A Unified Approach to Minimum Risk Training and Decoding

Abhishek Arun, Barry Haddow and Philipp Koehn
School of Informatics
University of Edinburgh
Edinburgh, EH8 9AB, UK
a.arun@sms.ed.ac.uk, {bhaddow,pkoehn}@inf.ed.ac.uk

Abstract

We present a unified approach to performing minimum risk training and minimum Bayes risk (MBR) decoding with BLEU in a phrase-based model. Key to our approach is the use of a Gibbs sampler that allows us to explore the *entire probability distribution* and maintain a strict probabilistic formulation across the pipeline. We also describe a new sampling algorithm called *corpus sampling* which allows us at training time to use BLEU instead of an approximation thereof. Our approach is theoretically sound and gives better (up to +0.6%BLEU) and more stable results than the standard MERT optimization algorithm. By comparing our approach to lattice MBR, we are also able to gain crucial insights about both methods.

1 Introduction

According to statistical decision theory, the optimal decision rule for any statistical model is the solution that minimizes its risk (expected loss). This solution is often referred to as the Minimum Bayes Risk (MBR) solution (Kumar and Byrne, 2004). Since machine translation (MT) models are typically evaluated by BLEU (Papineni et al., 2002), a loss function which rewards partial matches, the MBR solution is to be preferred to the Maximum A Posteriori (MAP) solution.

In most statistical MT (SMT) systems, MBR is implemented as a reranker of a list[1] of translations generated by a first-pass decoder. This decoder typically assigns unnormalised log probabilities (known as *scores*) to each translation hypothesis, so these scores must be converted to probabilities in order to apply MBR. In order to perform this conversion, it is first necessary to compute the normalization function Z. Since Z is defined as an intractable sum over all possible translations, it is approximated by summing over the translations in the list. The second step is to find the correct scale factor for the scores using a hyper-parameter search over held-out data. This is needed because the model parameters for the first-pass decoder are normally learnt using MERT (Och, 2003), which is invariant under scaling of the scores.

Both these steps are theoretically unsatisfactory methods of estimating the posterior probability distribution since the approximation to Z is an unbounded term and the scaling factor is an artificial way of inducing a probability distribution.

Recently, (Tromble et al., 2008; Kumar et al., 2009) have shown that using a search lattice to improve the estimation of the true probability distribution can lead to improved MBR performance. However, these approaches still rely on MERT for training the base model, and in fact introduce several extra parameters which must also be estimated using either grid search or a second MERT run. The lattice pruning required to make these techniques tractable is quite drastic, and is in addition to the pruning already performed during the search. Such extensive pruning is liable to render any probability estimates heavily biased (Blunsom and Osborne, 2008; Bouchard-Côté et al., 2009).

Here, we present a unified approach to training and decoding in a phrase-based translation model (Koehn et al., 2003) which keeps the objective constant across the translation pipeline and so obviates the need for any extra hyper-parameter fitting. We use the phrase-based Gibbs sampler of Arun et al. (2009) at training time to compute the gradient of our *minimum risk training* objective in order to apply first-order optimization techniques,

[1] We use the term list to denote any enumerable representation of translation hypotheses e.g n-best list, translation lattice or forest.

and at test time we use it to estimate the posterior distribution required by MBR (Section 3).

We experimented with two different objective functions for training (Section 4). First, following (Arun et al., 2009), we define our objective at the sentence-level using a sentence-level variant of BLEU. Then, in order to reduce the mismatch between training and test loss functions, we also tried directly optimising the expected corpus level BLEU, where we introduce a novel sampling technique, which we call *corpus sampling* to calculate the required expectations.

The methods presented in this paper are theoretically sound. Moreover, experimental evidence on three language pairs shows that our training regime is more stable than MERT, able to generalize better and generally leads to improvement in translation when used with sampling based MBR (Section 5). An added benefit is that the trained weights also lead to better performance when used with a beam-search based decoder.

2 Inference methods for MT

We assume a phrase-based machine translation model, defined with a log-linear form, with feature function vector **h** and parametrized by weight vector $\boldsymbol{\theta}$, as described in Koehn et al. (2003). The input sentence, f, is segmented into phrases, which are sequences of adjacent words. Each source phrase is translated into the target language, to produce an output sentence e and an alignment a representing the mapping from source to target phrases. Phrases are allowed to be reordered.

$$p(e,a|f;\boldsymbol{\theta}) = \frac{\exp\left[\boldsymbol{\theta}\cdot\mathbf{h}(e,a,f)\right]}{\sum_{\langle e',a'\rangle}\exp\left[\boldsymbol{\theta}\cdot\mathbf{h}(e',a',f)\right]} \quad (1)$$

MAP decoding under this model consists of finding the most likely output string, e^*:

$$e^* = \text{argmax}_e \sum_{a\in\triangle(e,f)} p(e,a|f) \quad (2)$$

where $\triangle(e,f)$ is the set of all derivations of output string e given source string f.

