Mixture-Model Adaptation for SMT

George Foster and **Roland Kuhn**
National Research Council Canada
first.last@nrc.gc.ca

Abstract

We describe a mixture-model approach to adapting a Statistical Machine Translation System for new domains, using weights that depend on text distances to mixture components. We investigate a number of variants on this approach, including cross-domain versus dynamic adaptation; linear versus loglinear mixtures; language and translation model adaptation; different methods of assigning weights; and granularity of the source unit being adapted to. The best methods achieve gains of approximately one BLEU percentage point over a state-of-the art non-adapted baseline system.

1 Introduction

Language varies significantly across different genres, topics, styles, etc. This affects empirical models: a model trained on a corpus of car-repair manuals, for instance, will not be well suited to an application in the field of tourism. Ideally, models should be trained on text that is representative of the area in which they will be used, but such text is not always available. This is especially the case for bilingual applications, because parallel training corpora are relatively rare and tend to be drawn from specific domains such as parliamentary proceedings.

In this paper we address the problem of adapting a statistical machine translation system by adjusting its parameters based on some information about a test domain. We assume two basic settings. In *cross-domain* adaptation, a small sample of parallel in-domain text is available, and it is used to optimize for translating future texts drawn from the same domain. In *dynamic* adaptation, no domain information is available ahead of time, and adaptation is based on the current source text under translation. Approaches developed for the two settings can be complementary: an in-domain development corpus can be used to make broad adjustments, which can then be fine tuned for individual source texts.

Our method is based on the classical technique of mixture modeling (Hastie et al., 2001). This involves dividing the training corpus into different components, training a model on each part, then weighting each model appropriately for the current context. Mixture modeling is a simple framework that encompasses many different variants, as described below. It is naturally fairly low dimensional, because as the number of sub-models increases, the amount of text available to train each, and therefore its reliability, decreases. This makes it suitable for discriminative SMT training, which is still a challenge for large parameter sets (Tillmann and Zhang, 2006; Liang et al., 2006).

Techniques for assigning mixture weights depend on the setting. In cross-domain adaptation, knowledge of both source and target texts in the in-domain sample can be used to optimize weights directly. In dynamic adaptation, training poses a problem because no reference text is available. Our solution is to construct a multi-domain development sample for learning parameter settings that are intended to generalize to new domains (ones not represented in the sample). We do not learn mixture weights directly with this method, because there is little hope

Proceedings of the Second Workshop on Statistical Machine Translation, pages 128–135,
Prague, June 2007.

that these would be well suited to new domains. Instead we attempt to learn how weights should be set as a function of distance. To our knowledge, this approach to dynamic adaptation for SMT is novel, and it is one of the main contributions of the paper.

A second contribution is a fairly broad investigation of the large space of alternatives defined by the mixture-modeling framework, using a simple genre-based corpus decomposition. We experimented with the following choices: cross-domain versus dynamic adaptation; linear versus loglinear mixtures; language and translation model adaptation; various text distance metrics; different ways of converting distance metrics into weights; and granularity of the source unit being adapted to.

The remainder of the paper is structured follows: section 2 briefly describes our phrase-based SMT system; section 3 describes mixture-model adaptation; section 4 gives experimental results; section 5 summarizes previous work; and section 6 concludes.

2 Phrase-based Statistical MT

Our baseline is a standard phrase-based SMT system (Koehn et al., 2003). Given a source sentence $\mathbf{s}$, this tries to find the target sentence $\hat{\mathbf{t}}$ that is the most likely translation of $\mathbf{s}$, using the Viterbi approximation:

$$\hat{\mathbf{t}} = \operatorname*{argmax}_{\mathbf{t}} p(\mathbf{t}|\mathbf{s}) \approx \operatorname*{argmax}_{\mathbf{t},\mathbf{a}} p(\mathbf{t},\mathbf{a}|\mathbf{s}),$$

where alignment $\mathbf{a} = (\tilde{s}_1, \tilde{t}_1, j_1), ..., (\tilde{s}_K, \tilde{t}_K, j_K)$; $\tilde{t}_k$ are target phrases such that $\mathbf{t} = \tilde{t}_1 \dots \tilde{t}_K$; $\tilde{s}_k$ are source phrases such that $\mathbf{s} = \tilde{s}_{j_1} \dots \tilde{s}_{j_K}$; and $\tilde{s}_k$ is the translation of the kth target phrase $\tilde{t}_k$.

