Minimum Error Rate Training in Statistical Machine Translation

Franz Josef Och
Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, CA 90292
och@isi.edu

Abstract

Often, the training procedure for statistical machine translation models is based on maximum likelihood or related criteria. A general problem of this approach is that there is only a loose relation to the final translation quality on unseen text. In this paper, we analyze various training criteria which directly optimize translation quality. These training criteria make use of recently proposed automatic evaluation metrics. We describe a new algorithm for efficient training an unsmoothed error count. We show that significantly better results can often be obtained if the final evaluation criterion is taken directly into account as part of the training procedure.

1 Introduction

Many tasks in natural language processing have evaluation criteria that go beyond simply counting the number of wrong decisions the system makes. Some often used criteria are, for example, F-Measure for parsing, mean average precision for ranked retrieval, and BLEU or multi-reference word error rate for statistical machine translation. The use of statistical techniques in natural language processing often starts out with the simplifying (often implicit) assumption that the final scoring is based on simply counting the number of wrong decisions, for instance, the number of sentences incorrectly translated in machine translation. Hence, there is a mismatch between the basic assumptions of the used statistical approach and the final evaluation criterion used to measure success in a task.

Ideally, we would like to train our model parameters such that the end-to-end performance in some application is optimal. In this paper, we investigate methods to efficiently optimize model parameters with respect to machine translation quality as measured by automatic evaluation criteria such as word error rate and BLEU.

2 Statistical Machine Translation with Log-linear Models

Let us assume that we are given a source ('French') sentence $\mathbf{f} = f_1^J = f_1, \ldots, f_j, \ldots, f_J$, which is to be translated into a target ('English') sentence $\mathbf{e} = e_1^I = e_1, \ldots, e_i, \ldots, e_I$. Among all possible target sentences, we will choose the sentence with the highest probability:[1]

$$\hat{\mathbf{e}}(\mathbf{f}) = \operatorname*{argmax}_{\mathbf{e}} \{\Pr(\mathbf{e}|\mathbf{f})\} \tag{1}$$

The argmax operation denotes the *search problem*, i.e. the generation of the output sentence in the target language. The decision in Eq. 1 minimizes the number of decision errors. Hence, under a so-called zero-one loss function this decision rule is optimal (Duda and Hart, 1973). Note that using a different loss function—for example, one induced by the BLEU metric—a different decision rule would be optimal.

[1] The notational convention will be as follows. We use the symbol $\Pr(\cdot)$ to denote general probability distributions with (nearly) no specific assumptions. In contrast, for model-based probability distributions, we use the generic symbol $p(\cdot)$.

As the true probability distribution $\Pr(\mathbf{e}|\mathbf{f})$ is unknown, we have to develop a model $p(\mathbf{e}|\mathbf{f})$ that approximates $\Pr(\mathbf{e}|\mathbf{f})$. We directly model the posterior probability $\Pr(\mathbf{e}|\mathbf{f})$ by using a log-linear model. In this framework, we have a set of M feature functions $h_m(\mathbf{e}, \mathbf{f}), m = 1, \ldots, M$. For each feature function, there exists a model parameter $\lambda_m, m = 1, \ldots, M$. The direct translation probability is given by:

$$\Pr(\mathbf{e}|\mathbf{f}) = p_{\lambda_1^M}(\mathbf{e}|\mathbf{f}) \quad (2)$$

$$= \frac{\exp[\sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}, \mathbf{f})]}{\sum_{e'^I_1} \exp[\sum_{m=1}^{M} \lambda_m h_m(e'^I_1, \mathbf{f})]} \quad (3)$$

In this framework, the *modeling problem* amounts to developing suitable feature functions that capture the relevant properties of the translation task. The *training problem* amounts to obtaining suitable parameter values λ_1^M. A standard criterion for log-linear models is the MMI (maximum mutual information) criterion, which can be derived from the maximum entropy principle:

$$\hat{\lambda}_1^M = \underset{\lambda_1^M}{\operatorname{argmax}} \left\{ \sum_{s=1}^{S} \log p_{\lambda_1^M}(\mathbf{e}_s|\mathbf{f}_s) \right\} \quad (4)$$

The optimization problem under this criterion has very nice properties: there is one unique global optimum, and there are algorithms (e.g. gradient descent) that are guaranteed to converge to the global optimum. Yet, the ultimate goal is to obtain good translation quality on unseen test data. Experience shows that good results can be obtained using this approach, yet there is no reason to assume that an optimization of the model parameters using Eq. 4 yields parameters that *are optimal* with respect to translation quality.

