Domain Differential Adaptation for Neural Machine Translation

Zi-Yi Dou, Xinyi Wang, Junjie Hu, Graham Neubig
Language Technologies Institute, Carnegie Mellon University
{zdou, xinyiw1, junjieh, gneubig}@cs.cmu.edu

Abstract

Neural networks are known to be data hungry and domain sensitive, but it is nearly impossible to obtain large quantities of labeled data for every domain we are interested in. This necessitates the use of domain adaptation strategies. One common strategy encourages generalization by aligning the global distribution statistics between source and target domains, but one drawback is that the statistics of different domains or tasks are inherently divergent, and smoothing over these differences can lead to sub-optimal performance. In this paper, we propose the framework of *Domain Differential Adaptation (DDA)*, where instead of smoothing over these differences we embrace them, directly modeling the difference between domains using models in a related task. We then use these learned domain differentials to adapt models for the target task accordingly. Experimental results on domain adaptation for neural machine translation demonstrate the effectiveness of this strategy, achieving consistent improvements over other alternative adaptation strategies in multiple experimental settings.[1]

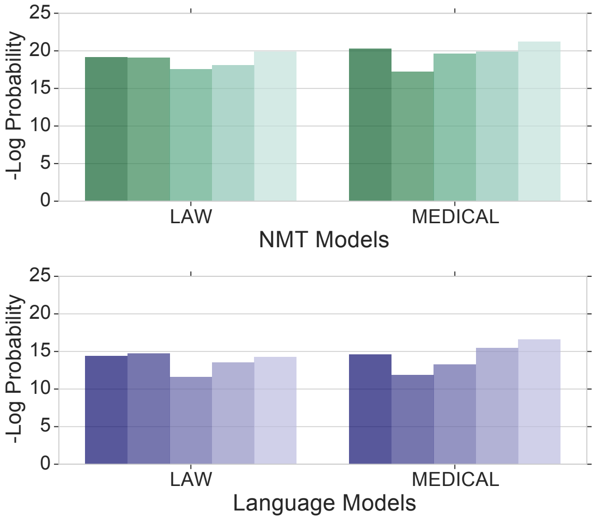

Figure 1: Mean log probabilities of NMT models and LMs trained on law and medical domains for the words (”needle”, ”hepatic”, ”complete”, ”justify”, ”suspend”). LM and NMT probabilities are correlated for each domain. (More examples in Section 5.1.)

1 Introduction

Most recent success of deep neural networks rely on the availability of high quality and labeled training data (He et al., 2017; Vaswani et al., 2017; Povey et al., 2018; Devlin et al., 2019). In particular, neural machine translation (NMT) models tend to perform poorly if they are not trained with enough parallel data from the test domain (Koehn and Knowles, 2017). However, it is not realistic to collect large amounts of parallel data in all possible domains due to the high cost of data collection. Moreover, certain domains by nature have far less data than others. For example, there is much more news produced and publicly available than more sensitive medical records. Therefore, it is essential to explore effective methods for utilizing out-of-domain data to train models that generalize well to in-domain data.

There is a rich literature in domain adaptation for neural networks (Luong and Manning, 2015; Tan et al., 2017; Chu et al., 2017; Ying et al., 2018). In particular, we focus on two lines of work that are conducive to *unsupervised* adaptation, where there is no training data available in the target domain. The first line of work focuses on aligning representations of data from different domains with the goal of improving data sharing across the two domains using techniques such as mean maximum discrepancy (Long et al., 2015) or adversarial training (Ganin et al., 2016; Sankaranarayanan and Balaji, 2017). However, these methods attempt to smooth over the differences in the domains by learning domain-invariant

[1] Code is available at https://github.com/zdou0830/DDA.

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 59–69
Hong Kong, China, November 4, 2019. ©2019 Association for Computational Linguistics
www.aclweb.org/anthology/D19-56%2d

features, and in the case when these differences are actually necessary for correctly predicting the output, this can lead to sub-optimal performance (Xie et al., 2017). Another line of research tries to directly integrate in-domain models of other tasks to adapt models in the target task. For example, Gulcehre et al. (2015) use pre-trained in-domain LMs with out-of-domain NMT models by directly using a weighted sum of probability distributions from both models, or fusing the hidden states of both models and fine-tuning. These LMs can potentially capture features of the in-domain data, but models of different tasks are inherently different and thus coming up with an optimal method for combining them is non-trivial.

