# A Comparison of Mixture and Vector Space Techniques for Translation Model Adaptation

**Boxing Chen** Boxing.Chen@nrc-cnrc.gc.ca
**Roland Kuhn** Roland.Kuhn@nrc-cnrc.gc.ca
**George Foster** George.Foster@nrc-cnrc.gc.ca
National Research Council Canada, Ottawa, Canada

**Abstract**

In this paper, we propose two extensions to the vector space model (VSM) adaptation technique (Chen et al., 2013b) for statistical machine translation (SMT), both of which result in significant improvements. We also systematically compare the VSM techniques to three mixture model adaptation techniques: linear mixture, log-linear mixture (Foster and Kuhn, 2007), and provenance features (Chiang et al., 2011). Experiments on NIST Chinese-to-English and Arabic-to-English tasks show that all methods achieve significant improvement over a competitive non-adaptive baseline. Except for the original VSM adaptation method, all methods yield improvements in the +1.7-2.0 BLEU range. Combining them gives further significant improvements of up to +2.6-3.3 BLEU over the baseline.

## 1 Introduction

The translation of a source-language expression to a target language might differ across genres, topics, national origins, and dialects, or the author's or publication's style. The word "domain" is often used to indicate a particular combination of all these factors (Chen et al., 2013b). Statistical machine translation (SMT) systems are trained on bilingual parallel and monolingual data. The training data vary across domains, and translations across domains are unreliable. Therefore, we can often get better performance by adapting the SMT system to the test domain.

Domain adaptation (DA) techniques for SMT systems have been widely studied. Approaches that have been tried for SMT model adaptation include self-training, data selection, data weighting, context-based DA, and topic-based DA. etc. We will review these techniques in the next Section. Among all these approaches, data weighting has received most attention, it assigns each data item a weight according to its closeness to the in-domain data. Mixture model and vector space model adaptation (Chen et al., 2013b) are two data weighting techniques. Both of them assume the existence of $N$ bilingual sub-corpora as training data, and have weights tuned on an in-domain development corpus (dev).

Mixture model adaptation assigns weights at corpus-level; sub-models trained on different domain data sets are combined linearly or log-linearly (Foster and Kuhn, 2007). Provenance features (Chiang et al., 2011) compute a separate set of lexical weights for each sub-corpus, and then log-linearly combined. While vector space model adaptation (Chen et al., 2013b) assigns weights at phrase-level; it weights each phrase pair in the training data with the similarity score between the phrase pair and the in-domain dev data.

In this paper, we first propose two extensions to the original vector space model adaptation, thereby invent two new data weighting adaptation methods based on the vector space model. The first one is grouped VSM adaptation, which classifies the in-domain data to several groups, such as general-domain, domain-specific phrase pair, then assigns several weights to the phrase pair with the similarity scores between the phrase pair and the dev data sub-sets. The second one is called distributional VSM adaptation, which directly uses the phrase pair's distribution across sub-corpora as decoding features. The two methods both significantly improve the SMT performance over the original VSM adaptation.

The second contribution of this paper is a systematic comparison of two groups of DA methods under the most common scenario for SMT: that the training material is heterogeneous, and an in-domain development set is available. The first group of adaptation technique is mixture model, includes linear mixtures, log-linear mixtures, provenance features; the other group of adaptation technique is vector space model, includes original vector space model (VSM), and the newly proposed grouped VSM and distributional adaptation. The mixture model and vector space adaptation techniques both rely on information about the subcorpora from which the data originate. However, a key difference is that vector space model methods capture each phrase pair's distribution across subcorpora, while in mixture models, a phrase pair's distribution is the prevalence of the pair within each subcorpus. Given this difference, it is interesting to have an systematic comparison between them.

Another small but nice contribution is that inspired from (Chiang et al., 2011) and (George Foster and Kuhn, 2013), we use a simple form of smoothing for log-linear mixture adaptation, which significantly improves performance over the non-smoothed log-linear mixtures.

Experiments on NIST Chinese-to-English and Arabic-to-English tasks show that 1) each technique yields significant improvement over a competitive but non-adaptive baseline; 2) the largest improvements are for five of the six methods (those other than the original VSM), with improvements for these five all in the range +1.7-2.0 BLEU; 3) combining some of these techniques yields further significant improvement. The best combination yields improvements of +2.6-3.3 BLEU over the baseline.

## 2 Reviewing of SMT adaptation techniques

Most approaches to DA can be classified into one of five categories: self-training, data selection, data weighting, context-based DA, and topic-based DA. Recently, several new approaches have also been studied.

With self-training (Ueffing and Ney, 2007; Chen et al., 2008; Schwenk, 2008; Bertoldi and Federico, 2009), an MT system trained on general domain data is used to translate in-domain monolingual data. The resulting target sentences or bilingual sentence pairs are then used as additional training data.

Data selection approaches (Zhao et al., 2004; Lü et al., 2007; Moore and Lewis, 2010; Axelrod et al., 2011; Duh et al., 2013) search for monolingual or bilingual parallel sentences that are similar to the in-domain data according to some criterion, then add them to the training data.