To model $p(\mathbf{t},\mathbf{a}|\mathbf{s})$, we use a standard loglinear approach:

$$p(\mathbf{t},\mathbf{a}|\mathbf{s}) \propto \exp\left[\sum_i \alpha_i f_i(\mathbf{s},\mathbf{t},\mathbf{a})\right] \quad (1)$$

where each $f_i(\mathbf{s},\mathbf{t},\mathbf{a})$ is a feature function, and weights α_i are set using Och's algorithm (Och, 2003) to maximize the system's BLEU score (Papineni et al., 2001) on a development corpus. The features used in this study are: the length of $\mathbf{t}$; a single-parameter distortion penalty on phrase reordering in $\mathbf{a}$, as described in (Koehn et al., 2003); phrase translation model probabilities; and 4-gram language model probabilities $\log p(\mathbf{t})$, using Kneser-Ney smoothing as implemented in the SRILM toolkit.

Phrase translation model probabilities are features of the form: $\log p(\mathbf{s}|\mathbf{t},\mathbf{a}) \approx \sum_{k=1}^{K} \log p(\tilde{s}_k|\tilde{t}_k)$. We use two different estimates for the conditional probabilities $p(\tilde{t}|\tilde{s})$ and $p(\tilde{s}|\tilde{t})$: relative frequencies and "lexical" probabilities as described in (Zens and Ney, 2004). In both cases, the "forward" phrase probabilities $p(\tilde{t}|\tilde{s})$ are not used as features, but only as a filter on the set of possible translations: for each source phrase $\tilde{s}$ that matches some ngram in $\mathbf{s}$, only the 30 top-ranked translations $\tilde{t}$ according to $p(\tilde{t}|\tilde{s})$ are retained.

To derive the joint counts $c(\tilde{s},\tilde{t})$ from which $p(\tilde{s}|\tilde{t})$ and $p(\tilde{t}|\tilde{s})$ are estimated, we use the phrase induction algorithm described in (Koehn et al., 2003), with symmetrized word alignments generated using IBM model 2 (Brown et al., 1993).

3 Mixture-Model Adaptation

Our approach to mixture-model adaptation can be summarized by the following general algorithm:

1. Split the corpus into different components, according to some criterion.

2. Train a model on each corpus component.

3. Weight each model according to its fit with the test domain:

   - For cross-domain adaptation, set parameters using a development corpus drawn from the test domain, and use for all future documents.
   - For dynamic adaptation, set global parameters using a development corpus drawn from several different domains. Set mixture weights as a function of the distances from corpus components to the current source text.

4. Combine weighted component models into a single global model, and use it to translate as described in the previous section.

We now describe each aspect of this algorithm in more detail.

3.1 Corpus Decomposition

We partition the corpus into different genres, defined as being roughly identical to corpus source. This is the simplest way to exploit heterogeneous training material for adaptation. An alternative, which we have not explored, would be to cluster the corpus automatically according to topic.

3.2 Component Models

We adapt both language and translation model features within the overall loglinear combination (1).

To train translation models on each corpus component, we used a global IBM2 model for word alignment (in order to avoid degradation in alignment quality due to smaller training corpora), then extracted component-specific relative frequencies for phrase pairs. Lexical probabilities were also derived from the global IBM2 model, and were not adapted.

The procedure for training component-specific language models on the target halves of each corpus component is identical to the procedure for the global model described in section 2. In addition to the component models, we also used a large static global model.

3.3 Combining Framework

The most commonly-used framework for mixture models is a linear one:

$$p(x|h) = \sum_c \lambda_c p_c(x|h) \quad (2)$$

where $p(x|h)$ is either a language or translation model; $p_c(x|h)$ is a model trained on component c, and λ_c is the corresponding weight. An alternative, suggested by the form of the global model, is a loglinear combination:

$$p(x|h) = \prod_c p_c(x|h)^{\alpha_c}$$

where we write α_c to emphasize that in this case the mixing parameters are global weights, like the weights on the other features within the loglinear model. This is in contrast to linear mixing, where the combined model $p(x|h)$ receives a loglinear weight, but the weights on the components do not participate in the global loglinear combination. One consequence is that it is more difficult to set linear weights using standard minimum-error training techniques, which assume only a "flat" loglinear model.

3.4 Distance Metrics

We used four standard distance metrics to capture the relation between the current source or target text q and each corpus component.[1] All are monolingual—they are applied only to source text or only to target text.

