A Look inside the ITC-irst SMT System

M. Cettolo, M. Federico, N. Bertoldi, R. Cattoni and B. Chen
ITC-irst - Centro per la Ricerca Scientifica e Tecnologica
38050 Povo (Trento), ITALY
cettolo@itc.it

Abstract

This paper presents a look inside the ITC-irst large-vocabulary SMT system developed for the NIST 2005 Chinese-to-English evaluation campaign.

Experiments on official NIST test sets provide a thorough overview of the performance of the system, supplying information on how single components contribute to the global performance.

The presented system exhibits performance comparable to that of the best systems participating in the NIST 2002-2004 MT evaluation campaigns: on the three test sets, achieved BLEU scores are 26.35%, 26.92% and 28.13%, respectively.

1 Introduction

At this time, statistical MT (SMT) has empirically proven to be a very, if not the most, competitive approach. Similarly to what happened with automatic speech recognition and other human language technologies, the systematic application of empirical evaluations over time, both within and across research laboratories, has dramatically boosted progress in this field.

Competition and the exchange of knowledge have considerably sped up the selection process of *ideas* which steadily emerge in the SMT community. In fact, the common pursuing of the same theoretical framework has produced more complex systems, which seems rather inevitable when the same technology is pushed towards its limits.

This paper presents a look inside the ITC-irst large-vocabulary SMT system developed for the NIST 2005 Chinese-to-English evaluation campaign. The presented system provides performance comparable to that of the best systems participating in the NIST 2004 MT evaluation campaign.

The paper is organized as follows. Section 2 presents the general log-linear framework to SMT and gives an overview of the phrase-based SMT architecture of the ITC-irst system. Hence, Section 3 lists the steps implementing the pre- and post-processing stages of text data. Sections 4, 5 and 6 cover training procedures and phrase extraction. Finally, after the sketch in Section 7 about rules with which explicit suggestions can be provided to the decoder, Section 8 throughout presents experimental results that allowed our system to reach state-of-the-art MT performance.

2 The ITC-irst SMT System

Given a source string $\mathbf{f}$ and a target string $\mathbf{e}$, the framework of maximum entropy (Berger et al., 1996) provides a mean to directly address the posterior probability $\Pr(\mathbf{e} \mid \mathbf{f})$. By introducing the hidden word *alignment* variable $\mathbf{a}$, the usual SMT optimization criterion is expressed by:

$$\begin{aligned}\mathbf{e}^* &= \arg\max_{\mathbf{e}} \Pr(\mathbf{e} \mid \mathbf{f}) \\ &= \arg\max_{\mathbf{e}} \sum_{\mathbf{a}} \Pr(\mathbf{e}, \mathbf{a} \mid \mathbf{f}) \\ &\approx \arg\max_{\mathbf{e},\mathbf{a}} \Pr(\mathbf{e}, \mathbf{a} \mid \mathbf{f}) \qquad (1)\end{aligned}$$

The conditional distribution $\Pr(\mathbf{e}, \mathbf{a} \mid \mathbf{f})$ is determined through suitable real valued feature functions $h_r(\mathbf{e}, \mathbf{f}, \mathbf{a}), r = 1 \dots R$, and takes the parametric form:

$$p_\lambda(\mathbf{e}, \mathbf{a} \mid \mathbf{f}) \propto \exp\{\sum_{r=1}^{R} \lambda_r h_r(\mathbf{e}, \mathbf{f}, \mathbf{a})\} \qquad (2)$$

In our system (Bertoldi et al., 2004), the following extension of IBM Model 4 (Brown et al., 1993) to phrases is introduced: $\mathbf{e}$ is intended as a sequence of target phrases $\tilde{e}_1 \dots \tilde{e}_l$; *fertilities* $\phi_0, \dots, \phi_l$ tell the number of source positions covered by target phrases, including the `null` word; *permutations* $\pi_0, \dots, \pi_l$ indicate the corresponding source positions; *tablets* $\tau_0, \dots, \tau_l$ indicate the corresponding lists of source words. Notice that any target phrase $\tilde{e}_i$ might either not cover any source position ($\phi_i = 0$) or might cover a set of consecutive source positions ($\phi_i > 0$). The target `null` word might cover any set of source positions.
Hence, the following 7 feature functions are considered:

- target language model
- fertility model of `null` word
- permutation model of `null` word
- fertility model of target phrases
- translation model for phrases and `null word`
- negative permutation model, which models non-monotone coverage of source positions
- positive permutation model, which models monotone coverage of source positions.