Summing over all the derivations is intractable, making approximations necessary. The most common of these approximations is the *Viterbi* approximation, which simply chooses the most likely derivation $\langle e^*, a^*\rangle$. This approximation can be computed in polynomial time via dynamic programming (DP). Though fast and effective for many problems, it has two serious drawbacks for probabilistic inference. First, the error incurred by the Viterbi maximum with respect to the true model maximum is unbounded. Second, the DP solution requires substantial pruning and restricts the use of non-local features. The latter problem persists even in the *variational* approximations of Li et al. (2009), which attempt to solve the former.

2.1 Gibbs sampling for phrase-based MT

An alternate approximate inference method for phrase-based MT without any of the previously mentioned drawbacks is the Gibbs sampler (Geman and Geman, 1984) of Arun et al. (2009) which draws samples from the posterior distribution of the translation model. For the work presented in this paper, we use this sampler.

The sampler produces a sequence of samples, $\mathcal{S}_1^N = (e_1, a_1)\dots(e_N, a_N)$, that are drawn from the distribution $p(e,a|f)$. These samples can be used to estimate the expectation of a function $h(e,a,f)$ as follows:

$$\mathbb{E}_{p(a,e|f)}[h] = \lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N} h(a_i,e_i,f) \quad (3)$$

3 Decoding

In this work, we are interested in performing MBR decoding with BLEU. We define the MBR decision rule following Tromble et al. (2008):

$$e^* = \arg\max_{e\in\epsilon_H}\sum_{e'\in\epsilon_E}\text{BLEU}_e(e')p(e'|f) \quad (4)$$

where ϵ_H refers to the hypothesis space from which translations are chosen, ϵ_E refers to the evidence space used for calculating risk and $\text{BLEU}_e(e')$ is a gain function that indicates the reward of hypothesising e' when the reference solution is e.

To perform MBR decoding using the sampler, let the function h in Equation 3 be the indicator function $h = \delta(a,\hat{a})\delta(e,\hat{e})$. Then, Equation 3 provides an estimate of $p(\hat{a},\hat{e}|f)$, and using $h = \delta(e,\hat{e})$ marginalizes over all derivations a', yielding an estimate of $p(\hat{e}|f)$. MBR is computed at the sentence-level while BLEU is a corpus-level metric, so instead we use a sentence-level approximation of BLEU.[2]

The sampler can be used to perform two other decoding tasks: the mode of the estimated distribution $p(\hat{a},\hat{e}|f)$ is the maximum derivation (MaxDeriv) solution while the mode of $p(\hat{e}|f)$ is the maximum translation (MaxTrans) solution.

[2]The ngram precision counts are smoothed by adding 0.01 for n $>$ 1

4 Minimum Risk Training

In order to train models suitable for use with Max-Trans or MBR decoding, we need to employ a training method which takes account of the whole distribution. To this end, we employ minimum risk training to find weights θ for Equation 1 that minimize the expected loss on the training set. We consider two variants of minimum risk training: *sentence sampling* optimizes an objective defined at the sentence level and *corpus sampling* a corpus-based objective.

4.1 Sentence sampling

Since BLEU, the metric we care about, is a gain function, our objective function maximizes the expected gain of our model. The expected gain, $\mathcal{G}$ of a probabilistic translation model on a corpus $\mathcal{D}$, defined with respect to the gain function $\text{BLEU}_{\hat{e}}(e)$ is given by

$$\mathcal{G} = \sum_{\langle \hat{e}, f\rangle \in \mathcal{D}} \sum_{e,a} p(e,a|f)\text{BLEU}_{\hat{e}}(e) \tag{5}$$

where $\hat{e}$ is the reference translation, e is a hypothesis translation and BLEU refers to the sentence-level approximation of the metric.