Data weighting approaches can be seen as a generalization of data selection: instead of making a binary include vs. not-include decision about a given sentence or sentence pair, one weights each data item according to its closeness to the in-domain data. This can be applied at corpus, sentence, or phrase level. At corpus level, linear and log-linear mixture combine sub-models trained on different domain data sets linearly or log-linearly (Foster and Kuhn, 2007). Log-linear mixture DA can employ the same discriminative tuning algorithm used to combine the log-linear high-level components of a typical SMT system (the translation model, language model, reordering model, etc.), but linear mixture DA seems to work better. Thus, (Foster et al., 2010; Sennrich, 2012; Chen et al., 2013a; George Foster and Kuhn, 2013) studied linear mixture adaptation. (Koehn and Schroeder, 2007), instead, combined the sub-models via alternative paths. Provenance features (Chiang et al., 2011) compute a separate set of lexical weights for each sub-corpus, then all these lexical weights are combined log-linearly. (Razmara et al., 2012) used ensemble decoding to mix multiple translation models. (Sennrich et al., 2013; Cui et al., 2013) extended the mixture model method to multi-domain DA.

At sentence level, (Matsoukas et al., 2009) used a rich feature set to compute weights for each sentence in the training data. A sentence from a corpus whose domain is close to that of the in-domain dev set would receive a high weight. At a finer level of granularity, (Foster et al., 2010) used a rich feature set to compute phrase pair weights. Vector space model adaptation (Chen et al., 2013b) is another phrase-level data weighting approach; it weights each phrase pair with the similarity score between the in-domain dev data and each phrase pair in the training data based on vector space model in which vectors (for the entire dev data, or for each phrase pair) represent domain profiles.

The cache-based method (Tiedemann, 2010; Gong et al., 2011) is a form of context-based adaptation technique. In the tradition of (Kuhn and De Mori, 1990) it uses a cache to consider the local or document-level context when choosing translations. (Carpuat et al., 2013) employed word sense disambiguation, using local context to distinguish the translations for different domains.

Work on topic-based DA includes (Tam et al., 2007), where latent semantic analysis (LSA) models topics for SMT adaptation, (Eidelman et al., 2012; Hewavitharana et al., 2013) which employs a latent Dirichlet allocation (LDA) topic model, and (Eva Hasler and Koehn, 2012), which employs hidden topic Markov models (HTMMs), adding a sentence topic distribution as an SMT system feature.

Other DA approaches include mining translations from comparable data to translate OOVs and capturing new senses in new domains (Daume III and Jagarlamudi, 2011; Irvine et al., 2013). (Dou and Knight, 2012; Zhang and Zong, 2013) learned bilingual lexica or phrase tables from in-domain monolingual data with a decipherment method, then incorporated them into the SMT system.

## 3 Mixture model adaptation

There are several adaptation scenarios for SMT, of which the most common is 1) The training material is heterogeneous, with some parts of it that are not too far from the test domain; 2) A bilingual dev set drawn from the test domain is available. A common approach to DA for this scenario is: 1) split the training data into sub-corpora by domain; 2) train sub-models or features on each sub-corpus; 3) weight these sub-models or features via machine learning or SMT tuning algorithms.

In the rest of the paper, we assume that the training data are split into $N$ sub-corpora. We apply various adaptation techniques to a phrase-based SMT system whose translation model incorporates forward and backward phrase translation probabilities and forward and backward lexical weights. Linear and log-linear mixture techniques are applied to phrase translation probabilities, provenance features to the lexical weights.

### 3.1 Linear mixture model

Given a set of phrase tables, each trained on one of the $N$ sub-corpora and with “forward” and “backward” probabilities for phrase pairs, these sub-models can be combined linearly. Let us consider the “backward” probability $p(s|t)$ of source-language phrase $s$ being generated by target-language phrase $t$ (for all the adaptation techniques, each technique is applied symmetrically in “backward” and “forward” directions). For a set of $p_i(s|t)$, each trained on a sub-corpus $d_i$, the mixture model is

$$p(s|t) = \sum_{i=1}^{N} \alpha_i \, p_i(s|t) \tag{1}$$

To set weights $\alpha_i$, we first extract a set of phrase pairs from an in-domain dev set using standard techniques [1]. This yields a joint distribution $\tilde{p}$, which we use to define a maximum likelihood objective as in Equation 2. The weights can then be learned efficiently using the EM algorithm. This estimation approach was first proposed in (Foster et al., 2010).

$$\hat{\alpha} = \arg\max_{\alpha} \sum_{s,t} \tilde{p}_i(s,t) \log \sum_{i=1}^{N} \alpha_i \, p_i(s|t) \tag{2}$$

A problem with this approach is that large phrase tables with good phrase pair coverage get assigned high weights by EM. When such tables are out-of-domain, this will cause translation performance to suffer: the in-domain translations for which they got credit under EM will be replaced by out-of-domain alternatives to which they assign higher probability. To correct this bias in favour of large corpora, (George Foster and Kuhn, 2013) proposed sub-sampling phrase tables. Thus, before running EM, we randomly select roughly the same number of phrase pairs covered by the in-domain dev set from each phrase table to learn the weights. While, the actual interpolated phrase table uses the full data.