While feature functions are estimated from a word-aligned parallel corpus and from monolingual texts in the target language, scaling factors of the log-linear model are estimated by a *minimum error training* procedure (Cettolo and Federico, 2004).

Figure 1 illustrates how the translation of a source string is performed.

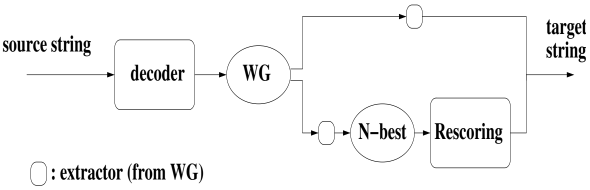

Figure 1: The ITC-irst SMT system.

In the first pass, a search algorithm (decoder) computes a word graph of translation hypotheses. Hence, either the best translation hypothesis is directly extracted from the word graph, or an N-best list of translations is computed (Tran et al., 1996). In the second pass, the N-best translations are reranked by applying additional knowledge sources as well expressed with feature functions. Finally, the top ranking translation is output.

The decoder exploits dynamic programming, i.e. the optimal solution is computed by expanding and recombining previously computed partial theories. A theory is described by its *state*, which is the only information needed for its expansion. If two expanded theories share the same state then they are recombined, i.e. only the best scoring one is stored for further expansions. To limit the large number of generated theories some approximations are introduced:

Beam search: at each expansion less promising theories are pruned by any of the following criteria:

- *threshold pruning*: the theory's score is smaller than the current optimum score times a given threshold;
- *histogram pruning*: the theory's score is not among the top K best scores.

The same criteria, but with different thresholds, are applied to all theories covering the same set of source positions, and to all theories with the same output length.

Reordering constraints: covered source positions are selected by applying the so-called IBM constraint, which limits to 3 the number of vacant positions on the left hand; moreover, the maximum distortion is also limited to some value V. Two settings are used, $V = 1$ and $V = 5$, which correspond to monotone and non-monotone search, respectively. Notice that phrase-based translation permits anyway some intra-phrase re-ordering.

3 Pre- and Post-processing

The following pre-processing steps aim at normalizing source and target texts both for training and testing:

Tokenization. Words are separated from punctuation, with the exception of acronyms and abbreviations.

Splitting. Long sentences represent a problem in training word-alignment models, as sentence length impacts on computational complexity and parameter estimation. Hence, long parallel sentences are split into shorter portions (chunks). Likelihood of chunks is measured with IBM Model-1 lexicon statistics. Candidate positions for splitting are selected according to strong punctuation and sentence length. The splitting procedure is binary, i.e. each sentence is split into two chunks, and recursive, i.e. the process is repeated on each chunk. The search for the best splitting point is exhaustive. Sentence/chunk length below a given threshold is the termination condition.

Number normalization. Numbers written in textual form are transformed into digits, with few exceptions, e.g. the word "million". Ordinal numbers and percentages are also managed in this way.

Case normalization. All words are put in lowercase. This also applies to Western words appearing in Chinese texts.

Chinese word segmentation. This step is performed with ICTCLAS, a publicly available tool developed at the Institute of Computing Technology, Beijing, (Zhang et al., 2003).

Post-processing basically involves *case restoration* of the English output, i.e. recovering word case information of proper names, words after strong punctuation, etc. We used the `disambig` tool[1], fed with a 3-gram case-sensitive language model, estimated on the same sample used for the target language model.

4 Word-based Alignment

After preprocessing, Viterbi alignments based on IBM Model-4 from source to target words, and vice-versa, are computed by means of the GIZA++ toolkit (Och and Ney, 2003). The corpus provided to GIZA++ to train the IBM Model-1 only (for sentence splitting and re-ranking) contains all the resources available for the NIST 2004 evaluation campaign, large data condition. For computational reasons long sentences have been cut off. The corpus provided to GIZA++ for the full training up to IBM Model-4 includes all the resources available with the exception of the `LDC2004E12` corpus *UN Chinese-English Parallel Text Version 2*: using this large resource does not appear to improve system performance.

Table 1 reports statistics on training data.

