Training and Evaluating Error Minimization Rules for Statistical Machine Translation

Ashish Venugopal
School of Computer Science
Carnegie Mellon University
arv@andrew.cmu.edu

Andreas Zollmann
School of Computer Science
Carnegie Mellon University
zollmann@cs.cmu.edu

Alex Waibel
School of Computer Science
Carnegie Mellon University
waibel@cs.cmu.edu

Abstract

Decision rules that explicitly account for non-probabilistic evaluation metrics in machine translation typically require special training, often to estimate parameters in exponential models that govern the search space and the selection of candidate translations. While the traditional Maximum A Posteriori (MAP) decision rule can be optimized as a piecewise linear function in a greedy search of the parameter space, the Minimum Bayes Risk (MBR) decision rule is not well suited to this technique, a condition that makes past results difficult to compare. We present a novel training approach for non-tractable decision rules, allowing us to compare and evaluate these and other decision rules on a large scale translation task, taking advantage of the high dimensional parameter space available to the phrase based Pharaoh decoder. This comparison is timely, and important, as decoders evolve to represent more complex search space decisions and are evaluated against innovative evaluation metrics of translation quality.

1 Introduction

State of the art statistical machine translation takes advantage of exponential models to incorporate a large set of potentially overlapping features to select translations from a set of potential candidates. As discussed in (Och, 2003), the direct translation model represents the probability of target sentence 'English' $\mathbf{e} = \mathbf{e_1} \ldots \mathbf{e_I}$ being the translation for a source sentence 'French' $\mathbf{f} = \mathbf{f_1} \ldots \mathbf{f_J}$ through an *exponential*, or *log-linear* model

$$p_\lambda(\mathbf{e}|\mathbf{f}) = \frac{\exp(\sum_{k=1}^{m} \lambda_k * h_k(\mathbf{e}, \mathbf{f}))}{\sum_{\mathbf{e}' \in \mathbf{E}} \exp(\sum_{k=1}^{m} \lambda_k * h_k(\mathbf{e}', \mathbf{f}))} \quad (1)$$

where $\mathbf{e}$ is a single candidate translation for $\mathbf{f}$ from the set of all English translations $\mathbf{E}$, λ is the parameter vector for the model, and each h_k is a feature function of $\mathbf{e}$ and $\mathbf{f}$. In practice, we restrict $\mathbf{E}$ to the set *Gen*($\mathbf{f}$) which is a set of highly likely translations discovered by a decoder (Vogel et al., 2003). Selecting a translation from this model under the Maximum A Posteriori (*MAP*) criteria yields

$$\text{transl}_\lambda(\mathbf{f}) = \arg\max_{\mathbf{e}} p_\lambda(\mathbf{e}|\mathbf{f}) \; . \quad (2)$$

This decision rule is optimal under the zero-one loss function, minimizing the Sentence Error Rate (Mangu et al., 2000). Using the log-linear form to model $p_\lambda(\mathbf{e}|\mathbf{f})$ gives us the flexibility to introduce overlapping features that can represent global context while decoding (searching the space of candidate translations) and rescoring (ranking a set of candidate translations before performing the $\arg\max$ operation), albeit at the cost of the traditional source-channel generative model of translation proposed in (Brown et al., 1993).

A significant impact of this paradigm shift, however, has been the movement to leverage the flexibility of the exponential model to maximize performance with respect to automatic evaluation met-

Proceedings of the ACL Workshop on Building and Using Parallel Texts, pages 208–215,
Ann Arbor, June 2005.

rics. Each evaluation metric considers different aspects of translation quality, both at the sentence and corpus level, often achieving high correlation to human evaluation (Doddington, 2002). It is clear that the decision rule stated in (1) does not reflect the choice of evaluation metric, and substantial work has been done to correct this mismatch in criteria. Approaches include integrating the metric into the decision rule, and learning λ to optimize the performance of the decision rule. In this paper we will compare and evaluate several aspects of these techniques, focusing on Minimum Error Rate (MER) training (Och, 2003) and Minimum Bayes Risk (MBR) decision rules, within a novel training environment that isolates the impact of each component of these methods.