### 3.2 Log-linear mixture model

Like the linear mixture model, the log-linear mixture model is made up of sub-models trained on each of the sub-corpora. However, it combines the sub-models multiplicatively by directly using those sub-models as features in the SMT log-linear framework. Thus, one is tuning directly to the desired objective function (e.g., BLEU) instead of to likelihood, as EM does.

The standard implementation of log-linear mixtures has a serious disadvantage compared to linear mixtures: it performs badly when there are many sub-corpora, and the sub-models are not smoothed. Within a log-linear combination, all sub-models must agree on successful hypotheses. Linear mixtures implement a kind of "or", while log-linear models implement a kind of "and": if $p_i(s|t)$ is zero because phrase pair $(s,t)$ did not occur in sub-corpus $d_i$, $p(s|t)$ will be estimated as zero. One can assign and tune small positive probabilities for the $p_i(s|t)$ for missing phrase pairs, but this tends not to work very well. "Small" models thus need to be down-weighted heavily to avoid excessive vetoing; whatever useful information they might possess is also discarded (George Foster and Kuhn, 2013). Linear mixture models are far more forgiving of phrase pairs that are missing from some of the sub-corpora. Inspired from (Chiang et al., 2011) and (George Foster and Kuhn, 2013), to address this problem, we propose the following smoothing scheme:

$$\hat{p}_i(s|t) = (1-\lambda)p_i(s|t) + \lambda p_a(s|t) \tag{3}$$

where $p_a(s|t)$ is trained on all training data, and $\lambda$ is tuned on held-out data. The sub-model weights $\hat{p}_i(s|t)$ are then learned under the standard SMT log-linear framework. Experiments (see 5.3) show that this simple smoothing greatly improves the performance of log-linear mixture adaptation.

[1] We use the models trained on the whole training data to align the dev set. This can be done with mgiza (http://www.kyloo.net/software/doku.php/mgiza)

### 3.3 Provenance features

Provenance features (Chiang et al., 2011) are applied to lexical weights. There are slight variations in computing lexical weights (Foster et al., 2006), they all use forward and backward word translation probabilities $T(\underline{s}|\underline{t})$ and $T(\underline{t}|\underline{s})$ estimated from the word-aligned parallel text. The conditional probability for word pair $(\underline{s}, \underline{t})$ in a translation table is computed as below:

$$T(\underline{s}|\underline{t}) = \frac{count(\underline{s}, \underline{t})}{\sum_{\underline{s}_i} count(\underline{s}_i, \underline{t})}. \tag{4}$$

We adopt the approach proposed in (Zens and Ney, 2004) to compute the lexical weights. It assumes that all source words are conditionally independent:

$$p_{lw}(s|t) = \prod_{i=1}^{n} p(\underline{s}_i|t) \tag{5}$$

and adopts a "noisy-or" combination, so that

$$p(\underline{s}_i|t) = 1 - \prod_{j=1}^{m} (1 - T(\underline{s}_i|\underline{t}_j)) \tag{6}$$

where $n$ and $m$ are number of source and target words in the phrase pair $(s, t)$ respectively.

To compute the provenance features, we first estimate the word translation tables $T(\underline{s}|\underline{t})$ and $T(\underline{t}|\underline{s})$ trained on the $N$ sub-corpora. However, many word pairs are unseen for the word translation table of a given sub-corpus. Following (Chiang et al., 2011), we smooth the translation tables:

$$\hat{T}_i(\underline{s}|\underline{t}) = (1 - \lambda)T_i(\underline{s}|\underline{t}) + \lambda T_a(\underline{s}|\underline{t}) \tag{7}$$

where $T_a(\underline{s}|\underline{t})$ is the word translation table trained on all training data. $T_i(\underline{s}|\underline{t})$ is the conditional probability for word pair $(\underline{s}, \underline{t})$ in a translation table extracted from the $i$th sub-corpus. After we obtain the smoothed word translation lexicons for each sub-corpora, we compute the lexical weights using Equation 5, therefore, we obtain $2 \times N$ provenance features.

## 4 Extensions to vector space model (VSM) adaptation

The original vector space model (VSM) adaptation was proposed in (Chen et al., 2013b), two new variants are proposed in this paper.

### 4.1 Original VSM adaptation

The version of VSM in (Chen et al., 2013b) compares the domain vector profile of the in-domain dev set with a profile for the phrase pairs extracted from the training data. The similarity score for these two vectors is used as a decoding feature. This VSM variant will be called "original" VSM.

The domain vector for phrase-pair $(s, t)$ is a vector where each entry reflects the contribution of a particular sub-corpus to the phrase pair. It is defined as a vector of standard $\text{tf}(s, t) \cdot \text{idf}(s, t)$ weights, where $\text{tf}(s, t)$ is the raw joint count $c_i(s, t)$ in the corpus $d_i$ normalized by dividing by the maximum raw count of any phrase pair extracted in the corpus $d_i$.[2] Let

[2]We tried normalizing with the total count or not normalizing: normalizing with the maximum count works better in practice.

$$\mathrm{tf}_i\left(s,t\right) = \frac{c_i\left(s,t\right)}{\max\left\{c_i\left(s_j,t_k\right),\left(s_j,t_k\right)\in d_i\right\}}. \tag{8}$$

The idf $(s,t)$ is the inverse document frequency. We use the standard formula:

$$\mathrm{idf}\left(s,t\right) = log\left(\frac{N}{\mathrm{df}\left(s,t\right)} + C\right), \tag{9}$$

where df$(s,t)$ is the number of sub-corpora that $(s,t)$ appears in, and $C$ is an empirically determined smoothing term.

