Fine-Tuning for Neural Machine Translation with Limited Degradation across In- and Out-of-Domain Data

Praveen Dakwale p.dakwale@uva.nl
Informatics Institute, University of Amsterdam

Christof Monz c.monz@uva.nl
Informatics Institute, University of Amsterdam

Abstract

Neural machine translation is a recently proposed approach which has shown competitive results to traditional MT approaches. Similar to other neural network based methods, NMT also suffers from low performance for the domains with less available training data. Domain adaptation deals with improving performance of a model trained on large general domain data over test instances from a new domain. Fine-tuning is a fast and simple domain adaptation method which has demonstrated substantial improvements for various neural network based tasks including NMT. However, it suffers from drastic performance degradation on the general or source domain test sentences, which is undesirable in real-time applications. To address this problem of drastic degradation, in this paper, we propose two simple modifications to the fine-tuning approach, namely multi-objective learning and multi-output learning which are based on the "Knowledge distillation" framework. Experiments on English-German translations demonstrate that our approaches achieve results comparable to simple fine-tuning on the target domain task with comparatively little loss on the general domain task.

1 Introduction

Standard neural MT (Bahdanau et al., 2015) is an end-to-end neural network which allows for easy training of the NMT system without the need to separately train large phrase table and n-gram language models. However, neural methods including NMT, are known to be data-hungry and do not generalize well for rare events. As a result, NMT systems trained only on a specific domain with less available parallel data quickly overfit and do not perform better or even comparable to standard statistical MT approaches (Zoph et al., 2016).

Research in domain adaptation deals with the problem of improving the performance of a model trained on a general domain data over test instances from a new domain. A simple solution to achieve better performance on both domains will be to train the network on the combined in-domain and general domain parallel data. However, as already discussed in (Freitag and Al-Onaizan, 2016) there are two problems with this approach. First, re-training the model from scratch on the combined data is time consuming and second due to the relatively small size of in-domain data, the learned model will be biased towards the general domain and may not perform comparatively on the in-domain test instances. Moreover, this solution is not efficient in real life applications because in situations where a general domain model is already deployed in an application, the original training data may not be available at the production time.

A fast and efficient method for domain adaptation for neural methods is "Fine-tuning" which has also been recently applied for Neural Machine translation (Freitag and Al-Onaizan, 2016; Chu et al., 2017). In fine-tuning, a neural network which is already trained on large general domain data is further trained on the small data available for the new target domain (also called in-domain data). Fine-tuning provides significant improvements as compared to both only in-domain training or only out-of-domain (general domain data) training. This is because pre-initialization with the parameters trained on large data prevents overfitting on the small in-domain data while at the same time, training on the new data performs a complete transform of the parameters space corresponding to the events observed in the new data. However, as a result of this transformation to the new parameter space, performance of the resulting model decreases drastically for the test instances from the general domain as there is no guidance for the general domain events while fine-tuning (Li and Hoiem, 2016). In a real-time application such a degradation of translation performance on either of the tasks is undesirable because even after adaptation one would like to use the model for translating sentences from both domains.

To address this problem of drastic degradation by fine tuning, in this paper, we propose two simple modifications to the fine-tuning approach. Both approaches are based on the "Knowledge distillation" framework of (Hinton et al., 2014) where a smaller "student" network learns to mimic a large "teacher" network by minimizing the loss between the output distributions of the two networks. We are motivated by the idea that new tasks can be learned without degrading performance on old tasks, by preserving the parameters shared between the two tasks and fine-tuning the task specific parameters with respect to the supervision from the corresponding labels or distributions (Li and Hoiem, 2016).

Our first modification is a simple multi-objective learning which involves *fine-tuning* a general domain model on a small in-domain data while at the same time minimizing the loss between the output distributions of the "student" network (the model learned by fine-tuning) and the baseline teacher network (model trained on large general domain data). In our second modification, we propose adding multiple output layers to the "student" network corresponding to the different tasks (domains) and learning task-specific parameters for both domains using only the in-domain data while simply fine-tuning the parameters shared between the two tasks. Our experiments demonstrate that both the proposed approaches multi-objective *fine-tuning* and multi-output *fine-tuning*, achieve translation performance comparable to vanilla fine-tuning on the target domain task and at the same time suffer little degradation on the original general domain task.

In Section 2 we discuss the related work on domain adaptation for traditional statistical MT as well as recent approaches in neural MT. We briefly introduce neural MT architecture of (Bahdanau et al., 2015) and Knowledge distillation framework of (Hinton et al., 2014) in Section 3. We introduce and explain our approaches in section 4 and finally discuss experiments and results in section 5 and 6 respectively.