Using the probabilistic formulation of Equation 1, the optimization of the objective in (5) is facilitated by the fact that it is continuous and differentiable with respect to the model parameters θ to give

$$\frac{\partial \mathcal{G}}{\partial \theta_k} = \sum_{\substack{\langle \hat{e}, f\rangle \\ \in \mathcal{D}}} \sum_{e,a} \text{BLEU}_{\hat{e}}(e) \frac{\partial p}{\partial \theta_k} \tag{6}$$

$$\text{where } \frac{\partial p}{\partial \theta_k} = \left(h_k - \mathbb{E}_{p(e,a|f)}[h_k]\right) p(e,a|f)$$

Since the gradient is expressed in terms of expectations of feature values, it can easily be calculated using the sampler and then first-order optimization techniques can be applied to find optimal values of θ. Because of the noise introduced by the sampler, we used stochastic gradient descent (SGD), with a learning rate that gets updated after each step proportionally to difference in successive gradients (Schraudolph, 1999).

While our initial formulation of minimum risk training is similar to that of Arun et al. (2009), in preliminary experiments we observed a tendency for translation performance on held-out data to quickly increase to a maximum and then plateau. Hypothesizing that we were being trapped in local maxima as $\mathcal{G}$ is non-convex, we decided to employ *deterministic annealing* (Rose, 1998) to smooth the objective function to ensure that the optimizer explored as large a region as possible of the space before it settled on an optimal weight set. Our instantiation of deterministic annealing (DA) is based on the work of Smith and Eisner (2006), and involves the addition of an entropic prior to the objective in Equation 5 to give

$$\hat{\mathcal{G}} = \sum_{\langle \hat{e}, f\rangle \in \mathcal{D}} \left[\left(\sum_{e,a} p(e,a|f)\text{BLEU}_{\hat{e}}(e) \right) + T.H(p) \right]$$

where $H(p)$ is the entropy of the probability distribution $p(e,a|f)$, and $T > 0$ is a temperature paramater which is gradually lowered as the optimization progresses according to some *annealing schedule*.

Differentiating with respect to θ_k then shows that the annealed gradient is given by the following expression:

$$\sum_{\substack{\langle \hat{e}, f\rangle \\ \in \mathcal{D}}} \sum_{e,a} \left(\text{BLEU}_{\hat{e}}(e) - T(1 + \log p)\right) \frac{\partial p}{\partial \theta_k}$$

$$\text{where } \frac{\partial p}{\partial \theta_k} = \left(h_k - \mathbb{E}_{p(e,a|f)}[h_k]\right) p(e,a|f)$$

A high value of T leads the optimizer to find weights which describe a fairly flat distribution, whereas a lower value of T pushes the optimizer towards a more peaked distribution. We perform 10 to 20 iterations of SGD at each temperature.

In their deterministic annealing formulation, (Smith and Eisner, 2006; Li and Eisner, 2009), express the parameterization of the distribution $\boldsymbol{\theta}$ as $\gamma\hat{\boldsymbol{\theta}}$ (where γ is the *scaling factor*) and perform optimization in two steps, the first optimizing $\hat{\boldsymbol{\theta}}$ and the second optimizing γ. We experimented with this two stage optimization process, but found that simply performing an unconstrained optimization on $\boldsymbol{\theta}$ gave better results.

4.2 Corpus sampling

While the objective functions in Equations 5 and 4.1 use a sentence-level variant of BLEU, the model's test-time performance is evaluated with corpus level BLEU. The lack of correlation between sentence-level BLEU and corpus BLEU is well-known (Chiang et al., 2008a). Therefore, in an effort to address this issue, we tried maximizing expected corpus BLEU directly.

In other words, given a training corpus of the form $\langle \mathcal{C}_F, \mathcal{C}_{\hat{E}} \rangle$ where $\mathcal{C}_F$ is a set of source sentences and $\mathcal{C}_{\hat{E}}$ its corresponding reference translations, we consider a gain function defined on the hypothesized translation $\mathcal{C}_E$ of the input $\mathcal{C}_F$ with respect to $\mathcal{C}_{\hat{E}}$.

The objective in equation 5 therefore becomes:

$$\mathcal{G} = \sum_{\mathcal{C}_E} P(\mathcal{C}_E|\mathcal{C}_F)\text{BLEU}_{\mathcal{C}_{\hat{E}}}(\mathcal{C}_E) \qquad (7)$$

The pair $(\mathcal{C}_E, \mathcal{C}_F)$ is denoted as a *corpus sample* corresponding to a sequence $(e^1, a^1), \ldots, (e^N, a^N)$ of derivations of the corresponding source strings $f^1, \ldots, f^N$ of source corpus $\mathcal{C}_F$.