The *tf/idf* metric commonly used in information retrieval is defined as $\cos(\mathbf{v}_c, \mathbf{v}_q)$, where $\mathbf{v_c}$ and $\mathbf{v_q}$ are vectors derived from component c and document q, each consisting of elements of the form: $-\tilde{p}(w) \log \tilde{p}_{doc}(w)$, where $\tilde{p}(w)$ is the relative frequency of word w within the component or document, and $p_{doc}(w)$ is the proportion of components it appears in.

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) is a technique for implicitly capturing the semantic properties of texts, based on the use of Singular Value Decomposition to produce a rank-reduced approximation of an original matrix of word and document frequencies. We applied this technique to all documents in the training corpus (as opposed to components), reduced the rank to 100, then calculated the projections of the component and document vectors described in the previous paragraph into the reduced space.

Perplexity (Jelinek, 1997) is a standard way of evaluating the quality of a language model on a test text. We define a perplexity-based distance metric $p_c(q)^{1/|q|}$, where $p_c(q)$ is the probability assigned to q by an ngram language model trained on component c.

The final distance metric, which we call *EM*, is based on expressing the probability of q as a word-level mixture model: $p(q) = \prod_{i=1}^{|q|} \sum_c d_c p_c(w_i|h_i)$, where $q = w_1 \ldots w_{|q|}$, and $p_c(w|h)$ is the ngram probability of w following word sequence h in component c. It is straighforward to use the EM algorithm to find the set of weights $\hat{d}_c, \forall c$ that maximizes the likelihood of q. The weight $\hat{d}_c$ is defined as the distance to component c. For all experiments described below, we used a probability difference threshold of 0.001 as the EM convergence criterion.

[1] Although we refer to these metrics as distances, most are in fact proximities, and we use the convention throughout that higher values mean closer.

3.5 Learning Adaptive Parameters

Our focus in this paper is on adaptation via mixture weights. However, we note that the usual loglinear parameter tuning described in section 2 can also be considered adaptation in the cross-domain setting, because learned preferences for word penalty, relative LM/TM weighting, etc, will reflect the target domain. This is not the case for dynamic adaptation, where, in the absence of an in-domain development corpus, the only information we can hope to glean are the weights on adapted models compared to other features of the system.

The method used for adapting mixture weights depends on both the combining framework (loglinear versus linear), and the adaptive setting (cross-domain versus dynamic), as described below.

3.5.1 Setting Loglinear Mixture Weights

When using a loglinear combining framework as described in section 3.3, mixture weights are set in the same way as the other loglinear parameters when performing cross-domain adaptation. Loglinear mixture models were not used for dynamic adaptation.

3.5.2 Setting Linear Mixture Weights

For both adaptive settings, linear mixture weights were set as a function of the distance metrics described in section 3.4. Given a set of metrics $\{D_1, \ldots, D_m\}$, let $d_{i,c}$ be the distance from the current text to component c according to metric D_i. A simple approach to weighting is to choose a single metric D_i, and set the weights in (2) to be proportional to the corresponding distances:

$$\lambda_c = d_{i,c} / \sum_{c'} d_{i,c'}. \tag{3}$$

Because different distance metrics may capture complementary information, and because optimal weights might be a non-linear function of distance, we also experimented with a linear combination of metrics transformed using a sigmoid function:

$$\lambda_c = \sum_{i=1}^{m} \frac{\beta_i}{1 + exp(a_i(b_i - d_{i,c}))} \tag{4}$$

where β_i reflects the relative predictive power of D_i, and the sigmoid parametes a_i and b_i can be set to selectively suppress contributions from components that are far away. Here we assume that β_i absorbs a normalization constant, so that the λ_c's sum to 1. In this approach, there are three parameters per distance metric to learn: β_i, a_i, and b_i. In general, these parameters are also specific to the particular model being adapted, ie the LM or the TM.

To optimize these parameters, we fixed global loglinear weights at values obtained with Och's algorithm using representative adapted models based on a single distance metric in (3), then used the Downhill Simplex algorithm (Press et al., 2002) to maximize BLEU score on the development corpus. For tractability, we followed standard practice with this technique and considered only monotonic alignments when decoding (Zens and Ney, 2004).

The two approaches just described avoid conditioning λ_c explicitly on c. This is necessary for dynamic adaptation, since any genre preferences learned from the development corpus cannot be expected to generalize. However, it is not necessary for cross-domain adaptation, where the genre of the development corpus is assumed to represent the test domain. Therefore, we also experimented with using Downhill Simplex optimization to *directly* learn the set of linear weights λ_c that yield maximum BLEU score on the development corpus.

