Multi-Domain Neural Machine Translation through Unsupervised Adaptation

M. Amin Farajian[(1,2)], **Marco Turchi**[(1)], **Matteo Negri**[(1)], **Marcello Federico**[(1)]
[(1)]Fondazione Bruno Kessler, Trento, Italy
[(2)]University of Trento, Trento, Italy
{farajian,turchi,negri,federico}@fbk.eu

Abstract

We investigate the application of Neural Machine Translation (NMT) under the following three conditions posed by real-world application scenarios. First, we operate with an input stream of sentences coming from many different domains and with no predefined order. Second, the sentences are presented without domain information. Third, the input stream should be processed by a single generic NMT model. To tackle the weaknesses of current NMT technology in this unsupervised multi-domain setting, we explore an efficient instance-based adaptation method that, by exploiting the similarity between the training instances and each test sentence, dynamically sets the hyperparameters of the learning algorithm and updates the generic model on-the-fly. The results of our experiments with multi-domain data show that local adaptation outperforms not only the original generic NMT system, but also a strong phrase-based system and even single-domain NMT models specifically optimized on each domain and applicable only by violating two of our aforementioned assumptions.

1 Introduction

The progress towards a more pervasive integration of machine translation (MT) into industrial translation workflows has to confront two interconnected problems. On one side, MT technology should be able to guarantee a high level of flexibility to deliver good-quality output in a wide range of use scenarios (language combinations, genres, domains). On the other side, the infrastructures required to reach this objective should be scalable enough to enable the industrial deployment of MT at reasonable cost.

The first problem is a well known one in (statistical) MT: regardless of the paradigm adopted, performance is bounded by the similarity between training and test data. The scenario addressed in this paper, in which the input stream comes from a variety of different domains, is a typical example where models trained on generic parallel corpora suffer from data diversity. Indeed, processing sentences from diverse domains becomes more and more difficult when the distance from the training instances increases. The more the domains a system is exposed to, the higher the chance to experience drops in translation quality under unseen conditions. To cope with this issue, MT systems should be flexible enough to adapt to a variety of linguistic differences (*e.g.* lexical, structural) between different data points.

The second problem is more practical: in absence of flexible models, multi-domain translation scenarios call for infrastructures based on multiple specialised systems, each of which is tuned to maximise performance in a given domain. This solution, however, has two evident drawbacks: *i)* domain-specific models can only be invoked by input sentences presented along with domain information, so that each instance is processed by the right model, and *ii)* each time a new domain has to be covered, a new dedicated model has to be trained with domain-specific data. In real-world application scenarios, however, translation requests rarely come with domain information, the notion of domain is *per se* fuzzy, domain-specific data can be hard to acquire and, most importantly, architectures' costs and scalability are of utmost concern. When maintenance costs and architecture scalability come into play, a preferable solution would be to rely on one single model, capable to adapt on-the-fly to input streams of diverse data,

Proceedings of the Conference on Machine Translation (WMT), Volume 1: Research Papers, pages 127–137

without any supervision.

Neural machine translation (Bahdanau et al., 2014), which has recently become the dominant approach in MT, is not immune to the aforementioned problems. Domain drifts are in fact hard to manage by NMT, due to its inherent characteristics. Different from the phrase-based paradigm, in which training data is explicitly memorised and used in the form of basic translation constituents (phrases), NMT generates a more implicit representation of the data, by compressing and distributing the information over its internal parameters. Moreover, given the amount of data and time required for training, high flexibility and fast adaptation capabilities of single generic models become key requirements to unleash NMT's potential in industry applications.

To pursue these objectives, we investigate the application of an unsupervised method to adapt on-the-fly a generic NMT model (M_g) and improve translation quality for an input stream of diverse, multi-domain sentences presented in random order. Our approach is based on a retrieval mechanism that, given an input sentence q, extracts from the pool of parallel data the top (*source*, *target*) pairs in terms of similarity between the *source* and q. The retrieved pairs are then used to fine-tune the model (M_q), which is then applied to translate q. Finally, the adapted model is reset to the original parameters (M_g), the next input sentence is read, and so on. In order to learn more efficiently from the retrieved set, we introduce a dynamic method that, based on the similarity between the test sentence and the retrieved pairs, decides about the hyperparameters to be used by the learning algorithm (*i.e.* learning rate and number of epochs).

In our experiments with multi-domain data, we observe significant improvements by our approach over the generic NMT system, a strong generic phrase-based MT system, and also specialised NMT models fine-tuned on each domain using domain-specific data. In particular, by dynamically setting the model hyperparameters, our solution is able to outperform the strong pool of domain-specific NMT systems by +2.8 BLEU scores, in overall.

