Discriminative Corpus Weight Estimation for Machine Translation

Spyros Matsoukas and **Antti-Veikko I. Rosti** and **Bing Zhang**
BBN Technologies, 10 Moulton Street, Cambridge, MA 02138
{smatsouk,arosti,bzhang}@bbn.com

Abstract

Current statistical machine translation (SMT) systems are trained on sentence-aligned and word-aligned parallel text collected from various sources. Translation model parameters are estimated from the word alignments, and the quality of the translations on a given test set depends on the parameter estimates. There are at least two factors affecting the parameter estimation: domain match and training data quality. This paper describes a novel approach for automatically detecting and down-weighing certain parts of the training corpus by assigning a weight to each sentence in the training bitext so as to optimize a discriminative objective function on a designated tuning set. This way, the proposed method can limit the negative effects of low quality training data, and can adapt the translation model to the domain of interest. It is shown that such discriminative corpus weights can provide significant improvements in Arabic-English translation on various conditions, using a state-of-the-art SMT system.

1 Introduction

Statistical machine translation (SMT) systems rely on a training corpus consisting of sentences in the source language and their respective reference translations to the target language. These parallel sentences are used to perform automatic word alignment, and extract translation rules with associated probabilities. Typically, a parallel training corpus is comprised of collections of varying quality and relevance to the translation problem of interest. For example, an SMT system applied to broadcast conversational data may be trained on a corpus consisting mostly of United Nations and newswire data, with only a very small amount of in-domain broadcast news/conversational data. In this case, it would be desirable to down-weigh the out-of-domain data relative to the in-domain data during the rule extraction and probability estimation. Similarly, it would be good to assign a lower weight to data of low quality (e.g., poorly aligned or incorrectly translated sentences) relative to data of high quality.

In this paper, we describe a novel discriminative training method that can be used to estimate a weight for each sentence in the training bitext so as to optimize an objective function – expected translation edit rate (TER) (Snover et al., 2006) – on a held-out development set. The training bitext typically consists of millions of (parallel) sentences, so in order to ensure robust estimation we express each sentence weight as a function of sentence-level features, and estimate the parameters of this mapping function instead. Sentence-level features may include the identifier of the collection or genre that the sentence belongs to, the number of tokens in the source or target side, alignment information, etc. The mapping from features to weights can be implemented via any differentiable function, but in our experiments we used a simple perceptron. Sentence weights estimated in this fashion are applied directly to the phrase and lexical counts unlike any previously published method to the author's knowledge. The tuning framework is developed for phrase-based SMT models, but the tuned weights are also applicable to the training of a hierarchical model. In cases where the tuning set used for corpus weight estimation is a close match to the test set, this method yields significant gains in TER, BLEU (Papineni et al., 2002), and METEOR (Lavie and Agarwal, 2007) scores over a state-of-the-art hierarchical baseline.

The paper is organized as follows. Related work on data selection, data weighting, and model adaptation is presented in Section 2. The corpus weight

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 708–717,
Singapore, 6-7 August 2009. ©2009 ACL and AFNLP

approach and estimation algorithm are described in Section 3. Experimental evaluation of the approach is presented in Sections 4 and 5. Section 6 concludes the paper with a few directions for future work.

2 Related Work

Previous work related to corpus weighting may be split into three categories: data selection, data weighting, and translation model adaptation. The first two approaches may improve the quality of the word alignment and prevent phrase-pairs which are less useful for the domain to be learned. The model adaptation, on the other hand, may boost the weight of the more relevant phrase-pairs or introduce translations for unseen source phrases.

Resnik and Smith (2003) mined parallel text from the web using various filters to identify likely translations. The filtering may be viewed as a data selection where poor quality translation are discarded before word alignment. Yasuda et al. (2008) selected subsets of an existing parallel corpus to match the domain of the test set. The discarded sentence pairs may be valid translations but they do not necessarily improve the translation quality on the test domain. Mandal et al. (2008) used active learning to select suitable training data for human translation. Hildebrand et al. (2005) selected comparable sentences from parallel corpora using information retrieval techniques.