To calculate the domain similarity score between a phrase pair and the in-domain data, we (following (Chen et al., 2013b)) compute the Bhattacharya coefficient (BC) (Bhattacharyya, 1943). To map the BC score onto a range from 0 to 1, we normalize each weight in the vector by dividing it by the sum of the weights. Thus, we get the probability of a phrase pair in the $i$th sub-corpus:

$$p_i(s,t) = \frac{\mathrm{tf}_i(s,t)\cdot\mathrm{idf}(s,t)}{\sum_{j=1}^{j=N}\mathrm{tf}_j(s,t)\cdot\mathrm{idf}(s,t)} \tag{10}$$

$$p_i(s,t) = \frac{\mathrm{tf}_i(s,t)}{\sum_{j=1}^{j=N}\mathrm{tf}_j(s,t)} \tag{11}$$

For the in-domain dev set, we first run word alignment and phrase extraction in the usual way, then sum the distribution of each phrase pair $(s_j,t_k)$ extracted from the dev across sub-corpora to represent its domain information. The $i$th component of the dev domain vector is thus

$$w_i(dev) = \sum_{j=0}^{j=J}\sum_{k=0}^{k=K} c_d\left(s_j,t_k\right)\mathrm{tf}_i(s_j,t_k)\cdot\mathrm{idf}(s_j,t_k) \tag{12}$$

$J,K$ are the total number of source/target phrases extracted from the dev data respectively. $c_d\left(s_j,t_k\right)$ is the joint count of phrase pair $(s_j,t_k)$ found in the dev set. $p_i(dev)$ is normalized similarly.

$$p_i(dev) = \frac{w_i(dev)}{\sum_{j=1}^{j=N} w_j(dev)} \tag{13}$$

The Bhattacharya coefficient (BC) is defined as:

$$BC(dev;s,t) = \sum_{i=0}^{i=N}\sqrt{p_i(dev)\cdot p_i(s,t)} \tag{14}$$

### 4.2 Grouped VSM

The original VSM adaptation computes the in-domain domain vector with a set of phrase pairs extracted from the in-domain dev set. However, even a highly domain-specific dev set contains both domain-specific phrase pairs and general-domain phrase pairs. For example, in "*I have two computers, one is a Dell laptop, another one is a HP desktop*", the phrases "*Dell laptop*", and "*HP desktop*" are from the computer domain, while the phrases "*I have*", "*two*", "*one is*", "*another one is*" are general. Therefore, we devised a new form of VSM which groups the in-domain phrase pairs into subsets, which are then used for VSM adaptation. That is we partition the phrase-pairs of the dev set into $K$ subsets, then derive a domain vector and similarity score for each, resulting in $K$ similarity scores as decoding features.

After initial attempts using $K$-medoid and $K$-mean clustering to group the phrase pairs, which we abandoned because these two algorithms were computationally expensive and yielded poor DA, we implemented a simple, cheap grouping algorithm. This algorithm can classify the phrase pairs into 2 or $N+1$ subsets, where $N$ is the number of sub-corpora. We first create $N+1$ pseudo-domain vectors, each with $N$ entries: the first $N$ vectors form an $N \times N$ identity matrix, and the last vector represents the uniform probability vector - each of its entries is $1/N$ as in Equation 15. Then, we calculate the similarity score for the domain profile vector for each phrase pair in the dev set with each of the $N+1$ pseudo-vectors. The phrase pairs are then grouped into $N+1$ subsets according to their biggest similarity score. Thus, each phrase pair in the dev set is assigned to the domain for which it has the largest component, except for general phrase pairs, which are assigned to domain $N+1$.

$$[1, 0..., 0],\ [0, 1..., 0],\ ..., [0, ..., 1, ..., 0], [0, 0, ..., 1],\ and\ [\tfrac{1}{N}, \tfrac{1}{N}, ..., \tfrac{1}{N}]. \tag{15}$$

$$lab(s,t) = \underset{i=1...N+1}{\operatorname{argmax}}\ BC(pvec_i; s, t) \tag{16}$$

where $lab(s,t)$ is the subset label of phrase pair $(s,t)$, and $pvec_i$ is the $i$th pseudo-vector in Equation 15.

If we want to split the dev phrase pairs into only two groups - domain-specific and general-domain - we merge the first $N$ subsets into a domain-specific set, while the last subset of phrase pairs is the general-domain set. To group the phrase pairs into $N$ subsets, i.e. if we don't want to have a general-domain sub-set, we remove the last, uniform pseudo-domain vector from the procedure.

After grouping the phrase-pairs of the dev set into $K$ subsets, we then compute a domain vector and similarity score for each using the algorithm in the section 4.1. Then, the $K$ similarity scores are used as decoding features.