2 Related work

Domain adaptation for Phrase-based MT has been studied excessively in last few years. The main approaches in domain adaptation involves either data selection (Moore and Lewis, 2010; Axelrod et al., 2011), instance weighting (Matsoukas et al., 2009; Foster et al., 2010) or multiple model interpolation (Nakov, 2008; Bisazza et al., 2011; Sennrich, 2011). The model interpolation is the most effective, however a complex approach, as it requires training multiple models including phrase-tables, re-ordering and language models corresponding to each of the domains and then learning the corresponding interpolation weights. In case of SMT, Wei wang et al. (2012) has goals similar to that of our approach i.e. to adapt an SMT system for newer domain while at the same time preserve the original generic performance. This approach pre-classifies

the incoming sentence and accordingly select the corresponding model weights.

For domain adaptation in neural machine translation, most of the recent research has focused mainly on "fine-tuning" over the additional in-domain data. This is due to the fact that the end-to-end architecture of NMT enables fast and efficient training without complexity of re-training individual component for each domain. The first attempt for applying fine-tuning to NMT was Luong and Manning (2015), where they simply adapted an NMT system trained on large general domain data by further training on a small in-domain data. Recently, (Freitag and Al-Onaizan, 2016) extended the fine-tuning approach by using an ensemble of the baseline general domain model and the fine-tuned model. They proposed that the ensemble provides better translation on new domain as well as retain much of the performance on the general domain. Their experiments using TED talks (Cettolo et al., 2012) as in-domain data demonstrated improvements on the in-domain and less performance drop on the general domain. However, as we will discuss in section 6, our experiments demonstrate that even with an ensemble, the performance of the fine-tuned model still degrades significantly on the general domain task, especially on a target domain such as medical documents for which vocabulary, style and lexical translation probabilities are highly different from source domain of news articles. On the other hand, our approach of using the "baseline supervision" while fine-tuning, not only retains general domain performance better than ensemble for two different target domains but also provides a faster and efficient decoding procedure by avoiding the requirement to compute the two output distributions separately as required in the ensemble approach.

Another recent approach for NMT domain adaptation is the "mixed fine-tuning" by Chu et al. (2017). They proposed to fine-tune the baseline model on a parallel corpus which is mix of the in-domain and general domain corpora instead of only in-domain data. They also perform multi-domain NMT by augmenting all the corpora with domain tags. However, as they pointed out "mixed fine-tuning" takes longer training time than vanilla fine-tuning depending on the size of the "mixed" data. Also, this requires robust data selection heuristics to extract only relevant sentences from the general domain and avoid selection of noise. On the other hand, in our approach we completely remove any dependence on the general domain data while fine-tuning. Similary, Sennrich et al. (2016) adapt to the new domain by back translating the target in-domain sentences, adding the new data to the training corpus and fine-tuning on the extended bitext.

A related line of research to our approach is in computer vision where the supervision from the baseline model based on the "Knowledge distillation" framework has been extensively used in various domain adaptation paradigms such as "Deep domain transfer" (Tzeng et al., 2015), "Knowledge adaptation" (Ruder et al., 2017) and "Learning without forgetting" (Li and Hoiem, 2016). The "Learning without forgetting" approach of Li and Hoiem (2016) is very similar to our approach since they propose to reduce the general domain performance degradation by adding multiple nodes in the output layer of a convolutional neural network and fine-tune the parameters on the in-domain data using the supervision from the baseline model. However, we differ from (Li and Hoiem, 2016) in that we not only apply this method to neural machine translation, but we also experiment by training with multiple objectives as well as multiple output layers instead of simply adding new nodes corresponding to the new classes.

3 Background

3.1 Neural Machine Translation

We employ an NMT system based on Luong et al. (2015) which is a simple encoder-decoder network. The encoder is a multi-layer recurrent network (we use LSTMs) which converts an

input sentence x = $[(x_1, x_2, ..., x_n)]$ into a sequence of hidden states $[(h_1, h_2, ..., h_n)]$.

$$h_i = f_{enc}(x_i, h_{i-1}) \tag{1}$$

Here, f_{enc} is an LSTM unit. The decoder is another multi-layer recurrent network which predicts a target sequence $y = (y_1, y_2, ...y_m)$. Each word in the sequence is predicted based on the last target word y_{i-1}, the current hidden state of the decoder s_j and the context vector c_j. The context vector c_j in turn is calculated using an attention mechanism Luong et al. (2015) as weighted sum of annotations of the encoder states h_i's.

$$c_j = \sum_{i=1}^{n} \alpha_{ji} h_i \tag{2}$$

where α_{ji} are attention weights corresponding to each encoder hidden state output h_i calculated as follows :

$$\alpha_{ji} = \frac{exp(a(s_{j-1}, h_i))}{\sum_{k=1}^{n} exp(a(s_{j-1}, h_k))} \tag{3}$$

s_j is the decoder hidden state generated by LSTM units similar to the encoder.

$$s_j = f_{dec}(s_{j-1}, y_{j-1}, c_j) \tag{4}$$

Given, the target hidden state and the context vector, a simple concatenation combines the information from both vectors into an attentional hidden state $\tilde{s}_t$.