2 Related Works

Domain adaptation has been extensively studied in machine translation. The existing works in this field mostly rely on the assumption of knowing the target domain in advance and having in-domain training data of reasonable size. This dataset is then used to train specific models that are interpolated with generic ones using standard log-linear methods (Koehn and Schroeder, 2007) or mixture models (Foster and Kuhn, 2007).

In line with the work presented in this paper, (Eck et al., 2004) and (Zhao et al., 2004) proposed to perform an instance selection step in which for each test document/sentence a small set of similar documents/sentences is retrieved from the pool of training data and used to build more specific language models. (Hildebrand et al., 2005) further extended this approach and proposed to build local translation models using the set of retrieved sentence pairs.

As in SMT, recent works in domain adaptation for neural MT share the assumption of knowing the target domain in advance and report significant improvements by adapting the generic system to the target domain as an offline step (Luong and Manning, 2015). More recently, (Li et al., 2016)[1] proposed an instance-based adaptation technique for NMT in which for each translation segment a set of similar sentence pairs is retrieved. This small training set is then used to update the model before translating the given test sentence. In order to reduce the cost of computing the similarity of the test segment and all the sentences in the pool, they suggest a three-step process in which, as the first step, all the sentence pairs containing at least one of the test words are retrieved. This large set is then filtered by measuring the similarity between the source sentences and the test using the Dice coefficient measure (Dice, 1945), and keeping the top 1000. The remaining sentences are then ranked based on their Levenshtein distance to the test sentence. To cope with the risk of training a model on sentence pairs with low similarity to the test sentence, the first best retrieved sentence pair is used only if its similarity is higher than a threshold (*i.e.* 0.4), otherwise they use the top-128 training pairs. Moreover, to keep under control the possibility of overfitting, they suggest to train the system for only one epoch over the retrieved set.

To further investigate the potential of the instance selection approach, we study if properly

[1]This work is a non-archival document that has not been peer reviewed for publication. For the sake of completeness we reported it in this paper and compared our results with our reimplementation of this method.

setting the hyperparameters in the learning algorithm can boost NMT performance. For this purpose, we take a different direction from (Li et al., 2016) by proposing a strategy to dynamically select the number of epochs and learning rate based on the similarity between the best retrieved sentence pair and the test segment. We empirically prove the effectiveness of this method in a multi-domain application scenario and show that only our adaptive approach is able to produce better translation quality than the PBMT system and the strong pool of specialised NMT systems.

3 Neural Machine Translation

We build our adaptive NMT approach on top of the state-of-the-art sequence-to-sequence model proposed by (Bahdanau et al., 2014). This model relies on a two-step process: first, a recurrent neural network encodes the source sentence word by word into a sequence of hidden states; then, another recurrent neural network decodes the source hidden sequence into the target string. Both the encoder and decoder networks are implemented with *gated recurrent units* (Cho et al., 2014). In particular, the decoder network operates like a language model: it predicts the next target word from the last target word, the last hidden state of the decoder, and a convex combination of the encoder hidden states. The weights of this convex combination are dynamically computed through a simple feed-forward network, called *attention model*. Intuitively, similarly to a word alignment model, the attention model informs the decoder about the encoder hidden states corresponding to the next target word. The decoder actually predicts a full distribution over the target language vocabulary. Thus, the generation of a translation requires sampling at each step the most probable target word from the distribution and then feeding it back to the decoder as input for the next step. The decoding phase is initialised with a conventional delimiter symbol and terminates when the same symbol is output. Better translations are actually produced by integrating the decoder generative process with a beam search, that considers multiple input and output word hypotheses at each step. Training of the presented NMT architecture involves estimating many parameters, such as word embedding matrices, GRU layers in both the encoder and decoder networks, and the attention model weights. Training is carried out via maximum-likelihood estimation over a large collection of parallel sentences. In particular, optimization is performed via stochastic gradient descent (SGD), by iterating over batches of training data randomly shuffled after each epoch (Goodfellow et al., 2016). More formally, starting from a random initialisation of the parameters, at each iteration a batch B is extracted and each parameter w is moved one step in the opposite direction of the mean gradient of the log-likelihood (L), evaluated on the entries of B:

$$\Delta w = -\eta \frac{1}{\mid B \mid} \sum_{(s,t) \in B} \frac{\partial L(s,t)}{\partial w} \qquad (1)$$

The size of the step Δw is moderated by a *learning rate* η which can either be fixed for all parameters and all iterations, or vary along one or both dimensions (Goodfellow et al., 2016). During training, the SGD procedure typically goes through several so-called epochs, *i.e.* the number of times the whole training data is processed.

The above presented training procedure is also used to adapt an already trained NMT model to a new task for which representative training data is available (Luong and Manning, 2015). In this paper, we investigate the application of the adaptation procedure under extreme conditions, that is when the training data is made of few samples.