Lu et al. (2007) proposed weighting comparable portions of the parallel text before word alignment based on information retrieval. The relevant portions of the parallel text were given a higher integer weight in GIZA++ word alignment. Similar effect may be achieved by replicating the relevant subset in the training data.

Lu et al. (2007) also proposed training adapted translation models which were interpolated with a model trained on the entire parallel text. Snover et al. (2008) used cross-lingual information retrieval to identify possible bias-rules to improve the coverage on the source side. These rules may cover source phrases for which no translations were learned from the available parallel text.

Koehn and Schroeder (2007) described a procedure for domain adaptation that was using two translation models in decoding, one trained on in-domain data and the other on out-of-domain data. Phrase translation scores from the two models where combined in a log-linear fashion, with weights estimated based on minimum error rate training (Och, 2003) on a designated tuning set.

The method described in this paper can also be viewed as data filtering or (static) translation adaptation, but it has the following advantages over previously published techniques:

1. The estimated corpus weights are *discriminative* and are computed so as to directly optimize an MT performance metric on a pre-defined development set. Unlike the domain adaptation technique in (Koehn and Schroeder, 2007), which also estimates the adaptation parameters discriminatively, our proposed method does not require a manual specification of the in-domain and out-of-domain training data collections. Instead, it automatically determines which collections are most relevant to the domain of interest, and increases their weight while decreasing the weight assigned to less relevant collections.

2. All sentences in the parallel corpus can influence the translation model, as opposed to filtering/discarding data. However, the proposed method can still assign very low weights to parts of the corpus, if it determines that it helps improve MT performance.

3. The framework used for estimating the corpus weights can be easily extended to support discriminative alignment link-level weights, thus allowing the system to automatically identify which portions of the training sentences are most useful.

Naturally, as with any method, the proposed technique has certain limitations. Specifically, it is only concerned with influencing the translation rule probabilities via the corpus weights; it does not change the set of rules extracted. Thus, it is unable to add new translation rules as in Snover et al. (2008). Also, it can potentially lead to parameter over-fitting, especially if the function that maps sentence features to weights is complex and based on a large number of parameters, or if the development set used for estimating the mapping function does not match the characteristics of the test set.

3 Corpus Weights Estimation

3.1 Feature Extraction

The purpose of feature extraction is to identify, for each sentence in the parallel training data, a set of features that can be useful in estimating a weight that is correlated with quality or relevance to the MT task at hand. Starting from sentence-aligned, word-aligned parallel training data, one could extract various types of sentence-level features. For example, we could specify features that describe the two sides of the parallel data or the alignment between them, such as collection id, genre id, number of source tokens, number of target tokens, ratio of number of source and target tokens, number of word alignment links, fraction of source tokens that are unaligned, and fraction of target tokens that are unaligned. Additionally, we could include information retrieval (IR) related features that reflect the relevance of a training sentence to the domain of interest, e.g., by measuring vector space model (VSM) distance of the sentence to the current tuning set, or its log likelihhod with respect to an in-domain language model.

Note that the collection and genre identifiers (ids) mentioned above are bit vectors. Each collection in the training set is mapped to a number. A collection may consist of sentences from multiple genres (e.g., newswire, web, broadcast news, broadcast conversations). Genres are also mapped to a unique number across the whole training set. Then, given a sentence in the training bitext, we can extract a binary vector that contains two non-zero bits, one indicating the collection id, and another denoting the genre id.

It is worth mentioning that in the experiments reported later in this paper we made use of only the collection and genre ids as features, although the framework supports general sentence-level features.

3.2 Mapping Features to Weights

As mentioned previously, one way to map a feature vector to a weight is to use a perceptron. A multi-layer neural network may also be used, but at the expense of slower training. In this work, all of the experiments carried out made use of a perceptron mapping function. However, it is also possible to cluster the training sentences into classes by training a Gaussian mixture model (GMM) on their respective feature vectors[1]. Then, given a feature vector we can compute the (posterior) probability that it was generated by one of the N Gaussians in the GMM, and use this N-dimensional vector of posteriors as input to the perceptron. This is similar to having a neural network with a static hidden layer and Gaussian activation functions.