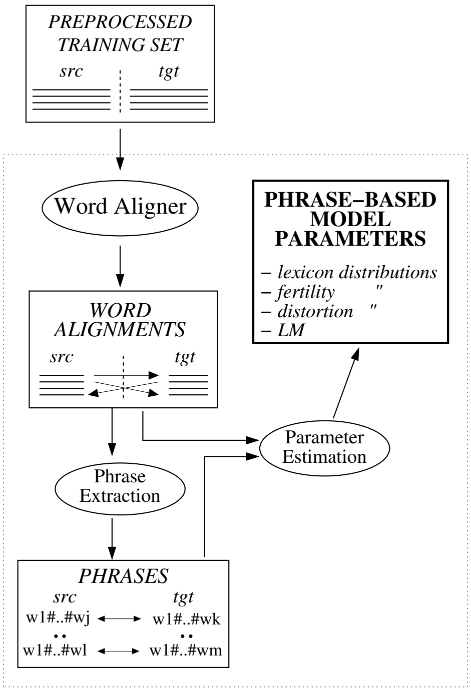

Figure 2: Training of the phrase-based model: estimation of bi-directional word-alignment, phrase extraction, estimation of phrase-based features.

5 Phrase extraction and model training

Starting from the parallel training corpus provided with direct and inverted alignments (Figure 2), phrase-pair statistics are extracted as follows.
Given a sentence pair $(\mathbf{f}, \mathbf{e})$, of lengths m and l, respectively, and its direct and inverted alignments $\mathbf{a}$ and $\mathbf{b}$, we define the union alignment by:

$$\mathbf{c} = \{(j,i) : a_j = i \vee b_i = j\} \subseteq \{1,..m\} \times \{1,..,l\}$$

Phrase-pairs extracted from $(\mathbf{f}, \mathbf{e})$ correspond to subintervals of the source and target positions, $J = [j_1, j_2]$ and $I = [i_1, i_2]$, such that the union alignment $\mathbf{c}$ links all positions of J into I and all positions of I into J. In general, phrases are extracted with maximum length in the source and target defined by the parameters J_{max} and I_{max}. Such a set $\mathcal{P}$ of phrase-pairs is efficiently computed by the algorithm shown in Figure 3.

The algorithm exploits some support arrays which can be computed in linear time:

- $minC_e[1\,..\,m]/maxC_e[1\,..\,m]$, which map each source position j into the min/max target position i according to $\mathbf{c}$
- $minC_f[1\,..\,l]/maxC_e[1\,..\,l]$, which map each target i into the min/max source j according to $\mathbf{c}$

Notice that the value 0 is used when $\mathbf{c}$ does not provide any correspondence.

The algorithm works as follows. In lines (1-10), for each target interval $I = [i_1, i_2]$ of length $\leq I_{max}$ it finds an interval $J' = [j'_1, j'_2]$ such that:

- $length(J') \leq J_{max}$
- target positions corresponding to J' fall within I

In lines (11-18), all intervals $J = [j_1, j_2]$ extending J' are computed, s.t.:

- $J \subseteq [1\,..\,m]$ and length$(J) \leq J_{max}$
- $J \setminus J'$ only includes words not aligned with any target word,

and phrase pair $(f[j_1\,..\,j_2], e[i_1\,..\,i_2])$ is added to $\mathcal{P}$. Time complexity of the algorithm is $O(l\, I_{max} {J_{max}}^2)$.

Given a training sample provided with best direct and inverse alignments

$$\{(\mathbf{f}^s, \mathbf{e}^s, \mathbf{a}^s, \mathbf{b}^s) : s = 1, \ldots, S\}$$

all phrase-pairs are collected through the above algorithm.

[1] `www.speech.sri.com/projects/srilm/manpages/`

```
EXTRACT-PHRASES($f[], e[], minC_e[], maxC_e[], minC_f[], idmaxC_f[]$)
$\mathcal{P} \leftarrow \emptyset$
for $i_1 \leftarrow 1$ to $l$
    do $j'_1 \leftarrow m$ ; $j'_2 \leftarrow 0$
       for $i_2 \leftarrow i_1$ to $min(l, i_1 + I_{max} - 1)$
           do $j'_1 \leftarrow min(j'_1, minC_f[i_2])$ ; $j'_2 \leftarrow max(j'_2, maxC_f[i_2])$
              if $j'_2 > 0 \wedge j'_2 - j'_1 < J_{max}$
                 then $i'_1 \leftarrow l$ ; $i'_2 \leftarrow 0$
                      for $j \leftarrow j'_1$ to $j'_2$
                          do $i'_1 \leftarrow min(i'_1, minC_e[j])$ ; $i'_2 \leftarrow max(i'_2, maxC_e[j])$
                      if $i_1 \geq i'_1 \wedge i'_2 \geq i_2$
                         then $j_1 \leftarrow j'_1$ ; $j_2 \leftarrow j'_1$
                              repeat
                                    repeat $push(\mathcal{P}, (f[j_1..j_2], e[i_1..i_2]))$
                                           $j_1 \leftarrow j_1 - 1$
                                    until $j_1 < 1 \vee (j_2 - j_1) \geq J_{max} \vee maxC_e[j_1] > 0$
                                    $j_1 \leftarrow j'_1; j_2 \leftarrow j_2 + 1$
                              until $j_2 > m \vee (j_2 - j_1) \geq J_{max} \vee maxC_e[j_2] > 0$
return $\mathcal{P}$
```

