Using Joint Models for Domain Adaptation in Statistical Machine Translation

Nadir Durrani Hassan Sajjad Shafiq Joty Ahmed Abdelali Stephan Vogel
Qatar Computing Research Institute – Hamad Bin Khalifa University
{ndurrani,hsajjad,sjoty,aabdelali,svogel}@qf.org.qa

Abstract

Joint models have recently shown to improve the state-of-the-art in machine translation (MT). We apply EM-based mixture modeling and data selection techniques using two joint models, namely the Operation Sequence Model or OSM — an ngram-based translation and reordering model, and the Neural Network Joint Model or NNJM — a continuous space translation model, to carry out domain adaptation for MT. The diversity of the two models, OSM with inherit reordering information and NNJM with continuous space modeling makes them interesting to be explored for this task. Our contribution in this paper is fusing the existing known techniques (linear interpolation, cross-entropy) with the state-of-the-art MT models (OSM, NNJM). On a standard task of translating German-to-English and Arabic-to-English IWSLT TED talks, we observed statistically significant improvements of up to +0.9 BLEU points.

1 Introduction

Parallel data required to train Statistical Machine Translation (SMT) systems is often inadequate, and is typically collected opportunistically from wherever it is available. The conventional wisdom is that more data improves the translation quality. Additional data however, may not be best suited for tasks such as translating TED talks (Cettolo et al., 2014) or patents (Fujii et al., 2010) or educational content (Abdelali et al., 2014), and often come with the challenges of dealing with word-sense ambiguities and stylistic variance of other domains. When additional data, later referred as *out-domain*, is much larger than in-domain, the resultant distribution can get biased towards out-domain, yielding a sub-optimal system. Domain adaptation aims to preserve the identity of the in-domain data while using the best of the out-domain data. This is done by selecting a subset from the out-domain data, which is closer to the in-domain (Matsoukas et al., 2009; Moore and Lewis, 2010), or by re-weighting the probability distribution in favor of the in-domain data (Foster and Kuhn, 2007; Sennrich, 2012).

Bilingual sequence models (Mariño et al., 2006) have shown to be effective in improving the quality of machine translation and have achieved state-of-the-art performance recently (Le et al., 2012; Durrani et al., 2013; Devlin et al., 2014). Their ability to capture non-local dependencies makes them superior to the traditional phrase-based models, which do not consider contextual information across phrasal boundaries. Two such models that we explore in this paper are (*i*) the *Operation Sequence Model* or *OSM* (Durrani et al., 2011) — a markov translation model that integrates reordering, and (*ii*) the *Neural Network Joint Model* or *NNJM* (Devlin et al., 2014) — a continuous space model that learns neural network over augmented streams of source and target sequences. Both models are used as additional language model (LM) features inside the SMT decoder.

The diversity of the two models, i.e., OSM with embedded reordering information and NNJM with continuous space modeling, makes them interesting to be explored for domain adaptation. The LM-like nature of the two models provides motivations to apply methods such as perplexity optimization for model weighting and cross-entropy-based ranking for data selection. In this paper, we explore both avenues. Firstly, we train models (OSM and NNJM) from each domain separately and then interpolate them (*i*) linearly using Expectation-Maximization or EM-based weighting, (*ii*) using log-linear model inside the SMT pipeline. Secondly, we use cross-entropy difference (Moore and Lewis, 2010) between in- and out-domain models to perform data selection for domain adaptation.

The bilingual property of the OSM and NNJM models gives them an edge over traditional LM-based methods, which do not capture source and target domain relevance jointly. The embedded reordering information modeled in OSM helps it to preserve reordering characteristic of the in-domain data. Capturing reordering variation across domains have been shown to be beneficial also by Chen et al. (2013a). NNJM adds a different dimension to it by semantically generalizing the data using distributed representation of words (Bengio et al., 2003).

We evaluated our systems on a standard task of translating IWSLT TED talks for German-to-English (DE-EN) and Arabic-to-English (AR-EN) language pairs. Below is a summary of our main findings:

Model Weighting:

- Linearly interpolating OSM models through EM-based weighting gave average BLEU (Papineni et al., 2002) improvements of up to +0.6 for DE-EN and +0.9 for AR-EN.
- Log-linear variant performed better in the case of NNJM giving an average improvements of +0.4 BLEU points for DE-EN and +0.5 for AR-EN.
- Linear interpolation for NNJM models was slightly behind its log-linear variant.

Data Selection:

- OSM-based selection performed better for AR-EN task giving an average improvement of +0.7
- NNJM performed better at the DE-EN task giving an average improvement of +0.6 points.
- Both OSM- and NNJM-based selection gave slightly better results than Modified-Moore-Lewis (MML) selection (Axelrod et al., 2011).