Given the many choices available in mapping features to weights, we will describe the mapping function in general terms. Let $\mathbf{f}_i$ be the $n \times 1$ feature vector corresponding to sentence i. Let $\phi(\mathbf{x}; \boldsymbol{\lambda})$ denote a function $\mathbb{R}^n \to (0,1)$ that is parameterized in terms of the parameter vector $\boldsymbol{\lambda}$ and maps a feature vector $\mathbf{x}$ to a scalar weight in $(0,1)$. The goal of the automatic corpus weight estimation procedure is to estimate the parameter vector $\boldsymbol{\lambda}$ so as to optimize an objective function on a development set.

3.3 Training with Weighted Corpora

Once the sentence features have been mapped to weights, the translation rule extraction and probability estimation can proceed as usual, but with weighted counts. For example, let $w_i = \phi(\mathbf{f}_i; \boldsymbol{\lambda})$ be the weight assigned to sentence i. Let (s,t) be a source-target phrase pair that can be extracted from the corpus, and $\mathcal{A}(s)$ and $\mathcal{B}(t)$ indicating the sets of sentences that s and t occur in. Then,

$$P(s|t) = \frac{\sum_{j \in \mathcal{A}(s) \cap \mathcal{B}(t)} w_j c_j(s,t)}{\sum_{j \in \mathcal{B}(t)} w_j c_j(t)} \qquad (1)$$

where $c_j(\cdot)$ denotes the number of occurrences of the phrase (or phrase-pair) in sentence j.

3.4 Optimizing the Mapping Function

Estimation of the parameters $\boldsymbol{\lambda}$ of the mapping function ϕ can be performed by directly optimizing a suitable objective function on a development set. Ideally, we would like to estimate the parameters of the mapping function so as to directly optimize an automatic MT performance evaluation metric, such as TER or BLEU on the full translation search space. However, this is extremely computationally intensive for two reasons: (a) optimizing in the full translation search space requires a new decoding pass for each iteration of optimization; and (b) a direct optimization of TER or

[1] Note that in order to train such a GMM it may be necessary to first apply a decorrelating, dimensionality reducing, transform (e.g., principal component analysis) to the features.

BLEU requires the use of a derivative free, slowly converging optimization method such as MERT (Och, 2003), because these objective functions are not differentiable.

In our case, for every parameter vector update we need to essentially retrain the translation model (reestimate the phrase and lexical translation probabilities based on the updated corpus weights), so the cost of each iteration is significantly higher than in a typical MERT application. For these reasons, in this work we chose to minimize the expected TER over a translation N-best on a designated tuning set, which is a continuous and differentiable function and can be optimized with standard gradient descent methods in a small number of iterations. Note, that using expected TER is not the only option here; any criterion that can be expressed as a continuous function of the phrase or lexical translation probabilities can be used to optimize $\boldsymbol{\lambda}$.

Given an N-best of translation hypotheses over a development set of S sentences, we can define the expected TER as follows

$$\mathcal{T} = \frac{\sum_{s=1}^{S} \sum_{j=1}^{N_s} p_{sj} \epsilon_{sj}}{\sum_{s=1}^{S} r_s} \qquad (2)$$

where N_s is the number of translation hypotheses available for segment s; ϵ_{sj} is the minimum raw edit distance between hypothesis j of segment s (or h_{sj}, for short) and the reference translation(s) corresponding to segment s; r_s is the average number of reference translation tokens in segment s, and p_{sj} is the posterior probability of hypothesis h_{sj} in the N-best. The latter is computed as follows

$$p_{sj} = \frac{e^{\gamma L_{sj}}}{\sum_{k=1}^{N_s} e^{\gamma L_{sk}}} \qquad (3)$$

where L_{sj} is the total log likelihood of hypothesis h_{sj}, and γ is a tunable scaling factor that can be used to change the dynamic range of the likelihood scores and hence the distribution of posteriors over the N-best. The hypothesis likelihood L_{sj} is typically computed as a dot product of a decoding weight vector and a vector of various "feature" scores, such as log phrase translation probability, log lexical translation probability, log n-gram language model probability, and number of tokens in the hypothesis. However, in order to simplify this presentation we will assume that it contains a single translation model score, the log phrase translation probability of source given target. This score is a sum of log conditional probabilities, similar to the one defined in Equation 1. Therefore, L_{sj} is indirectly a function of the training sentence weights.