Although the sampler described in Section 2 generates samples at the sentence level, we can use it to generate corpus samples by applying the following procedure (see Figure 1). For each source sentence f^i in the corpus, we generate a sequence of samples $(e^i_1, a^i_1), \ldots, (e^i_n, a^i_n)$ using the sampler. From each of these sequences of samples, we then *resample* new sequences of derivation samples, one for each source sentence in the corpus. The first corpus sample is then obtained by iterating through the source sentences and taking the first resampled derivation for each sentence, then the second corpus sample by taking the second resampled derivation, and so on. The resampling step is necessary to eliminate any biases due to the order of the generated samples.

The corpus sampling procedure invariably generates a set of samples which are all distinct and so would give us a uniform estimate of the probability distribution $P(\mathcal{C}_E|\mathcal{C}_F)$. However this is not a problem since we are not interested in evaluating the actual distribution; we just need to calculate expectations of feature values and BLEU scores over the distribution. The feature values of a corpus sample are the average of the feature values of its constituting derivations and its BLEU score is computed based on the yield of its derivations.

When training using corpus sampling we process the training corpus in batches $\langle \mathcal{C}_F, \mathcal{C}_{\hat{E}} \rangle$, treating each batch as a corpus in its own right, and updating the weights after each batch.

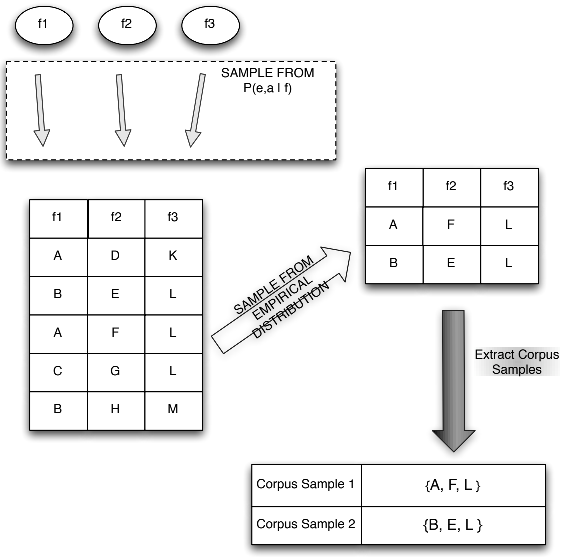
The gradient for the objective function in (7) is:

$$\frac{\partial \mathcal{G}}{\partial \theta_k} = \sum_{\mathcal{C}_E} \text{BLEU}_{\mathcal{C}_{\hat{E}}}(\mathcal{C}_E) \frac{\partial P}{\partial \theta_k}$$

$$\text{where } \frac{\partial P}{\partial \theta_k} = \left(h^{\mathcal{C}}_k - \mathbb{E}_{P(\mathcal{C}_E|\mathcal{C}_F)}[h^{\mathcal{C}}_k]\right) P(\mathcal{C}_E|\mathcal{C}_F)$$

where $h^{\mathcal{C}}_k$ is the k-th component of a corpus sample feature vector.

During deterministic annealing for sentence sampling, the entropy term is computed over the distribution $p(e, a|f)$ of each individual sentence. While corpus sampling, we are considering the distribution $P(\mathcal{C}_E|\mathcal{C}_F)$ but the estimated distribution is always uniform. So we define the entropic prior term over the distribution $p(e, a|f)$ of the sentences making up the corpus sample.



f1	f2	f3
A	D	K
B	E	L
A	F	L
C	G	L
B	H	M

f1	f2	f3
A	F	L
B	E	L

Corpus Sample 1	{A, F, L }
Corpus Sample 2	{B, E, L }

Figure 1: Example illustrating the extraction of 2 corpus samples for a corpus of source sentences f1, f2, f3. In the first step, we sample 5 derivations for each source sentence. We then resample 2 derivations from the empirical distributions of each source sentence.

The annealed corpus sampling objective is therefore:

$$\sum_{\mathcal{C}_E} P(\mathcal{C}_E|\mathcal{C}_F)\text{BLEU}_{\mathcal{C}_{\hat{E}}}(\mathcal{C}_E) + \frac{T}{|\mathcal{C}_F|} \sum_{f \in \mathcal{C}_F} H(p(e, a|f))$$

The gradient of this objective is of similar form to the sentence sampling gradient in Equation (6).