The rest of the paper is organized as follows. Section 2 briefly describes the OSM and the NNJM models. Section 3 describes mixture model and data selection techniques that we apply using the OSM and the NNJM models to carry out adaptation. Section 4 presents the results. Section 5 discusses related work and Section 6 concludes the paper.

2 Joint Sequence Models

In this section, we revisit Operation Sequence and Neural Network Joint models briefly.

2.1 Operation Sequence Model

The Operation Sequence Model (OSM) is a bilingual model that couples translation and reordering by representing them as a sequence of operations. An operation either generates source

and/or target word(s) or performs reordering by inserting gaps and jumping forward and backward. A bilingual sentence pair (T, S) and its word-alignment A is transformed deterministically to a heterogeneous sequence of translation and reordering operations $(o_1, o_2, \ldots, o_J)$. A Markov model is then learned over these sequences:

$$P_{osm}(T, S) = P(o_1, .., o_J) \approx \prod_{j=1}^{J} P(o_j | o_{j-n+1} ... o_{j-1})$$

For example, the German-English sentence pair shown in Figure 1 can be converted into the following sequence of operations:

Generate (Wir, We) – Generate (haben, have) – Insert Gap – Generate (genommen, taken) – Jump Back (1) – Generate (sie, them) – Generate (aus, out) – Generate (ihrer , of their) – Generate (ursprünglichen, natural) – Generate (Pyramids, pyramid)

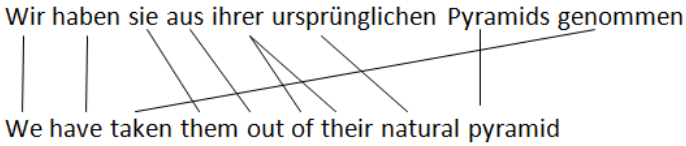

Figure 1: Sample German-English Sentence with Alignments

The generation is carried out in the order of target (English in this case). Gaps and jumps are inserted on the source side. Unaligned source and target words are handled through **Generate Source Only** and **Generate Target Only** operations, respectively. Discontinuous source and target units are handled through other operations; see Durrani et al. (2011) for details about the operations and the algorithm to convert a word-aligned corpus into sequences of operations.

Mixing lexical generation and reordering, each (translation or reordering) decision conditions on $n-1$ previous (translation or reordering) decisions. This allows the model to learn very rich translation and reordering patterns. Moreover, the model is based on minimal translation units (MTUs) and considers source and target contextual information across phrasal boundaries, thus addressing phrasal independence assumption and spurious segmentation problems in traditional phrase-based MT.

2.2 Neural Network Joint Model

In recent years, there has been a great deal of effort dedicated to neural networks (NNs) and word embeddings with applications to MT and other areas in NLP (Bengio et al., 2003; Auli et al., 2013; Kalchbrenner and Blunsom, 2013; Gao et al., 2014; Schwenk, 2012; Collobert et al., 2011; Mikolov et al., 2013; Socher et al., 2013; Hinton et al., 2012). A bilingual Neural Network Joint model for MT was recently proposed by Devlin et al. (2014). It learns a feed-forward neural network from augmented streams of source and target sequences. For a bilingual sentence pair (S, T), NNJM defines a conditional probability distribution:

$$P(T|S) \approx \prod_{i=1}^{|T|} P(t_i | t_{i-1} \ldots t_{i-n+1}, \mathbf{s_i})$$

where, $\mathbf{s_i}$ is an m-word source window for a target word t_i based on the one-to-one alignment between T and S. Each input word in the context has a D dimensional (continuous-valued)

vector representation in the shared look-up table $L \in \mathbb{R}^{|V_i| \times D}$, where V_i is the input vocabulary. The context of the sequence is represented by a concatenated vector $\mathbf{x_n} \in \mathbb{R}^{(m+n-1)D}$, which is then passed through non-linear hidden layers to learn a high-level representation. The output layer is a `softmax` over the output vocabulary V_o:

$$P(y_n = k|\mathbf{x_n}, \theta) = \frac{exp\ (\mathbf{w}_k^T \phi(\mathbf{x_n}))}{\sum_{m=1}^{|V_o|} exp\ (\mathbf{w}_m^T \phi(\mathbf{x_n}))}$$

where $\phi(\mathbf{x_n})$ defines the non-linear transformations of $\mathbf{x_n}$, and $\mathbf{w}_k$ are the weights from the outermost hidden layer to the output layer. By setting m and n to be sufficiently large, NNJM can capture long-range cross-lingual dependencies between words.

3 Domain Adaptation

The ability to learn rich lexical and reordering patterns by OSM, the generalization power of NNJM, and their strong empirical results in MT gives us a strong motivation to use them for the problem of domain adaptation. However, the OSM and NNJM models trained on a plain concatenation of in-domain data with large and diverse multi-domain data are suboptimal. When other domains are sufficiently larger and/or different than the in-domain, the probability distribution can skew away from the target domain resulting in poor performance. The goal in domain adaptation is to do restrict this drift while still using the best of the available data.