In order to minimize the expected TER $\mathcal{T}$, we need to compute the derivative of $\mathcal{T}$ with respect to the mapping function parameters $\boldsymbol{\lambda}$. Using the chain rule, we get equations (4)-(8), where the summation in Equation 6 is over all source-target phrase pairs in the derivation of hypothesis h_{sm}, ξ is the decoding weight assigned to the log phrase translation score, and the summation in Equation 7 is over all training sentences[2].

Thus, in order to compute the derivative of the objective function we first need to calculate $\frac{\partial \ln P(s_k|t_k)}{\partial \boldsymbol{\lambda}}$ for every phrase pair (s_k, t_k) in the translation N-best based on Equations 7 and 8, which requires time proportional to the number of occurrences of these phrases in the parallel training data. After that, we can compute $\frac{\partial L_{sm}}{\partial \boldsymbol{\lambda}}$ for each hypothesis h_{sm}, based on Equation 6. Finally, we calculate $\frac{\partial \ln p_{sj}}{\partial \boldsymbol{\lambda}}$ and $\frac{\partial \mathcal{T}}{\partial \boldsymbol{\lambda}}$ based on Equations 5 and 4, respectively.

3.5 Implementation Issues

In our system, the corpus weights were trained based on N-best translation hypotheses generated by a phrase-based MT system on a designated tuning set. Each translation hypothesis in the N-best has a score that is a (linear) function of the following log translation probabilities: target phrase given source phrase, source phrase given target phrase, and lexical smoothing term. Additionally, each hypothesis specifies information about its derivation, i.e., which source-target phrase pairs it consists of. Therefore, given an N-best, we can identify the set of unique phrase pairs and use this information in order to perform a filtered accumulation of the statistics needed for calculating the derivative in Equation 8. This reduces the storage needed for the sufficient statistics significantly.

Minimization of the expected TER of the N-best hypotheses was performed using the limited-memory BFGS algorithm (Liu and Nocedal, 1989). Typically, the parameter vector $\boldsymbol{\lambda}$ required about 30 iterations of LBFGS to converge.

Since the N-best provides only a limited representation of the MT hypothesis search space, we regenerated the N-best after every 30 iterations

[2] In the general case where L_{sj} includes other translation scores, e.g., lexical translation probabilities, the derivative $\frac{\partial L_{sm}}{\partial \boldsymbol{\lambda}}$ will have to include additional terms.

$$\frac{\partial \mathcal{T}}{\partial \boldsymbol{\lambda}} = \sum_{s=1}^{S} \sum_{j=1}^{N_s} \frac{\partial \mathcal{T}}{\partial \ln p_{sj}} \frac{\partial \ln p_{sj}}{\partial \boldsymbol{\lambda}} = \left(\frac{1}{\sum_{s=1}^{S} r_s} \right) \sum_{s=1}^{S} \sum_{j=1}^{N_s} p_{sj} \epsilon_{sj} \frac{\partial \ln p_{sj}}{\partial \boldsymbol{\lambda}} \tag{4}$$

$$\frac{\partial \ln p_{sj}}{\partial \boldsymbol{\lambda}} = \sum_{m=1}^{N_s} \frac{\partial \ln p_{sj}}{\partial L_{sm}} \frac{\partial L_{sm}}{\partial \boldsymbol{\lambda}} = \gamma \left(\frac{\partial L_{sj}}{\partial \boldsymbol{\lambda}} - \sum_{m=1}^{N_s} p_{sm} \frac{\partial L_{sm}}{\partial \boldsymbol{\lambda}} \right) \tag{5}$$