5 Experiments

5.1 Training Data and Preparation

The experiments in this section were performed using the Europarl section of the French-English and German-English parallel corpora from the WMT09 shared translation task (Callison-Burch et al., 2009), as well as 300k parallel Arabic-English sentences from the NIST MT evaluation training data.[3] For all language pairs, we constructed

[3]The Arabic-English training data consists of the eTIRR corpus (LDC2004E72), the Arabic news corpus (LDC2004T17), the Ummah corpus (LDC2004T18), and the

a phrase-based translation model as described in Koehn et al. (2003), limiting the phrase length to 5. The target side of the parallel corpus was used to train 3-gram language models. For the German and French systems, the DEV2006 set was used for model tuning and the first half of TEST2007 (in-domain) for heldout testing. Final testing was performed on NEWS-DEV2009B (out-of-domain) and the first half of TEST2008 (in-domain). For the Arabic system, the MT02 set (10 reference translations) was used for tuning and MT03 and MT05 (4 reference translations, each) were used for held-out testing and final testing respectively. To reduce the size of the phrase table, we used the association-score technique suggested by Johnson et al. (2007). Translation quality is reported using case-insensitive BLEU.

5.2 Baseline

Our baseline system is phrase-based Moses (Koehn et al., 2007) with feature weights trained using MERT. Moses and the Gibbs sampler use identical feature sets.[4]

The MERT optimization algorithm uses multiple random restarts to avoid getting stuck in a poor local optima. Therefore, every time MERT is run, it produces a slightly different final weight vector leading to varying test set results. While this characteristic of MERT is typically ignored, we account for it by performing MERT training 10 times for each of the 3 language pairs, decoding the test sets with each of the 10 optimized weight sets. We present the best and the worst test set results along with the mean and the standard deviation (σ) of these results in Table 1. We report results using the Moses implementation of Viterbi, nbest MBR and lattice MBR decoding (Kumar et al., 2009). [5] For both nbest and lattice MBR decoding, the hypothesis set was composed of the top 1000 unique translations produced by the Viterbi decoder, and the same 1000 translations were used as evidence set for nbest MBR.

As Table 1 shows, translation results using MERT optimized weights vary markedly from one tuning run to the other, with results varying from a range of 0.3% BLEU to 1.3% BLEU when using Viterbi decoding. We also see that, bar in-domain German to English, MBR decoding gives a small improvement on all other datasets.

Surprisingly, lattice MBR only gives improvements on two datasets and actually leads to a *drop* in performance on the other 3 datasets. We discuss possible reasons for this in Section 6.

5.3 Sentence sampling

At training time, the optimization algorithm is initialized with zero weights and the sampler is initialized with a random derivation from Moses. To get rid of any initialization biases, the first 100 samples are discarded.[6] We then run the sampler for 1000 iterations after which we perform *reheating* whereby the distribution is progressively flattened. Samples are not collected during this period. Reheating allows the sampler more mobility around the search space thus possibly escaping any local optima it might be trapped in. We subsequently run the sampler for 1000 more iterations. We denote this procedure as running 2 *chains* of the sampler. We use batch sizes of 96 randomly selected sentences for SGD optimization.

During DA, our cooling schedule is an exponentially decaying one with decay rate set to 0.9, performing 20 iterations of SGD optimization at each temperature setting. Five training runs were performed and the BLEU scores averaged. The feature weights were output every 50 iterations and performance measured on the heldout set by running the sampler as a decoder. At decode time, we use the same sampler configurations as during training but run 2 chains each for 5000 iterations.

For MBR decoding, we use the entirety of this sample set as our evidence set and use the top 1000 most probable translations as the hypothesis set.

5.4 Corpus sampling

For our corpus sampling experiments, we sample using the same procedure as in sentence sampling but using 2 chains of 2000 iterations. We then resample 2000 corpus samples from the empirical distribution estimated from the first 4000 samples. For Arabic-English training, we used batch sizes of 100 randomly selected sentences for experiments without DA and batches of 400 random

sentences with confidence $c > 0.995$ in the ISI automatically extracted web parallel corpus (LDC2006T02).

[4]We use 5 translation model scores, distance-based distortion, language model and word penalty. The reordering limit is set to 6 for all experiments.

[5]For nbest and lattice MBR decoding, we optimized for the scaling factor using a grid-search on held-out data. For lattice MBR decoding, we optimized the lattice density and set the p and r parameters as per Tromble et al. (2008).

[6]This procedure is referred to as *burn-in* in the MCMC literature.