We analyze the operation corpus as generated by the corpus conversion algorithm of Durrani et al. (2011) in OSM training. It provides useful insights on the amount of reordering, number of (source word) insertions and (target word) deletions that are carried out in the bilingual corpus. We use this information to motivate our study. Table 1 shows some statistics about the operations in several datasets. We report probabilities of Jumps (**Jump Forward** and **Jump Back (*)** operations), Gaps (**Insert Gap** operation), Insertions of source words (**Generate Source Only (X)** operation to handle unaligned source words) and Deletions of target words (**Generate Target Only (Y)** operation to handle unaligned target words) in each domain.

Domain	Jumps	Gaps	Deletions	Insertions
German-to-English				
iwslt	0.17	0.09	0.06	0.04
news	0.21	0.13	0.05	0.07
europarl	0.22	0.14	0.07	0.06
common crawl	0.19	0.11	0.12	0.11
Arabic-to-English				
iwslt	0.17	0.09	0.07	0.05
UN	0.21	0.12	0.07	0.08

Table 1: Probabilities of Jumps, Gaps, Insertion and Deletion operations in each domain.

The probabilities of Jumps and Gaps in the in-domain IWSLT data are lower than other domains in both German-to-English and Arabic-to-English language pairs. This indicates that lesser amount of reordering is required in the in-domain data. Because other domains are significantly larger than the in-domain data, the resulting distribution would get biased towards doing more reordering than desired. For example **Insert Gap** operation in Europral and UN data is much probable than IWSLT (compare column **Gaps** in Table 1). Similarly the probability of insertions carried out in the in-domain data is less than the other domains. Therefore, the resulting models

would favor more insertions than preferred by the in-domain data. Table 1 does not show statistics on different vocabularies, but lexical variance between domains is obviously another cause of divergence from the in-domain data, which previous methods have also tackled. In this work, we additionally address the reordering variance across domains. These statistics, although, collected from the operation corpus on which the OSM model is trained, can be reflected on the NNJM training as well which uses same word-alignments to generate the stream of source and target n-grams.

In this paper we study two directions to perform domain adaptation in MT. We apply mixture modeling, a well-established model weighting technique, to re-weight the models in favor of the in-domain data. More specifically, we first train OSM and NNJM models on different domains and then use an EM-based interpolation to optimize the weights based on an in-domain tuning set. We also use the two models to rank sequences for data selection using *cross entropy difference*. In the next two subsections we discuss these in detail.

3.1 Model Weighting

We use both OSM and NNJM models as an additional language model feature inside the decoder. A domain-adapted version of the model, biased towards the in-domain data, can help assigning higher scores to the hypotheses that represent lexical choices and reordering patterns preferred by the in-domain data. We train OSM and NNJM models from each domain separately and learn the relative weights of the models using linear and log-linear interpolation methods. For linear interpolation, we compute weights by optimizing perplexity on in-domain tuning set[1] using a standard EM-based algorithm as described below:

Model Weighting by EM: Let $\theta_d \in \{\theta_1, \dots \theta_D\}$ represent a model (e.g., OSM, NNJM) trained on domain d, where D is the total number of domains. The probability of a sequence $\mathbf{x}_n$ can be written as a mixture of D probability densities, each coming from a different model:

$$P(\mathbf{x}_n|\theta, \lambda) = \sum_{d=1}^{D} P(\mathbf{x}_n|z_n = d, \theta_d)\ \lambda_d$$

where $P(\mathbf{x}_n|z_n = d, \theta_d)$ represents the probability of $\mathbf{x}_n$ assigned by model θ_d, and the mixture weights λ_d satisfy $0 \leq \lambda_d \leq 1$ and $\sum_{d=1}^{D} \lambda_d = 1$. In our setting, $\theta = \{\theta_1, \dots \theta_D\}$ is known, and we can use EM to learn the mixture weights. The expected complete data log likelihood is given by:

$$E[L(\lambda)] = \sum_{n=1}^{N} \sum_{d=1}^{D} r_{nd}\ log\ [P(\mathbf{x}_n|z_n = d, \theta_d)\lambda_d]$$

where $r_{nd} = P(z_n = d|\mathbf{x}_n, \theta_d, \lambda_d^{t-1})$ is the responsibility that domain d takes for data point n given the mixing weight in the previous step λ_d^{t-1}. In the E-step, we compute r_{nd} and we update λ in the M-step. More specifically:

E-step: Compute $r_{nd}^t = \frac{\lambda_d^{t-1} P(\mathbf{x}_n|z_n=d,\theta_d)}{\sum_{d'=1}^{D} \lambda_{d'}^{t-1} P(\mathbf{x}_n|z_n=d,\theta_{d'})}$