The goal of this paper is to investigate alternative training criteria and corresponding training algorithms, which are directly related to translation quality measured with automatic evaluation criteria. In Section 3, we review various automatic evaluation criteria used in statistical machine translation. In Section 4, we present two different training criteria which try to directly optimize an error count. In Section 5, we sketch a new training algorithm which efficiently optimizes an unsmoothed error count. In Section 6, we describe the used feature functions and our approach to compute the candidate translations that are the basis for our training procedure. In Section 7, we evaluate the different training criteria in the context of several MT experiments.

3 Automatic Assessment of Translation Quality

In recent years, various methods have been proposed to automatically evaluate machine translation quality by comparing hypothesis translations with reference translations. Examples of such methods are word error rate, position-independent word error rate (Tillmann et al., 1997), generation string accuracy (Bangalore et al., 2000), multi-reference word error rate (Nießen et al., 2000), BLEU score (Papineni et al., 2001), NIST score (Doddington, 2002). All these criteria try to approximate human assessment and often achieve an astonishing degree of correlation to human subjective evaluation of fluency and adequacy (Papineni et al., 2001; Doddington, 2002).

In this paper, we use the following methods:

- multi-reference word error rate (mWER): When this method is used, the hypothesis translation is compared to various reference translations by computing the edit distance (minimum number of substitutions, insertions, deletions) between the hypothesis and the closest of the given reference translations.

- multi-reference position independent error rate (mPER): This criterion ignores the word order by treating a sentence as a bag-of-words and computing the minimum number of substitutions, insertions, deletions needed to transform the hypothesis into the closest of the given reference translations.

- BLEU score: This criterion computes the geometric mean of the precision of n-grams of various lengths between a hypothesis and a set of reference translations multiplied by a factor BP($\cdot$) that penalizes short sentences:

$$\text{BLEU} = \text{BP}(\cdot) \cdot \exp\left(\sum_{n=1}^{N} \frac{\log p_n}{N}\right)$$

  Here p_n denotes the precision of n-grams in the hypothesis translation. We use $N = 4$.

- NIST score: This criterion computes a weighted precision of n-grams between a hypothesis and a set of reference translations multiplied by a factor BP'$(\cdot)$ that penalizes short sentences:

$$\text{NIST} = \text{BP'}(\cdot) \cdot \sum_{n=1}^{N} w_n$$

  Here w_n denotes the weighted precision of n-grams in the translation. We use $N = 4$.

Both, NIST and BLEU are accuracy measures, and thus larger values reflect better translation quality. Note that NIST and BLEU scores are not additive for different sentences, i.e. the score for a document cannot be obtained by simply summing over scores for individual sentences.

4 Training Criteria for Minimum Error Rate Training

In the following, we assume that we can measure the number of errors in sentence $\mathbf{e}$ by comparing it with a reference sentence $\mathbf{r}$ using a function $\text{E}(\mathbf{r}, \mathbf{e})$. However, the following exposition can be easily adapted to accuracy metrics and to metrics that make use of multiple references.

We assume that the number of errors for a set of sentences $\mathbf{e}_1^S$ is obtained by summing the errors for the individual sentences: $E(\mathbf{r}_1^S, \mathbf{e}_1^S) = \sum_{s=1}^{S} E(\mathbf{r}_s, \mathbf{e}_s)$.