The main intuition behind our method is that models with different data requirements, namely LMs and NMT models, exhibit similar behavior when trained on the same domain, but there is little correlation between models trained on data from different domains (as demonstrated empirically in Figure 1). Because of this, directly adapting an out-of-domain NMT model by integrating an in-domain LM (i.e. with methods in Gulcehre et al. (2015)) may be sub-optimal, as the in-domain and out-of-domain NMT may not be highly correlated. However, the *difference* between LMs from two different domains will likely be similar to the *difference* between the NMT models. Based on these observations, we propose a new unsupervised adaptation framework, Domain Differential Adaptation (DDA), that utilizes models of a related task to capture *domain differences*. Specifically, we use LMs trained with in-domain and out-of-domain data, which gives us hints about how to compensate for domain differences and adapt an NMT model trained on out-of-domain parallel data. Although we mainly examine NMT in this paper, the general idea can be applied to other tasks as well.

We evaluate DDA in two different unsupervised domain adaptation settings on four language pairs. DDA demonstrates consistent improvements of up to 4 BLEU points over an unadapted NMT baseline, and up to 2 BLEU over an NMT baseline adapted using existing methods. An analysis reveals that DDA significantly improves the NMT model's ability to generate words more frequently seen in in-domain data, indicating that DDA is a promising approach to domain adaptation of NMT and neural models in general.

2 Background

2.1 Neural Language Models

Given a sequence of tokens $\mathbf{y} = (y_1, y_2, \cdots, y_N)$, LMs compute a probability of the sequence $p(\mathbf{y})$ by decomposing it into the probability of each token y_t given the history $(y_1, y_2, \cdots, y_{t-1})$. Formally, the probability of the sequence $\mathbf{y}$ is calculated as:

$$p(\mathbf{y}) = \prod_{t=1}^{N} p(y_t | y_1, y_2, \cdots, y_{t-1}).$$

LMs are comonly modeled using some variety of recurrent neural networks (RNN; (Hochreiter and Schmidhuber, 1997; Cho et al., 2014)), where at each timestep t, the network first outputs a context-dependent representation s_t^{LM}, which is then used to compute the conditional distribution $p(y_t|y_{<t})$ using a softmax layer. During training, gradient descent is used to maximize the log-likelihood of the monolingual corpus Y:

$$\max_{\theta_{LM}} \sum_{\mathbf{y}^i \in Y} \log p(\mathbf{y}^i; \theta_{LM}).$$

2.2 Neural Machine Translation Models

Current neural machine translation models are generally implemented in the encoder-decoder framework (Sutskever et al., 2014; Cho et al., 2014), where the encoder generates a context vector for each source sentence $\mathbf{x}$ and the decoder then outputs the translation $\mathbf{y}$, one target word at a time.

Similarly to LMs, NMT models would also generate hidden representation s_t^{NMT} at each timestep t, and then compute the conditional distribution $p(y_t|y_{<t}, \mathbf{x})$ using a softmax layer. Both encoder and decoder are jointly trained to maximize the log-likelihood of the parallel training corpus (X, Y):

$$\max_{\theta_{NMT}} \sum_{(\mathbf{x}^i, \mathbf{y}^i) \in (X,Y)} \log p(\mathbf{y}^i | \mathbf{x}^i; \theta_{NMT}).$$

During decoding, NMT models generate words one by one. Specifically, at each time step t, the NMT model calculates the probability of next word $p_{\text{NMT}}(y_t|y_{<t}, \mathbf{x})$ for each of all previous hypotheses $\{y_{\leq t-1}^{(i)}\}$. After appending the new word to the previous hypothesis, new scores would be calculated and top K ones are selected as new hypotheses $\{y_{\leq t}^{(i)}\}$.

3 Domain Differential Adaptation

In this section, we propose two approaches under the overall umbrella of the DDA framework: Shallow Adaptation (DDA-Shallow) and Deep Adaptation (DDA-Deep). At a high level, both methods capture the domain difference by two LMs, trained on in-domain (LM-in) and out-of-domain (LM-out) monolingual data respectively. Without access to in-domain parallel data, we want to adapt the NMT model trained on out-of-domain parallel data (NMT-out) to approximate the NMT model trained on in-domain parallel data (NMT-in).

In the following sections, we assume that LM-in, LM-out as well as the NMT-out model have been pretrained separately before being integrated.

3.1 Shallow Adaptation

Given LM-in, LM-out, and NMT-out, our first method, *i.e. shallow adaptation* (DDA-Shallow), combines the three models only at *decoding* time. As we have stated above, at each time step t, NMT-out would generate the probability of the next word $p_{\text{NMT-out}}(y_t|y_{<t}, \mathbf{x})$ for each of all previous hypotheses $\{y^{(i)}_{<t}\}$. Similarly, language models LM-in and LM-out would output probabilities of the next word $p_{\text{LM-in}}(y_t|y_{<t})$ and $p_{\text{LM-out}}(y_t|y_{<t})$, respectively.