M-step: Update $\lambda_d^t = \frac{1}{N} \sum_{n=1}^{N} r_{nd}^t$

Once we have learned the relative weights of the models based on the in-domain tuning data, we can linearly interpolate the models as:

[1]The tuning-set is required to be word-aligned and then converted into a sequence of operations (for OSM) and augmented streams of source and target strings (for NNJM) to compute model-wise perplexities.

$$P_{osm}(T,S) \approx \prod_{j=1}^{J} \sum_{d} \lambda_d P(o_j | o_{j-n+1} ... o_{j-1}, \theta_d)$$

$$P_{nnjm}(T|S) \approx \prod_{i=1}^{|T|} \sum_{d} \lambda_d P(t_i | t_{i-1} \dots t_{i-n+1}, \mathbf{s_i}, \theta_d)$$

An alternative way to combine the models is through log-linear interpolation by optimizing weights, directly on BLEU, along with other features inside of the SMT pipeline.

3.2 Data Selection

An alternative to model weighting is data selection, which attempts to filter out harmful data from the training corpus rather than down weighting it. Data selection could be useful in a scenario with memory constraints. However, a down-side of this approach is that it requires extensive amount of experimentation to find an optimal cut-off point.

In this paper, we select data using differences in cross entropy as proposed by Moore and Lewis (2010). More specifically, we first train a model (OSM or NNJM) on the in-domain corpus, and then train another model on the out-domain data of equal size. Then we score the out-domain data using:

$$score(x) = H_I(x) - H_O(x)$$

where x is a sequence of operations $(o_1, \dots, o_n)$ in the case of OSM and an augmented stream of source and target sequences $(t_1, \dots, t_n, \mathbf{s}_i)$ in the case of NNJM. H_D is the cross-entropy between a model and the empirical n-gram distribution in the domain D. We train a 5-gram OSM and a 14-gram NNJM with 5-grams on target-side and 4-grams on each side of the source word that is aligned with the target word t_i. The bilingual characteristic of the models makes it comparable to the MML method which trains source- and target-side language models from in- and out-domains separately and take a sum of cross-entropy differences over each side of the corpus:

$$score(s,t) = [H_{I-src}(s) - H_{O-src}(s)] + [H_{I-tgt}(t) - H_{O-tgt}(t)]$$

where s and t are sequences of source and target strings respectively. Out-domain models are trained by randomly selecting corpora of same size as that of the in-domain data.

4 Experiments

Data: We used TED talks (Cettolo et al., 2014) as our in-domain corpus. For German-to-English (DE-EN), we used the data made available for WMT'14.[2] This contains News, Europarl and Common Crawl as out-domain data. For Arabic-English (AR-EN), we used the UN corpus as out-domain data. We concatenated dev- and test-2010 for tuning and used test2011-2013 for evaluation. Table 2 shows the size of the training and test data used.

NNJM Settings: The NNJM models were trained using NPLM[3] toolkit (Vaswani et al., 2013) with the following settings. We used a target context of 5 words and an aligned source window of 9 words, forming a joint stream of 14-grams for training. We restricted source and target side vocabularies to 20K and 40K most frequent words. We used an input embedding layer of 150

[2] http://www.statmt.org/wmt14/

[3] http://nlg.isi.edu/software/nplm/

German-English				Arabic-English			
Corpus	Sent.	Tok$_{DE}$	Tok$_{EN}$	Corpus	Sent.	Tok$_{AR}$	Tok$_{EN}$
iwslt	177K	3.3M	3.5M	iwslt	186K	2.7M	1.8M
news	200K	5.1M	5.0M	un	3.7M	12.4M	12.3M
ep	1.9M	48.7M	51.0M	-	-	-	-
cc	2.3M	53.9M	57.5M	-	-	-	-
Test Set	Sent.	Tok$_{DE}$	Tok$_{EN}$	Corpus	Sent.	Tok$_{AR}$	Tok$_{EN}$
tune	2452	42K	44K	tune	2456	48K	52K
test-11	1433	22K	23K	test-11	1199	21K	24K
test-12	1700	25K	26K	test-12	1702	30K	32K
test-13	1363	19K	20K	test-13	1169	26K	28K

Table 2: Statistics of the German-English and Arabic-English training corpora in terms of Sentences and Tokens (Source/Target). Tokens are represented in Millions. ep = Europarl, cc = Common Crawl, un = United Nations

and an output embedding layer of 750. Only one hidden layer is used with NCE[4] to allow faster training and decoding. Training was done using mini-batch size of 1000 and using 100 noise samples. We train the out-domain NNJM models using the same vocabulary as the in-domain vocabulary. All models were trained for 25 epochs.