4 Unsupervised Instance-based Adaptation

We consider the scenario in which translation requests from a variety of domains are presented in random order to a single generic system. We also assume that each input sentence is presented without information about the domain it comes from. In this setting, the system needs to perform an unsupervised adaptation step on-the-fly and produce the translation.

To this aim, we experiment with an approach in which, given a generic NMT model (M_g), the pool of parallel data (C_p), and a sentence to be translated (q), the following three steps are performed: (1) q is used as a query to retrieve from C_p a set of (*source, target*) pairs (C_q) in which the *source* is similar to q; (2) this set is used to locally adapt the hyperparameters (HP_q) of M_g; (3) the resulting locally-tuned model (M_q) is applied to translate q. If C_q is empty, the generic model is used to translate q. The pseudo code of this approach is shown in Algorithm 1.

Algorithm 1 RetrieveAdaptTranslate

▷ M_g: generic NMT model
▷ S: stream of sentences to be translated
▷ RT: text retrieval module
▷ C_q: retrieved parallel sentences
▷ q^*: translated segment
procedure RAT(M_g, RT, S)
    ▷ For each segment to be translated
    while pop q from S **do**
        ▷ Local copy of the generic model
        M_q:=M_g
        ▷ Instance selection
        C_q:=**Retrieve**(RT,q)
        if $C_q \neq \varnothing$ **then**
            ▷ Model optimization
            HP_q:=**SetHP**(C_q, q)
            M_q:=**Adapt**(M_g, C_q, HP_q)
        ▷ Translate the segment with M_q
        q^*:=**Translate**(M_q, q)
        ▷ Post the translated segment
        Post q^*

Instance selection and parameter optimization are the two key steps of this algorithm. On one hand, since instance selection aims to retrieve the most relevant sentences from C_p, the similarity measure plays an important role as the quality of the material used for local tuning directly affects the next processing steps. In this paper, we use Lucene (McCandless et al., 2010), an open-source information retrieval library that is highly optimized for text search purposes. However, since the similarity measure used in Lucene is based on *tf-idf* counts (Baeza-Yates and Ribeiro-Neto, 2011), it does not consider the order of the words and n-grams in the query and in the retrieved sentences, which is an important aspect for MT model training. In order to take advantage also of this information, we first query Lucene to retrieve a large set of candidates and then re-score them using the sentence-level BLEU (Chen and Cherry, 2014), so that the sentences with higher BLEU score are ranked first. Finally, the top-n similar sentences are used to update the model. This approach is reasonably fast, since it takes advantage of Lucene in searching in a large set of data and then computes the BLEU scores on just few candidates.

The optimization phase, on the other hand, needs to effectively adapt the model parameters with a very small set of parallel data featuring different levels of similarity. In order to tune at best its parameters with respect to the input sentence q, the system has in fact to learn as much as possible from highly similar points in C_q, and keep under control the risk of overfitting in the case of instances with low similarity. The learning rate and number of times the system iterates over the retrieved sentence pairs hence become crucial aspects during optimization. Differently from (Li et al., 2016), who keeps these factors fixed, in this paper we propose a simple yet effective method that dynamically decides about the hyperparameters of the learning algorithm (*i.e.* HP_q) based on the relevance of the retrieved sentence pairs to the input segment. To this aim, we define two functions that for the retrieved samples with high similarity to the test segment increase the learning rate and number of epochs, so that the system can leverage more the information of the training set and vice versa. The idea is to overfit more the NMT system on sentences that are similar to the test sentence while avoiding drastic changes in case of tuning with low similarity sentence pairs. In Section 6, we investigate the effect of dynamically setting these parameters on the final performance of the system. As the results show, our dynamic method significantly improves the performance of the adaptive system, outperforming the strong PBMT system and the oracle NMT systems fine-tuned specifically to each domain.

5 Experimental Setup

5.1 Data

Our experiments are carried out on an English to French translation task, where the training data is a collection of publicly available corpora from different domains: European Central Bank (ECB), Gnome, JRC-Acquis (JRC), KDE4, OpenOffice (OOffice), PHP, Ubuntu, and translated UN documents (UN-TM).[2] Since the size of these corpora is relatively small for training robust MT systems, in particular NMT solutions, we added the News Commentary data from WMT'13[3](WMT_nc), as well as the CommonCrawl (CommonC.) and Europarl corpora as out-domain data, so to reach a total of ∼5.8M sentence pairs.