For DDA-Shallow, the candidates proposed by NMT-out are rescored considering scores given by LM-in and LM-out. Specifically, at each decoding timestep t, the probability of the next generated word y_t, is obtained by an interpolation of log-probabilities from LM-in, LM-out into NMT-out.

Formally, the log probability of y_t is

$$\begin{aligned}&\log\left(p(y_t)\right) \propto \log\left(p_{\text{NMT-out}}(y_t|y_{<t}, \mathbf{x})\right)\\&+ \beta\left[\log\left(p_{\text{LM-in}}(y_t|y_{<t})\right) - \log\left(p_{\text{LM-out}}(y_t|y_{<t})\right)\right],\end{aligned} \tag{1}$$

where β is a hyper-parameter.[2]

Intuitively, Equation 1 encourages the model to generate more words in the target domain as well as reduce the probability of generating words in the source domain.

3.2 Deep Adaptation

DDA-Shallow only functions during decoding time so there is almost no learning involved. In addition, hyper-parameter β is the same for all words, which limits the model's flexibility. Our second more expressive *deep adaptation* (DDA-Deep) method enables the model to learn how to make predictions based on the hidden states of LM-in, LM-out, and NMT-out. We freeze the parameters of the LMs and only fine-tune the fusion strategy and NMT parameters.

Formally, at each time step t, we have three hidden states $\mathrm{s}^{(t)}_{\text{LM-out}}$, $\mathrm{s}^{(t)}_{\text{LM-in}}$, and $\mathrm{s}^{(t)}_{\text{NMT-out}}$. We then concatenate them and use a gating strategy to combine the three hidden states:

$$s^{(t)}_{\text{concat}} = \left[\mathrm{s}^{(t)}_{\text{LM-out}}; \mathrm{s}^{(t)}_{\text{LM-in}}; \mathrm{s}^{(t)}_{\text{NMT-out}}\right], \tag{2.1}$$

$$g^{(t)}_{\text{LM-out}}, g^{(t)}_{\text{LM-in}}, g^{(t)}_{\text{NMT-out}} = F\left(s^{(t)}_{\text{concat}}\right), \tag{2.2}$$

$$\begin{aligned}s^{(t)}_{\text{DA}} &= g^{(t)}_{\text{LM-out}} \odot \mathrm{s}^{(t)}_{\text{LM-out}} + g^{(t)}_{\text{LM-in}} \odot \mathrm{s}^{(t)}_{\text{LM-in}}\\&+ g^{(t)}_{\text{NMT-out}} \odot \mathrm{s}^{(t)}_{\text{NMT-out}}.\end{aligned} \tag{2.3}$$

Here F is a linear transformation and $\odot$ stands for elementwise multiplication. As the three gating values g, we use matrices of the same dimension as the hidden states. This design gives the model more flexibility in combining the states.

One potential problem of training with only out-of-domain parallel corpora is that our method cannot learn a reasonable strategy to predict in-domain words, since it would never come across them during training or fine-tuning. In order to solve this problem, we copy some in-domain monolingual data from target side to source side as in Currey et al. (2017) to form pseudo in-domain parallel corpora. The pseudo in-domain data is concatenated with the original dataset when training the models.

4 Experiments

4.1 Setup

Datasets. We test both DDA-Shallow and DDA-Deep in two different data settings. In the first setting we use the dataset of Koehn and Knowles (2017), training on the law, medical and IT datasets of the German-English OPUS corpus[3] (Tiedemann, 2012). The standard splits contain 2K development and test sentences in each domain, and about 715K, 1M and 337K training sentences respectively. In the second setting, we train our models on the WMT-14 datasets[4] (Bojar

[2] Note that this quantity is simply proportional to the log probability, so it is important to re-normalize the probability after interpolation to ensure $\sum_k p(y_t = k) = 1$.

[3] http://opus.nlpl.eu

[4] https://www.statmt.org/wmt14/translation-task.html

Method	**De-En**						**Cs-En**	**De-En**
	LAW		MED		IT		WMT	
	MED	IT	LAW	IT	LAW	MED	TED	TED
w/o copying monolingual data								
Koehn and Knowles (2017)	12.1	3.5	3.9	2.0	1.9	6.5	-	-
Baseline	13.60	4.34	4.57	3.29	4.30	8.56	24.25	24.00
LM-Shallow	13.74	4.41	4.54	3.41	4.29	8.15	24.29	24.03
DDA-Shallow	16.39*	5.49*	5.89*	4.51*	5.87*	10.29*	26.52*	25.53*
w/ copying monolingual data								
Baseline	17.14	6.14	5.09	4.59	5.09	10.65	25.60	24.54
LM-Deep	17.74	6.01	5.16	4.87	5.01	11.88	25.98	25.12
DDA-Deep	18.02†	6.51*	5.85*	5.39*	5.52†	12.48*	26.44*	25.46†
w/ back-translated data								
Baseline	22.89	13.36	9.96	8.03	8.68	13.71	30.12	28.88
LM-Deep	23.58	14.04	10.02	9.05	8.48	15.08	30.34	28.72
DDA-Deep	23.74	13.96	10.74*	8.85	9.28*	16.40*	30.69	28.85