$$\frac{\partial L_{sm}}{\partial \boldsymbol{\lambda}} = \sum_{(s_k,t_k) \in h_{sm}} \frac{\partial L_{sm}}{\partial \ln P(s_k|t_k)} \frac{\partial \ln P(s_k|t_k)}{\partial \boldsymbol{\lambda}} = \sum_{(s_k,t_k) \in h_{sm}} \xi \frac{\partial \ln P(s_k|t_k)}{\partial \boldsymbol{\lambda}} \tag{6}$$

$$\frac{\partial \ln P(s_k|t_k)}{\partial \boldsymbol{\lambda}} = \sum_{i} \frac{\partial \ln P(s_k|t_k)}{\partial w_i} \frac{\partial w_i}{\partial \boldsymbol{\lambda}} \tag{7}$$

$$\frac{\partial \ln P(s_k|t_k)}{\partial w_i} = \frac{\sum_{j \in \mathcal{A}(s_k) \cap \mathcal{B}(t_k)} \delta\,(j-i)\, c_j(s_k,t_k)}{\sum_{j \in \mathcal{A}(s_k) \cap \mathcal{B}(t_k)} w_j c_j(s_k,t_k)} - \frac{\sum_{j \in \mathcal{B}(t_k)} \delta\,(j-i)\, c_j(t_k)}{\sum_{j \in \mathcal{B}(t_k)} w_j c_j(t_k)} \tag{8}$$

$$\delta(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases} \tag{9}$$

of LBFGS training, merging new hypotheses with translations from previous iterations. The overall training procedure is described in more detail below:

1. Initialize parameter vector $\boldsymbol{\lambda}$ to small random values, so that all training sentences receive approximately equal weights.

2. Initialize phrase-based MT decoding weights to previously tuned values.

3. Perform weighted phrase rule extraction as described in Equation 1, to estimate the phrase and lexical translation probabilities.

4. Decode the tuning set, generating N-best.

5. Merge N-best hypotheses from previous iterations to current N-best.

6. Tune decoding weights so as to minimize TER on merged N-best, using a derivative free optimization method. In our case, we used Powell's algorithm (Powell, 1964) modified by Brent as described in (Brent, 1973) [3].

7. Identify set of unique source-target phrase pairs in merged N-best.

8. Extract sufficient statistics from training data for all phrases identified in step 7.

9. Run the LBFGS algorithm to minimize the expected TER in the merged N-best, using the derivative equations described previously.

10. Assign a weight to each training sentence based on the $\boldsymbol{\lambda}$ values optimized in 9.

11. Go to step 3.

Typically, the corpus weights converge in about 4-5 main iterations. The calculation of the derivative is parallelized to speed up computation, requiring about 10 minutes per iteration of LBFGS.

4 Experimental Setup

In this section we describe the setup that was used for all experiments reported in this paper. Specifically, we provide details about the training data, development sets, and MT systems (phrase-based and hierarchical).

4.1 Training Data

All MT training experiments made use of an Arabic-English corpus of approximately 200 million tokens (English side). Most of the collections in this corpus are available through the Linguistic Data Consortium (LDC) and are regularly part of the resources specified for the constrained data track of the NIST MT evaluation[4].

[3] This method was first used for N-best based parameter optimization in (Ostendorf et al., 1991).

[4] For a list of the NIST MT09 constrained training condition resources, see `http://www.itl.nist.gov/iad/mig/tests/mt/2009/MT09_ConstrainedResources.pdf`

The corpus includes data from multiple genres, as shown in Table 1. The “Sakhr” newswire collection is a set of Arabic-to-English and English-to-Arabic data provided by Sakhr Software, totaling about 30.8 million tokens, and is only available to research teams participating in the Defense Advanced Research Projects Agency (DARPA) Global Autonomous Language Exploitation (GALE) program. The “LDC Gigaword (ISI)” collection was produced by automatically detecting and extracting portions of parallel text from the monolingual LDC Arabic and English Gigaword collections, using a method developed at the Information Sciences Institute (ISI) of the University of Southern California.