Figure 3: Phrase-pair extraction algorithm.

Chinese running words	71M
English running words	77M
Chinese vocabulary	157K
English vocabulary	214K
phrase pairs len 4	41M
pruned phrase pairs len 4	5.2M
phrase pairs len 8	99M
pruned phrase pairs len 8	8.4M

Table 1: Statistics of training data and of extracted phrases for length 4 and 8.

Phrase pruning

More reliable phrase-pairs can be obtained by filtering out pairs for which:

- lengths of source and target differ too much
- strong punctuation is not preserved between source and target
- frequency is below a given threshold, set to 2.

Parameter estimation

Given a sample of phrase-pairs, phrase-translation probabilities $\Pr(\tilde{f} \mid \tilde{e}, \phi)$ and phrase-fertility $\Pr(\phi \mid \tilde{e})$ probabilities are computed by applying the Witten-Bell smoothing, as follows:

$$\tilde{p}(\phi \mid \tilde{e}) = \frac{N(\phi, \tilde{e})}{N(\tilde{e}) + D(\tilde{e})} \quad (3)$$

$$\tilde{p}(\tilde{f} \mid \phi, \tilde{e}) = \frac{N(\tilde{f}, \phi, \tilde{e})}{N(\phi, \tilde{e}) + D(\phi, \tilde{e})} \quad (4)$$

where $N(\cdot)$ indicates the number of occurrences, $D(\tilde{e})$ is the number of different ϕ observed with $\tilde{e}$, and $D(\phi, \tilde{e})$ is the number of different source phrases $\tilde{f}$ observed with ϕ and $\tilde{e}$.

Additional pruning is applied by only taking for each source phrase the most probable target translations up to .95 of the total probability and no more than 30.

Finally, as additional feature function, also the inverted translation probability $\Pr(\tilde{e} \mid \tilde{f})$ is computed analogously.

Language Model

Target language models (LMs) used by the decoder and re-ranking modules are, respectively, estimated from 3-gram and 4-gram statistics by applying the *modified Kneser-Ney* smoothing method (Goodman and Chen, 1998). LMs are estimated with an in-house software toolkit which also provides a compact binary representation of the LM.

6 Tuning and Re-Ranking

6.1 Decoder Tuning

The decoder exploits a log-linear interpolation of seven feature functions. The corresponding weights can be estimated by optimizing the BLEU score (Papineni et al., 2001), as proposed in (Och, 2004). We apply an iterative procedure described in (Cettolo and Federico, 2004), which uses the *simplex* algorithm for multi-variate function optimization. Figure 4 shows the procedure. By using some values $\{\lambda_1, \lambda_2 \ldots\}$ at step t, the decoder translates sentences of a development set. The set of 1-best translation hypotheses is compared with the reference translations, in terms of BLEU score. On the basis of this value, the simplex algorithm selects a new configuration $\{\lambda_1, \lambda_2 \ldots\}$ which goes toward a (local) minimum of the function. The procedure is iterated until a convergence criterion is met and eventually the optimal parameters are generated.

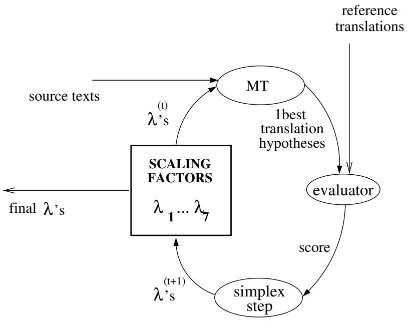

Figure 4: Estimation of MT parameters.

6.2 Re-Ranking Module

Once the parameters of the MT decoder have been optimized, it is able to generate word graphs at its best. From each word graph N-best translation hypotheses are extracted.