Our goal is to obtain a minimal error count on a representative corpus $\mathbf{f}_1^S$ with given reference translations $\hat{\mathbf{e}}_1^S$ and a set of K different candidate translations $\mathbf{C}_s = \{\mathbf{e}_{s,1}, \ldots, \mathbf{e}_{s,K}\}$ for each input sentence $\mathbf{f}_s$.

$$\begin{aligned} \hat{\lambda}_1^M &= \operatorname*{argmin}_{\lambda_1^M} \left\{ \sum_{s=1}^{S} E(\mathbf{r}_s, \hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M)) \right\} \quad (5) \\ &= \operatorname*{argmin}_{\lambda_1^M} \left\{ \sum_{s=1}^{S} \sum_{k=1}^{K} E(\mathbf{r}_s, \mathbf{e}_{s,k}) \delta(\hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M), \mathbf{e}_{s,k}) \right\} \end{aligned}$$

with

$$\hat{\mathbf{e}}(\mathbf{f}_s; \lambda_1^M) = \operatorname*{argmax}_{\mathbf{e} \in \mathbf{C}_s} \left\{ \sum_{m=1}^{M} \lambda_m h_m(\mathbf{e}|\mathbf{f}_s) \right\} \quad (6)$$

The above stated optimization criterion is not easy to handle:

- It includes an argmax operation (Eq. 6). Therefore, it is not possible to compute a gradient and we cannot use gradient descent methods to perform optimization.
- The objective function has many different local optima. The optimization algorithm must handle this.

In addition, even if we manage to solve the optimization problem, we might face the problem of overfitting the training data. In Section 5, we describe an efficient optimization algorithm.

To be able to compute a gradient and to make the objective function smoother, we can use the following error criterion which is essentially a smoothed error count, with a parameter α to adjust the smoothness:

$$\hat{\lambda}_1^M = \operatorname*{argmin}_{\lambda_1^M} \left\{ \sum_{s,k} E(\mathbf{e}_{s,k}) \frac{p(\mathbf{e}_{s,k}|\mathbf{f})^\alpha}{\sum_k p(\mathbf{e}_{s,k}|\mathbf{f})^\alpha} \right\} \quad (7)$$

In the extreme case, for $\alpha \to \infty$, Eq. 7 converges to the unsmoothed criterion of Eq. 5 (except in the case of ties). Note, that the resulting objective function might still have local optima, which makes the optimization hard compared to using the objective function of Eq. 4 which does not have different local optima. The use of this type of smoothed error count is a common approach in the speech community (Juang et al., 1995; Schlüter and Ney, 2001).

Figure 1 shows the actual shape of the smoothed and the unsmoothed error count for two parameters in our translation system. We see that the unsmoothed error count has many different local optima and is very unstable. The smoothed error count is much more stable and has fewer local optima. But as we show in Section 7, the performance on our task obtained with the smoothed error count does not differ significantly from that obtained with the unsmoothed error count.

5 Optimization Algorithm for Unsmoothed Error Count

A standard algorithm for the optimization of the unsmoothed error count (Eq. 5) is Powells algorithm combined with a grid-based line optimization method (Press et al., 2002). We start at a random point in the K-dimensional parameter space