Machine Translation Settings: We followed Birch et al. (2014) to train a Moses system Koehn et al. (2007) with the following settings: maximum sentence length of 80, Fast-Align (Dyer et al., 2013) for word-alignments, an interpolated Kneser-Ney smoothed 5-gram language model (Schwenk and Koehn, 2008) with KenLM (Heafield, 2011) for querying, lexicalized reordering (Galley and Manning, 2008) and other default parameters. We used Moses implementations of OSM and NNJM as a part of their respective baseline systems. Arabic OOVs were translated using an unsupervised transliteration module (Durrani et al., 2014b) in Moses. We used k-best batch MIRA (Cherry and Foster, 2012) for tuning.[5]

4.1 Results: Model Weighting

We first discuss the results of applying mixture modeling approach. The MT systems are trained on a concatenation of all in- and out-domain data. The OSM and NNJM models used in baseline MT systems were also trained on the concatenated data.

Linear interpolation (OSM$_{ln}$) based on EM-weighting shows significant improvements with average BLEU gains of +0.6 in DE-EN and +0.9 in AR-EN over the baseline system B$_{cat}$ (see Table 3).[6] One reason for better gains in AR-EN is the fact that the out-domain UN data

[4]Training NNJM with backpropagation could be prohibitively slow because for each training instance, the softmax layer requires a summation over the entire output vocabulary One way to avoid this repetitive computation is to use a Noise Contrastive Estimation or NCE (Gutmann and Hyvärinen, 2010) of the loss function. NCE has been recently used in neural language models (Vaswani et al., 2013; Mnih and Teh, 2012).

[5]All systems were tuned three times.

[6]We carried out additional experiments by linearly interpolating class-based OSM models Durrani et al. (2014a). We used the `mkcls` utility in GIZA to cluster source and target vocabularies into 50 classes. Class-based OSM models were trained on each domain and interpolated in the same way as we did for the word forms. This however, did not yield any significant improvements on top of what was already achieved from the interpolation of word-based OSM. We also tried interpolating POS, morph and lemma-based OSM-models but did not gain any further improvement. Results are ommitted from the paper.

OSM Interpolation (German-English)				
System	test11	test12	test13	Avg.
B_{cat}	35.8	31.1	27.6	31.5
OSM_{ln}	36.6 **+0.8**	31.9 **+0.8**	27.7 +0.1	32.1 **+0.6**
OSM_{lg}	35.4 -0.4	31.1 $\pm$ 0.0	27.4 -0.2	31.3 -0.2
OSM Interpolation (Arabic-English)				
B_{cat}	26.4	29.2	29.9	28.5
OSM_{ln}	27.3 **+0.9**	30.0 **+0.8**	30.8 **+0.9**	29.4 **+0.9**
OSM_{lg}	25.8 -0.6	28.7 -0.5	29.4 -0.5	28.0 -0.5

Table 3: OSM Interpolation OSM_{ln} = Linear, OSM_{lg} = Log-linear

NNJM Interpolation (German-English)				
System	test11	test12	test13	Avg.
B_{cat}	35.6	31.3	27.4	31.4
$NNJM_{ln}$	36.2 **+0.6**	31.8 **+0.5**	27.1 -0.3	31.7 +0.3
$NNJM_{lg}$	36.1 **+0.5**	32.1 **+0.8**	27.2 -0.2	31.8 **+0.4**
NNJM Interpolation (Arabic-English)				
B_{cat}	26.6	29.4	30.1	28.7
$NNJM_{ln}$	26.7 +0.1	30.2 **+0.8**	30.3 +0.2	29.1 **+0.4**
$NNJM_{lg}$	26.8 +0.2	30.2 **+0.8**	30.5 **+0.4**	29.2 **+0.5**

Table 4: NNJM Interpolation $NNJM_{ln}$ = Linear, $NNJM_{lg}$ = Log-linear

is much harmful for the task at hand. On the contrary additional data in DE-EN is helpful (see also the results in next section for more information). Log-linear interpolation of OSM models (OSM_{lg}) performs much worse than B_{cat} in both language pairs. In the log-linear model, all sub-models are queried separately. An operation sequence from the out-domain data that is unknown to the in-domain OSM, gets high probability[7] and is ranked higher in the search space. On the contrary, the same gets down-weighted in a linearly interpolated global model.

Both linear and log-linear interpolation of the NNJM models showed improvements over the baseline system B_{cat} (refer to Table 4). Log-linear interpolation ($NNJM_{lg}$) performed slightly better in both cases. Notice that $NNJM_{lg}$ does not face the same problem as OSM_{lg} because all NNJM models are trained using the in-domain vocabulary with a low probability assigned to the out-domain UNKs.[8] See Joty et al. (2015) for more details on our novel handling

[7] Due to probability mass assigned to UNK sequences.