Table 1: Translation accuracy (BLEU; Papineni et al. (2002)) under different settings. The first three rows list the language pair, the source domain, and the target domain. "LAW", "MED" and "IT" represent law, medical and IT domains, respectively. We use compare-mt (Neubig et al., 2019) to perform significance tests (Koehn, 2004) and statistical significance compared with the best baseline is indicated with $*$ ($p < 0.005$) and † ($p < 0.05$).

et al., 2014) which contain data from several domains and test on the multilingual TED test sets of Duh (2018).[5] We consider two language pairs for this setting, namely Czech and German to English. The Czech-English and German-English datasets consist of about 1M and 4.5M sentences respectively and the development and test sets contain about 2K sentences. Byte-pair encoding (Sennrich et al., 2016b) is employed to process training data into subwords with a vocabulary size of 50K for both settings.

Models. NMT-out is a 500 dimensional 2-layer attentional LSTM encoder-decoder model (Bahdanau et al., 2015) implemented on top of OpenNMT (Klein et al., 2017). LM-in and LM-out are also 2-layer LSTMs with hidden sizes of 500. Here we mainly test on RNN-based models, but there is nothing architecture-specific in our methods preventing them from being easily adapted to other architectures such as the Transformer model (Vaswani et al., 2017).

Baselines. We compare our methods with three baseline models: 1) Shallow fusion and deep fusion (Gulcehre et al., 2015): they directly combine LM-in with NMT-out[6]. Shallow fusion combines LM-in and NMT-out during decoding while deep fusion learns to combine hidden states of LM-in and NMT-out. We denote shallow fusion and deep fusion as "LM-Shallow" and "LM-Deep". 2) The copied monolingual data model (Currey et al., 2017) which copies target in-domain monolingual data to the source side to form synthetic in-domain data. 3) Back-translation (Sennrich et al., 2016a) which enriches the training data by generating synthetic in-domain parallel data via a target-to-source NMT model which is trained on a out-of-domain corpus.

4.2 Main Results

4.2.1 Adapting Between Domains

The first 6 result columns of Table 1 show the experimental results on the OPUS dataset. We can see the LM-Shallow model can only marginally improve and sometimes even harms the performance of baseline models. On the other hand, our proposed DDA-Shallow model can outperform the baseline significantly by over 2 BLEU points. This reinforces the merit of our main idea of explicitly modeling the *difference* between domains, instead of simply modeling the target domain itself.

Under the setting where additional copied in-domain data is added into the training set, both LM-Deep and DDA-Deep perform better than the baseline model, with DDA-Deep consistently outperforming the LM-Deep method, indicating the presence of an out-of-domain LM is helpful. We also compare with back-translation, a

[5] http://www.cs.jhu.edu/ kevinduh/a/multitarget-tedtalks

[6] To ensure the fairness of comparison, we use our gating formula (Equation (2.2)) and fine-tune all parts of NMT-out for deep fusion.

strong baseline for domain adaptation. We obtain back-translated data via a target-to-source NMT model and concatenate the back-translated data with the original training data to train models. Again, DDA generally brings improvements over the baseline and LM-Deep with back-translated data.

4.2.2 Adapting from a General Domain to a Specific Domain

The last two result columns of Table 1 show the experimental results in the WMT-TED setting. As we can see, in this data setting our baseline performance is much stronger than the first setting. Similarly to the previous setting, DDA-Shallow can significantly improve the baseline model by over 2 BLEU points. However, the DDA-Deep model cannot outperform baselines by a large margin, probably because the baseline models are strong when adapting from a general domain to a specific domain and thus additional adaptation strategies can only lead to incremental improvements.

5 Analysis

5.1 Domain Differences between NMT Models and LMs

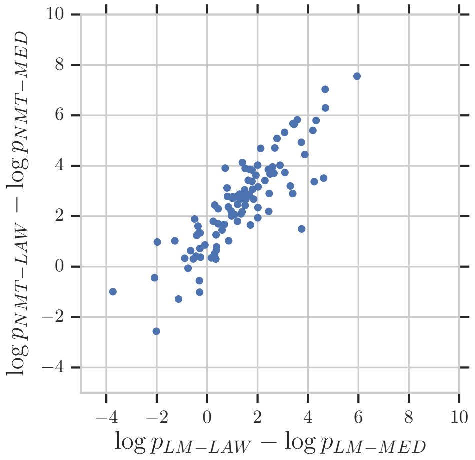

Figure 2: Correlation between $\log p_{\text{NMT-LAW}} - \log p_{\text{NMT-MED}}$ and $\log p_{\text{LM-LAW}} - \log p_{\text{LM-MED}}$. We decode each model on the medical set by feeding in the gold labels and calculate the mean of total log probabilities. We plot 100 words that appear frequently in both domains.