2 Addressing Evaluation Metrics

We now describe competing strategies to address the problem of modeling the evaluation metric within the decoding and rescoring process, and introduce our contribution towards training non-tractable error surfaces. The methods discussed below make use of *Gen*($\mathbf{f}$), the approximation to the complete candidate translation space $\mathbf{E}$, referred to as an *n-best list*. Details regarding *n-best list* generation from decoder output can be found in (Ueffing et al., 2002).

2.1 Minimum Error Rate Training

The predominant approach to reconciling the mismatch between the MAP decision rule and the evaluation metric has been to train the parameters λ of the exponential model to correlate the *MAP* choice with the maximum score as indicated by the evaluation metric on a development set with known references (Och, 2003). We differentiate between the decision rule

$$\mathrm{transl}_\lambda(\mathbf{f}) = \arg\max_{\mathbf{e} \in \mathit{Gen}(\mathbf{f})} p_\lambda(\mathbf{e}|\mathbf{f}) \qquad (3a)$$

and the training criterion

$$\hat{\lambda} = \arg\min_{\lambda} \mathit{Loss}(\mathrm{transl}_\lambda(\vec{\mathbf{f}}), \vec{\mathbf{r}}) \qquad (3b)$$

where the Loss function returns an evaluation result quantifying the difference between the English candidate translation $\mathrm{transl}_\lambda(\mathbf{f})$ and its corresponding reference $\mathbf{r}$ for a source sentence $\mathbf{f}$. We indicate that this loss function is operating on a sequence of sentences with the vector notation. To avoid overfitting, and since MT researchers are generally blessed with an abundance of data, these sentences are from a separate development set.

The optimization problem (3b) is hard since the $\arg\max$ of (3a) causes the error surface to change in steps in $\mathbb{R}^m$, precluding the use of gradient based optimization methods. Smoothed error counts can be used to approximate the $\arg\max$ operator, but the resulting function still contains local minima. Grid-based line search approaches like Powell's algorithm could be applied but we can expect difficultly when choosing the appropriate grid size and starting parameters. In the following, we summarize the optimization algorithm for the unsmoothed error counts presented in (Och, 2003) and the implementation detailed in (Venugopal and Vogel, 2005).

- Regard $\mathit{Loss}(\mathrm{transl}_\lambda(\vec{\mathbf{f}}), \vec{\mathbf{r}})$ as defined in (3b) as a function of the parameter vector λ to optimize and take the $\arg\max$ to compute $\mathrm{transl}_\lambda(\vec{\mathbf{f}})$ over the translations *Gen*($\mathbf{f}$) according to the *n-best* list generated with an initial estimate λ^0.

- The error surface defined by Loss (as a function of λ) is piecewise linear with respect to a single model parameter λ_k, hence we can determine exactly where it would be useful (values that change the result of the $\arg\max$) to evaluate λ_k for a given sentence using a simple line intersection method.

- Merge the list of useful evaluation points for λ_k and evaluate the corpus level $\mathit{Loss}(\mathrm{transl}_\lambda(\vec{\mathbf{f}}), \vec{\mathbf{r}})$ at each one.

- Select the model parameter that represents the lowest Loss as k varies, set λ_k and consider the parameter λ_j for another dimension j.

This training algorithm, referred to as minimum error rate (MER) training, is a greedy search in each dimension of λ, made efficient by realizing that within each dimension, we can compute the points at which changes in λ actually have an impact on Loss. The appropriate considerations for termination and initial starting points relevant to any greedy search procedure must be accounted for. From the

nature of the training procedure and the *MAP* decision rule, we can expect that the parameters selected by MER training will strongly favor a few translations in the *n-best* list, namely for each source sentence the one resulting in the best score, moving most of the probability mass towards the translation that it believes should be selected. This is due to the decision rule, rather than the training procedure, as we will see when we consider alternative decision rules.