From each specific domain a set of size 500 sentence pairs is randomly selected as development set, and 1,000 sentence pairs are used as held-

[2] All the corpora are available in http://opus.lingfil.uu.se
[3] http://www.statmt.org/wmt13/

	Segments	Tok/Typ	Avg. Len
ECB	142.4K	76.7	20.5
Gnome	236.1K	102.1	7.2
JRC	678.9K	146.3	15.4
KDE4	160.7K	25.3	6.4
OOffice	32.9K	40.8	11.2
PHP	36.7K	26.0	6.5
Ubuntu	7.5K	5.1	5.2
UN-TM	37.6K	69.6	21.9
WMT_nc	189.1K	65.3	24.6
CommonC.	2.6M	80.3	20.9
Europarl	1.7M	364.3	22.9

Table 1: Statistics of the English side of the training corpora.

out test corpus. Duplicated sentence pairs are removed from each corpus separately, resulting in a total of 3,527 dev and 6,962 test corpora for all the domains. To analyze the performance of the system on generic data, two subsets of size 500 and 1000 sentence pairs are randomly selected from the WMT'13 test data as dev and test corpora. The statistics of the training and test corpora are reported in Tables 1 and 2, respectively, showing that the considered domains are extremely diverse in terms of average sentence length and average word frequency. The *Avg. Sim* column in Table 2 reports the average similarity of the test sentences and the source side of the most relevant sentence pair retrieved from the pool of training data. The scores are computed using the sentence-level BLEU (Chen and Cherry, 2014). Since our adaptation approach updates the model by leveraging these retrieved sentences, their average similarity can be a reliable indicator for predicting the performance gain after adaptation. In other words, the system can learn more from the retrieved samples in the case of corpora with higher sentence similarity (*e.g.* Gnome) than the datasets with lower average BLEU score (*e.g.* WMT). In Section 6 we analyze these features and their impact on the system performance.

Finally, the analysis of the characteristics of Gnome, KDE4, OpenOffice, PHP, and Ubuntu, which are often referred to as IT domain corpora, evidences another important issue in developing domain-specific MT systems. As the statistics of Table 1 show, these corpora are extremely diverse in terms of average sentence length and word frequency, which are likely to correspond to different

	Segments	Tok/Typ	Avg. Sim
ECB	1,000	5.5	50.5
Gnome	982	3.8	70.2
JRC	757	5.1	54.7
KDE4	988	7.0	34.8
OOffice	976	5.8	30.3
PHP	352	4.1	55.7
Ubuntu	997	2.7	27.7
UN-TM	910	7.1	65.1
WMT	1,000	2.2	11.9

Table 2: Statistics of the English side of the test corpora.

levels of difficulty for MT and, in turn, to large differences in final translation quality.

5.2 Neural MT System

All our experiments with NMT are conducted with an in-house developed and maintained branch of the Nematus toolkit[4] which is an implementation of the attentional encoder-decoder architecture (Bahdanau et al., 2014). Since handling large vocabularies is one of the main bottlenecks for the existing NMT systems, state-of-the-art approaches are trained on corpora in which the less frequent words are segmented into their sub-word units (Sennrich et al., 2016) by applying a modified version of the byte pair encoding (BPE) compression algorithm (Gage, 1994). This makes the NMT systems capable of dealing with new and rare words. As recommended in (Sennrich et al., 2016), in order to increase the consistency in segmenting the source and target text, we combined both sides of the training data, and set the number of merge rules to 89,500, resulting in vocabularies of size 78K and 86K tokens respectively for English and French. We use mini-batches of size 100, word embeddings of size 500, and GRU layers of size 1,024. The maximum sentence length is set to 50. The models are trained using Adagrad (Duchi et al., 2011) by reshuffling the training set at each epoch, and are evaluated every 10,000 mini-batches with BLEU (Papineni et al., 2002).

5.3 Terms of Comparison

We compare our adaptive NMT system with a generic NMT and a strong PBMT system trained on the *pool* of all the training data. For training the PBMT system we used the open source Moses

[4]https://github.com/rsennrich/nematus

toolkit (Koehn et al., 2007). The word alignment models were trained with FastAlign (Dyer et al., 2013). We trained the 5-gram language models with the KenLM toolkit (Heafield et al., 2013) on the target side of the pooled corpora. Feature weights were tuned with batch MIRA (Cherry and Foster, 2012) to maximize BLEU on the dev set. Details of the generic NMT system are described in Section 5.2. In Table 3, the results on the dev set are reported. Although trained on the same dataset, it is interesting to note that, the performance of the generic NMT system is by far lower than the PBMT system. A possible explanation is that the PBMT system can explicitly memorise and use translation options learned from the training data, while the NMT system generates a more implicit representation of the data. This might have a fundamental role in domain-specific and very repetitive datasets.