### 4.3 Distributional VSM

This is the second new version of VSM. Instead of computing a similarity score between the domain vectors of the phrase pair and the dev set, we propose to use the entries of the phrase pair's domain vector directly as decoder log-linear features. Those features, i.e. the probability distribution in Equation 11, indicate each phrase pair's distribution across sub-corpora. Therefore, we call this method distributional VSM adaptation. This method has a few advantages compared to the original VSM adaptation. In the original VSM adaptation, we need a similarity function to compute the domain similarity between the dev set and phrase pair, such as the Bhattacharya coefficient in Equation 14. To select a good similarity function needs additional experiments or a priori knowledge, like (Chen et al., 2013b) did. And there are also some free parameters needs to be tuned before-hand, such as $C$ in Equation 9. But the distributional VSM adaptation can avoid these additional experiments, moreover, experiments (see 5.3) show that this method greatly improves the performance over the original VSM adaptation.

## 5 Experiments

### 5.1 Data setting

We carried out experiments in two different settings, both involving data from NIST Open MT 12.[3] The first setting is based on data from the Chinese to English constrained track, comprising about 283 million English running words. We manually grouped the training data into 14 corpora according to genre and origin. Table 1 summarizes information about the training,

[3]http://www.nist.gov/itl/iad/mig/openmt12.cfm

| corpus | # segs | # en tok | % | genres |
|---|---|---|---|---|
| fbis | 250K | 10.5M | 3.7 | nw |
| financial | 90K | 2.5M | 0.9 | financial |
| gale_bc | 79K | 1.3M | 0.5 | bc |
| gale_bn | 75K | 1.8M | 0.6 | bn ng |
| gale_nw | 25K | 696K | 0.2 | nw |
| gale_wl | 24K | 596K | 0.2 | wl |
| hkh | 1.3M | 39.5M | 14.0 | Hansard |
| hkl | 400K | 9.3M | 3.3 | legal |
| hkn | 702K | 16.6M | 5.9 | nw |
| isi | 558K | 18.0M | 6.4 | nw |
| lex&ne | 1.3M | 2.0M | 0.7 | lexicon |
| others_nw | 146K | 5.2M | 1.8 | nw |
| sinorama | 282K | 10.0M | 3.5 | nw |
| un | 5.0M | 164M | 58.2 | un |
| TOTAL | 10.1M | 283M | 100.0 | (all) |
| devtest | | | | |
| tune | 1,506 | 161K | | nw wl |
| NIST06 | 1,664 | 189K | | nw bn ng |
| NIST08 | 1,357 | 164K | | nw wl |

Table 1: NIST Chinese-English data. In the *genres* column: nw=newswire, bc=broadcast conversation, bn=broadcast news, wl=weblog, ng=newsgroup, un=United Nations proceedings.

development and test sets; we show the sizes of the training subcorpora in number of words as a percentage of all training data. The development set (*tune*) was taken from the NIST 2005 evaluation set, augmented with some web-genre material reserved from other NIST corpora. NIST MT 06 and 08 test sets are used as our two test sets.

The second setting uses NIST 12 Arabic to English data, but excludes the UN data. There are about 47.8 million English running words in these training data. We manually grouped the training data into 7 groups according to genre and origin. Table 2 summarizes information about the training, development and test sets. We use the evaluation sets from NIST 06, 08, and 09 as our development set and two test sets, respectively. All Chinese and Arabic dev and test sets have 4 references.

## 5.2 System

Experiments were carried out with an in-house, state-of-the-art phrase-based system. The whole corpora were first word-aligned using IBM2, HMM, and IBM4 models, then split to subcorpora according to genre and origin; the phrase table was the union of phrase pairs extracted from these alignments, with a length limit of 7. The translation model (TM) was Kneser-Ney smoothed in both directions (Chen et al., 2011). We use the hierarchical lexicalized reordering model (RM) (Galley and Manning, 2008), with a distortion limit of 7, lexical weighting in both directions, word count, a distance-based reordering model, a 4-gram language model (LM) trained on the target side of the parallel data, and a 6-gram English *Gigaword* LM. The system was tuned with batch lattice MIRA (Cherry and Foster, 2012).

## 5.3 Results

This paper has described three modifications to existing techniques: smoothing for log-linear mixtures and two new versions of VSM - grouped VSM and distributional VSM. First, we did

| corpus | # segs | # en toks | % | genres |
|---|---|---|---|---|
| gale_bc | 57K | 1.6M | 3.3 | bc |
| gale_bn | 45K | 1.2M | 2.5 | bn |
| gale_ng | 21K | 491K | 1.0 | ng |
| gale_nw | 17K | 659K | 1.4 | nw |
| gale_wl | 24K | 590K | 1.2 | wl |
| isi | 1,124K | 34.7M | 72.6 | nw |
| other_nw | 224K | 8.7M | 18.2 | nw |
| TOTAL | 1,512K | 47.8M | 100.0 | (all) |
| devtest | | | | |
| NIST06 | 1,664 | 202K | | nw wl |
| NIST08 | 1,360 | 205K | | nw wl |
| NIST09 | 1,313 | 187K | | nw wl |

Table 2: NIST Arabic-English data. In the *genres* column: nw=newswire, bc=broadcast conversation, bn=broadcast news, ng=newsgroup, wl=weblog.