Data Origin	Style	Size (K tokens)
LDC pre-GALE	U. Nations	118049
	Newswire	2700
	Treebank	685
LDC post-GALE	Newswire	14344
	Treebank	292
	Web	478
	Broad. News	573
	Broad. Conv.	1003
Web-found text	Lexicons	436
	Quran	406
Sakhr	Newswire	30790
LDC Gigaword (ISI)	Newswire	29169

Table 1: Composition of the Arabic-English parallel corpus used for MT training.

It is easy to see that most of the parallel training data are either newswire or from United Nations. The amount of web text or broadcast news/conversations is only a very small fraction of the total corpus. In total, there are 31 collections in the training bitext. Some collections (especially those released recently by LDC for the GALE project) consist of data from multiple genres. The total number of unique genres (or data types) in the training set is 10.

Besides the above bitext, we also used approximately 8 billion words of English text for language model (LM) training (3.7B words from the LDC Gigaword corpus, 3.3B words of web-downloaded text, and 1.1B words of data from CNN archives). This data was used to train two language models: an entropy-pruned trigram LM, used in decoding, and an unpruned 5-gram LM used in N-best rescoring. Kneser-Ney smoothing was applied to the n-grams in both cases.

4.2 Development Sets

The development sets used for tuning and testing the corpus weights and other MT settings were comprised of documents from previous Arabic-English NIST MT evaluation sets and from GALE development/evaluation sets.

Specifically, the newswire Tune and Test sets consist of documents from the following collections: the newswire portion of NIST MT04, MT05, MT06, and MT08 evaluation sets, the GALE Phase 1 (P1) and Phase 2 (P2) evaluation sets, and the GALE P2 and P3 development sets. The web Tune and Test sets are made of documents from NIST MT06 and MT08, the GALE P1 and P2 evaluation sets, the GALE P2 and P3 development sets, and a held-out portion of the GALE year 1 quarter 4 web training data release.

The audio Tune and Test sets consist of roughly equal parts of news and conversations broadcast from November 2005 through May 2007 by major Arabic-speaking television and radio stations (e.g., Al-Jazeera, Al-Arabiya, Syrian TV), totaling approximately 14 hours of speech. The audio was processed through automated speech recognition (ASR) in order to produce (errorful) transcripts that were used as input to all MT decoding experiments reported in this paper. However, the corpus weight estimation was carried out based on N-best MT of the Arabic audio reference transcriptions (i.e., the transcripts had no speech recognition errors, and contained full punctuation).

It is important to note that some of the documents in the above devsets have multiple reference translations (usually 4), while others have only one. Most of the documents in the newswire sets have 4 references, but unfortunately the web and audio sets have, on average, less than 2 reference translations per segment. More details are listed in Table 2.

Another important note is that, although the audio sets consist of both broadcast news (BN) and broadcast conversations (BC), we did not perform BN or BC-specific tuning. Corpus weights and MT decoding parameters were optimized based on a single Tune set, on a mix of BN and BC data. However, when we report speech translation results in later sections, we break down the perfor-

Genre	Tune			Test		
	#segs	#tokens	#refs/seg	#segs	#tokens	#refs/seg
Newswire	1994	72359	3.94	3149	115700	3.67
Web	3278	99280	1.69	4425	125795	2.08
Audio BN	897	32990	1.00	1530	53067	1.00
Audio BC	765	24607	1.00	1416	44435	1.00

Table 2: Characteristics of the tuning (Tune) and validation (Test) sets used for development on Arabic newswire, web, and audio. The audio sets include material from both broadcast news and broadcast conversations.

mance by genre.

4.3 MT Systems

Experiments were performed using two types of statistical MT systems: a phrase-based system, similar to Pharaoh (Koehn, 2004), and a state-of-the-art, hierarchical string-to-dependency-tree system, similar to (Shen et al., 2008).

The phrase-based MT system employs a pruned 3-gram LM in decoding, and can optionally generate N-best unique translation hypotheses which are used to estimate the corpus weights, as described in Section 3.

