Automatic Evaluation Measures for Statistical Machine Translation System Optimization

Arne Mauser, Saša Hasan and Hermann Ney

Human Language Technology and Pattern Recognition
Lehrstuhl für Informatik 6, Computer Science Department
RWTH Aachen University, D-52056 Aachen, Germany
{mauser,hasan,ney}@cs.rwth-aachen.de

Abstract

Evaluation of machine translation (MT) output is a challenging task. In most cases, there is no single correct translation. In the extreme case, two translations of the same input can have completely different words and sentence structure while still both being perfectly valid. Large projects and competitions for MT research raised the need for reliable and efficient evaluation of MT systems. For the funding side, the obvious motivation is to measure performance and progress of research. This often results in a specific measure or metric taken as primarily evaluation criterion. Do improvements in one measure really lead to improved MT performance? How does a gain in one evaluation metric affect other measures? This paper is going to answer these questions by a number of experiments.

1. Introduction

Evaluation of machine translation (MT) output is a challenging task. In most cases, there is no single correct translation. In the extreme case, two translations of the same input can have completely different words and sentence structure while still both being perfectly valid.

Large projects and competitions for MT research raised the need for reliable and efficient evaluation of MT systems. For the funding side, the obvious motivation is to measure performance and progress of research. This often results in a specific measure or metric taken as primarily evaluation criterion.

MT research is therefor forced to improve in these specific measures. For statistical MT (SMT), translation systems are usually optimized for an automatic evaluation measure. A number of these methods already exist and new evaluation measures are frequently proposed to the MT community.

Do improvements in one measure really lead to improved MT performance? How does a gain in one evaluation metric affect other measures? This paper is going to answer these questions by a number of experiments.

The work is organized as follows: in the next section, we describe the statistical approach to machine translation. This is followed by the description of the parameter tuning in Section 2.. We then look at the evaluation measures in Section 3.. The task, system setup and results are discussed in Section 4..

1.1. Log-linear model

The problem of statistical machine translation is to find the translation $e_1^I = e_1 \dots e_i \dots e_I$ of a given source language sentence $f_1^J = f_1 \dots f_j \dots f_J$. Among all possible target language sentences, we will choose the sentence with the highest probability:

$$\hat{e}_1^{\hat{I}} = \operatorname*{argmax}_{I,e_1^I} \left\{ Pr(e_1^I|f_1^J) \right\} \quad (1)$$

$$(2)$$

Using a log-linear model (Och and Ney, 2002), we obtain:

$$Pr(e_1^I|f_1^J) = \frac{\exp\left(\sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J)\right)}{\sum\limits_{e'^{I'}_1} \exp\left(\sum_{m=1}^M \lambda_m h_m(e'^{I'}_1, f_1^J)\right)} \quad (3)$$

The denominator represents a normalization factor that depends only on the source sentence f_1^J. Therefore, we can omit it during the search process. As a decision rule, we obtain:

$$\hat{e}_1^{\hat{I}} = \operatorname*{argmax}_{I,e_1^I} \left\{ \sum_{m=1}^M \lambda_m h_m(e_1^I, f_1^J) \right\} \quad (4)$$

This is a generalization of the source-channel approach. It has the advantage that additional models $h(\cdot)$ can be easily integrated into the overall system. The process of obtaining the λ values will be described in Section 2..

1.2. Phrase-based approach

The basic idea of phrase-based translation is to segment the given source sentence into phrases, then translate each phrase and finally compose the target sentence from these phrase translations. Formally, we define a segmentation of a given sentence pair (f_1^J, e_1^I) into K blocks:

$$k \rightarrow s_k := (i_k; b_k, j_k), \text{ for } k = 1 \dots K. \quad (5)$$

Here, i_k denotes the last position of the k^{th} target phrase; we set $i_0 := 0$. The pair (b_k, j_k) denotes the start and end positions of the source phrase that is aligned to the k^{th} target phrase; we set $j_0 := 0$. Phrases are defined as nonempty contiguous sequences of words. We constrain the segmentations so that all words in the source and the target sentence are covered by exactly one phrase. Thus, there are no gaps and there is no overlap.