In this section, we visualize the correlation between $\log p_{\text{NMT-in}} - \log p_{\text{NMT-out}}$ and $\log p_{\text{LM-in}} - \log p_{\text{LM-out}}$. We treat the law domain as the target domain and the medical domain as the source domain. Specifically, we train four models NMT-LAW, NMT-MED, LM-LAW, LM-MED with law and medical data and decode each model on the medical set by feeding in the gold labels and calculate the mean of total log probabilities, then plot the correlation of 100 words that appear most frequently in both domains. Figure 2 shows that the difference between NMT models and LMs are roughly correlated, which supports the main motivation of the DDA framework.

5.2 Fusing Different Parts of the Models

In this section, we try to fuse different parts of LMs and NMT models. Prior works have tried different strategies such as fusing the hidden states of LMs with NMT models (Gulcehre et al., 2015) or combining multiple layers of a deep network (Peters et al., 2018). Therefore, it would be interesting to find out which combination of hidden vectors in our DDA-Deep method would be more helpful. Specifically, we try to fuse word embeddings, hidden states and output probabilities.

Components	LAW-MED	MED-LAW
Word-Embed	17.43	5.26
Hidden States	18.02	5.85
Word-Embed & Hidden States	18.00	5.79

Table 2: Performance of DDA-Deep when fusing different parts of models on the law and medical datasets.

We conduct experiments on the law and medical datasets in OPUS, and experimental results are shown in Table 2. We find that generally fusing hidden states is better than fusing word embeddings, and fusing hidden states together with word embeddings does not show any improvements over simply fusing hidden states alone. These results indicate that combining the higher-level information captured by the encoder states is more advantageous for domain adaptation. Also, we found that directly using DDA-Deep to fuse output probabilities was unstable even after trying several normalization techniques, possibly because of the sensitivity of output probabilities.

5.3 Analysis of the Adaptation Effect

In this section, we quantitatively and qualitatively analyze the effect of our proposed DDA framework on adapting the NMT model to in-domain

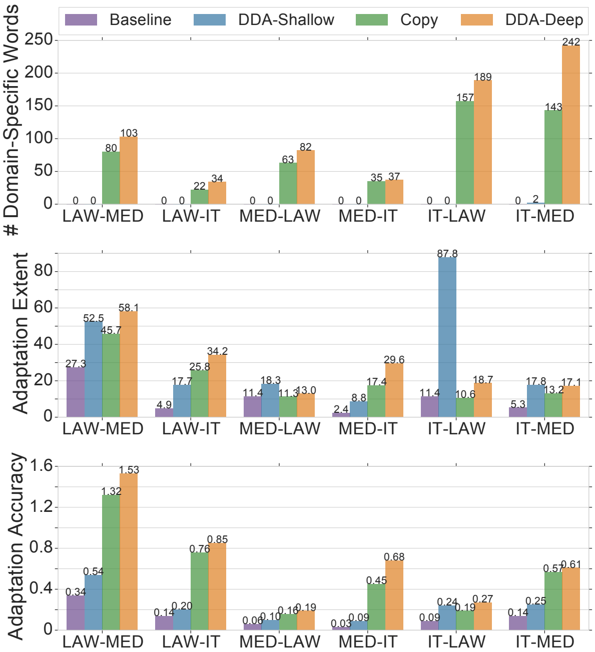

Figure 3: Number of generated domain-specific subwords, scores of adaptation extent and adaptation accuracy for each method. *Top*: count of words only exist in in-domain data produced by different models; *Middle*: adaptation extent of different models; *Bottom*: adaptation accuracy of different models.

data. We conduct analysis on the level of the subwords that were used in the MT system, and study whether our methods can generate in-domain subwords that have never appeared or appeared less frequently in the out-of-domain dataset as well as whether our methods can generate these in-domain subwords accurately.

First, we focus on domain-specific subwords, *i.e.* subwords that appear *exclusively* in the in-domain data. The counts of these subwords are shown in Figure 3. In general, both the baseline and DDA-Shallow struggle at generating subwords that never appear in the out-of-domain parallel data. On the other hand, copying monolingual data performs better in this facet, because it exposes the model to subwords that appear only in the in-domain data. DDA-Deep generates the largest number of in-domain subwords among the four models, indicating the effectiveness of our method.