2.2 The Minimum Bayes Risk Decision Rule

The Minimum Bayes Risk Decision Rule as proposed by (Mangu et al., 2000) for the Word Error Rate Metric in speech recognition, and (Kumar and Byrne, 2004) when applied to translation, changes the decision rule in (2) to select the translation that has the lowest expected loss $\mathbf{E}[Loss(\mathbf{e}, \mathbf{r})]$, which can be estimated by considering a weighted $Loss$ between $\mathbf{e}$ and the elements of the *n-best* list, the approximation to $\mathbf{E}$, as described in (Mangu et al., 2000). The resulting decision rule is:

$$\mathrm{transl}_\lambda(\mathbf{f}) = \arg\min_{\mathbf{e} \in Gen(\mathbf{f})} \sum_{\mathbf{e}' \in Gen(\mathbf{f})} Loss(\mathbf{e}, \mathbf{e}') p_\lambda(\mathbf{e}'|\mathbf{f}) \ . \tag{4}$$

(Kumar and Byrne, 2004) explicitly consider selecting both $\mathbf{e}$ and $\mathbf{a}$, an alignment between the English and French sentences. Under a phrase based translation model (Koehn et al., 2003; Marcu and Wong, 2002), this distinction is important and will be discussed in more detail. The representation of the evaluation metric or the $Loss$ function is in the decision rule, rather than in the training criterion for the exponential model. This criterion is hard to optimize for the same reason as the criterion in (3b): the objective function is not continuous in λ. To make things worse, it is more expensive to evaluate the function at a given λ, since the decision rule involves a sum over all translations.

2.3 MBR and the Exponential Model

Previous work has reported the success of the MBR decision rule with fixed parameters relating independent underlying models, typically including only the language model and the translation model as features in the exponential model.

We extend the MBR approach by developing a training method to optimize the parameters λ in the exponential model as an explicit form for the conditional distribution in equation (1). The training task under the MBR criterion is

$$\lambda^* = \arg\min_\lambda Loss(\mathrm{transl}_\lambda(\vec{\mathbf{f}}), \vec{\mathbf{r}}) \tag{5a}$$

where

$$\mathrm{transl}_\lambda(\mathbf{f}) = \arg\min_{\mathbf{e} \in Gen(\mathbf{f})} \sum_{\mathbf{e}' \in Gen(\mathbf{f})} Loss(\mathbf{e}, \mathbf{e}') p_\lambda(\mathbf{e}'|\mathbf{f}) \ . \tag{5b}$$

We begin with several observations about this optimization criterion.

- The MAP optimal λ^* are not the optimal parameters for this training criterion.
- We can expect the error surface of the MBR training criterion to contain larger sections of similar altitude, since the decision rule emphasizes consensus.
- The piecewise linearity observation made in (Papineni et al., 2002) is no longer applicable since we cannot move the log operation into the expected value.

3 Score Sampling

Motivated by the challenges that the MBR training criterion presents, we present a training method that is based on the assumption that the error surface is locally non-smooth but consists of local regions of similar $Loss$ values. We would like to focus the search within regions of the parameter space that result in low $Loss$ values, simulating the effect that the MER training process achieves when it determines the merged error boundaries across a set of sentences.

Let $Score(\lambda)$ be some function of $Loss(\mathrm{transl}_\lambda(\vec{\mathbf{f}}), \vec{\mathbf{r}})$ that is greater or equal zero, decreases monotonically with $Loss$, and for which $\int (Score(\lambda) - \min_{\lambda'} Score(\lambda')) d\lambda$ is finite; e.g., $1 - Loss(\mathrm{transl}_\lambda(\vec{\mathbf{f}}), \vec{\mathbf{r}})$ for the word-error rate (WER) loss and a bounded parameter space. While sampling parameter vectors λ and estimating $Loss$ in these points, we will constantly refine our estimate of the error surface and thereby of the $Score$ function. The main idea in our score

sampling algorithm is to make use of this $Score$ estimate by constructing a probability distribution over the parameter space that depends on the $Score$ estimate in the current iteration step i and sample the parameter vector λ^{i+1} for the next iteration from that distribution. More precisely, let $\widehat{Sc}^{(i)}$ be the estimate of $Score$ in iteration i (we will explain how to obtain this estimate below). Then the probability distribution from which we sample the parameter vector to test in the next iteration is given by:

$$p(\lambda) = \frac{\widehat{Sc}^{(i)}(\lambda) - \min_{\lambda'} \widehat{Sc}^{(i)}(\lambda')}{\int (\widehat{Sc}^{(i)}(\lambda) - \min_{\lambda'} \widehat{Sc}^{(i)}(\lambda'))\, d\lambda} \quad (6)$$

This distribution produces a sequence $\lambda^1, \ldots, \lambda^n$ of parameter vectors that are more concentrated in areas that result in a high $Score$. We can select the value from this sequence that generates the highest $Score$, just as in the MER training process.