For a given sentence pair (f_1^J, e_1^I) and a given segmentation s_1^K, we define the bilingual phrases as:

$$\tilde{e}_k := e_{i_{k-1}+1} \dots e_{i_k} \quad (6)$$

$$\tilde{f}_k := f_{b_k} \dots f_{j_k} \quad (7)$$

The segmentation s_1^K is introduced as a hidden variable in the translation model. Therefore, it would be theoretically correct to sum over all possible segmentations. In practice, we use the maximum approximation for this sum. As a result, the models $h(\cdot)$ depend not only on the sentence pair (f_1^J, e_1^I), but also on the segmentation s_1^K, i.e., we have models $h(f_1^J, e_1^I, s_1^K)$. Note that the segmentation also contains the information on the phrase-level reordering.

1.3. Models used during search

When searching for the best translation for a given input sentence, we use a log-linear combination of several models (also called feature functions) as decision function.

More specifically the models are: a phrase translation model, a word-based translation model, word and phrase penalty, a target language model and a reordering model. A detailed description of the models used can be found in (Mauser et al., 2006).

2. Minimum Error Rate Training

Phrase table probabilies themselves already give a fairly good represetation of our training data. Translating unseen data however, requires the system to be more flexible. Depending on the task, for example longer or shorter translation might be preferable. For lower quality bilingual data, we would want the system to rely a little more on the monolingual target language model than on the actual translation probabilities.

For this purpose, the scaling factors λ in equation are usually set for a specific translation task. This is done by optimizing the factors with respect to a loss function. If the loss function in the subjectively perceived translation quality, we have to adjust the weights manually. If this loss function is an automatic evaluation measure, we can use an automated procedure that will find a good solution four our parameters. This procedure is referred to as Minimum Error Rate Training (MERT) (Och, 2003).

In the experiments, we used the downhill simplex algorithm (Press et al., 2002) to optimize the system weights for a specific evaluation measure.

3. Evaluation Measures

This sections briefly describes the Evaluation measures considered in this work. We selected the most commonly used metrics which were suitable to perform the experiments.

Most evaluation measures show a reasonable correlation with human judgement but so far, it remains an open question, if an improvement in one of these measures will also lead to improvements in the translation quality.

3.1. Word Error Rate (WER)

The Edit distance or Levenshtein distance on word level is the minimum number of word insertions, substitutions and deletions necessary to transform the candidate translation into the reference translation. All three operations are assumed to have identical costs. Reordering is not permitted. Swapping parts of a sentence, even within grammatical rules, results in a series of insertions, deletions and/or substitutions.

The number of edit operations is divided by the number of words in the reference. If the hypothesis is longer than the reference, this can result in an error rate larger than 1. Therefor, WER has a bias towards shorter hypotheses.

When multiple reference translations are given, the reported error for a translation hypothesis is the minimum error over all references.

3.2. Position-independent Word Error Rate (PER)

Where WER is extreme to the fact, that it requires exactly the same order of the words in hypothesis and reference the position-independent word error rate (PER) (Tillmann et al., 1997) neglects word order completely. It measures the difference in the count of the words occurring in hypothesis and reference. The resulting number is divided by the number of words in the reference.

3.3. Translation Edit Rate (TER)

The TER (Snover et al., 2006) is an error measure counts the number of edits required to change a system output into one of the given translation references. The background is to measure the amount of human work that would be required to post-edit the translations proposed by the system into the reference. In contrast to WER, movements of blocks are allowed and counted as one edit with equal costs to insertions, deletions and substitutions of single words. The number of edit operations is divided by the average number of reference words.

3.4. BLEU

Proposed by (Papineni et al., 2002), the BLEU criterion measures the similarity of n-grams count vectors of the reference translations in the candidate translation. If multiple references are present the counts are collected of all translations. The typical length of the n-gram is 4, with shorter n-grams being also counted and then interpolated. BLEU is a precision measure, higher values indicate better results. If the no n-gram of maximum length matches between translation hypothesis and reference, the BLEU score will be zero.

In addition, there is a brevity penalty to attenuate the strong bias towards short sentences. Variants of the brevity penalty exist with respect to the reference length used. The original IBM-BLEU used the length of the reference which was closest in length to the translation hypothesis. This is the variant that we use here.