A final variant on setting linear mixture weights is a hybrid between cross-domain and dynamic adaptation. In this approach, both the global loglinear weights and, if they are being used, the mixture parameters β_i, a_i, b_i are set to characterize the test domain as in cross-domain adaptation. When translating, however, distances to the current source text are used in (3) or (4) instead of distances to the in-domain development corpus. This obviously limits the metrics used to ones that depend only on source text.

4 Experiments

All experiments were run on the NIST MT evaluation 2006 Chinese data set. Table 1 summarizes the corpora used. The training corpus was divided into seven components according to genre; in all cases these were identical to LDC corpora, with the exception of the *Newswire* component, which was amalgamated from several smaller corpora. The target genre for cross-domain adaptation was newswire, for which high-quality training material is available. The cross-domain development set *NIST04-nw* is the newswire subset of the NIST 2004 evaluation set, and the dynamic adaptation development set *NIST04-mix* is a balanced mixed-genre subset of NIST 2004. The NIST 2005 evaluation set was used for testing cross-domain adaptation, and the NIST 2006 evaluation set (both the "GALE" and "NIST" parts) was used to test dynamic adaptation.

Because different development corpora are used for cross-domain and dynamic adaptation, we trained one static baseline model for each of these adaptation settings, on the corresponding development set.

All results given in this section are BLEU scores.

role	corpus	genres	sent
train	FBIS04	nw	182k
	HK Hans	proceedings	1,375k
	HK Laws	legal	475k
	HK News	press release	740k
	Newswire	nw	26k
	Sinorama	news mag	366k
	UN	proceedings	4,979k
dev	NIST04-nw	nw	901
	NIST04-mix	nw, sp, ed	889
test	NIST05	nw	1,082
	NIST06-GALE	nw, ng, bn, bc	2,276
	NIST06-NIST	nw, ng, bn	1,664

Table 1: Corpora. In the *genres* column: nw = newswire, sp = speeches, ed = editorial, ng = newsgroup, bn = broadcast news, and bc = broadcast conversation.

4.1 Linear versus Loglinear Combination

Table 2 shows a comparison between linear and loglinear mixing frameworks, with uniform weights used in the linear mixture. Both types of mixture model are better than the baseline, but the linear mixture is slightly better than the loglinear mixture. This is quite surprising, because these results are on the *development* set: the loglinear model tunes its component weights on this set, whereas the linear model only adjusts global LM and TM weights. We speculated that this may have been due to non-smooth component models, and tried various smoothing schemes, including Kneser-Ney phrase table smoothing similar to that described in (Foster et al., 2006), and binary features to indicate phrase-pair presence within different components. None helped, however, and we conclude that the problem is most likely that Och's algorithm is unable to find a good maximimum in this setting. Due to this result, all experiments we describe below involve linear mixtures only.

combination	adapted model		
	LM	TM	LM+TM
baseline	30.2	30.2	30.2
loglinear mixture	30.9	31.2	31.4
uniform linear mixture	31.2	31.1	31.8

Table 2: Linear versus loglinear combinations on NIST04-nw.

4.2 Distance Metrics for Weighting

Table 3 compares the performance of all distance metrics described in section 3.4 when used on their own as defined in (3). The difference between them is fairly small, but appears to be consistent across LM and TM adaptation and (for the LM metrics) across source and target side matching. In general, LM metrics seem to have a slight advantage over the vector space metrics, with EM being the best overall. We focus on this metric for most of the experiments that follow.

metric	source text		target text	
	LM	TM	LM	TM
tf/idf	31.3	31.3	31.1	31.1
LSA	31.5	31.6		
perplexity	31.6	31.3	31.7	31.5
EM	31.7	31.6	32.1	31.3

Table 3: Distance metrics for linear combination on the NIST04-nw development set. (Entries in the top right corner are missing due to lack of time.)

Table 4 shows the performance of the parameterized weighting function described by (4), with source-side EM and LSA metrics as inputs. This is compared to direct weight optimization, as both these techniques use Downhill Simplex for parameter tuning. Unfortunately, neither is able to beat the performance of the normalized source-side EM metric on its own (reproduced on the first line from table 3). In additional tests we verified that this also holds for the test corpus. We speculate that this disappointing result is due to compromises made in order to run Downhill Simplex efficiently, including holding global weights fixed, using only a single starting point, and running with monotone decoding.

weighting	LM	TM
EM-src, direct	31.7	31.6
EM-src + LSA-src, parameterized	31.0	30.0
direct optimization	31.7	30.2

Table 4: Weighting techniques for linear combination on the NIST04-nw development set.