The exact method of obtaining the $Score$ estimate $\widehat{Sc}$ is crucial: If we are not careful enough and guess too low values of $\widehat{Sc}(\lambda)$ for parameter regions that are still unknown to us, the resulting sampling distribution p might be zero in those regions and thus potentially optimal parameters might never be sampled. Rather than aiming for a consistent estimator of $Score$ (i.e., an estimator that converges to $Score$ when the sample size goes to infinity), we design $\widehat{Sc}$ with regard to yielding a suitable sampling distribution p.

Assume that the parameter space is bounded such that $min_k \leq \lambda_k \leq max_k$ for each dimension k, We then define a set of pivots $\mathcal{P}$, forming a grid of points in $\mathbb{R}^m$ that are evenly spaced between min_k and max_k for each dimension k. Each pivot represents a region of the parameter space where we expect generally consistent values of $Score$. We do not restrict the values of λ_m to be at these pivot points as a grid search would do, rather we treat the pivots as landmarks within the search space.

We approximate the distribution $p(\lambda)$ with the discrete distribution $p(\lambda \in \mathcal{P})$, leaving the problem of estimating $|\mathcal{P}|$ parameters. Initially, we set p to be uniform, i.e., $p^{(0)}(\lambda) = 1/|\mathcal{P}|$. For subsequent iterations, we now need an estimate of $Score(\lambda)$ for each pivot $\lambda \in \mathcal{P}$ in the discrete version of equation (6) to obtain the new sampling distribution p. Each iteration i proceeds as follows.

- Sample $\widetilde{\lambda^i}$ from the discrete distribution $p^{(i-1)}(\lambda \in \mathcal{P})$ obtained by the previous iteration.
- Sample the new parameter vector λ^i by choosing for each $k \in \{1, \ldots, m\}$, $\lambda^i_k := \widetilde{\lambda^i_k} + \varepsilon_k$, where ε_k is sampled uniformly from the interval $(-d_k/2, d_k/2)$ and d_k is the distance between neighboring pivot points along dimension k. Thus, λ^i is sampled from a region around the sampled pivot.
- Evaluate $Score(\lambda^i)$ and distribute this score to obtain new estimates $\widehat{Sc}^{(i)}(\lambda)$ for all pivots $\lambda \in \mathcal{P}$ as described below.
- Use the updated estimates $\widehat{Sc}^{(i)}$ to generate the sampling distribution $p^{(i)}$ for the next iteration according to

$$p^{(i)}(\lambda) = \frac{\widehat{Sc}^{(i)}(\lambda) - \min_{\lambda'} \widehat{Sc}^{(i)}(\lambda')}{\sum_{\lambda \in \mathcal{P}} (\widehat{Sc}^{(i)}(\lambda) - \min_{\lambda'} \widehat{Sc}^{(i)}(\lambda'))} .$$

The score $Score(\lambda^i)$ of the currently evaluated parameter vector does not only influence the score estimate at the pivot point of the respective region, but the estimates at all pivot points. The closest pivots are influenced most strongly. More precisely, for each pivot $\lambda \in \mathcal{P}$, $\widehat{Sc}^{(i)}(\lambda)$ is a weighted average of $Score(\lambda^1), \ldots, Score(\lambda^i)$, where the weights $w^{(i)}(\lambda)$ are chosen according to

$$\begin{aligned} w^{(i)}(\lambda) &= \mathrm{infl}^{(i)}(\lambda) \times \mathrm{corr}^{(i)}(\lambda) \quad \text{with} \\ \mathrm{infl}^{(i)}(\lambda) &= \mathrm{mvnpdf}(\lambda, \lambda^i, \Sigma) \quad \text{and} \\ \mathrm{corr}^{(i)}(\lambda) &= 1/p^{(i-1)}(\lambda) . \end{aligned}$$