$$\tilde{s}_t = tanh(W_c[c_t; h_t]) \tag{5}$$

This attentional vector $\tilde{s}_t$ is then projected to the output vocabulary size using a linear transformation and then passed through a softmax layer to produce the output probability of each word in the target vocabulary.

$$p(y_j|y_1, ...y_{j-1}, \mathbf{x}) = softmax(W_s\ \tilde{s}_t) \tag{6}$$

The probability of the sentence is modelled as product of the probability of each target word.

$$p(\mathbf{y}) = \prod_{j}^{m} p(y_j|y_1, ...y_{j-1}, \mathbf{x}) \tag{7}$$

The end-to-end network is trained by maximizing log-likelihood over the training data. The log-likelihood loss is defined as

$$L_{NLL}(\theta) = -\sum_{j=1}^{n} \sum_{k=1}^{|V|} (y_j = k) * log(p(y_j = k|x : \theta)) \tag{8}$$

Where y_j is the output distribution generated by the network at each timestep and *'k'* is the true class label, i.e., the reference target word at each timestep which is selected from a fixed vocabulary *'V'*. The outer summation represent that the total loss is computed as the sum over complete target sequence.

3.2 Knowledge Distillation

Knowledge Distillation is a framework proposed by Hinton et al. (2014) for training compressed "student" networks by using supervision from a large teacher network. Assuming, we have a teacher network with large dimension sizes, trained on a large amount of data, a smaller student network with much smaller dimension sizes can be trained to perform comparable or even better than the teacher by learning to mimic the output distributions of the teacher network on the same data. This is usually done by minimizing the cross-entropy or KL-divergence loss between the two distributions. Formally, if we have a teacher network trained on the same data and with a learned distribution $q(y|x;\theta_T)$, the student network (model parameters represented by θ) can be trained by minimizing the following loss:

$$L_{KD}(\theta,\theta_T) = -\sum_{k=1}^{|V|} KLdiv\Big(q(y|x;\theta_T)\ p(y|x;\theta)\Big) \tag{9}$$

where θ_T is the parameter distribution of the teacher network. Commonly, this loss is interpolated with the log-likelihood loss which is calculated with regard to the target labels for the in-domain data

$$L(\theta,\theta_T) = (1-\lambda)L_{NLL}(\theta) + \lambda L_{KD}(\theta,\theta_T) \tag{10}$$

In order to allow the student network to encode the similarities among the output classes, (Hinton et al., 2014) suggests to generate a smoother distribution called "soft-targets" by increasing the temperature of the softmax of both teacher and student network.

Note that "Knowledge distillation" for model compression has been applied for neural machine translation by (Kim and Rush, 2016). However, we use the supervision from teacher network for domain adaptation while fine-tuning.

4 Domain adaptation with baseline supervision

As described in Section 1, in fine-tuning a model pre-trained on general domain data is further trained on the in-domain data which largely shifts the parameter space of the model to the target domain resulting in a performance degradation on the general domain test instances. We propose that fine-tuning on the target domain using an objective similar to one defined in equation (10) using the supervision from both general and the target domain can avoid the performance degradation on the general domain while achieving performance comparable to vanilla fine-tuning on the target domain. We explore this idea by proposing two modifications to the fine-tuning approach.

4.1 Multi-objective fine-tuning (MCL)

In the first modification, we train the network on a joint objective which includes the supervision provided by the hard target labels in the in-domain data as well as the output distribution of the general domain model on the in-domain data. We believe that with such a joint objective, the network can learn a parameter space common to both domains.

Assuming that we have trained an NMT model on large general domain data, we first record the output distribution of this model on all the in-domain data, then fine-tune this baseline model on the in-domain data by minimizing the log likelihood loss between the target references and the output distribution of the network. However, at the same time, for each sentence, we also minimize the KL-divergence (or cross-entropy) loss between recorded teacher distribution and the distribution produced by the student network as shown in Figure 1. Let the general domain data on which the baseline model is trained be represented by x_{out} and the in-domain

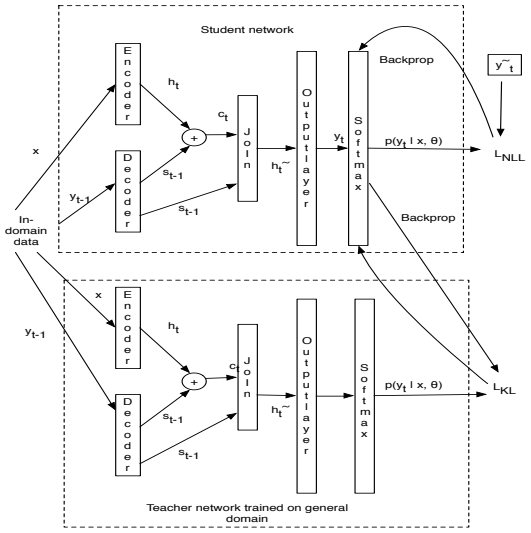

Figure 1: **Multi-objective fine-tuning**. Both the teacher and student network have same architecture. Student network is initialized with parameters trained for teacher network. While fine-tuning parameters of teacher network are frozen.