The hierarchical MT system performs decoding with the same 3-gram LM, generates N-best of unique translation hypotheses, and then rescores them using a large, unpruned 5-gram LM in order to select the best scoring translation. It is worth mentioning that this hierarchical MT system provides a very strong baseline; it achieves a case-sensitive BLEU score of 52.20 on the newswire portion of the NIST MT08 evaluation set, which is similar to the score of the second-best system that participated in the unconstrained data track of the NIST MT08 evaluation.

Both types of models were trained on the same word alignments generated by GIZA++ (Och and Ney, 2003).

5 Results

In this section we report results on the Arabic newswire, web, and audio development sets, using both phrase-based and hierarchical MT systems, in terms of TER, BLEU[5], and METEOR (Lavie and Agarwal, 2007). Whenever corpus weights are used, they were estimated on the designated Tune set using the phrase-based MT system. Only the collection and genre ids were used as sentence features in order to estimate the corpus weights. As mentioned in Section 4.1, the training bitext consists of 31 collections and 10 genres, so each training sentence was assigned a 41-dimensional binary vector indicating its particular collection/genre combination. That vector was then mapped into a single weight using a perceptron.

5.1 Phrase-based MT

Results using the phrase-based MT system are shown in Table 3. In all cases, the decoding weights were optimized so as to minimize TER on the designated Tune set. On newswire, the discriminative corpus weights provide 0.8% absolute gain in TER, in both Tune and Test sets. On web, the TER gain is 0.9% absolute on Tune and 0.5% on Test. On the audio Test set, the TER gain is 0.5% on BN and 1.4% on BC. Significant improvements were also obtained in the BLEU and METEOR scores, on all sets and conditions.

5.2 Hierarchical MT

Results using the hierarchical MT system are shown in Table 4. The hierarchical system used different tuning criteria in each genre. On newswire, the decoding weights were optimized so as to maximize BLEU, while on web and audio the tuning was based on $0.5\text{TER} + 0.5(1 - \text{BLEU})$ (referred to as TERBLEU in what follows). Note that these were the criteria for tuning the decoding weights; whenever corpus weights were used, they were taken from the phrase-based system.

It is interesting to see that gains from discriminative corpus weights carry over to the more powerful hierarchical MT system. On newswire Test, the gain in BLEU is 0.8; on web Test, the gain in TERBLEU is 0.3. On the audio Test set, the corpus weights provide 0.7 and 0.75 TERBLEU reduction on BN and BC, respectively. As with the

[5] The brevity penalty was calculated using the formula in the original IBM paper, rather than the more recent definition implemented in the NIST mteval-v11b.pl script.

Set	Corpus Weights	Newswire			Web		
		TER	BLEU	MTR	TER	BLEU	MTR
Tune	No	42.3	48.2	67.5	60.0	21.9	51.3
	Yes	41.5	49.6	68.7	59.1	22.8	52.3
Test	No	43.2	46.2	66.5	58.6	24.2	52.2
	Yes	42.4	47.5	67.8	58.1	25.4	52.9

(a) Results on Arabic text.

Set	Corpus Weights	BN			BC		
		TER	BLEU	MTR	TER	BLEU	MTR
Tune	No	56.0	22.9	55.5	57.3	21.7	55.0
	Yes	55.0	25.0	57.1	56.1	23.6	56.4
Test	No	53.0	25.3	57.7	55.9	22.9	55.4
	Yes	52.5	26.6	58.8	54.5	24.7	56.8

(b) Results on Arabic audio.

Table 3: Phrase-based trigram decoding results on the Arabic text and audio development sets. Decoding weights were optimized on the Tune set in order to directly minimize TER. Corpus weights were also optimized on Tune set, but based on expected TER.

phrase-based system, all metrics improve from the use of corpus weights, in all sets/conditions.

6 Conclusions

We have described a novel approach for estimating a weight for each sentence in a parallel training corpus so as to optimize MT performance of a phrase-based statistical MT system. The sentence weights influence MT performance by being applied to the phrase and lexical counts during translation rule extraction and probability estimation.

In order to ensure robust training of the weights, we expressed them as a function of sentence-level features. Then, we defined the process for optimizing the parameters of that function based on the expected TER of a translation hypothesis N-best on a designated tuning set.