Each entry of the N-best list is characterized by the 7 scores mentioned above, computed during the MT decoding. For the sake of re-ranking, additional feature functions are used which can potentially introduce new knowledge in the translation process. We have evaluated the impact of the following additional models:

IBM model 1 as defined in (Brown et al., 1993). It should capture lexical co-occurrences in the source and target strings.

Length of the target string. It should favour longer hypothesis, which are intrinsically penalized in a statistical framework with respect to those whose scores are defined by less factors.

4gr target LM trained on the same data used for training the 3gr LM used during the decoding. It should better cover the fluency of the target string.

3gr test LM estimated on reference translations of test sets of different evaluation campaigns.

For re-ranking N-best lists, weights of feature scores can be again estimated by means of a minimum error rate procedure similar to that employed for optimizing the parameters of the decoder. Figure 5 illustrates the scheme for the estimation of optimal re-ranking weights.

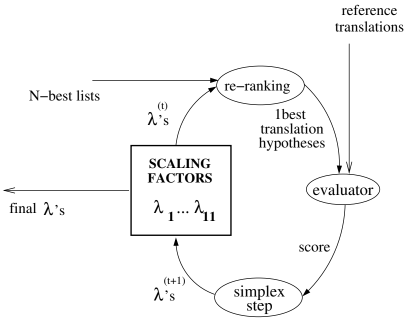

Figure 5: Estimation of weights for re-ranking

7 Rules

A specific feature has been added to the log-linear model in order to accept translation suggestions in an on-line mode. The purpose of providing such translation alternatives is either to exploit external knowledge which becomes available only at runtime or to enforce what is already present in the training data.

Translation alternatives are automatically generated by analyzing the input sentence and by exploiting a set of human-designed rules, which can possibly take into account the input context and are weighted according to their reliability.

Around fifty phrase-level rules have been written to translate syntax-fixed input expressions, such as news headings and time and numerical expressions. As examples, time expressions include the month name, the week-day name, the time of the day or a time interval; numerical expressions include percentages, fractions, large numbers, and so on.

set	#doc	Chinese	English
NIST02	100	24K	28K
NIST03	100	26K	29K
NIST04	200	51K	61K

Table 2: Test set statistics: number of documents and running words. For English, the gold reference is only considered.

8 Experimental Results

Experiments described in this section aim at providing a thorough overview of the performance of the ITC-irst MT system, supplying information on how single components contribute to the global performance.

8.1 Training and Test Data

Performance evaluation are mainly reported on the test set used in the NIST 2004 MT evaluation, for

	decoder						
	baseline	+V=5	+| phrase |=8	+rules	+filt	+2TM	+NE
1best	25.48	25.92	27.01	27.71	28.05	28.32	28.50
1000-best re-ranking:							
flat (a,b,c,d)	26.96	27.59	28.17	28.77	28.95	29.04	29.30
est. (a,b,c,d)							29.40

Table 5: Translation results (Bleu score) for different setups of the decoder. The evaluation is case-insensitive.

	baseline
1best	25.48
1000-best re-ranking:	
IBM mdl 1 (a)	26.30
tgt len (b)	25.83
tgt 4grLM (c)	25.86
3grTstLM (d)	25.52
flat (a,b,c)	26.83
flat (a,b,c,d)	26.96

Table 3: Re-ranking module performance (Bleu score): additional features are evaluated on their own and in combination. The evaluation is case-insensitive.

	decoder	
	baseline	tuned
1best	25.48	26.03
1000-best re-ranking:		
flat (a,b,c)	26.83	26.49
est (a,b,c)	26.98	26.99

Table 4: Optimized vs. baseline decoder: performance (Bleu score) on 1-best and after 1000-best re-ranking. The evaluation is case-insensitive.

	true-case	
case-ins.	3gr	3gr+headings
29.40	27.86	28.13

Table 6: Case restoration module performance.

the Chinese-to-English task.

Details on training (Table 1) and test (Table 2) data can be found in the NIST web site[2]. System development was done by measuring improvements on the NIST 2004 test set, in terms of BLEU score (Papineni et al., 2001), which adequately correlates with human subjective evaluations. The best final system was also run on NIST 2002 and 2003 test sets, for checking purposes. Each test set provides four reference translations.