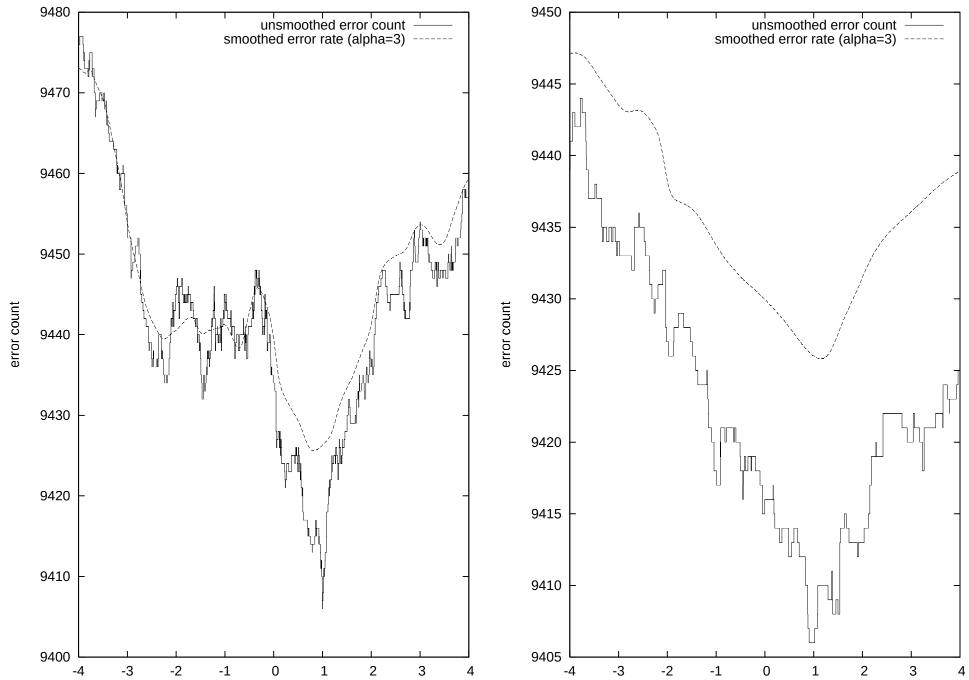

Figure 1: Shape of error count and smoothed error count for two different model parameters. These curves have been computed on the development corpus (see Section 7, Table 1) using $1,600$ alternatives per source sentence. The smoothed error count has been computed with a smoothing parameter $\alpha = 3$.

and try to find a better scoring point in the parameter space by making a one-dimensional line minimization along the directions given by optimizing one parameter while keeping all other parameters fixed. To avoid finding a poor local optimum, we start from different initial parameter values. A major problem with the standard approach is the fact that grid-based line optimization is hard to adjust such that both good performance and efficient search are guaranteed. If a fine-grained grid is used then the algorithm is slow. If a large grid is used then the optimal solution might be missed.

In the following, we describe a new algorithm for efficient line optimization of the unsmoothed error count (Eq. 5) using a log-linear model (Eq. 3) which is guaranteed to find the optimal solution. The new algorithm is much faster and more stable than the grid-based line optimization method.

Computing the most probable sentence out of a set of candidate translation $\mathbf{C} = \{\mathbf{e}_1, \ldots, \mathbf{e}_k\}$ (see Eq. 6) along a line $\lambda_1^M + \gamma \cdot d_1^M$ with parameter γ results in an optimization problem of the following functional form:

$$\hat{\mathbf{e}}(\hat{\mathbf{f}}; \gamma) = \underset{\mathbf{e} \in \mathbf{C}}{\operatorname{argmin}} \; \{t(\mathbf{e}, \mathbf{f}) + \gamma \cdot m(\mathbf{e}, \mathbf{f})\} \qquad (8)$$

Here, $t(\cdot)$ and $m(\cdot)$ are constants with respect to γ. Hence, every candidate translation in $\mathbf{C}$ corresponds to a line. The function

$$f(\gamma; \mathbf{f}) = \min_{\mathbf{e} \in \mathbf{C}} \; \{t(\mathbf{e}, \mathbf{f}) + \gamma \cdot m(\mathbf{e}, \mathbf{f})\} \qquad (9)$$

is piecewise linear (Papineni, 1999). This allows us to compute an efficient exhaustive representation of that function.

In the following, we sketch the new algorithm to optimize Eq. 5: We compute the ordered sequence of linear intervals constituting $f(\gamma; \mathbf{f})$ for every sentence $\mathbf{f}$ together with the incremental change in error count from the previous to the next interval. Hence, we obtain for every sentence $\mathbf{f}$ a sequence $\gamma_1^{\mathbf{f}} < \gamma_2^{\mathbf{f}} < \ldots < \gamma_{N_{\mathbf{f}}}^{\mathbf{f}}$ which denote the interval boundaries and a corresponding sequence for the change in error count involved at the corresponding interval boundary $\Delta E_1^{\mathbf{f}}, \Delta E_2^{\mathbf{f}}, \ldots, \Delta E_{N_{\mathbf{f}}}^{\mathbf{f}}$. Here, $\Delta E_n^{\mathbf{f}}$ denotes the change in the error count at position $(\gamma^{\mathbf{f}}_{n-1} + \gamma^{\mathbf{f}}_{n})/2$ to the error count at position $(\gamma^{\mathbf{f}}_{n} + \gamma^{\mathbf{f}}_{n+1})/2$. By merging all sequences $\gamma^{\mathbf{f}}$ and $\Delta E^{\mathbf{f}}$ for all different sentences of our corpus, the complete set of interval boundaries and error count changes on the whole corpus are obtained. The optimal γ can now be computed easily by traversing the sequence of interval boundaries while updating an error count.