Similarly to (Luong and Manning, 2015), in order to improve the performance of the generic NMT system on the target domains, we separately adapted multiple instances of the generic NMT model to each specific domain (using only the corresponding training data). This is done by using the same configurations and training criteria used for the generic model, described in Section 5.2. We refer to these strong systems as *oracles*, because they exploit knowledge of the domain labels both at the training and test time. As we see in Table 3, this offline adaptation significantly improves the performance of the NMT system, resulting in translations with higher quality than the strong PBMT system. However, as mentioned earlier, this approach requires information of the target domain and assumes having sufficient amount of time and data to train domain-specific systems for each test domain.

The recently proposed approach by (Li et al., 2016) is the first attempt in this field that tries to cope with these limitations. In this paper we implement this method and compare it with our adaptive strategy (*i.e. Adaptive Baseline*), which differs in how the retrieved sentences are ranked (*i.e.* sentence-level BLEU) and in using only the top-1 retrieved pair for updating the model. As shown in Table 3, our method performs identically to the Li et al. (2016) system that uses a larger number of training samples in the case of low similarities. So, for efficiency reasons, in all our experiments we use our approach and we keep only the first best sentence pair for updating the model.

6 Unsupervised Neural MT Adaptation

In this section we discuss the dynamic setting of the hyperparameters of the learning algorithm and their impact on the performance of the system.

6.1 Model Adaptation: Learning Rate

Once the set of relevant sentence pairs is extracted, we need to update the generic model accordingly. As described in Section 3, the learning rate controls the contribution of the new information for updating the model parameters by determining the magnitude of the update steps. Deciding about the learning rate value is very important for the SGD algorithm in general, but becomes even more crucial in our scenario where we need to adjust the parameters by using only a small training set. In order to approximate the optimal value, we performed a set of experiments on our dev set in which the learning rate is gradually increased until the overall performance of the system starts to degrade (Figure 2).

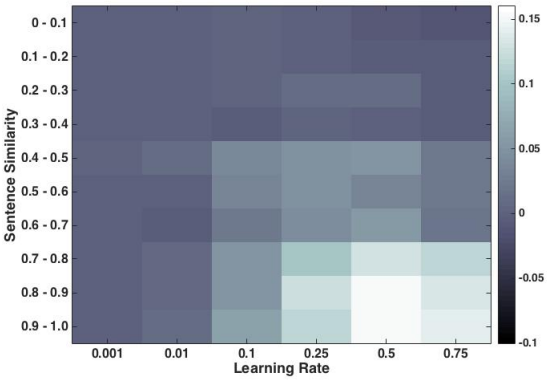

Figure 1: The effect of different learning rates on the adaptive system performance on the dev set. Darker cells show performance degradations or no gain over the *NMT generic*, while brighter cells represent larger gains over the generic system.

However, if the similarity between the retrieved sentence and the test sentence is low, by applying larger learning rates we run the risk of making drastic parameter changes in the wrong direction, which can result in lower quality translations and global performance degradations. The low results of the system when using learning rate of 0.75 empirically confirms this.

	PBMT	NMT		Li *et al.*	Adaptive		
		Generic	Oracle		Baseline	Dynamic Lrate	Dynamic Lrate-Epochs
Overall	54.6	45.7	55.9	53.2	53.2	53.6	57.5
ECB	56.1	46.3	56.3	50.9	51.1	51.0	53.7
Gnome	88.1	62.3	90.5	75.2	74.6	79.0	91.2
JRC	65.7	59.9	64.7	67.3	66.6	66.8	70.2
KDE4	50.1	45.7	54.3	50.5	50.4	50.4	53.1
OOffice	37.5	32.0	41.5	35.7	36.5	36.2	39.3
PHP	46.6	29.7	39.3	37.1	39.8	39.6	48.0
Ubuntu	50.1	47.9	47.7	49.1	49.5	51.5	53.1
UN-TM	72.8	54.4	78.4	70.0	70.3	70.3	77.6
WMT	26.7	30.5	26.8	29.0	29.7	30.6	30.3

Table 3: Comparison of the performance of generic PBMT and NMT, domain-specific oracles, and the adaptive NMT systems on the dev corpora in terms of BLEU.

To further analyze the effect of different learning rates on sentences with different levels of similarity, we measured the average performance gain of the adaptive system (over *NMT generic* in terms of sentence-level BLEU) in each similarity range when using different learning rates (Figure 1). The darker cells represent settings in which there is a drop in performance or no gain over the generic system, while brighter cells correspond to the configurations where the performance of the adaptive system is higher. We observe that updating the model with large learning rates on less relevant samples results in performance degradation (*i.e.* the darker cells in the top-right side of the figure). This suggests to apply more conservative learning rates to less relevant training samples, while increasing it to larger values for higher similarity levels. Based on this analysis, we developed a dynamic learning rate method that, for each similarity range in Figure 1, selects the learning rate that provides the largest gain over the generic NMT model (*i.e. Adaptive Dynamic-Lrate*). For instance, it uses the learning rate of 0.01 for the sentences in the similarity range of [0.0-0.1], and sets the learning rate to 0.5 for the samples with the similarity between 0.9 and 1.0. The results of this system are reported in Table 3. By comparing these results against the best performing system with fixed learning rate (*i.e. Adaptive Baseline*), we see that dynamically setting the learning rate improves the performance by +0.4 BLEU points[5] in overall (53.6 vs 53.2), which further reduces the gap between the generic and specialized NMTs.