8.2 Baseline and Re-ranking Results

The baseline decoder performs a monotone search (see Section 2) on phrases long up to 4 words (Section 5), employing weights equal to 1 in the log-linear interpolation of the 7 feature functions mentioned in Section 2. Table 3 shows performance of the baseline system both evaluating its first bests and re-ranking the 1000-best translation hypotheses extracted by its word graphs. Re-ranking was done by adding scores of the additional features described in Section 6.2. Improvements are given at the level both of single new features and of their flat combination. "Flat" means that re-ranking weights were not estimated through any automatic procedure, but were empirically fixed to 1 with few exceptions. Whenever re-ranking weights have been estimated by means of the algorithm of Section 6.2, the experiment will be referred as "est".

Estimation of optimal decoder weights (Section 6) yields to performance reported in Table 4. The optimal decoder works better than the baseline only at the 1-best level; on the contrary, after the re-ranking step, performances of optimal and baseline decoders are practically the same. Since the optimization of the decoder weights is costly, from here on the decoder will employ always empirically fixed weights for log-linear interpolating statistical models.

8.3 Decoding Advances

Table 5 provides performance of different decoders, incrementally upgraded as described in the following, both at the 1-best level, and after the 1000-best re-ranking. In addition to the baseline one, the other evaluated decoders are:

V=5: instead of monotone search, phrase reordering is possible;

| **phrase** |=**8**: the maximum length of phrases is extended from 4 to 8.

[2] www.nist.gov/speech/tests/mt/

rules: decoder is fed with suggestions from the rules briefly mentioned in Section 7.

filt: the smart filtering of phrases mentioned in Section 5 is performed during the phrase selection stage.

2TM: in addition to the direct phrase-based translation model, the log-linear interpolation also includes the inverse phrase-based translation model;

NE: training data is augmented with named entities from the available resource `LDC2003E01`.

The final decoder improves the baseline by almost 12% at 1best level and by almost 9% after the 1000-best re-ranking. The 1000-best lists provided by the final decoder were also re-ranked with weights estimated through the minimum error training procedure of Section 6.2, yielding a further, although quite small, improvement.

8.4 Case Restoration Results

Table 6 provides performance of the case restoration module referred in Section 3, applied to the set of translations hypothesized by the final best decoder.

The degradation of the BLEU score from case-insensitive to true-case evaluation is quite limited. Moreover, exploiting the knowledge that the headings of newswire are cased in a different way with respect to the body of news, a special filter was appended to the post-processing, allowing a further performance improvement.

8.5 Performance on NIST Test Sets

Finally, also NIST 2002 and 2003 test sets were automatically translated through our best system. Table 7 collects measured scores, both case insensitive and true-case. Performance of the baseline system are reported for reference purposes.

test set	system	case-ins.	true-case
NIST04	baseline	25.48	24.19
	final	29.40	28.13
NIST03	baseline	24.47	22.98
	final	28.07	26.92
NIST02	baseline	24.36	22.58
	final	27.86	26.35

Table 7: Bleu scores on the official test sets of the last three NIST evaluations.

It has to be taken into account that the 2004 test set, unlike those of 2002 and 2003, does not only includes newswire texts, but also other kind of sources, i.e. speeches and editorials, for which MT seems to perform better. In fact, the 29.40 BLEU score (no true case) decreases to 28.49 if only the portion with newswire data is considered.

Anyway, improvements over the baseline on the three test sets are indeed comparable, ranging, in the true-case evaluation, from 16.2 to 17.1%.

9 Acknowledgements

This work has been funded by the European Union under the integrated project TC-STAR - Technology and Corpora for Speech to Speech Translation - (IST-2002-FP6-506738, http://www.tc-star.org).

References

A.L. Berger, S.A. Della Pietra, and V.J. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71.

N. Bertoldi, R. Cattoni, M. Cettolo, and M. Federico. 2004. The ITC-irst Statistical Machine Translation System for IWSLT-2004. In *Proceedings of IWSLT*, Kyoto, Japan.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–313.

M. Cettolo and M. Federico. 2004. Minimum Error Training of Log-Linear Translation Models. In *Proceedings of IWSLT*, Kyoto, Japan.

J. Goodman and S. Chen. 1998. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, August.

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

F.J. Och. 2004. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL*, Sapporo, Japan.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Research Report RC22176, IBM Research Division, Thomas J. Watson Research Center.

B. H. Tran, F. Seide, and V. Steinbiss. 1996. A Word Graph based N-Best Search in Continuous Speech Recognition. In *Proceedings of ICLSP*, Philadelphia, PA, USA.

H.-P. Zhang, Q. Liu, X. Q. Cheng, H. Zhang, and Hong-Kui Yu. 2003. Chinese Lexical Analysis Using Hierarchical Hidden Markov Model. In *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*. Sapporo, Japan, July.