Source	warum wurde Ab- ili- fy zugelassen ?
Reference	why has Ab- ili- fy been approved ?
Baseline	reasons was received why a reminder was accepted ?
DDA-Shallow	why has been approved?
Copy	why ,
DDA-Deep	why was Ab- ili- fy authorised ?

Table 3: Translation examples under the law to medical adaptation setting.

Second, we propose two subword-level evaluation metrics that study whether the models can generate in-domain subwords and if the generated in-domain subwords are correct. We first define metric "Adaptation Extent" (AE) as follows:

$$\mathrm{AE} = \frac{1}{|V|} \sum_{w \in V} \frac{\mathrm{freq\_in}(w)}{\mathrm{freq\_out}(w)} \mathrm{count}(w), \qquad (3)$$

where V is the whole vocabulary, $\mathrm{freq\_in}(w)$ and $\mathrm{freq\_out}(w)$ represent the frequency of subword w in both in-domain and out-of-domain corpora, and $\mathrm{count}(w)$ measures how many times subword w appears in the translation result.

We define "Adaptation Accuracy" (AA) in a similar way:

$$\mathrm{AA} = \frac{1}{|V|} \sum_{w \in V} \frac{\mathrm{freq\_in}(w)}{\mathrm{freq\_out}(w)} F1(w), \qquad (4)$$

where $F1$ denotes the F1-score of subword w. In order to avoid dividing by zero, we use add-one smoothing when calculating $\mathrm{freq\_out}(w)$. While AE measures the *quantity* of in-domain subwords the models can generate, AA tells us the *quality* of these subwords, namely whether the in-domain subwords form meaningful translations.

We plot the AE and AA scores of our methods as well as the baselines in Figure 3. The AE scores demonstrate that both DDA-Shallow and DDA-Deep adapt the model to a larger extent compared to other baselines even though DDA-Shallow fails to generate domain-specific subwords. In addition, the AA scores reveal that DDA-Deep outperforms other methods in terms of adaptation accuracy while DDA-Shallow is relatively weak in this respect. However, it should be noted that there is still large gap between deep adaptation method and the upper bound where the gold reference is used as a "translation"; the upper bound is about 10 for each setting.

We also we sample some translation results and show them in Table 3 to qualitatively demonstrate the differences between the methods. We could

Strategy	LAW-MED	MED-LAW
LM-in + LM-out	18.02	6.51
two LMs-in	17.60	6.06
two LMs-out	17.42	6.03
two LMs-general	17.64	6.22

Table 4: Performance of ensembling different LMs on the law and medical datasets. LMs-general are trained with both in-domain and out-of-domain datasets.

see that by modifying the output probabilities, the DDA-Shallow strategy has the ability to adjust tokens translated by the baseline model to some extent, but it is not capable of generating the domain-specific subwords "Ab- i li- fy". However, the DDA-Deep strategy can encourage the model to generate domain-specific tokens and make the translation more correct.

All of the above quantitative and qualitative results indicate that our strategies indeed help the model adapt from the source to target domains.

5.4 Necessity of Integrating both LMs

In this section, we further examine the necessity of integrating both in-domain and out-of-domain LMs. Although previous experimental results partially support the statement, we perform more detailed analysis to ensure the gain in BLEU points is because of the joint contribution of LM-in and LM-out.

Ensembling LMs. Ensembling multiple models is a common and broadly effective technique for machine learning, and one possible explanation for our success is that we are simply adding more models into the mix. To this end, we compare DDA-Deep with three models: the first one integrates NMT-out with two LMs-in trained with different random seeds and the second one integrates NMT-out with two LMs-out; we also integrate two general-domain LMs which are trained on both the in-domain and out-of-domain data and compare the performance. The experimental results are shown in Table 4.

We can see that DDA-Deep achieves the best performance compared with the three other models, demonstrating the gain in BLEU is not simply because of using more models.

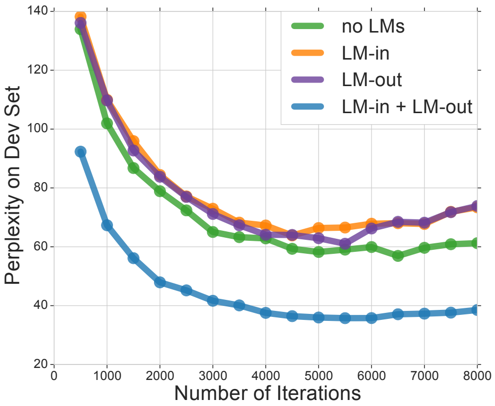

Figure 4: Perplexity on the development set for each method under the continued training setting. "no LMS", "LM-in", "LM-out" and "LM-in + LM-out" denote the baseline model, LM-Deep with LM-in, LM-Deep with LM-out and DDA-Deep respectively.