Here, $\mathrm{mvnpdf}(x, \mu, \Sigma)$ denotes the m-dimensional multivariate-normal probability density function with mean μ and covariance matrix Σ, evaluated at point x. We chose the covariance matrix $\Sigma = \mathrm{diag}(d_1^2, \ldots, d_m^2)$, where again d_k is the distance between neighboring grid points along dimension k. The term $\mathrm{infl}^{(i)}(\lambda)$ quantifies the influence of the evaluated point λ^i on the pivot λ, while $\mathrm{corr}^{(i)}(\lambda)$ is a correction term for the bias introduced by having sampled λ^i from $p^{(i-1)}$.

Smoothing uncertain regions In the beginning of the optimization process, there will be pivot regions that have not yet been sampled from and for which not even close-by regions have been sampled yet. This will be reflected in the low sum of influence terms $\text{infl}^{(1)}(\lambda) + \cdots + \text{infl}^{(i)}(\lambda)$ of the respective pivot points λ. It is therefore advisable to discount some probability mass from $p^{(i)}$ and distribute it over pivots with low influence sums (reflecting low confidence in the respective score estimates) according to some smoothing procedure.

4 N-Best lists in Phrase Based Decoding

The methods described above make extensive use of *n-best* lists to approximate the search space of candidate translations. In phrase based decoding we often interpret the MAP decision rule to select the top scoring path in the translation lattice. Selecting a particular path means in fact selecting the pair $\langle \mathbf{e}, \mathbf{s} \rangle$, where $\mathbf{s}$ is a segmentation of the the source sentence $\mathbf{f}$ into phrases and alignments onto their translations in $\mathbf{e}$. Kumar and Byrne (2004) represent this decision explicitly, since the $Loss$ metrics considered in their work evaluate alignment information as well as lexical (word) level output. When considering lexical scores as we do here, the decision rule minimizing 0/1 loss actually needs to take the sum over all potential segmentations that can generate the same word sequence. In practice, we only consider the high probability segmentation decisions, namely the ones that were found in the *n-best* list. This gives the *0/1 loss* criterion shown below.

$$\text{transl}_\lambda(\mathbf{f}) = \arg\max_{\mathbf{e}} \sum_{\mathbf{s}} p_\lambda(\mathbf{e}, \mathbf{s}|\mathbf{f}) \qquad (7)$$

The *0/1 loss* criterion favors translations that are supported by several segmentation decisions. In the context of phrase-based translations, this is a useful criterion, since a given lexical target word sequence can be correctly segmented in several different ways, all of which would be scored equally by an evaluation metric that only considers the word sequence.

5 Experimental Framework

Our goal is to evaluate the impact of the three decision rules discussed above on a large scale translation task that takes advantage of multidimensional features in the exponential model. In this section we describe the experimental framework used in this evaluation.

5.1 Data Sets and Resources

We perform our analysis on the data provided by the 2005 ACL Workshop in Exploiting Parallel Texts for Statistical Machine Translation, working with the French-English Europarl corpus. This corpus consists of 688031 sentence pairs, with approximately 156 million words on the French side, and 138 million words on the English side. We use the data as provided by the workshop and run lower casing as our only preprocessing step. We use the 15.5 million entry phrase translation table as provided for the shared workshop task for the French-English data set. Each translation pair has a set of 5 associated phrase translation scores that represent the maximum likelihood estimate of the phrase as well as internal alignment probabilities. We also use the English language model as provided for the shared task. Since each of these decision rules has its respective training process, we split the workshop test set of 2000 sentences into a development and test set using random splitting. We tried two decoders for translating these sets. The first system is the Pharaoh decoder provided by (Koehn et al., 2003) for the shared data task. The Pharaoh decoder has support for multiple translation and language model scores as well as simple phrase distortion and word length models. The pruning and distortion limit parameters remain the same as in the provided initialization scripts, i.e., $DistortionLimit = 4, BeamThreshold = 0.1, Stack = 100$. For further information on these parameter settings, confer (Koehn et al., 2003). Pharaoh is interesting for our optimization task because its eight different models lead to a search space with seven free parameters. Here, a principled optimization procedure is crucial. The second decoder we tried is the CMU Statistical Translation System (Vogel et al., 2003) augmented with the four translation models provided by the Pharaoh system, in the following called CMU-Pharaoh. This system also leads to a search space with seven free parameters.