| | Chinese-English | | Arabic-English | |
|---|---|---|---|---|
| | MT06 | MT08 | MT08 | MT09 |
| baseline | 36.0 | 29.4 | 46.4 | 49.2 |
| log-lin w/o smooth | 35.8 | 29.0 | 47.7** | 50.0** |
| log-lin w/ smooth | 37.9**++ | 31.1**++ | 48.2**+ | 50.6**++ |

Table 3: Results of log-linear mixtures with or without smoothing. */** or +/++ means result is significantly better than baseline or system without smoothing ($p < 0.05$ or $p < 0.01$, respectively).

experiments to see if these three modifications yield statistically significant improvements over the original techniques. Our evaluation metric is IBM BLEU (Papineni et al., 2002), which performs case-insensitive matching of n-grams up to n = 4. Following (Koehn, 2004), we use the bootstrap-resampling test for significance testing.

We set the $\lambda$ in Equation 3 and Equation 7 to 0.01 by looking at performance on the Arabic-English dev set. Table 3 shows the results of log-linear mixture model with or without smoothing. Without smoothing, log-linear mixture model DA hurt the performance of Chinese-to-English on both test sets, but gave moderate improvements on Arabic-to-English. The Chinese task has more (14) sub-corpora than the Arabic one, and they seem more diverse; as suggested earlier, the “veto power” of small sub-corpora seems to be particularly harmful for Chinese. Fortunately, the smoothing technique described earlier yielded significant improvements over the baseline for all four test sets at level $p < 0.01$, with absolute improvements from +1.4-1.9 BLEU: also a significant improvement over the adaptive but unsmoothed system. Thus, smoothing makes log-linear mixture DA perform much better.

Table 4 shows experiments with grouped VSM DA. We classify the phrase pairs extracted from in-domain dev set into $K$ groups. The $K = 1$ case is equivalent to original VSM. The $K = 2$ case labels every phrase pair as either domain-specific or general-domain. The $K = N$ case means that every phrase pair is labeled with the number of the closest sub-corpus (there are $N$ sub-corpora). Finally, the $K = N + 1$ case labels every phrase pair either with the number of the closest sub-corpus, or as general-domain (the $N + 1$ group). Performance steadily improves as the number of groups increases.

| | Chinese-English | | Arabic-English | |
|---|---|---|---|---|
| | MT06 | MT08 | MT08 | MT09 |
| baseline | 36.0 | 29.4 | 46.4 | 49.2 |
| $K = 1$ | 37.0** | 30.3** | 47.7** | 50.1** |
| $K = 2$ | 37.3**+ | 30.6** | 47.9** | 50.3**+ |
| $K = N$ | 37.9**++ | 31.2**++ | 48.2**++ | 50.8**++ |
| $K = N + 1$ | 38.2**++ | 31.6**++ | 48.3**++ | 51.0**++ |

Table 4: Grouped VSM adaptation. */** or +/++ means result is significantly better than baseline or original VSM adaptation system, i.e., $K = 1$. ($p < 0.05$ or $p < 0.01$, respectively).

| | Chinese-English | | Arabic-English | |
|---|---|---|---|---|
| | MT06 | MT08 | MT08 | MT09 |
| baseline | 36.0 | 29.4 | 46.4 | 49.2 |
| vsm | 37.0** | 30.3** | 47.7** | 50.1** |
| distr. feat. | 38.0**++ | 31.2**++ | 48.0**+ | 50.9**++ |

Table 5: The results of baseline, original VSM adaptation and distributional VSM adaptation.

Table 5 compares original VSM and distributional VSM. It shows that the original VSM adaptation reported in (Chen et al., 2013b) does yield improvement over a non-adaptive baseline. However, if instead of computing a domain similarity score, we directly maximize BLEU on the dev set by tuning the weights of distribution features, we got further significant improvements. On the Chinese task, the further improvements were +0.7-0.8 BLEU, while on Arabic, the further improvements were smaller but still significant: +0.3-0.4 BLEU. (Cherry, 2013) showed that in the case of reordering features, directly maximizing BLEU outperforms maximum entropy optimization; the experiments in Table 5 yield a similar conclusion. Given that recently developed tuning algorithms such as MIRA can handle a very large feature set, we may consider having all possible features directly tuned to maximize BLEU (or similar criteria).

Now, we compare all six DA techniques, Table 6 reports the results. All techniques improved on all test sets across two language pairs over the non-adaptive baseline, and all these improvements are significant. From the average absolute improvement across test sets, the original VSM adaptation yield improvement of around 1.1 BLEU on average. And it is inferior to the remaining five DA techniques; these five techniques obtained similar improvement over the baseline, around +1.7-2.0 BLEU. Linear mixtures did best on Arabic MT08, provenance features did best on Chinese MT06, while $N + 1$ grouped VSM did best on the other two sets. So there is no clear winner among these techniques.

Our last experiment studies whether all these DA techniques exploit the same information, or if they are somewhat complementary. The results are reported in Table 7. “Single best” is a strong baseline, since it involves the best result in each column of Table 6 rather than the best single row. We first figured out the best mixture model combination, by exploring all 4 possible combinations of the three mixture model adaptation methods. We found that combination of linear mixture and provenance features got the best performance, even better than using all three mixture model methods together. The possible reason is that they are likely to be complementary, since the former adapts phrase translation probabilities and the latter adapts lexical weights. This combination did very well on Arabic-English but gave only small improvements on Chinese-English.