4.3 Cross-Domain versus Dynamic Adaptation

Table 5 shows results for cross-domain adaptation, using the source-side EM metric for linear weighting. Both LM and TM adaptation are effective, with test-set improvements of approximately 1 BLEU point over the baseline for LM adaptation and somewhat less for TM adaptation. Performance also improves on the NIST06 out-of-domain test set (although this set includes a newswire portion as well). However, combined LM and TM adaptation is not better than LM adaptation on its own, indicating that the individual adapted models may be capturing the same information.

model	dev	test	
	nist04-nw	nist05	nist06-nist
baseline	30.2	30.3	26.5
EM-src LM	31.7	31.2	27.8
EM-src TM	31.6	30.9	27.3
EM-src LM+TM	32.5	31.2	27.7

Table 5: Cross-Domain adaptation results.

Table 6 contains results for dynamic adaptation, using the source-side EM metric for linear weighting. In this setting, TM adaptation is much less effective, not significantly better than the baseline; performance of combined LM and TM adaptation is also lower. However, LM adaptation improves over the baseline by up to a BLEU point. The performance of cross domain adaptation (reproduced from table 5 on the second line) is slightly better for the in-domain test set (NIST05), but worse than dynamic adaptation on the two mixed-domain sets.

model	dev	test		
	nist04-mix	nist05	nist06-nist	nist06-gale
baseline	31.9	30.4	27.6	12.9
cross LM	n/a	31.2	27.8	12.5
LM	32.8	30.8	28.6	13.4
TM	32.4	30.7	27.6	12.8
LM+TM	33.4	30.8	28.5	13.0

Table 6: Dynamic adaptation results, using src-side EM distances.

model	NIST05
baseline	30.3
cross EM-src LM	31.2
cross EM-src TM	30.9
hybrid EM-src LM	30.9
hybrid EM-src TM	30.7

Table 7: Hybrid adaptation results.

Table 7 shows results for the hybrid approach described at the end of section 3.5.2: global weights are learned on NIST04-nw, but linear weights are derived dynamically from the current test file. Performance drops slightly compared to pure cross-domain adaptation, indicating that it may be important to have a good fit between global and mixture weights.

4.4 Source Granularity

The results of the final experiment, to determine the effects of source granularity on dynamic adaptation, are shown in table 8. Source-side EM distances are applied to the whole test set, to genres within the set, and to each document individually. Global weights were tuned specifically for each of these conditions. There appears to be little difference among these approaches, although genre-based adaptation perhaps has a slight advantage.

granularity	dev	test		
	nist04-mix	nist05	nist06-nist	nist06-gale
baseline	31.9	30.4	27.6	12.9
file	32.4	30.8	28.6	13.4
genre	32.5	31.1	28.9	13.2
document	32.9	30.9	28.6	13.4

Table 8: The effects of source granularity on dynamic adaptation.

5 Related Work

Mixture modeling is a standard technique in machine learning (Hastie et al., 2001). It has been widely used to adapt language models for speech recognition and other applications, for instance using cross-domain topic mixtures, (Iyer and Ostendorf, 1999), dynamic topic mixtures (Kneser and Steinbiss, 1993), hierachical mixtures (Florian and Yarowsky, 1999), and cache mixtures (Kuhn and De Mori, 1990).

Most previous work on adaptive SMT focuses on the use of IR techniques to identify a relevant subset of the training corpus from which an adapted model can be learned. Byrne et al (2003) use cosine distance from the current source document to find relevant parallel texts for training an adapted translation model, with background information for smoothing alignments. Hildebrand et al (1995) describe a similar approach, but apply it at the sentence level, and use it for language model as well as translation model adaptation. They rely on a perplexity heuristic to determine an optimal size for the relevant subset. Zhao et al (2004) apply a slightly different sentence-level strategy to language model adaptation, first generating an nbest list with a baseline system, then finding similar sentences in a monolingual target-language corpus. This approach has the advantage of not limiting LM adaptation to a parallel corpus, but the disadvantage of requiring two translation passes (one to generate the nbest lists, and another to translate with the adapted model).

Ueffing (2006) describes a *self-training* approach that also uses a two-pass algorithm. A baseline system generates translations that, after confidence filtering, are used to construct a parallel corpus based on the test set. Standard phrase-extraction techniques are then applied to extract an adapted phrase table from the system's own output.