[8] In order to reduce the training time and to learn better word representations, neural models are trained on most frequent vocabulary words only and low frequency words are represented under a class of unknown words, `unk`. This results in a large number of n-gram sequences containing at least one `unk` word and thereby, makes `unk` a highly probable word for the model. As a result of this discrepancy, sentences with more number of `unk` words will be selected. To solve this problem we created a separate class for out-domain $\texttt{unk}_o$ words. We train the in-domain model by adding a few dummy sequences containing $\texttt{unk}_o$ occurring on both source and target sides ensuring that out-domain unknown words get minimal probabilities.

%age	German-English			Arabic-English		
	MML	**OSM**	**NNJM**	**MML**	**OSM**	**NNJM**
0%	35.4	35.4	35.4	27.2	27.2	27.2
5%	36.0	36.0	36.2	27.6	27.7	27.6
10%	36.2	36.3	36.5	26.9	27.3	27.1
20%	36.4	36.8	36.9	26.8	27.0	27.0
40%	36.3	36.6	36.7	26.6	26.8	26.6
100%	35.6	35.6	35.6	26.6	26.6	26.6

Table 5: MML, OSM and NNJM-based data selection, evaluated using test2011

Data Selection (German-English)				
System	test11	test12	test13	Avg.
$B_{100\%}$	35.8	31.1	27.6	31.5
$B_{0\%}$	35.4	31.3	25.5	30.7
$MML_{20\%}$	36.4 **+0.6**	31.4 +0.3	27.7 +0.1	31.8 +0.3
$OSM_{20\%}$	36.8 **+1.0**	31.5 **+0.4**	27.7 +0.1	32.0 **+0.5**
$NNJM_{20\%}$	36.9 **+1.1**	31.6 **+0.5**	27.7 +0.1	32.1 **+0.6**
Data Selection (Arabic-English)				
$B_{100\%}$	26.4	29.2	29.9	28.5
$B_{0\%}$	27.2	30.0	30.2	29.1
$MML_{5\%}$	27.6 **+0.4**	30.5 **+0.5**	31.0 **+0.8**	29.7 **+0.6**
$OSM_{5\%}$	27.7 **+0.5**	30.6 **+0.6**	31.0 **+0.8**	29.8 **+0.7**
$NNJM_{5\%}$	27.6 **+0.4**	30.5 **+0.5**	31.1 **+0.9**	29.7 **+0.6**

Table 6: Data Selection

of UNK words in the NNJM model.

4.2 Results: Data Selection

We selected 0%, 2.5%, 5%, 10%, 20%, 40% and 100% out-domain data and evaluated on test2011 to select the best percentage. See Table 5 for results on each selected percentage. Table 6 shows that the out-domain data is helpful in the case of DE-EN and harmful in the case of AR-EN; compare $B_{100\%}$ (all data) versus $B_{0\%}$ (in-domain data only). MML-selection improves the baseline by +0.3 and +0.6 in case of DE-EN and AR-EN respectively. OSM and NNJM-based selection gave similar improvements with slightly better results than MML. We found that the amount of overlap in data selected by the three models is roughly 63% in DE-EN and 71% in AR-EN.

5 Related Work

Previous work on domain adaptation in MT can be broken down broadly into two main categories namely *data selection* and *model adaptation*.

5.1 Data Selection

Data selection has shown to be an effective way to discard poor quality or irrelevant training instances, which when included in the MT systems, hurts its performance. The idea is to score the out-domain data using model trained from the in-domain data and apply a cut-off based on the resulting scores. The MT system can then be trained on a subset of the out-domain data that is closer to in-domain. Selection based methods can be helpful to reduce computational cost when training is expensive and also when memory is constrained. Data selection was earlier done for language modeling using information retrieval techniques (Hildebrand et al., 2005) and using perplexity measure (Moore and Lewis, 2010). Axelrod et al. (2011) further extended the work of Moore and Lewis (2010) to translation model adaptation by using both source side and target side language models. Duh et al. (2013) used recurrent neural network language model instead of an ngram-based language model to do the same. Translation model features were used recently by Liu et al. (2014); Hoang and Sima'an (2014) to do data selection.

5.2 Model Adaptation

The downside of data selection is that finding an optimal cut-off threshold is a time consuming process. Therefore rather than filtering less useful data, an alternative way is to down-weight it and boost the data closer to the in-domain. It is robust than selection since it takes advantage of the complete out-domain data with intelligent weighting towards the in-domain. Matsoukas et al. (2009) proposed a classification-based sentence weighting method for adaptation. Foster et al. (2010) extended this by weighting phrases rather than sentence pairs. Other researchers have carried out weighting by merging phrase-tables through linear interpolation (Finch and Sumita, 2008; Nakov and Ng, 2009) or log-linear combination (Foster and Kuhn, 2009; Bisazza et al., 2011; Sennrich, 2012) and through phrase training based adaptation (Mansour and Ney, 2013). Chen et al. (2013b) used vector space model for adaptation at phrase level. Every phrase pair is represented as a vector where every entry in the vector reflects its relatedness with each domain. Chen et al. (2013a) also applied mixture model adaptation for reordering model. Joty et al. (2015) performed model weighting by regularizing the loss function towards the in-domain model directly inside neural network training. They also used NNJM model as their basis.