data be represented by x_{in}, then the final learning objective becomes :

$$\begin{aligned} &L(\theta, \theta_T) = \\ &(1-\lambda)\sum_{k=1}^{|V|}(y_j = k) \times log(p(y_j = k|x_{in} : \theta)) \\ &+ \lambda(-\sum_{k=1}^{|V|} KLdiv\Big(q(y|x;\theta_T)\ p(y|x_{in};\theta)\Big)) \end{aligned} \tag{11}$$

Where θ_T is learned over the general domain data using the standard learning objective in equation 8. Note that as discussed in section 3.2 both distributions here are obtained by increasing the temperature of the softmax as defined in equation 6. This is done by dividing the input by a constant hyper-parameter ϵ

$$p_\epsilon(y|x;\theta) = softmax(\frac{W_s\ \tilde{s}_t}{\epsilon}) \tag{12}$$

Note that Equation 11 clearly indicates that the proposed approach does not require the original data while fine-tuning but only the parameters of the trained baseline model.

4.2 Multiple-output layer fine tuning (MLL)

Though learning on a joint objective as discussed in section 4.1 can reduce the degradation on the general domain task to some extent, a much better solution in order to retain the performance on the old-task could be to preserve task specific parameters corresponding the old-task and at the same time slightly transform parameters shared across the two tasks. Therefore, as a second modification, we propose modifying the baseline model by adding parameter specific to the new task and learning the task-specific parameters with respective learning objectives.

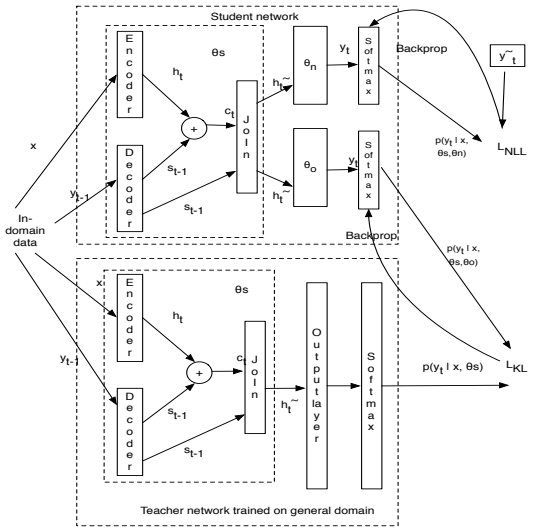

Figure 2: **Multi-output-layer learning**. Student network has two output layers. The additional layer is trained wrt distribution from teacher network

In this work, we consider only the parameters of the final output-layer (W_s as defined in equation 6) of the NMT network as task-specific, while all the parameters corresponding to encoder, decoder, attention mechanism and the concatenation layer in equation 5 are considered to be shared. Let θ_s to be the set of all the shared parameters, θ_o and θ_n to be the task-specific output-layer parameters corresponding to the old (general domain) and new (in-domain) task.

Training the general domain baseline model results in initial-learning of the shared parameters θ_s and the output layer for the out-of-domain task θ_o. For training the in-domain student model, we first modify the network by adding another output layer to the standard NMT network as shown in Figure 2. Similar to multi-objective fine-tuning, we first note the output distribution of the general domain teacher model on the "in-domain" data. Then the shared parameters for the student network corresponding to encoder-decoder and attention mechanism are initialized with θ_s which are learned from the baseline model. Similarly, the parameters of the output layer corresponding the old-domain task are also initialized with parameters θ_o learned on the general domain task. Parameters specific to the in-domain task θ_n are initialized randomly. Then we first train the new parameters θ_n using the ground-truth with the standard log-likelihood objective as defined in equation 8. This is a simple warm-up procedure which enhances the fine-tuning performance (Li and Hoiem, 2016). During this warm-up, we freeze θ_s and θ_n. Finally all the parameters are fine-tuned by minimizing the joint objective. Consider x_n, y_n to be a sentence pair from the in-domain data. Then y_o be the recorded output of the "teacher" model for the new data, i.e.,

$$y_o = q(x_n, \theta_s, \theta_o) \tag{13}$$

For the "student" network, let y'_o and y'_n be the output distributions from the old and new output-layer respectively corresponding to θ'_o and θ'_n

$$y'_o = p(x_n, \theta'_s, \theta'_o) \tag{14}$$

$$y'_n = p(x_n, \theta'_s, \theta'_n) \tag{15}$$

Then similar to equation 11, we define two objectives. First is the standard log-likelihood loss for the shared parameters θ_s and the parameters for new task θ_n with respect to the target references in the in-domain data

$$L_n = -y_n \times log(p(x_n|\theta'_s, \theta'_n)) \tag{16}$$

The second objective is the KL-divergence between the output distribution of the old-layer of the student network with respect to the recorded distribution of the teacher network on the same in-domain data.

$$L_o = -\sum_{k=1}^{|V|} KLdiv\Big(y_o, p(y'_o|x_n; \theta'_s, \theta'_o)\Big) \tag{17}$$

The student network is finally trained on the joint objective defined as:

$$L_{combined} = (1 - \lambda)L_n + \lambda L_o \tag{18}$$

While decoding, the output layer corresponding to the domain label of the test sentences is used to compute the output distribution.