[5]The difference is statistically significant with $p < 0.05$.

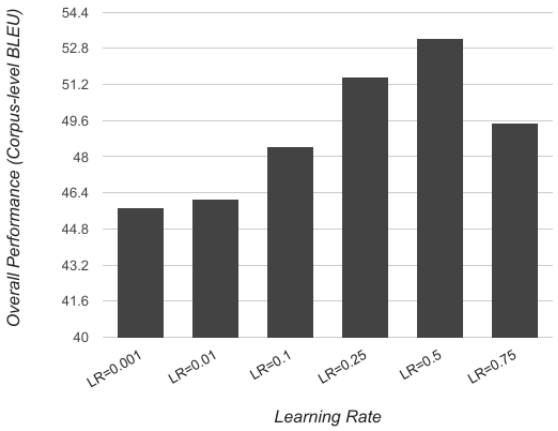

Figure 2: The effect of different learning rates on the adaptive system performance on dev data.

6.2 Model Adaptation: Number of Epochs

As described in Section 3, during training/adaptation the training samples are processed iteratively and the network parameters are updated accordingly. The number of epochs plays an important role in this process. Setting it to a large value may result in overfitting, which limits the generalization capability of the model, while performing only few epochs results in underfitting, where the system does not learn effectively from the samples. In order to analyze the effect of this factor on the final performance of the system, we run another set of experiments in which the maximum number of epochs is gradually increased until the overall results start to degrade (similar to what we did in the analysis of the learning rates in the previous section). In

	PBMT	NMT Generic	NMT Oracle	Adaptive Lrate-Epochs
Overall	54.5	44.9	55.6	**58.4**
ECB	58.6	46.5	58.0	**58.7**
Gnome	90.5	61.5	93.8	**94.9**
JRC	66.3	56.5	62.6	**69.7**
KDE4	50.6	46.4	55.7	**56.2**
OOffice	37.1	31.8	39.9	**40.3**
PHP	47.0	33.4	39.7	**50.5**
Ubuntu	45.8	45.3	46.9	**48.8**
UN-TM	69.7	52.1	75.7	**75.9**
WMT	26.0	**30.7**	26.6	**30.7**

Table 4: Comparison of the generic PBMT, NMT, and domain-specific NMT oracles against the adaptive system with dynamic learning rate and dynamic number of epochs. The reported BLEU scores are computed on the test corpora.

these experiments we used the dynamic learning rates described in Section 6.1. We observed that increasing the maximum number of epochs up to 9 helps to improve the overall performance of the system, while using larger number of updates leads in performance degradation due to the aforementioned overfitting issue.

Similarly to the experiments in Section 6.1, the relation between the number of epochs and the sentence similarity is first explored and, then, it is used to devise an approach that can automatically set the number of updates. This analysis suggests to set the number of epochs proportional to the level of similarity of the training instance.

The results of our adaptive system using dynamic learning rate and number of epochs (*i.e. Adaptive Dynamic Lrate-Epochs*) on the dev set are reported in Table 3. As the results show, dynamically deciding about the number of epochs improves the performance of the system by +3.9 BLEU scores, outperforming all our adaptive systems and the approach of (Li et al., 2016) by a large margin. More detailed analysis of the system shows that dynamically setting the number of epochs is in particular beneficial for the domains with high similarity, where it allows the system to leverage more the information of the training sample by performing more updates. In fact, the significant improvements over *Adaptive Dynamic Lrate* in the domains with high sentence similarities (*e.g.* +12.2 in case of Gnome and +7.4 in case of UN-TM) and the smaller gains in the domains with low similarities (*e.g.* +1.7 in case of Ubuntu) empirically proves this. Our investigation into the correlation of the performance gain by the adaptive system and similarity of the retrieved sentence, shows that there is a correlation of 0.9 between these two factors, further supporting our domain-wise analysis.

The results of the experiments on the test set are reported in Table 4 and show that our adaptive system outperforms the generic NMT system and both the strong PBMT system and domain-specific oracles by a large margin (+14.5, +3.9 and +2.8). This confirms that local adaptation of the generic NMT models to small set of relevant training samples can effectively improve the final performance of the system, making it a reasonable solution for the multi-domain application scenarios where maintaining several domain-specific MT engines is not feasible.