Then, we add VSM adaptation method by a form of greedy search: beginning with the best

| | Chinese-English | | Arabic-English | | |
|---|---|---|---|---|---|
| | MT06 | MT08 | MT08 | MT09 | avg $\Delta$ |
| baseline | 36.0 | 29.4 | 46.4 | 49.2 | — |
| linmix | 37.8 | 31.2 | **48.8** | 50.9 | +1.9 |
| smoothed log-linmix | 37.9 | 31.1 | 48.2 | 50.6 | +1.7 |
| provenance | **38.4** | 31.4 | 48.2 | 50.6 | +1.9 |
| original VSM | 37.0 | 30.3 | 47.7 | 50.1 | +1.1 |
| distr. VSM | 38.0 | 31.2 | 48.0 | 50.9 | +1.8 |
| $N + 1$ gr. VSM | 38.2 | **31.6** | 48.3 | **51.0** | +2.0 |

Table 6: The comparison of all 7 adaptation techniques. The $\Delta$ is computed on all four test sets.

mixture model combination and then incorporating the technique that yields the highest BLEU improvement over the system it's added to. For the next combination, we add $N + 1$ grouped VSM adaptation and observe significant improvements over "single best" for all test sets. For the final combination, we add distributional VSM DA, gaining +2.9 BLEU on both Chinese MT06 and MT08, +3.3 BLEU on Arabic MT08 and +2.6 BLEU on Arabic MT09 over the non-adaptive baseline. Adding the two VSM adaptation methods to the best mixture combinations, further significant improvements were obtained on three out of four test sets. We were unable to obtain further gains by adding the original VSM adaptation technique to the system: apparently, the combination of four techniques shown in the last row of the table covers all the information exploited by the six DA techniques described in this paper.

| | Chinese-English | | Arabic-English | | avg |
|---|---|---|---|---|---|
| | MT06 | MT08 | MT08 | MT09 | $\Delta$ |
| baseline | 36.0 | 29.4 | 46.4 | 49.2 | — |
| single best | 38.4 | 31.6 | 48.8 | 51.0 | +2.2 |
| linmix | 37.8 | 31.2 | 48.8 | 50.9 | +1.9 |
| prov. (provenance) | 38.4 | 31.4 | 48.2 | 50.6 | +1.9 |
| linmix & prov. | 38.6 | 31.7+ | 49.3**++ | 51.4*++ | +2.5 |
| linmix & prov. & gr. VSM | 38.7* | 32.0*+ | 49.6**+ | 51.6** | +2.7 |
| linmix & prov. & gr. VSM & distr. VSM | 38.9** | 32.3**+# | 49.7**# | 51.8**# | +2.9 |

Table 7: Combinations of techniques. */** or +/++ means result is significantly better than single best result in Table 6 ("single best" row) or system without newly added technique (i.e., than previous row) ($p < 0.05$ or $p < 0.01$, respectively). # means the result in the final combination (the last row) is significantly better than the best mixture model combination ("linmix & prov." row) at level $p < 0.05$.

## 6 Conclusions

We have proposed two extensions to the original vector space model (VSM) adaptation, which are distributional VSM, and grouped VSM. They both significantly improved over the original VSM adaptation. We also improved log-linear mixture significantly by smoothing the mixture components. Then we systematically compared the VSM adaptation techniques to the mixture model adaptation techniques, which includes linear mixture, log-linear mixture and provenance features. According to BLEU, the original VSM DA has weaker performance than the other techniques, yielding around 1.1 improvement on average over a non-adaptive baseline. The remaining five techniques obtained similar improvements across four test sets in two large-scale data conditions, in the range +1.7-2.0 BLEU on average; there is no clear winner among them.

Although the improvements obtained from these techniques are not strictly additive, combining a subset of them yields further significant improvement: we obtained +2.6-3.3 BLEU improvement on average over the non-adaptive baseline with our best combination. In future work, we will try combining language model and reordering model adaptation with the techniques we have described here.

## Acknowledgement

This research was supported in part by DARPA contract HR0011-12-C-0014 under subcontract to Raytheon BBN Technologies.

## References

Axelrod, A., He, X., and Gao, J. (2011). Domain adaptation via pseudo in-domain data selection. In *EMNLP 2011*.

Bertoldi, N. and Federico, M. (2009). Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of the 4th Workshop on Statistical Machine Translation*, Athens. WMT.

Bhattacharyya, A. (1943). On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109.

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 *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1435–1445, Sofia, Bulgaria. Association for Computational Linguistics.

Chen, B., Foster, G., and Kuhn, R. (2013a). Adaptation of reordering models for statistical machine translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 938–946, Atlanta, Georgia. Association for Computational Linguistics.

Chen, B., Kuhn, R., and Foster, G. (2013b). Vector space model for adaptation in statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1285–1293, Sofia, Bulgaria. Association for Computational Linguistics.

Chen, B., Kuhn, R., Foster, G., and Johnson, H. (2011). Unpacking and transforming feature functions: New ways to smooth phrase tables. In *MTSummit*.