Other work on domain adaptation includes but not limited to studies that focus on topic modeling (Eidelman et al., 2012; Hasler et al., 2014), dynamic adaptation where no in-domain data is available (Sennrich et al., 2013; Mathur et al., 2014) and sense disambiguation (Carpuat et al., 2013).

6 Conclusion

We targeted an unexplored area of using bilingual language models for domain adaptation. We applied model weighting and data selection techniques using OSM and NNJM models. Both methods were shown to be effective in the target translation tasks. Interpolating multi-domain models gave an average improvement of up to +0.9 BLEU points using OSM and +0.5 using NNJM. We also used NNJM and OSM models for data selection using differences in cross entropy and showed improvements of up to +0.6 BLEU points. The code will be contributed to Moses git repository.

References

Abdelali, A., Guzman, F., Sajjad, H., and Vogel, S. (2014). The AMARA corpus: Building parallel language resources for the educational domain. In *LREC'14*, Reykjavik, Iceland.

Auli, M., Galley, M., Quirk, C., and Zweig, G. (2013). Joint language and translation modeling

with recurrent neural networks. In *EMNLP-2013*, pages 1044–1054, Seattle, Washington, USA. ACL.

Axelrod, A., He, X., and Gao, J. (2011). Domain adaptation via pseudo in-domain data selection. EMNLP '11, pages 355–362, Edinburgh, UK. ACL.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Birch, A., Huck, M., Durrani, N., Bogoychev, N., and Koehn, P. (2014). Edinburgh SLT and MT system description for the IWSLT 2014 evaluation. In *Proceedings of the 11th International Workshop on Spoken Language Translation*, IWSLT '14, Lake Tahoe, CA, USA.

Bisazza, A., Ruiz, N., and Federico, M. (2011). Fill-up versus interpolation methods for phrase-based SMT adaptation. In Federico, M., Hwang, M.-Y., Rödder, M., and Stüker, S., editors, *IWSLT*, pages 136–143.

Carpuat, M., Daume III, H., Henry, K., Irvine, A., Jagarlamudi, J., and Rudinger, R. (2013). Sensespotting: Never let your parallel data tie you to an old domain. In *ACL*, pages 1435–1445, Sofia, Bulgaria.

Cettolo, M., Niehues, J., Stüker, S., Bentivogli, L., and Federico, M. (2014). Report on the 11th IWSLT Evaluation Campaign. *Proceedings of the International Workshop on Spoken Language Translation, Lake Tahoe, US*.

Chen, B., Foster, G., and Kuhn, R. (2013a). Adaptation of reordering models for statistical machine translation. NAACL-HLT '13, pages 938–946, Atlanta, Georgia.

Chen, B., Kuhn, R., and Foster, G. (2013b). Vector space model for adaptation in statistical machine translation. In *ACL (Volume 1: Long Papers)*, Sofia, Bulgaria.

Cherry, C. and Foster, G. (2012). Batch tuning strategies for statistical machine translation. NAACL-HLT '12, Montréal, Canada.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. volume 12, pages 2493–2537. JMLR. org.

Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R., and Makhoul, J. (2014). Fast and robust neural network joint models for statistical machine translation. In *ACL (Volume 1: Long Papers)*, pages 1370–1380.

Duh, K., Neubig, Graham, S. K., and Tsukada, H. (2013). Adaptation data selection using neural language models: Experiments in machine translation. In *ACL (Volume 2: Short Papers)*, Sofia, Bulgaria.

Durrani, N., Fraser, A., Schmid, H., Hoang, H., and Koehn, P. (2013). Can Markov Models Over Minimal Translation Units Help Phrase-Based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 399–405, Sofia, Bulgaria. Association for Computational Linguistics.

Durrani, N., Koehn, P., Schmid, H., and Fraser, A. (2014a). Investigating the Usefulness of Generalized Word Representations in SMT. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 421–432, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Durrani, N., Sajjad, H., Hoang, H., and Koehn, P. (2014b). Integrating an Unsupervised Transliteration Model into Statistical Machine Translation. In *Proceedings of the 15th Conference of the European Chapter of the ACL (EACL 2014)*, Gothenburg, Sweden.

Durrani, N., Schmid, H., and Fraser, A. (2011). A Joint Sequence Translation Model with Integrated Reordering. In *Proceedings of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT'11)*, Portland, OR, USA.

Dyer, C., Chahuneau, V., and Smith, N. A. (2013). A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of NAACL'13*.