Our further analysis on the outputs of the generic and the adaptive systems reveals that the adaptive system produces sentences that are more similar to the references in terms of length (*i.e.* 14.4 vs 15.1 words per sentence, compared to 15.3 of the reference). Moreover, by analyzing the errors of the systems using TER we note that the adaptive system effectively reduces the lexical errors by 14%. This shows that our adaptive approach helps the system to produce translations that are more similar to the reference both in terms of length and structure.

7 Further Analysis

In Table 5 we present four examples taken from the dev set, for which the performance of the *Generic*, *Baseline* and *Adaptive* [6] systems is compared. The first example (from ECB) shows a case that the retrieved source sentence is highly similar to the test sentence. Comparing the outputs produced by the three systems we see that the adaptation step in the *Baseline* helps the system to produce translations that are closer to the reference but it still translates "*such management*" literally to "*cetter gestion*" which is a less fluent translation than "*celle-ci*" in this context. This shows that in this case the system could not leverage all the information provided by the retrieved pair in just one

[6] To make it short, we refer to the adaptive system with dynamic learning rate and epochs (*Adaptive Dynamic Lrate-Epochs*) as *Adaptive*.

epoch. However, by performing more epochs the *Adaptive* system is able to effectively adapt to the retrieved pair and correctly translate the given test.

The second example (from KDE4) shows another case where the retrieved pair is highly similar to the test sentence on the surface level but is different in terms of semantics. The English word "*collapse*" is translated into "*réduire*" and "*Groupe*" in French, which are semantically different. However, given the retrieved pair, both the *Baseline* and *Adaptive* system learn to translate it to "*Groupe*" which is not a correct translation in the context of the test sentence.

The next example (from JRC) presents the issue of inconsistent translations. As we see, the retrieval method is able to retrieve a sentence pair whose its source side is identical to the test sentence but is translated differently. In this case both the *Baseline* and *Adaptive* system effectively learn the information given by the retrieved pair and learn to produce the exact translation as the retrieved target (*i.e. Ret.Trg*). However, since the translation provided by the retrieved pair is different than the reference, they are eventually penalized in terms of BLEU.

The last example (from WMT) shows a case of low-similar retrieved pair and the need for a more conservative adaptation. In this case the *Baseline* replaces the French term "*une aide*" with "*des services d'*", learned aggressively from the retrieved pair, while the *Adaptive* system learns to only replace "*aide*" with "*services*", leading in a higher quality translation. However, both the systems fail to learn the correct translation of "*and*" (*i.e.* "*et de*") in this context, which is provided by the retrieved target (*i.e. Ret.Trg*).

These observations open to interesting research avenues that we plan to address in future work. These include: *i)* developing semantic-aware retrieval methods that, in addition to the surface form of the sentences, also consider their semantic similarities (KDE4 example). *ii)* handling inconsistent translations to avoid being biased towards an specific translation where other correct translations exist for the given sentence or phrase (JRC example) *iii)* developing adaptation methods that are able to leverage more efficiently the small amount of information provided by the low similar retrieved training samples (WMT example).

8 Conclusion

In this paper we investigated an instance-based adaptive Neural MT approach that effectively handles translation requests from multiple domains in an unsupervised manner, that is without knowing the domain labels. Given an input sentence, it updates a background generic model on-the-fly by means of a single training instance selected among all available training data. Differently from previous works, we enhance this approach by proposing a method to dynamically set the hyperparameters of the learning algorithm (*i.e.* learning rate and number of epochs) before updating the model. When tested in a multi-domain scenario, our approach was able to significantly outperform the generic NMT and PBMT systems and the single-domain NMT models specifically optimized on each domain.

Acknowledgments

This work has been partially supported by the EC-funded H2020 projects QT21 (grant no. 645452) and ModernMT (grant no. 645487).

References

R. Baeza-Yates and B. Ribeiro-Neto. 2011. *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison Wesley.

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

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level BLEU. In *Proceedings of the Ninth Workshop on Statistical Machine Translation, WMT@ACL 2014, June 26-27, 2014, Baltimore, Maryland, USA*. pages 362–367. http://aclweb.org/anthology/W/W14/W14-3346.pdf.