It is straightforward to refine this algorithm to also handle the BLEU and NIST scores instead of sentence-level error counts by accumulating the relevant statistics for computing these scores (n-gram precision, translation length and reference length) .

6 Baseline Translation Approach

The basic feature functions of our model are identical to the alignment template approach (Och and Ney, 2002). In this translation model, a sentence is translated by segmenting the input sentence into phrases, translating these phrases and reordering the translations in the target language. In addition to the feature functions described in (Och and Ney, 2002), our system includes a phrase penalty (the number of alignment templates used) and special alignment features. Altogether, the log-linear model includes $M = 8$ different features.

Note that many of the used feature functions are derived from probabilistic models: the feature function is defined as the negative logarithm of the corresponding probabilistic model. Therefore, the feature functions are much more 'informative' than for instance the binary feature functions used in standard maximum entropy models in natural language processing.

For search, we use a dynamic programming beam-search algorithm to explore a subset of all possible translations (Och et al., 1999) and extract n-best candidate translations using A* search (Ueffing et al., 2002).

Using an n-best approximation, we might face the problem that the parameters trained are good for the list of n translations used, but yield worse translation results if these parameters are used in the dynamic programming search. Hence, it is possible that our new search produces translations with more errors on the training corpus. This can happen because with the modified model scaling factors the n-best list can change significantly and can include sentences not in the existing n-best list. To avoid this problem, we adopt the following solution: First, we perform search (using a manually defined set of parameter values) and compute an n-best list, and use this n-best list to train the model parameters. Second, we use the new model parameters in a new search and compute a new n-best list, which is combined with the existing n-best list. Third, using this extended n-best list new model parameters are computed. This is iterated until the resulting n-best list does not change. In this algorithm convergence is guaranteed as, in the limit, the n-best list will contain all possible translations. In our experiments, we compute in every iteration about 200 alternative translations. In practice, the algorithm converges after about five to seven iterations. As a result, error rate cannot increase on the training corpus.

A major problem in applying the MMI criterion is the fact that the reference translations need to be part of the provided n-best list. Quite often, none of the given reference translations is part of the n-best list because the search algorithm performs pruning, which in principle limits the possible translations that can be produced given a certain input sentence. To solve this problem, we define for the MMI training new pseudo-references by selecting from the n-best list all the sentences which have a minimal number of word errors with respect to any of the true references. Note that due to this selection approach, the results of the MMI criterion might be biased toward the mWER criterion. It is a major advantage of the minimum error rate training that it is not necessary to choose pseudo-references.

7 Results

We present results on the 2002 TIDES Chinese–English small data track task. The goal is the translation of news text from Chinese to English. Table 1 provides some statistics on the training, development and test corpus used. The system we use does not include rule-based components to translate numbers, dates or names. The basic feature functions were trained using the training corpus. The development corpus was used to optimize the parameters of the log-linear model. Translation results are reported on the test corpus.

Table 2 shows the results obtained on the development corpus and Table 3 shows the results obtained

Table 2: Effect of different error criteria in training *on the development corpus*. Note that better results correspond to larger BLEU and NIST scores and to smaller error rates. Italic numbers refer to results for which the difference to the best result (indicated in bold) is not statistically significant.

error criterion used in training	mWER [%]	mPER [%]	BLEU [%]	NIST	# words
confidence intervals	+/- 2.4	+/- 1.8	+/- 1.2	+/- 0.2	-
MMI	70.7	55.3	12.2	5.12	10382
mWER	**69.7**	*52.9*	15.4	5.93	10914
smoothed-mWER	*69.8*	*53.0*	15.2	5.93	10925
mPER	*71.9*	**51.6**	17.2	6.61	11671
smoothed-mPER	*71.8*	*51.8*	17.0	6.56	11625
BLEU	76.8	54.6	**19.6**	*6.93*	13325
NIST	73.8	*52.8*	18.9	**7.08**	12722

Table 1: Characteristics of training corpus (Train), manual lexicon (Lex), development corpus (Dev), test corpus (Test).