5.2 N-Best lists

As mentioned earlier, the model parameters λ play a large role in the search space explored by a pruning beam search decoder. These parameters affect the histogram and beam pruning as well as the future cost estimation used in the Pharaoh and CMU decoders. The initial parameter file for Pharaoh provided by the workshop provided a very poor estimate of λ, resulting in an n-best list of limited potential. To account for this condition, we ran Minimum Error Rate training on the development data to determine scaling factors that can generate a *n-best* list with high quality translations. We realize that this step biases the *n-best* list towards the MAP criteria, since its parameters will likely cause more aggressive pruning. However, since we have chosen a large N=1000, and retrain the MBR, MAP, and 0/1 loss parameters separately, we do not feel that the bias has a strong impact on the evaluation.

5.3 Evaluation Metric

This paper focuses on the BLEU metric as presented in (Papineni et al., 2002). The BLEU metric is defined on a corpus level as follows.

$$Score(\vec{\mathbf{e}}, \vec{\mathbf{r}}) = BP(\vec{\mathbf{e}}, \vec{\mathbf{r}}) * \exp(\frac{1}{N}\sum_{1}^{N}(\log p_n))$$

where p_n represent the precision of n-grams suggested in $\vec{\mathbf{e}}$ and BP is a brevity penalty measuring the relative shortness of $\vec{\mathbf{e}}$ over the whole corpus. To use the BLEU metric in the candidate pairwise loss calculation in (4), we need to make a decision regarding cases where higher order n-grams matches are not found between two candidates. Kumar and Byrne (2004) suggest that if any n-grams are not matched then the pairwise BLEU score is set to zero. As an alternative we first estimate corpus-wide n-gram counts on the development set. When the pairwise counts are collected between sentences pairs, they are added onto the baseline corpus counts to and scored by BLEU. This scoring simulates the process of scoring additional sentences after seeing a whole corpus.

5.4 Training Environment

It is important to separate the impact of the decision rule from the success of the training procedure. To appropriately compare the MAP, 0/1 loss and MBR decisions rules, they must all be trained with the same training method, here we use the Score Sampling training method described above. We also report MAP scores using the MER training described above to determine the impact of the training algorithm for MAP. Note that the MER training approach cannot be performed on the MBR decision rule, as explained in Section 2.3. MER training is initialized at random values of λ and run (successive greedy search over the parameters) until there is no change in the error for three complete cycles through the parameter set. This process is repeated with new starting parameters as well as permutations of the parameter search order to ensure that there is no bias in the search towards a particular parameter. To improve efficiency, pairwise scores are cached across requests for the score at different values of λ, and for MBR only the $\mathbf{E}[Loss(\mathbf{e}, \mathbf{r})]$ for the top twenty hypotheses as ranked by the model are computed.

6 Results

The results in Table 1 compare the BLEU score achieved by each training method on the development and test data for both Pharaoh and CMU-Pharaoh. Score-sampling training was run for 150 iterations to find λ for each decision rule. The MAP-MER training was performed to evaluate the effect of the greedy search method on the generalization of the development set results. Each row represents an alternative training method described in this paper, while the test set columns indicate the criteria used to select the final translation output $\vec{\mathbf{e}}$. The bold face scores are the scores for matching training and testing methods. The underlined score is the highest test set score, achieved by MBR decoding using the CMU-Pharaoh system trained for the MBR decision rule with the score-sampling algorithm. When comparing MER training for MAP-decoding with score-sampling training for MAP-decoding, score-sampling surprisingly outperforms MER training for both Pharaoh and CMU-Pharaoh, although MER training is specifically tailored to the MAP metric. Note, however, that our score-sampling algorithm has a considerably longer running time (several hours) than the MER algorithm (several minutes). Interestingly, within MER train-