Chen, B., Zhang, M., Aw, A., and Li, H. (2008). Exploiting n-best hypotheses for smt self-enhancement. In *Proceedings of ACL-08: HLT, Short Papers*, pages 157–160, Columbus, Ohio. Association for Computational Linguistics.

Cherry, C. (2013). Improved reordering for phrase-based translation using sparse features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 22–31, Atlanta, Georgia. Association for Computational Linguistics.

Cherry, C. and Foster, G. (2012). Batch tuning strategies for statistical machine translation. In *NAACL 2012*.

Chiang, D., DeNeefe, S., and Pust, M. (2011). Two easy improvements to lexical weighting. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 455–460, Portland, Oregon, USA. Association for Computational Linguistics.

Cui, L., Chen, X., Zhang, D., Liu, S., Li, M., and Zhou, M. (2013). Multi-domain adaptation for SMT using multi-task learning. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1055–1065, Seattle, Washington, USA. Association for Computational Linguistics.

Daume III, H. and Jagarlamudi, J. (2011). Domain adaptation for machine translation by mining unseen words. In *ACL 2011*.

Dou, Q. and Knight, K. (2012). Large scale decipherment for out-of-domain machine translation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 266–275, Jeju Island, Korea. Association for Computational Linguistics.

Duh, K., Neubig, G., Sudoh, K., and Tsukada, H. (2013). Adaptation data selection using neural language models: Experiments in machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 678–683, Sofia, Bulgaria. Association for Computational Linguistics.

Eidelman, V., Boyd-Graber, J., and Resnik, P. (2012). Topic models for dynamic translation model adaptation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 115–119, Jeju Island, Korea. Association for Computational Linguistics.

Eva Hasler, B. H. and Koehn, P. (2012). Sparse lexicalised features and topic adaptation for smt. In *Proceedings of IWSLT*, Hongkong.

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 (EMNLP)*, Boston.

Foster, G. and Kuhn, R. (2007). Mixture-model adaptation for SMT. In *Proceedings of the ACL Workshop on Statistical Machine Translation*, Prague. WMT.

Foster, G., Kuhn, R., and Johnson, H. (2006). Phrasetable smoothing for statistical machine translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Sydney, Australia.

Galley, M. and Manning, C. D. (2008). A simple and effective hierarchical phrase reordering model. In *EMNLP 2008*, pages 848–856, Hawaii.

George Foster, B. C. and Kuhn, R. (2013). Simulating discriminative training for linear mixture adaptation in statistical machine translation. In *MT Summit*, Nice, France.

Gong, Z., Zhang, M., and Zhou, G. (2011). Cache-based document-level statistical machine translation. In *EMNLP 2011*.

Hewavitharana, S., Mehay, D., Ananthakrishnan, S., and Natarajan, P. (2013). Incremental topic-based translation model adaptation for conversational spoken language translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 697–701, Sofia, Bulgaria. Association for Computational Linguistics.

Irvine, A., Quirk, C., and Daumé III, H. (2013). Monolingual marginal matching for translation model adaptation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1077–1088, Seattle, Washington, USA. Association for Computational Linguistics.

Koehn, P. (2004). Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas*, Georgetown University, Washington D.C. Springer-Verlag.

Koehn, P. and Schroeder, J. (2007). Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Prague, Czech Republic. Association for Computational Linguistics.

Kuhn, R. and De Mori, R. (1990). A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 12(6):570–583.

Lü, Y., Huang, J., and Liu, Q. (2007). Improving Statistical Machine Translation Performance by Training Data Selection and Optimization. In *Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Prague, Czech Republic.

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 (EMNLP)*, Singapore.

Moore, R. C. and Lewis, W. (2010). Intelligent selection of language model training data. In *ACL 2010*.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: A method for automatic evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia. ACL.

Razmara, M., Foster, G., Sankaran, B., and Sarkar, A. (2012). Mixing multiple translation models in statistical machine translation. In *ACL 2012*.

Schwenk, H. (2008). Investigations on large-scale lightly-supervised training for statistical machine translation. In *IWSLT 2008*.

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

Sennrich, R., Schwenk, H., and Aransa, W. (2013). A multi-domain translation model framework for statistical machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 832–840, Sofia, Bulgaria. Association for Computational Linguistics.

Tam, Y.-C., Lane, I., and Schultz, T. (2007). Bilingual-LSA Based LM Adaptation for Spoken Language Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, Prague, Czech Republic. ACL.

Tiedemann, J. (2010). Context adaptation in statistical machine translation using models with exponentially decaying cache. In *DANLP*.

Ueffing, N. and Ney, H. (2007). Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40.

Zens, R. and Ney, H. (2004). Improvements in phrase-based statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Boston. NAACL.

Zhang, J. and Zong, C. (2013). Learning a phrase-based translation model from monolingual data with application to domain adaptation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1425–1434, Sofia, Bulgaria. Association for Computational Linguistics.

Zhao, B., Eck, M., and Vogel, S. (2004). Language model adaptation for statistical machine translation with structured query models. In *Proceedings of the International Conference on Computational Linguistics (COLING) 2004*, Geneva.