5 Experimental settings

5.1 NMT parameters

In all our experiments, we use an NMT system based on Luong et al. (2015) and implemented using the Torch7 deep learning framework. It is a two layer unidirectional LSTM encoder-decoder with a global attention (dot product) mechanism. Both the encoder and decoder have input embedding and hidden layer of size 1000. As it is common practice, we limit the vocabulary sizes to 60k for the source and 40k for the target side. Parameters are optimized using stochastic gradient descent. We set the initial learning rate as 1 with a decay rate of 0.5 for each epoch after 5th epoch. Model weights are initialized uniformly within [-0.02, 0.02]. A dropout value of 0.2 is applied for each layer. The mini-batch size for out-of-domain is fixed to 64, while for each of the in-domain fine-tunings, we use a batch size of 32. We train for a maximum of 20 epochs and decode with standard beam search with beam size of 10. All models are trained on NVIDIA Titan-X (Pascal) devices.

For the in-domain trainings, we use the same vocabulary extracted from the baseline general domain bitext. Note that our multiple-output layer approach allows for use of a different vocabulary (that can be extracted from the new data) for the in-domain fine-tuning. We experimented with different values of interpolation weights λ (as used in equation 11 and 18) and also with temperature ϵ (defined in equation 12) and obtained the optimal values to be 0.9 and 2 respectively for these hyper-parameters for all the experiments.

5.2 Data

We conducted experiments on English-German translation. For all the settings we use WMT-2015 Bojar et al. (2015) training set as the general domain training data. This training set contains approximately 4.2 million parallel sentences from multiple sources Europarl, news commentary and common crawled articles. We constrain the maximum sequence length to be 80 and remove the sentences greater than this length along with the duplicates. Thus we are left with a bitext of size approximately 4 million, out of which we reserve 5000 sentence for perplexity validation and use the rest for training the general domain baseline model. We use newstest15 (official test set for WMT-2015) as general domain test set.

	Epo.	iwslt	newstest'15
Standard NMT baselines			
TED_{only}		17.95	
WMT_{only}		19.46*	18.54*
Combined training (WMT+ TED)		$22.86^{\blacktriangle}$ (+3.4)	$17.57^{\blacktriangledown}$ (-0.97)
Fine-tuning methods			
FT	1	$19.69^{\triangle}$(+0.23)	$13.94^{\blacktriangledown}$(-4.6)
	3	$21.90^{\blacktriangle}$(+2.44)	$12.68^{\blacktriangledown}$(-5.86)
FTEns		$22.90^{\blacktriangle}$(+3.44)	$16.70^{\blacktriangledown}$(-1.84)
Proposed approaches			
MCL	1	$21.77^{\blacktriangle}$(+2.21)	$16.95^{\blacktriangledown}$(-1.59)
	7	$22.19^{\blacktriangle}$(+2.73)	$16.55^{\blacktriangledown}$(-1.99)
MLL	1	$20.26^{\triangle}$(+0.8)	$18.24^{\triangledown}$(-0.3)
	9	$21.70^{\blacktriangle}$(+2.24)	$16.90^{\blacktriangledown}$(-1.64)

Table 1: BLEU scores for different approaches over TED data domain adaptation. iwslt = IWSLT (2011+2012+2013). * represents the baseline setting for these experiments $^{\blacktriangle}/_{\blacktriangledown}$ and $^{\triangle}/_{\triangledown}$ indicates a statistically significant gain/drop at $p < 0.01$ and $p < 0.05$ respectively over the baseline. TED_{only} = only in-domain training, WMT_{only} = only general domain training, FT = Vanilla fine-tuning, FTEns = Ensemble fine tuning, MCL = Multi-objective fine-tuning, MLL = Multi-output-layer fine-tuning

For in-domain training we consider two different domains namely the TED-talks bitext (IWSLT 2013) Cettolo et al. (2012) (approx 170k sentences) and EMEA corpus (Tiedemann, 2009) (approx 200k sentence) which is a parallel text of medical guidelines. From each of these, we reserve 5000 sentence for respective perplexity validation. For the TED talk domain, we use a combination of official test sets for IWSLT 2011, 2012 and 2013 as evaluation set and for EMEA, we reserve a set of 2000 sentences (excluded from training corpus) from the EMEA corpus as evaluation set. Results are reported in terms of case-insensitive BLEU-4 (Papineni et al., 2002). Approximate randomization (Noreen., 1989; Riezler and Maxwell, 2005) is used to detect statistically significant differences.