		Chinese	English
Train	Sentences	5 109	
	Words	89 121	111 251
	Singletons	3 419	4 130
	Vocabulary	8 088	8 807
Lex	Entries	82 103	
Dev	Sentences	640	
	Words	11 746	13 573
Test	Sentences	878	
	Words	24 323	26 489

on the test corpus. Italic numbers refer to results for which the difference to the best result (indicated in bold) is not statistically significant. For all error rates, we show the maximal occurring 95% confidence interval in any of the experiments for that column. The confidence intervals are computed using bootstrap resampling (Press et al., 2002). The last column provides the number of words in the produced translations which can be compared with the average number of reference words occurring in the development and test corpora given in Table 1.

We observe that if we choose a certain error criterion in training, we obtain in most cases the best results using the same criterion as the evaluation metric on the test data. The differences can be quite large: If we optimize with respect to word error rate, the results are mWER=68.3%, which is better than if we optimize with respect to BLEU or NIST and the difference is statistically significant. Between BLEU and NIST, the differences are more moderate, but by optimizing on NIST, we still obtain a large improvement when measured with NIST compared to optimizing on BLEU.

The MMI criterion produces significantly worse results on all error rates besides mWER. Note that, due to the re-definition of the notion of reference translation by using minimum edit distance, the results of the MMI criterion are biased toward mWER. It can be expected that by using a suitably defined n-gram precision to define the pseudo-references for MMI instead of using edit distance, it is possible to obtain better BLEU or NIST scores.

An important part of the differences in the translation scores is due to the different translation length (last column in Table 3). The mWER and MMI criteria prefer shorter translations which are heavily penalized by the BLEU and NIST brevity penalty.

We observe that the smoothed error count gives almost identical results to the unsmoothed error count. This might be due to the fact that the number of parameters trained is small and no serious overfitting occurs using the unsmoothed error count.

8 Related Work

The use of log-linear models for statistical machine translation was suggested by Papineni et al. (1997) and Och and Ney (2002).

The use of minimum classification error training and using a smoothed error count is common in the pattern recognition and speech

Table 3: Effect of different error criteria used in training *on the test corpus*. Note that better results correspond to larger BLEU and NIST scores and to smaller error rates. Italic numbers refer to results for which the difference to the best result (indicated in bold) is not statistically significant.

error criterion used in training	mWER [%]	mPER [%]	BLEU [%]	NIST	# words
confidence intervals	+/- 2.7	+/- 1.9	+/- 0.8	+/- 0.12	-
MMI	**68.0**	*51.0*	11.3	5.76	21933
mWER	*68.3*	*50.2*	13.5	6.28	22914
smoothed-mWER	*68.2*	*50.2*	13.2	6.27	22902
mPER	*70.2*	*49.8*	15.2	*6.71*	24399
smoothed-mPER	*70.0*	**49.7**	15.2	*6.69*	24198
BLEU	76.1	53.2	**17.2**	6.66	28002
NIST	73.3	*51.5*	*16.4*	**6.80**	26602

recognition community (Duda and Hart, 1973; Juang et al., 1995; Schlüter and Ney, 2001). Paciorek and Rosenfeld (2000) use minimum classification error training for optimizing parameters of a whole-sentence maximum entropy language model.

A technically very different approach that has a similar goal is the *minimum Bayes risk approach*, in which an optimal decision rule with respect to an application specific *risk/loss function* is used, which will normally differ from Eq. 3. The loss function is either identical or closely related to the final evaluation criterion. In contrast to the approach presented in this paper, the training criterion and the statistical models used remain unchanged in the minimum Bayes risk approach. In the field of natural language processing this approach has been applied for example in parsing (Goodman, 1996) and word alignment (Kumar and Byrne, 2002).

9 Conclusions

We presented alternative training criteria for log-linear statistical machine translation models which are directly related to translation quality: an unsmoothed error count and a smoothed error count on a development corpus. For the unsmoothed error count, we presented a new line optimization algorithm which can efficiently find the optimal solution along a line. We showed that this approach obtains significantly better results than using the MMI training criterion (with our method to define pseudo-references) and that optimizing error rate as part of the training criterion helps to obtain better error rate on unseen test data. As a result, we expect that actual 'true' translation quality is improved, as previous work has shown that for some evaluation criteria there is a correlation with human subjective evaluation of fluency and adequacy (Papineni et al., 2001; Doddington, 2002). However, the different evaluation criteria yield quite different results on our Chinese–English translation task and therefore we expect that not all of them correlate equally well to human translation quality.