Finally, Zhang et al (2006) cluster the parallel training corpus using an algorithm that heuristically minimizes the average entropy of source-side and target-side language models over a fixed number of clusters. Each source sentence is then decoded using the language model trained on the cluster that assigns highest likelihood to that sentence.

The work we present here is complementary to both the IR approaches and Ueffing's method because it provides a way of exploiting a pre-established corpus division. This has the potential to allow sentences having little surface similarity to the current source text to contribute statistics that may be relevant to its translation, for instance by raising the probability of rare but pertinent words. Our work can also be seen as extending all previous approaches in that it assigns weights to components depending on their degree of relevance, rather than assuming a binary distinction between relevant and non-relevant components.

6 Conclusion and Future Work

We have investigated a number of approaches to mixture-based adaptation using genres for Chinese to English translation. The most successful is to weight component models in proportion to maximum-likelihood (EM) weights for the current text given an ngram language model mixture trained on corpus components. This resulted in gains of around one BLEU point. A more sophisticated approach that attempts to transform and combine multiple distance metrics did not yield positive results, probably due to an unsucessful optmization procedure.

Other conclusions are: linear mixtures are more tractable than loglinear ones; LM-based metrics are better than VS-based ones; LM adaptation works well, and adding an adapted TM yields no improvement; cross-domain adaptation is optimal, but dynamic adaptation is a good fallback strategy; and source granularity at the genre level is better than the document or test-set level.

In future work, we plan to improve the optimization procedure for parameterized weight functions. We will also look at bilingual metrics for cross-domain adaptation, and investigate better combinations of cross-domain and dynamic adaptation.

References

Peter F. Brown, Stephen A. Della Pietra, Vincent Della J. Pietra, and Robert L. Mercer. 1993. The mathematics of Machine Translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312, June.

W. Byrne, S. Khudanpur, W. Kim, S. Kumar, P. Pecina, P. Virga, P. Xu, and D. Yarowsky. 2003. The JHU 2003 Chinese-English Machine Translation System. In *MT Summit IX*, New Orleans, September.

S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. 1990. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407.

Radu Florian and David Yarowsky. 1999. Dynamic non-local language modeling via hierarchical topic-based adaptation. In *ACL 1999*, pages 167–174, College Park, Maryland, June.

George Foster, Roland Kuhn, and Howard Johnson. 2006. Phrasetable smoothing for statistical machine translation. In *EMNLP 2006*, Sydney, Australia.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer.

Almut Silja Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 1995. Adaptation of the translation model for statistical machine translation based on information retrieval. In *EAMT 1995*, Budapest, May.

R. Iyer and M. Ostendorf. 1999. Modeling long distance dependence in language: Topic mixtures vs. dynamic cache models. In *IEEE Trans on Speech and Language Processing, 1999*.

Frederick Jelinek. 1997. *Statistical Methods for Speech Recognition*. MIT Press.

Reinhard Kneser and Volker Steinbiss. 1993. On the dynamic adaptation of stochastic language models. In *ICASSP 1993*, pages 586–589, Minneapolis, Minnesota. IEEE.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL 2003*, pages 127–133.

Roland Kuhn and Renato De Mori. 1990. A cache-based natural language model for speech recognition. *IEEE Trans on PAMI*, 12(6):570–583, June.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *ACL 2006*

Franz Josef Och. 2003. Minimum error rate training for statistical machine translation. In *ACL 2003*, Sapporo, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: A method for automatic evaluation of Machine Translation. Technical Report RC22176, IBM, September.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 2002. *Numerical Recipes in C++*. Cambridge University Press, Cambridge, UK.

Christoph Tillmann and Tong Zhang. 2006. A discriminative global training algorithm for statistical MT. In *ACL 2006*.

Nicola Ueffing. 2006. Self-training for machine translation. In *NIPS 2006 Workshop on MLIA*, Whistler, B.C., December.

Richard Zens and Hermann Ney. 2004. Improvements in phrase-based statistical machine translation. In *HLT/NAACL 2004*, Boston, May.

R. Zhang, H. Yamamoto, M. Paul, H. Okuma, K. Yasuda, Y. Lepage, E. Denoual, D. Mochihashi, A. Finch, and E. Sumita. 2006. The NiCT-ATR statistical machine translation system for the IWSLT 2006 evaluation. In *IWSLT 2006*.

Bing Zhao, Matthias Eck, and Stephan Vogel. 2004. Language model adaptation for statistical machine translation with structured query models. In *COLING 2004*, Geneva, August.