6 Results

We define multiple baselines to compare the proposed approaches. Firstly, we obtain the BLEU scores for a model trained only on general domain data and note its performance on the general domain (source domain) test set as well as on the in-domain test sets. Similarly, for baseline comparisons we train models only on the small in-domain data for each of the target domains. We compare our approaches to the vanilla fine-tuning and ensemble methods (Freitag and Al-Onaizan, 2016) in terms of the BLEU improvements on the in-domain test sets and degradation on the general domain test-set. Finally, we also train baseline models on the combined general and in-domain bitext and test it on both test sets. Note that this baseline act as ceiling for our approaches as this setting can only be trained when data from both general as well as target domain are available beforehand.

For the domain adaptation over TED data, Table 1 summarizes the BLEU scores over different settings and also the highest and lowest BLEU scores for fine-tuning as well as the proposed approaches. The performance of general-domain-only (WMT_{only}) model (19.46) is higher than that of in-domain-only (TED_{only}) model (17.95) for the iwslt test set. Hence, we consider BLEU scores of the general-domain-only model on the two test sets as our baselines.

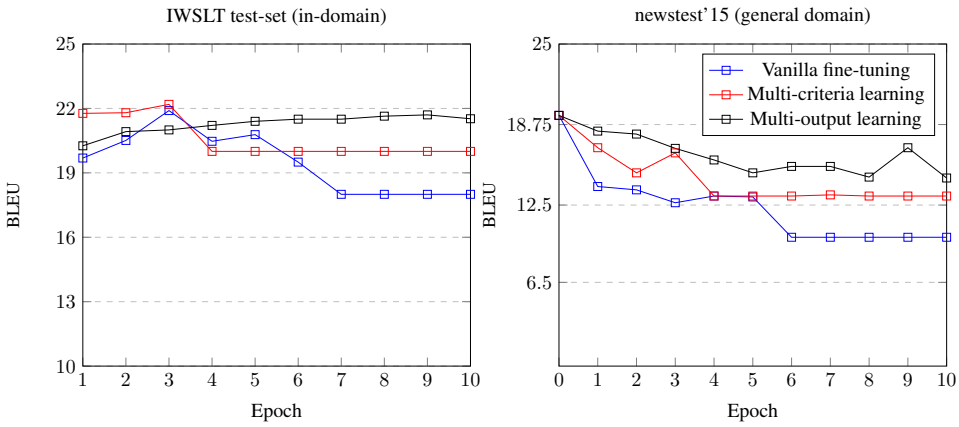

Figure 3: Epoch-wise BLEU (i) improvement over in-domain test set (TED) (ii) degradation over general domain (WMT)

To compare the gradual degradation of the vanilla fine-tuning with the proposed approaches, we report the BLEU scores corresponding to first epoch as well as highest BLEU score achieved for the in-domain test set. The last column shows the corresponding BLEU scores over the general domain test set.

Vanilla fine-tuning improves over the baseline by up to 2.4 BLEU in the third epoch for the target domain task. However, it drops substantially even in the first epoch by 4.6 BLEU on the general domain task and by 5.8 corresponding to third epoch (best model for target domain test-set). On the other hand, for the target domain, multi-objective learning improves over the baseline by up-to 2.7 BLEU (in third epoch) which is slightly better than vanilla fine-tuning while at the same time, performance degradation on the source domain is only -1.99 which is substantially less than that of fine-tuning. Similarly, for multi-output-layer learning, we observe improvements of up to 2.2 BLEU over the baseline on the target domain which is almost equal to fine-tuning and the drop in BLEU over the source domain for the 9th epoch is only 1.64. The ensemble method of (Freitag and Al-Onaizan, 2016) achieves an improvement of 1 BLEU over simple fine-tuning for in-domain test set (iwslt) which is higher than both proposed methods and also suffers similar degradation for general domain (newstest'15). This is due to the fact that though TED talk is considered to be a different domain, the vocabulary and style of the TED data is very similar to that of the general domain data. Also, the model trained on the combined data performs comparable to the best setting i.e, the ensemble method for both test sets. Figure 3 shows the epochwise comparison of BLEU score over source and target domain test sets for TED data domain adaptation. For the source domain fine-tuning performance drops rapidly starting from the first epoch. However, performance degradation of the two proposed approaches on general domain is relatively slow

In conclusion, We observe that for the TED data, our approaches show comparable improvement to fine-tuning while suffering less degradation over source domain. However, the ensemble method of (Freitag and Al-Onaizan, 2016) also demonstrates similar results. Therefore, we repeat our experiments for domain adaptation over the EMEA corpus for which the vocabulary is substantially different from the source domain of news article as compared to the TED data.