3.5. NIST

The NIST precision measure (Doddington, 2002) was intended as an improved version of BLEU. Unwanted effects of the brevity penalty of BLEU should be reduced and n-gram occurrences are weighted by their importance. The importance is computed by the frequency of the n-gram in the references. As for BLEU, multiple reference translations are pooled, but NIST considers frequently occurring n-grams to be less important than rare ones. The brevity penalty is designed to avoid BLEU's slight bias towards short candidates.

	BLEU	TER	WER	PER	NIST
NIST 2002	38.9	55.2	61.5	39.7	10.09
NIST 2003	37.1	56.9	63.1	40.5	9.78
NIST 2004	37.6	56.3	62.8	40.4	9.73
NIST 2005	35.9	56.7	63.1	39.7	9.63

Table 1: Overview of the translation results on all test sets.

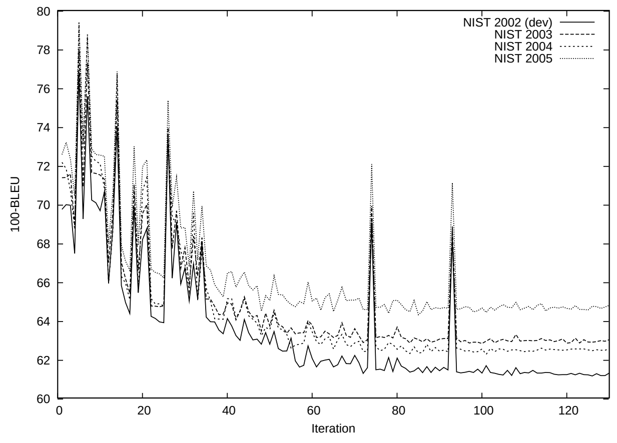

Figure 1: Optimization for BLEU and the effect on the test sets NIST 2003–2005. BLEU is displayed as error-rate to be consistent with other graphs.

4. Experiments

We conducted our experiments on the NIST MT Chinese-to-English task. The international evaluation held by the US National Institute of Standards and Technology (NIST) is focused on news translation of Arabic and Chinese to English. The bilingual training is provided by the Linguistic Data Consortion LDC and consists of newswire and news magazine translations, UN documents and parliamentary proceedings. In total, we have about 8 million sentence pairs or 250 million running words.

The evaluation corpus from 2002 is used as main development set. In most experiments we optimize the system weights on this corpus. The years 2003 to 2005 serve as test sets unless stated otherwise. Each corpus consists of about 1000 sentences and has 4 reference translations. All measures are computed neglecting case information. Table 1 gives an overview of the results on all corpora when optimizing for BLEU on the NIST 2002 set.

In order to determine, how well he system tuned for a specific measure generalizes to other datasets, we optimized a system for TER and BLEU and computed the other error measures after each iteration of the optimization procedure. The corresponding graphs are shown in Figure 1 for BLEU and Figure 2 for TER. The graphs clearly show, that improvements on the development set generalize well to the test sets.

The optimization procedure was run until the change in the objective function was below a threshold. Peaks visible in the graphs are artefacts of the optimization algorithm.

The second questions was, how well the improvement on one measure shows in other measures. To answer this questions, we examined the results on the NIST 2005 corpus. To reduce the number of graphs, we look again at BLEU and TER in Figures 4. and 4. respectively. While other measures largely improve in the process, some degradation can be seen in later iterations.

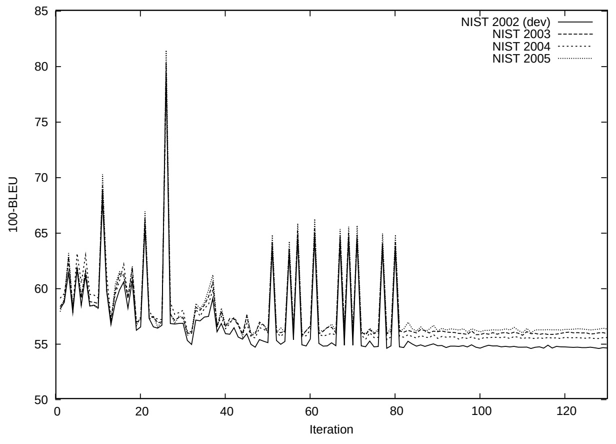

Figure 2: Optimization for TER and the effect on the test sets NIST 2003–2005.