training method	Dev. set sc.	test set sc. MAP	test set sc. 0/1 loss	test set sc. MBR
MAP MER (Pharaoh)	29.08	**29.30**	29.42	29.36
MAP score-sampl. (Pharaoh)	29.08	**29.41**	29.24	29.30
0/1 loss sc.-s. (Pharaoh)	29.08	29.16	**29.28**	29.30
MBR sc.-s. (Pharaoh)	29.00	29.11	29.08	**29.17**
MAP MER (CMU-Pharaoh)	28.80	**29.02**	29.41	29.60
MAP sc.-s. (CMU-Ph.)	29.10	**29.85**	29.75	29.55
0/1 loss sc.-s. (CMU-Ph.)	28.36	29.97	**29.91**	29.72
MBR sc.-s. (CMU-Ph.)	28.36	30.18	30.16	**30.28**

Table 1. Comparing BLEU scores generated by alternative training methods and decision rules

ing for Pharaoh, the 0/1 loss metric is the top performer; we believe the reason for this disparity between training and test methods is the impact of phrasal consistency as a valuable measure within the *n-best* list.

The relative performance of MBR score-sampling w.r.t. MAP and 0/1-loss score sampling is quite different between Pharaoh and CMU-Pharaoh: While MBR score-sampling performs worse than MAP and 0/1-loss score sampling for Pharaoh, it yields the best test scores across the board for CMU-Pharaoh. A possible reason is that the n-best lists generated by Pharaoh have a large percentage of lexically identical translations, differing only in their segmentations. As a result, the 1000-best lists generated by Pharaoh contain only a small percentage of unique translations, a condition that reduces the potential of the Minimum Bayes Risk methods. The CMU decoder, contrariwise, prunes away alternatives below a certain score-threshold during decoding and does not recover them when generating the n-best list. The n-best lists of this system are therefore typically more diverse and in particular contain far more unique translations.

7 Conclusions and Further Work

This work describes a general algorithm for the efficient optimization of error counts for an arbitrary $Loss$ function, allowing us to compare and evaluate the impact of alternative decision rules for statistical machine translation. Our results suggest the value and sensitivity of the translation process to the $Loss$ function at the decoding and reordering stages of the process. As phrase-based translation and reordering models begin to dominate the state of the art in machine translation, it will become increasingly important to understand the nature and consistency of *n-best* list training approaches. Our results are reported on a complete package of translation tools and resources, allowing the reader to easily recreate and build upon our framework. Further research might lie in finding efficient representations of Bayes Risk loss functions within the decoding process (rather than just using MBR to rescore n-best lists), as well as analyses on different language pairs from the available Europarl data. We have shown score sampling to be an effective training method to conduct these experiments and we hope to establish its use in the changing landscape of automatic translation evaluation. The source code is available at: `www.cs.cmu.edu/~zollmann/scoresampling/`

8 Acknowledgments

We thank Stephan Vogel, Ying Zhang, and the anonymous reviewers for their valuable comments and suggestions.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

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

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology and North*

American Association for Computational Linguistics Conference (HLT/NAACL), Edomonton, Canada, May 27-June 1.

Shankar Kumar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL)*, Boston,MA, May 27-June 1.

Lidia Mangu, Eric Brill, and Andreas Stolcke. 2000. Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *CoRR*, cs.CL/0010012.

Daniel Marcu and William Wong. 2002. A phrase-based, joint probability model for statistical machine translation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, Philadephia, PA, July 6-7.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the Association for Computational Linguistics*, Sapporo, Japan, July 6-7.

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

Nicola Ueffing, Franz Josef Och, and Hermann Ney. 2002. Generation of word graphs in statistical machine translation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*, Philadephia, PA, July 6-7.

Ashish Venugopal and Stephan Vogel. 2005. Considerations in mce and mmi training for statistical machine translation. In *Proceedings of the Tenth Conference of the European Association for Machine Translation (EAMT-05)*, Budapest, Hungary, May. The European Association for Machine Translation.

Stephan Vogel, Ying Zhang, Fei Huang, Alicia Tribble, Ashish Venogupal, Bing Zhao, and Alex Waibel. 2003. The CMU statistical translation system. In *Proceedings of MT Summit IX*, New Orleans, LA, September.