The proposed technique was evaluated in the context of Arabic-English translation, on multiple conditions. It was shown that encouraging results were obtained by just using collection and genre ids as features. Interestingly, the discriminative corpus weights were found to be generally applicable and provided gains in a state-of-the-art hierarchical string-to-dependency-tree MT system, even though they were trained using the phrase-based MT system.

Next step is to include other sentence-level features, as described in Section 3.1. Finally, the technique described in this paper can be extended to address the estimation of weights at the alignment link level, based on link-level features. We believe that this will have a larger impact on the lexical and phrase translation probabilities, since there is a large number of parallel training sentences that are partially correct, i.e., they contain parts that are aligned and translated correctly, and parts that are wrong. The current procedure tries to assign a single weight to such sentences, so there is no way to distinguish between the “good” and “bad” portions of each sentence. Pushing the weight estimation at the alignment link level will alleviate this problem and will make the discriminative training more targeted.

Acknowledgments

This work was supported by DARPA/IPTO Contract No. HR0011-06-C-0022 under the GALE program.

References

Richard P. Brent. 1973. *Algorithms for Minimization Without Derivatives*. Prentice-Hall.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 2005. Adaptation of the transla-

Set	Corpus Weights	Newswire			Web		
		TER	BLEU	MTR	TER	BLEU	MTR
Tune	No	39.5	54.4	70.3	58.2	25.2	53.8
	Yes	38.8	55.6	71.2	58.0	25.5	54.0
Test	No	40.7	52.1	69.3	57.0	28.3	54.7
	Yes	40.1	52.9	69.8	56.6	28.5	55.0

(a) Results on Arabic text.

Set	Corpus Weights	BN			BC		
		TER	BLEU	MTR	TER	BLEU	MTR
Tune	No	54.9	27.3	58.0	55.8	26.1	57.4
	Yes	53.6	28.2	59.0	54.9	26.9	58.0
Test	No	51.6	29.9	60.0	54.4	27.6	57.7
	Yes	50.7	30.4	60.7	53.2	27.9	58.7

(b) Results on Arabic audio.

Table 4: Hierarchical 5-gram rescoring results on the Arabic text and audio development sets. Decoding/rescoring weights were optimized on the Tune set in order to directly maximize BLEU (for newswire) or minimize TERBLEU (for web and audio). Corpus weights were the same as the ones used in the corresponding phrase-based decodings.

tion model for statistical machine translation based on information retrieval. In *Proceedings of the 10th Annual Conference of European Association for Machine Translation*, pages 133–142.

Philipp Koehn and Josh Schroeder. 2007. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227.

Philipp Koehn. 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*.

Alon Lavie and Abhaya Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231.

Dong C. Liu and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528.

Yajuan Lu, Jin Huang, and Qun Liu. 2007. Improving statistical machine translation performance by training data selection and optimization. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 343–350.

Arindam Mandal, Dimitra Vergyri, Wen Wang, Jing Zheng, Andreas Stolcke, Gokhan Tur, Dilek Hakkani-Tür, and Necip Fazil Ayan. 2008. Efficient data selection for machine translation. In *Proceedings of the Second IEEE/ACL Spoken Language Technology Workshop*, pages 261–264.

Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

M. Ostendorf, A. Kannan, S. Austin, O. Kimball, R. Schwartz, and J. R. Rohlicek. 1991. Integration of diverse recognition methodologies through reevaluation of nbest sentence hypotheses. In *Proceedings of the DARPA Workshop on Speech and Natural Language*, pages 83–87.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

M. J. D. Powell. 1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, pages 155–162.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29(3):349–380.

Libin Shen, Jinxi Xu, and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 577–585.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciula, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2008. Language and translation model adaptation using comparable corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 857–866.

Keiji Yasuda, Ruiqiang Zhang, Hirofumi Yamamoto, and Eiichiro Sumita. 2008. Method of selecting training data to build a compact and efficient translation model. In *Proceedings of the Third International Joint Conference on Natural Language Processing*, volume II, pages 655–660.