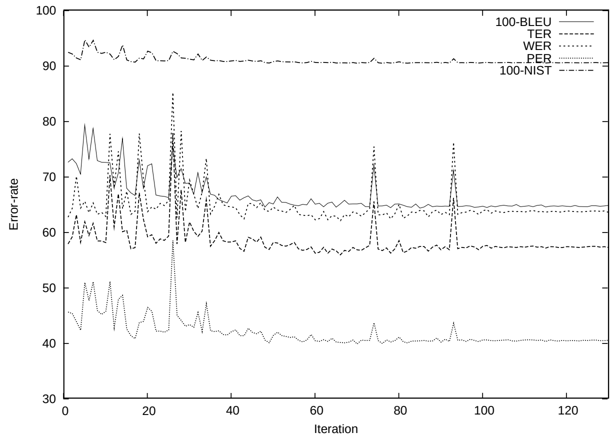

Figure 3: Optimization on BLEU and the effect on other error measures shown on the NIST 2005 set. NIST and BLEU are displayed as error-rates for better comparison.

In order to find a good overall criterion for system tuning, we examined the effect of optimizing on one measure and evaluating on all. The results for the NIST 2005 set are shown in Table 2. As expected from the initial generalization experiments, optimizing for one measure also leads to the best or near-best results on the test set.

While all offering comparable translation quality, the criteria differ largely in the sentence length of the resulting hypotheses. The count-based BLEU, NIST and also PER result in longer hypotheses. TER and especially WER lead to rather short hypotheses.

With the preference for sentence length being rather different, we also tried to optimize on the sum of a error-rate version of BLEU (100-BLEU) and TER. The result shows a good performance on all error measures, indicating that it could serve as a reasonable all-round criterion.

	Evaluation					
Optimize on ↓	BLEU	TER	WER	PER	NIST	Avg. length
BLEU	35.9	56.7	63.1	39.7	9.63	31.8
TER	34.6	55.7	62.0	40.4	9.41	29.2
WER	33.2	55.5	61.0	41.8	8.93	27.4
PER	35.1	57.3	64.5	39.8	9.59	31.9
NIST	35.8	56.2	63.1	39.5	9.66	31.3
BLEU+TER	35.4	55.8	62.2	39.8	9.56	30.2

Table 2: Error Rates on the NIST 2005 corpus when optimizing an evaluating with different measures. TER, WER, PER: lower values are better, BLEU, NIST: higher values are better.

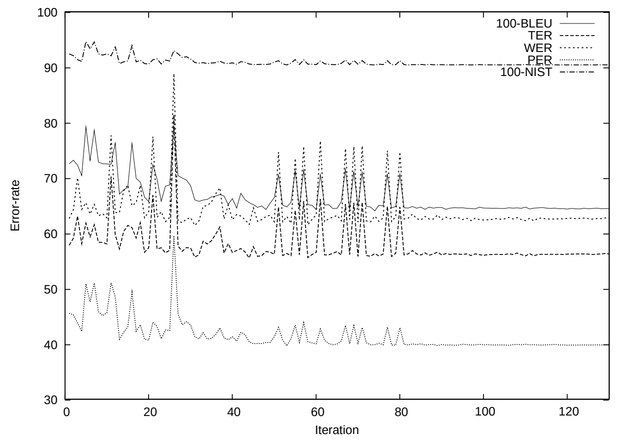

Figure 4: Optimization on TER and the effect on other error measures shown on the NIST 2005 set. NIST and BLEU are displayed as error-rates for better comparison.

5. Conclusions

We investigated the effects of optimizing a statistical MT system for different error measures. The results show, that modern evaluation metrics like BLEU or TER are robust in two aspects. First, improvements on a development set also lead to improvements an similar test sets. Second, improvements in one measure also improve other measures. This is was found not true for simpler measures like WER or PER. As a result, using an interpolation of BLEU and TER appeared to be a good overall choice for system tuning.

6. Acknowledgements

This paper is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023.

7. References

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

Arne Mauser, Richard Zens, Evgeny Matusov, Sasa Hasan, and Hermann Ney. 2006. The RWTH Statistical Machine Translation System for the IWSLT 2006 Evaluation. In *Proc. of the International Workshop on Spoken Language Translation*, pages 103–110, Kyoto, Japan.

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)*, pages 295–302, Philadelphia, PA, July.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wie-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 311–318, Philadelphia, PA, July.

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

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, August.

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