Table 3 summarizes our experiments for domain adaptation for EMEA data. First important observation the BLEU score for the EMEA_{only} model is substantially higher (6.5 BLEU

	Epo.	EMEA-test	newstest'15
Standard NMT baselines			
EMEA_{only}		20.08 *	
WMT_{only}		13.54	18.54*
Combined training (WMT+ EMEA)		$20.84^{\triangle}$(+0.76)	$17.60^{\blacktriangledown}$(-0.94)
Fine-tuning methods			
FT	1	12.34	$7.10^{\blacktriangledown}$(-11.44)
	6	$22.57^{\blacktriangle}$(+2.49)	$4.50^{\blacktriangledown}$(-14.04)
FTEns		$19.43^{\blacktriangledown}$(-0.65)	$13.63^{\blacktriangledown}$(-4.91)
Proposed approaches			
MCL	1	$18.14^{\blacktriangledown}$(-1.94)	$15.50^{\blacktriangledown}$(-3.04)
	11	**22.50**$^{\blacktriangle}$(+2.42)	**13.29**$^{\blacktriangledown}$(-5.25)
MLL	1	20.6(-0.02)	$17.17^{\blacktriangledown}$(-1.37)
	6	**22.33**$^{\blacktriangle}$(+2.25)	**14.67**$^{\blacktriangledown}$(-3.87)

Table 2: BLEU score for different approaches for EMEA data domain adaptation. * represents the baseline setting for these experiments. EMEA-test = test set for medical domain, EMEA_{only} = training only on in-domain medical data, WMT_{only} = training only on general domain data. Other abbreviations are same as Table 1

points) than the WMT_{only}(this is contrast to the experiments on TED data where the WMT_{only} model performed better than TED_{only} model). Hence, for this set of experiments we consider the general domain performance (20.08) as our baseline. Also, for medical domain adaptation, the model trained on the combined data performs only slightly better than the in-domain baseline over the EMEA test set while comparable to the general domain model on the WMT test set. This implies that this joint model is more biased towards the general domain and hence does not perform better than fine-tuning on the in-domain test set.

Vanilla fine-tuning shows an improvement of up to 2.5 BLEU (in the 6th epoch) over the in-domain-only model on EMEA test-set. However, the drop in performance over the general domain test set is drastic i.e. 11.3 in the first epoch and 14 BLEU in the 6th epoch. Also, though ensemble method suffers less degradation in performance over newstest'15 (4.9 BLEU), it performs slightly lower than the baseline for the EMEA test (-0.6) as compared to vanilla fine-tuning. This implies that a simple ensemble is more biased towards general domain. On the other hand, multi-objective learning performs comparable to fine-tuning on the EMEA test set (+2.4 improvement over baseline) while the drop in general domain performance (-5.25) is not as drastic as vanilla fine-tuning. Similarly, for multi-output-layer approach, while the improvement on EMEA test-set is comparable to both approaches, it shows least drop in performance for newstest'15. Figure 4 shows the comparison of BLEU scores over EMEA test sets and newstest'15 for medical data domain adaptation. Similar to TED data experiments, while fine-tuning performance drops rapidly on newstest'15, for the proposed methods, it is relatively slow.

In conclusion, for the medical domain, proposed methods of multi-objective and multi-output-layer learning show improvements comparable to fine-tuning on the target domain with relatively little loss on the source domain as compared to fine-tuning. On the other hand, ensemble method is biased towards the general domain and fails to add any improvement for the target domain.

Finally, we compare the average decoding time per sentence for ensemble decoding and multi-objective learning in Table 3. The decoding time for ensemble method is twice that of

	Average time(ms)
Fine tuning	0.131
Ensemble decoding	0.277
Multi-objective learning	0.147

Table 3: Average decoding time (in millisecond) on GPU devices per sentence for ensemble decoding and multi-objective learning. Average sentence length for the used test set is 20.9 tokens

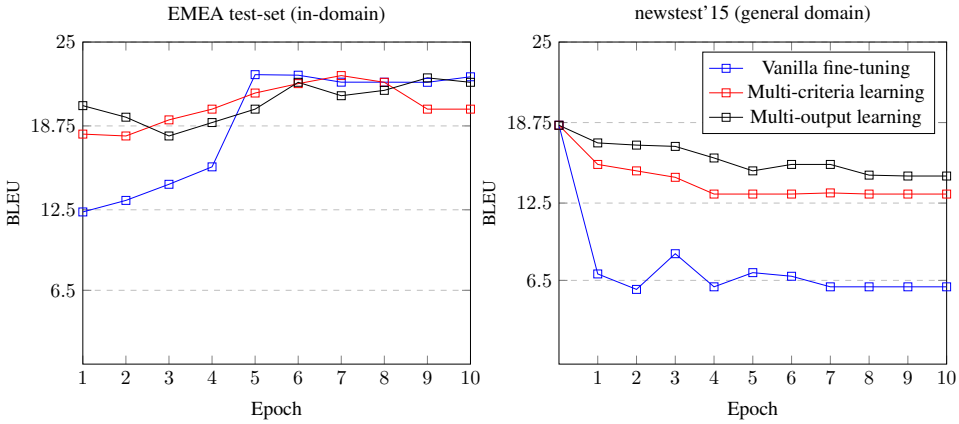

Figure 4: Epoch-wise BLEU (i) improvement over in-domain test set (EMEA) (ii) degradation over general domain test-set (WMT)

fine-tuning due to computation on two different models while for multi-objective learning it is same as that of fine-tuning.

