMT+LM (Shallow)	14.44	17.99	18.48	18.80	19.87	18.66
NMT+LM (Deep)	**15.69**	**19.34**	**20.17**	**20.23**	**21.34**	**20.56**

Table 3: Results on Tr-En. We show for each set separately to make it easier to compare to previously reported scores.

that deep fusion allows the model to be more robust to the domain mismatch between the TM and LM, thus suggests why deep fusion was more successful than shallow fusion in the experiments.

	Zh-En	Tr-En	De-En	Cs-En
Perplexity	223.68	163.73	78.20	78.20
Average g	0.23	0.12	0.28	0.31
Std Dev g	0.0009	0.02	0.003	0.008

Table 5: Perplexity of RNNLM's on development sets and the statistics of the controller gating mechanism g.

8 Conclusion and Future Work

In this paper, we propose and compare two methods for incorporating monolingual corpora into an existing NMT system. We empirically evaluate these approaches (shallow fusion and deep fusion) on low-resource En-Tr (TED/TEDx Subtitles), focused domain for En-Zh (SMS/Chat and conversational speech) and two high-resource language pairs: Cs-En and De-En. We show that with our approach on the Tr-En and Zh-En language pairs, the NMT models trained with deep fusion were able to achieve better results than the existing phrase-based statistical machine translation systems (up to a $+1.96$ BLEU points on En-Tr). We also observed up to a 0.47 BLEU score improvement for high resource language pairs such as De-En and Cs-En on the datasets provided in WMT'15 competition over our NMT baseline. This provides an evidence that our method can also improve the translation performance regardless of the amount of available parallel corpora.

Our analysis also revealed that the performance improvement from incorporating an external LM was highly dependent on the domain similarity between the monolingual corpus and the target task. In the case where the domain of the bilingual and monolingual corpora were similar (De-En, Cs-En), we observed improvement with both deep and shallow fusion methods. In the case where they were dissimilar (Zh-En), the improvement using shallow fusion were much smaller. This trend might also explain why deep fusion, which implements an adaptive mechanism for modulating information from the integrated LM, works better than shallow fusion. This analysis also suggests that future work on domain adaption of the language model may further improve translations.

References

[Bahdanau et al.2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[Cettolo et al.2012] Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. Wit3: Web inventory of transcribed and translated talks. *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268.

[Chiang2005] David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics.

[Cho et al.2014] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, October. to appear.

[Devlin et al.2014] Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. 2014. Fast and robust neural network joint models for statistical machine translation. In *Association for Computational Linguistics*.

[Goodfellow et al.2013] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1319–1327.

[Graves2011] Alex Graves. 2011. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pages 2348–2356.

[Hinton et al.2012] 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*.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

[Jean et al.2014] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2014. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*.

[Kalchbrenner and Blunsom2013] Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1700–1709. Association for Computational Linguistics.

[Kingma and Ba2014] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv:*`1412.6980 [cs.LG]`, December.

[Koehn et al.2003] Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

[Koehn2010] Philipp Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA.

[Luong et al.2014] Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

[Mikolov et al.2011] Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and Jan Cernocky. 2011. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201.

[Pascanu et al.2013] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*.

[Pascanu et al.2014] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio. 2014. How to construct deep recurrent neural networks. In *Proceedings of the Second International Conference on Learning Representations (ICLR 2014)*, April.

[Sak et al.2007] Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2007. Morphological disambiguation of turkish text with perceptron algorithm. In *Computational Linguistics and Intelligent Text Processing*, pages 107–118. Springer.

[Schuster and Paliwal1997] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *Signal Processing, IEEE Transactions on*, 45(11):2673–2681.

[Schwenk2007a] Holger Schwenk. 2007a. Continuous space language models. *Comput. Speech Lang.*, 21(3):492–518, July.

[Schwenk2007b] Holger Schwenk. 2007b. Continuous space language models. *Computer Speech & Language*, 21(3):492–518.

[Schwenk2012] Holger Schwenk. 2012. Continuous space translation models for phrase-based statistical machine translation. In Martin Kay and Christian Boitet, editors, *Proceedings of the 24th International Conference on Computational Linguistics (COLIN)*, pages 1071–1080. Indian Institute of Technology Bombay.

[Shuyo2010] Nakatani Shuyo. 2010. Language detection library for java.

[Sutskever et al.2014] Ilya Sutskever, Oriol Vinyals, and Quoc Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NIPS 2014)*, December.

[Tieleman and Hinton2012] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4.

[Yılmaz et al.2013] Ertugrul Yılmaz, Ilknur Durgar El-Kahlout, Burak Aydın, Zisan Sıla Ozil, and Coskun Mermer. 2013. Tubitak turkish-english submissions for iwslt 2013. *Proceedings of the 10th International Workshop on Spoken Language Translation (IWSLT)*, pages 152–159.

[Zeiler2012] Matthew D Zeiler. 2012. ADADELTA: An adaptive learning rate method. *arXiv:*`1212.5701 [cs.LG]`.