Continued Training. In this section, we attempt to gain more insights about the contribution of LM-in and LM-out by investigating how DDA-Deep behaves under a continued training setting, where a small number of in-domain parallel sentences are available. We first train the NMT-out model until convergence on the out-of-domain corpus, and then fine-tune it with DDA-Deep on the in-domain corpus. Here we use the medical and IT datasets as our out-of-domain and in-domain corpora respectively, mainly because the baseline model performs poorly under this setting. We randomly select $10,000$ parallel sentences in the in-domain dataset for continued training.

We freeze LM-in and LM-out as before and fine-tune the NMT-out model. The results are shown in Figure 4. We find that the perplexity of deep adaptation method on the development set drops more dramatically compared to baseline models. Figure 4 shows that integrating only LM-in or LM-out with the NMT model does not help, and sometimes even hurts the performance. This finding indicates that there indeed exists some correlation between LMs trained on different domains. Using both LM-in and LM-out together is essential for the NMT model to utilize the domain difference to adapt more effectively.

However, if we look at the BLEU points on the development set, DDA-deep with continued training performs much worse than the baseline model (13.36 vs. 15.61), as shown in Table 5 ($\beta = 0$). This sheds light on some limitations of our proposed method, which we will discuss in the next section.

Coverage penalty β	0.00	0.05	0.10	0.15	0.20	0.25	0.30
Baseline (no LMs)	15.61	16.28	17.26	**17.59**	17.21	16.39	15.96
LM-Deep (LM-out)	13.56	14.61	15.52	15.92	**15.98**	15.76	15.24
LM-Deep (LM-in)	12.00	13.36	14.56	15.10	15.62	**15.98**	15.57
DDA-Deep (LM-in + LM-out)	13.36	15.18	17.52	18.46	**18.62**	18.03	17.17

Table 5: BLEU points of models after continued training on the IT development dataset with different values of coverage penalty β.

5.5 Limitations of Current DDA Methods

Although our two proposed methods under the DDA framework achieve impressive results on unsupervised domain adaptation for NMT, the translation results still fall behind the gold reference by a large margin and the DDA-Deep performs much worse than the baseline model under a continued training setting as demonstrated in previous sections. In this section, we specify some limitations with our proposed methods and list a few future directions.

The objectives of LMs and NMT models are inherently different: LMs care more about the fluency whereas NMT models also need to consider translation adequacy, that is, the translations should faithfully reflect the source sentence (Tu et al., 2016). Therefore, directly integrating LMs with NMT models might have a negative impact on adequacy.

To verify this hypothesis, under the continued training setting we adopt a decoding-time coverage penalty (Wu et al., 2016), which is a simple yet effective strategy to reduce the number of dropped tokens. As shown in Table 5, the coverage penalty can improve the deep adaptation method by more than 5 BLEU points while the baseline model can only be improved by 2 BLEU points. The best DDA-Deep method outperforms the baseline by 1.03 BLEU points.

These results suggest some promising future directions for designing models under the DDA framework. Although current DDA methods can extract domain differences from two LMs, they cannot fully reduce the negative effect of LM objective on the NMT model. Therefore, it may be useful to add domain related priors that encourage the in-domain annd out-of-domain LMs to be more distinct, so that they can capture more domain-specific information. Another possible option is to add extra objectives to LM pretraining so that it can be fused with the NMT model more seamlessly.

6 Related Work

Finally, we overview related works in the general field of unsupervised domain adaptation, and then list some specific domain adaptation strategies for neural machine translation.

6.1 Unsupervised Domain Adaptation

Prior unsupervised domain adaptation methods for neural models mainly address the problem by aligning source domain and target domain by minimizing certain distribution statistics. For instance, Long et al. (2015) propose deep adaptation networks that minimize a multiple kernel maximum mean discrepancy (MK-MMD) between source and target domains. Sankaranarayanan and Balaji (2017) on the other hand utilize adversarial training to match different domains. Researchers have also tried to use language models for unsupervised domain adaptation. For example, Siddhant et al. (2019) propose to apply Embeddings from Language Models (ELMo) (Peters et al., 2018) and its variants in unsupervised transfer learning.

6.2 Domain Adaptation for NMT

Domain adaptation is an active research topic in NMT (Chu and Wang, 2018). Many previous works focus on the setting where a small amount of in-domain data is available. For instance, continued training (Luong and Manning, 2015; Freitag and Al-Onaizan, 2016) is one of the most popular methods, whose basic idea is to first train an NMT model on out-of-domain data and then fine-tune it on the in-domain data. Also, Wang et al. (2017) propose instance weighting methods for NMT domain adaptation problem, the main goal of which is to assign higher weights to in-domain data than out-of-domain data.