7 Conclusion and future work

In this paper, we proposed two modifications to the well-known fine-tuning method for domain adaptation for neural machine translation in order to retain the performance on the source domain. We observe that both proposed approaches achieve performance comparable to the vanilla fine-tuning while still retaining performance on the source (general) domain. Moreover, the decoding speeds of the proposed methods are same as fine-tuning as compared, while the ensemble method requires almost twice decoding time than the fine-tuning.

In this work, we mainly focused on adapting a general domain model to a new domain. As future work, we plan to experiment by extending the two approaches for iteratively adapting a single model for multiple domains.

Acknowledgments

This research was funded in part by the Netherlands Organization for Scientific Research (NWO) under project numbers 639.022.213 and 612.001.218.

References

Axelrod, A., He, X., and Gao, J. (2011). Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11,

pages 355–362. Association for Computational Linguistics.

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

Bisazza, A., Ruiz, N., and Federico, M. (2011). Fill-up versus interpolation methods for phrase-based smt adaptation. In *IWSLT*.

Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Huck, M., Hokamp, C., Koehn, P., Logacheva, V., Monz, C., Negri, M., Post, M., Scarton, C., Specia, L., and Turchi, M. (2015). Findings of the 2015 workshop on statistical machine translation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 1–46, Lisbon, Portugal. Association for Computational Linguistics.

Cettolo, M., Girardi, C., and Federico, M. (2012). Wit3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th EAMT Conference, 28-30 May 2012, Trento, Italy*.

Chu, C., Dabre, R., and Kurohashi, S. (2017). An empirical comparison of simple domain adaptation methods for neural machine translation. *CoRR*, abs/1701.03214.

Foster, G., Goutte, C., and Kuhn, R. (2010). Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459, Cambridge, MA. Association for Computational Linguistics.

Freitag, M. and Al-Onaizan, Y. (2016). Fast domain adaptation for neural machine translation. *CoRR*, abs/1612.06897.

Hinton, G., Vinyals, O., and Dean, J. (2014). Distilling the knowledge in a neural network. In *NIPS 2014 Deep Learning Workshop*.

Kim, Y. and Rush, A. M. (2016). Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas. Association for Computational Linguistics.

Li, Z. and Hoiem, D. (2016). Learning without forgetting. In *ECCV*.

Luong, M.-T. and Manning, C. D. (2015). Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal. Association for Computational Linguistics.

Matsoukas, S., Rosti, A.-V. I., and Zhang, B. (2009). Discriminative corpus weight estimation for machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 708–717, Singapore. Association for Computational Linguistics.

Moore, R. C. and Lewis, W. (2010). Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10. Association for Computational Linguistics.

Nakov, P. (2008). Improving english-spanish statistical machine translation: Experiments in domain adaptation, sentence paraphrasing, tokenization, and recasing. In *Proceedings of the Third Workshop on Statistical Machine Translation*, StatMT '08, pages 147–150, Stroudsburg, PA, USA. Association for Computational Linguistics.

Noreen., E. W. (1989). *Computer Intensive Methods for Testing Hypotheses. An Introduction*. Wiley-Interscience.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Riezler, S. and Maxwell, J. T. (2005). On some pitfalls in automatic evaluation and significance testing for mt. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*.

Ruder, S., Ghaffari, P., and Breslin, J. G. (2017). Knowledge adaptation: Teaching to adapt. *CoRR*, abs/1702.02052.

Sennrich, R. (2011). Combining multi-engine machine translation and online learning through dynamic phrase tables. In *15th Conference of the European Association for Machine Translation*.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany. Association for Computational Linguistics.

Tiedemann, J. (2009). News from opus - a collection of multilingual parallel corpora with tools and interfaces. In *Recent Advances in Natural Language Processing*.

Tzeng, E., Hoffman, J., Darrell, T., and Saenko, K. (2015). Simultaneous deep transfer across domains and tasks. In *International Conference in Computer Vision (ICCV)*.

Wei wang, Klaus Macherey, W. M., och, F., and Xu, P. (2012). Improved domain adaptation for statistical machine translation. In *Proceedings of the Tenth Biennial Conference of the Association for Machine Translation in the Americas*, San Diego, California.

Zoph, B., Yuret, D., May, J., and Knight, K. (2016). Transfer learning for low-resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.