The following important questions should be answered in the future:

- How many parameters can be reliably estimated using unsmoothed minimum error rate criteria using a given development corpus size? We expect that directly optimizing error rate for many more parameters would lead to serious overfitting problems. Is it possible to optimize more parameters using the smoothed error rate criterion?
- Which error rate should be optimized during training? This relates to the important question of which automatic evaluation measure is optimally correlated to human assessment of translation quality.

Note, that this approach can be applied to any evaluation criterion. Hence, if an improved automatic evaluation criterion is developed that has an even better correlation with human judgments than BLEU and NIST, we can plug this alternative criterion directly into the training procedure and optimize the model parameters for it. This means that improved translation evaluation measures lead directly to improved machine translation quality. Of course, the approach presented here places a high demand on the fidelity of the measure being optimized. It might happen that by directly optimizing an error measure in the way described above, weaknesses in the measure might be exploited that could yield better scores without improved translation quality. Hence, this approach poses new challenges for developers of automatic evaluation criteria.

Many tasks in natural language processing, for instance summarization, have evaluation criteria that go beyond simply counting the number of wrong system decisions and the framework presented here might yield improved systems for these tasks as well.

Acknowledgements

This work was supported by DARPA-ITO grant 66001-00-1-9814.

References

Srinivas Bangalore, O. Rambox, and S. Whittaker. 2000. Evaluation metrics for generation. In *Proceedings of the International Conference on Natural Language Generation*, Mitzpe Ramon, Israel.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. ARPA Workshop on Human Language Technology*.

Richhard O. Duda and Peter E. Hart. 1973. *Pattern Classification and Scene Analysis*. John Wiley, New York, NY.

Joshua Goodman. 1996. Parsing algorithms and metrics. In *Proceedings of the 34th Annual Meeting of the ACL*, pages 177–183, Santa Cruz, CA, June.

B. H. Juang, W. Chou, and C. H. Lee. 1995. Statistical and discriminative methods for speech recognition. In A. J. Rubio Ayuso and J. M. Lopez Soler, editors, *Speech Recognition and Coding - New Advances and Trends*. Springer Verlag, Berlin, Germany.

Shankar Kumar and William Byrne. 2002. Minimum bayes-risk alignment of bilingual texts. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA.

Sonja Nießen, Franz J. Och, G. Leusch, and Hermann Ney. 2000. An evaluation tool for machine translation: Fast evaluation for machine translation research. In *Proc. of the Second Int. Conf. on Language Resources and Evaluation (LREC)*, pages 39–45, Athens, Greece, May.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, PA, July.

Franz J. Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28, University of Maryland, College Park, MD, June.

Chris Paciorek and Roni Rosenfeld. 2000. Minimum classification error training in exponential language models. In *NIST/DARPA Speech Transcription Workshop*, May.

Kishore A. Papineni, Salim Roukos, and R. T. Ward. 1997. Feature-based language understanding. In *European Conf. on Speech Communication and Technology*, pages 1435–1438, Rhodes, Greece, September.

Kishore A. Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. Technical Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY, September.

Kishore A. Papineni. 1999. Discriminative training via linear programming. In *Proceedings of the 1999 IEEE International Conference on Acoustics, Speech & Signal Processing*, Atlanta, March.

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

Ralf Schlüter and Hermann Ney. 2001. Model-based MCE bound to the true Bayes' error. *IEEE Signal Processing Letters*, 8(5):131–133, May.

Christoph Tillmann, Stephan Vogel, Hermann Ney, Alex Zubiaga, and Hassan Sawaf. 1997. Accelerated DP based search for statistical translation. In *European Conf. on Speech Communication and Technology*, pages 2667–2670, Rhodes, Greece, September.

Nicola Ueffing, Franz Josef Och, and Hermann Ney. 2002. Generation of word graphs in statistical machine translation. In *Proc. Conference on Empirical Methods for Natural Language Processing*, pages 156–163, Philadelphia, PE, July.