Using LMs or monolingual data to address domain adaptation has been investigated by several researchers (Sennrich et al., 2016a; Currey et al., 2017; Hu et al., 2019). Moore and Lewis (2010);

Axelrod et al. (2011) use LMs to score the out-of-domain data and then select data that are similar to in-domain text based on the resulting scores, a paradigm adapted by Duh et al. (2013) to neural models. Gulcehre et al. (2015) propose two fusion techniques, namely shallow fusion and deep fusion, to integrate LM and NMT model. Shallow fusion mainly combines LM and NMT model during decoding while deep fusion integrates the two models during training. Researchers have also proposed to perform adaptation for NMT by retrieving sentences or n-grams in the training data similar to the test set (Farajian et al., 2017; Bapna and Firat, 2019). However, it can be difficult to find similar parallel sentences in domain adaptation settings.

7 Conclusion

We propose a novel framework of domain differential adaptation (DDA) that models the differences between domains with the help of models in a related task, based on which we adapt models for the target task. Two simple strategies under the proposed framework for neural machine translation are presented and are demonstrated to achieve good performance. Moreover, we introduce two subword-level evaluation metrics for domain adaptation in machine translation and analyses reveal that our methods can adapt models to a larger extent and with a higher accuracy compared with several alternative adaptation strategies.

However, as shown in our analysis, there are certain limitations for our current methods. Future directions include adding more prior knowledge into our methods as well as considering more sophisticated combining strategies. We will also validate our framework on other pairs of tasks, such as text summarization and language modeling.

Acknowledgements

We are grateful to anonymous reviewers for their helpful suggestions and insightful comments. We also thank Junxian He, Austin Matthews, Paul Michel for proofreading the paper.

This material is based upon work generously supported partly by the National Science Foundation under grant 1761548 and the Defense Advanced Research Projects Agency Information Innovation Office (I2O) Low Resource Languages for Emergent Incidents (LORELEI) program under Contract No. HR0011-15-C0114. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations (ICLR)*.

Ankur Bapna and Orhan Firat. 2019. Non-parametric adaptation for neural machine translation. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Workshop on Machine Translation (WMT)*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Chenhui Chu, Raj Dabre, and Sadao Kurohashi. 2017. An empirical comparison of domain adaptation methods for neural machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *International Conference on Computational Linguistics (COLING)*.

Anna Currey, Antonio Valerio Miceli Barone, and Kenneth Heafield. 2017. Copied monolingual data improves low-resource neural machine translation. In *Conference on Machine Translation (WMT)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Kevin Duh. 2018. The multitarget ted talks task. http://www.cs.jhu.edu/~kevinduh/a/multitarget-tedtalks/.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

M Amin Farajian, Marco Turchi, Matteo Negri, and Marcello Federico. 2017. Multi-domain neural machine translation through unsupervised adaptation. In *Conference on Machine Translation (WMT)*.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06897*.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.

Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask r-cnn. In *International Conference on Computer Vision (ICCV)*.

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

Junjie Hu, Mengzhou Xia, Graham Neubig, and Jaime Carbonell. 2019. Domain adaptation of neural machine translation by lexicon induction. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Workshop on Neural Machine Translation (WMT)*.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. 2015. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning (ICML)*.

Minh-Thang Luong and Christopher D Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *International Workshop on Spoken Language Translation (IWSLT)*.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Graham Neubig, Zi-Yi Dou, Junjie Hu, Paul Michel, Danish Pruthi, and Xinyi Wang. 2019. compare-mt: A tool for holistic comparison of language generation systems. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL) Demo Track*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Daniel Povey, Gaofeng Cheng, Yiming Wang, Ke Li, Hainan Xu, Mahsa Yarmohamadi, and Sanjeev Khudanpur. 2018. Semi-orthogonal low-rank matrix factorization for deep neural networks. In *Annual Conference of the International Speech Communication Association (INTERSPEECH)*.

Swami Sankaranarayanan and Yogesh Balaji. 2017. Generate to adapt: Aligning domains using generative adversarial networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Aditya Siddhant, Anuj Goyal, and Angeliki Metallinou. 2019. Unsupervised transfer learning for spoken language understanding in intelligent agents. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Ben Tan, Yu Zhang, Sinno Jialin Pan, and Qiang Yang. 2017. Distant domain transfer learning. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in opus. In *International Conference on Language Resources and Evaluation (LREC)*.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017. Instance weighting for neural machine translation domain adaptation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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 preprint arXiv:1609.08144*.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. 2017. Controllable invariance through adversarial feature learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Wei Ying, Yu Zhang, Junzhou Huang, and Qiang Yang. 2018. Transfer learning via learning to transfer. In *International Conference on Machine Learning (ICML)*.