Colin Cherry and George Foster. 2012. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL HLT '12, pages 427–436. http://dl.acm.org/citation.cfm?id=2382029.2382089.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and

Src	a participating NCB may decide to abstain from such management or to pool such management with one or more other participating NCBs .
Ref	une BCN participante peut décider de ne pas participer à cette gestion ou d' assumer celle-ci en commun avec une ou plusieurs autres BCN participantes .
Ret.Src	a euro area NCB may decide to abstain from such management or to pool such management with one or more other euro area NCBs .
Ret.Trg	une BCN de la zone euro peut décider de ne pas participer à cette gestion ou d' assumer celle-ci en commun avec une ou plusieurs autres BCN de la zone euro .
Generic	une BCN participante peut décider de s' abstenir de cette gestion ou de créer une telle gestion avec une ou plusieurs autres BCN participantes .
Baseline	une BCN participante peut décider de ne pas participer à cette gestion ou d' assumer cette gestion avec une ou plusieurs autres BCN participantes .
Adaptive	une BCN participante peut décider de ne pas participer à cette gestion ou d' assumer celle-ci en commun avec une ou plusieurs autres BCN participantes .
Src	collapse all categories
Ref	réduire toutes les catégories
Ret.Src	collapse all folders
Ret.Trg	Groupe tous les dossiers
Generic	réduire toutes les catégories
Baseline	Groupe toutes les catégories
Adaptive	Groupe toutes les catégories
Src	this Decision is addressed to the Member States .
Ref	les États membres sont destinataires de la présente décision .
Ret.Src	this Decision is addressed to the Member States .
Ret.Trg	la présente décision est adressée aux États membres .
Generic	les États membres sont destinataires de la présente décision .
Baseline	la présente décision est adressée aux États membres .
Adaptive	la présente décision est adressée aux États membres .
Src	in addition , they limited the right of individuals and groups to provide assistance to voters wishing to register .
Ref	de plus , ils ont limité le droit de personnes et de groupes de fournir une assistance aux électeurs désirant s' inscrire .
Ret.Src	in addition , we provide assistance and advice to real estate investors and promoters .
Ret.Trg	enfin , nous proposons des services d' assistance et de conseil aux investisseurs et promoteurs immobiliers .
Generic	en outre , ils ont limité le droit des personnes et des groupes à fournir une aide aux électeurs souhaitant s' inscrire .
Baseline	en outre , ils ont limité le droit des personnes et des groupes à fournir des services d' assistance aux électeurs souhaitant s' inscrire .
Adaptive	en outre , ils ont limité le droit des personnes et des groupes à fournir une assistance aux électeurs souhaitant s' inscrire .

Table 5: Translation examples for comparing Generic, Baseline and the Adaptive NMT systems.

Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR* abs/1406.1078. http://arxiv.org/abs/1406.1078.

Lee Raymond Dice. 1945. Measures of the amount of ecologic association between species. *Ecology* 26(3):297–302. http://www.jstor.org/pss/1932409.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* .

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of ibm model 2. In *In Proc. NAACL*.

Matthias Eck, Stephan Vogel, and Alex Waibel. 2004. Language model adaptation for statistical machine translation based on information retrieval. In *In Proceedings of the 4th International Conference on language resources and evaluation (LREC-2004*. pages 327–330.

George Foster and Roland Kuhn. 2007. Mixture-model adaptation for SMT. In *Proceedings of the Second Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Prague, Czech Republic, pages 128–135. http://www.aclweb.org/anthology/W/W07/W07-0217.

Philip Gage. 1994. A New Algorithm for Data Compression. *C Users J.* 12(2):23–38. http://dl.acm.org/citation.cfm?id=177910.177914.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified kneser-ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Sofia, Bulgaria, pages 690–696. http://www.aclweb.org/anthology/P13-2121.

Almut Silja Hildebrand, Mattias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the translation model for statistical machine translation based on information retrieval. In *Proceedings of the 10th annual conference of the European association for machine translation*. Budapest, Hungary, page 133142.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '07, pages 177–180. http://dl.acm.org/citation.cfm?id=1557769.1557821.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Prague, Czech Republic, pages 224–227. http://www.aclweb.org/anthology/W/W07/W07-0233.

Xiaoqing Li, Jiajun Zhang, and Chengqing Zong. 2016. One sentence one model for neural machine translation. *CoRR* abs/1609.06490. http://arxiv.org/abs/1609.06490.

Minh-Thang Luong and Christopher D Manning. 2015. Stanford Neural Machine Translation Systems for Spoken Language Domains. In *Proceedings of the International Workshop on Spoken Language Translation*.

M. McCandless, E. Hatcher, and O. Gospodnetić. 2010. *Lucene in Action*. Manning Pubs Co Series. Manning. https://books.google.it/books?id=XrJBPgAACAAJ.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, Philadelphia, Pennsylvania, USA, pages 311–318. https://doi.org/10.3115/1073083.1073135.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics.

Bing Zhao, Matthias Eck, and Stephan Vogel. 2004. Language model adaptation for statistical machine translation with structured query models. In *Proceedings of the 20th International Conference on Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, COLING '04. https://doi.org/10.3115/1220355.1220414.