Eidelman, V., Boyd-Graber, J., and Resnik, P. (2012). Topic models for dynamic translation model adaptation. ACL '12, pages 115–119, Jeju Island, Korea. ACL.

Finch, A. and Sumita, E. (2008). Dynamic model interpolation for statistical machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, Columbus, Ohio.

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. ACL.

Foster, G. and Kuhn, R. (2007). Mixture-model adaptation for smt. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07.

Foster, G. and Kuhn, R. (2009). Stabilizing minimum error rate training. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, StatMT '09, Athens, Greece.

Fujii, A., Utiyama, M., Yamamoto, M., and Utsuro, T. (2010). Overview of the patent translation task at the ntcir-8 workshop. In *In Proceedings of the 8th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-lingual Information Access*, pages 293–302.

Galley, M. and Manning, C. D. (2008). A Simple and Effective Hierarchical Phrase Reordering Model. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 848–856, Honolulu, Hawaii.

Gao, J., He, X., Yih, W.-t., and Deng, L. (2014). Learning continuous phrase representations for translation modeling. In *ACL (Volume 1: Long Papers)*, pages 699–709, Baltimore, Maryland. ACL.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Teh, Y. and Titterington, M., editors, *Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, volume 9 of *JMLR W&CP*, pages 297–304.

Hasler, E., Blunsom, P., Koehn, P., and Haddow, B. (2014). Dynamic topic adaptation for phrase-based mt. EACL, pages 328–337, Gothenburg, Sweden.

Heafield, K. (2011). KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom.

Hildebrand, A. S., Eck, M., Vogel, S., and Waibel, A. (2005). Adaptation of the translation model for statistical machine translation based on information retrieval. In *EAMT*, Budapest.

Hinton, G., Deng, L., Yu, D., Dahl, G., rahman Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*.

Hoang, C. and Sima'an, K. (2014). Latent domain translation models in mix-of-domains haystack. COLING: Technical Papers, Dublin, Ireland.

Joty, S., Sajjad, H., Durrani, N., Al-Mannai, K., Abdelali, A., and Vogel, S. (2015). How to Avoid Unwanted Pregnancies: Domain Adaptation using Neural Network Models. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, Portugal.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *EMNLP-2013*, Seattle, Washington, USA.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. (ACL'07), Prague, Czech Republic.

Le, H.-S., Allauzen, A., and Yvon, F. (2012). Continuous space translation models with neural networks. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–48, Montréal, Canada. Association for Computational Linguistics.

Liu, L., Hong, Y., Liu, H., Wang, X., and Yao, J. (2014). Effective selection of translation model training data. In *ACL (Volume 2: Short Papers)*, pages 569–573, Baltimore, Maryland. ACL.

Mansour, S. and Ney, H. (2013). Phrase training based adaptation for statistical machine translation. NAACL-HLT '13, Atlanta, Georgia.

Mariño, J. B., Banchs, R. E., Crego, J. M., de Gispert, A., Lambert, P., Fonollosa, J. A. R., and Costa-jussà, M. R. (2006). N-gram-Based Machine Translation. *Computational Linguistics*, 32(4):527–549.

Mathur, P., Venkatapathy, S., and Cancedda, N. (2014). Fast domain adaptation of smt models without in-domain parallel data. COLING 2014: Technical Papers, pages 1114–1123, Dublin, Ireland.

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: Volume 2*, EMNLP '09.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space.

Mnih, A. and Teh, Y. W. (2012). A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the International Conference on Machine Learning*.

Moore, R. C. and Lewis, W. (2010). Intelligent selection of language model training data. (ACL'10), Uppsala, Sweden.

Nakov, P. and Ng, H. T. (2009). Improved statistical machine translation for resource-poor languages using related resource-rich languages. In *EMNLP'09*, Singapore.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. (ACL'02), Philadelphia, PA, USA.

Schwenk, H. (2012). Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of COLING 2012: Posters*, pages 1071–1080, Mumbai, India.

Schwenk, H. and Koehn, P. (2008). Large and Diverse Language Models for Statistical Machine Translation. In *International Joint Conference on Natural Language Processing*, pages 661–666.

Sennrich, R. (2012). Perplexity minimization for translation model domain adaptation in statistical machine translation. EACL, Avignon, France.

Sennrich, R., Schwenk, H., and Aransa, W. (2013). A multi-domain translation model framework for statistical machine translation. In *ACL (Volume 1: Long Papers)*, pages 832–840, Sofia, Bulgaria. ACL.

Socher, R., Bauer, J., Manning, C. D., and Andrew Y., N. (2013). Parsing with compositional vector grammars. In *ACL (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria.

Vaswani, A., Zhao, Y., Fossum, V., and Chiang, D. (2013). Decoding with large-scale neural language models improves translation. In *EMNLP-2013*, pages 1387–1392.