	Viterbi				nMBR				lMBR			
	min	max	mean	σ	min	max	mean	σ	min	max	mean	σ
AR-EN MT05	43.7	44.3	44.0	0.17	44.2	44.5	44.4	0.13	44.2	44.6	**44.5**	0.12
FR-EN In	33.1	33.4	33.3	0.10	33.2	33.6	**33.4**	0.12	32.3	32.7	32.6	0.13
FR-EN Out	19.1	19.6	19.4	0.18	19.3	19.7	**19.5**	0.12	19.1	19.4	19.3	0.12
DE-EN In	27.6	27.9	**27.8**	0.10	27.6	27.9	27.7	0.10	27.2	27.5	27.4	0.10
DE-EN Out	14.9	16.2	15.7	0.33	15.0	16.3	15.7	0.33	15.3	16.4	**16.0**	0.30

Table 1: Baseline results - MERT trained models decoded using Viterbi, nbest MBR (nMBR) and lattice MBR (lMBR). MERT was run 10 times for each language pair. We report minimum, maximum, mean and standard deviation of test set BLEU scores across the 10 runs.

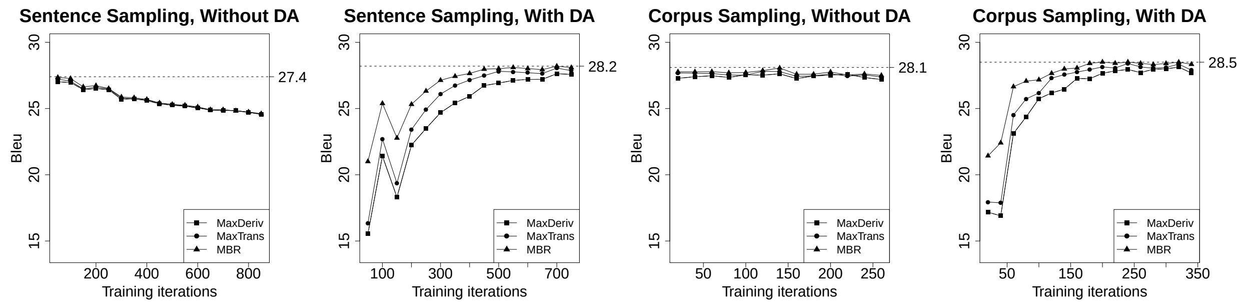


Figure 2: Heldout performance for German-English training averaged across 5 minimum risk training runs. Best scores achieved are indicated by dotted line.

sentences with DA. The size of the batches corresponds to the number of sentences that form a corpus sample. For German/French to English experiments, we used batches of 100 random sentences for training with and without DA. We perform 10 optimizations at each temperature setting during deterministic annealing. Test time conditions are identical to the sentence sampling ones and we measure performance on a held-out set after every 20 iterations of the learner.

5.5 Results

Figures 2 and 3 show the scores on the German-English and Arabic-English held-out sets respectively comparing all four training regimes: corpus vs sentence sampling, DA vs without DA. Results for French-English training are similar.

We focus our analysis on the Arabic-English experimental setup. Without deterministic annealing, the learner converges quickly, usually after just 20 iterations, after which performance degrades steadily. The magnitudes of the weights are large, sharpening the distribution. There is not much diversity amongst the sampled derivations, i.e. the entropy of the sample set is low. Therefore, all 3 decoding regimes give very similar results. With the addition of the entropic prior, the model is slow to converge before the so-called *phase transition* occurs (usually after around 50 iterations), after which performance goes up to reach a peak (45.2 BLEU) higher than that without the prior (44.2 BLEU), before steadily declining. The entropic prior encourages diversity among the sample set, especially at high temperature settings.

In the presence of diversity, the benefits of marginalization over derivations is clear: MaxTrans does better than MaxDeriv and MBR does best, confirm recent findings of (Blunsom et al., 2008; Arun et al., 2009) that MaxTrans improves over MaxDeriv decoding for models trained to account for multiple derivations. As the temperature decreases to zero, the model sharpens, effectively intent on maximizing one-best performance and thus voiding the benefits of MaxTrans and MBR. Figures 2 and 3 also show that corpus sampling improves over sentence sampling, although not by much (+ 0.3 BLEU).

5.6 Comparison with MERT baseline

Having established the superiority of the pipeline of expected corpus BLEU training with DA followed by MBR decoding over other alternatives considered, we compare it to the best results obtained with MERT optimized Moses (bold scores from Table 1). To account for sampler variance during both training and decoding, we average scores across 50 runs; 10 decoding runs each using the best weight set from 5 training runs. Results

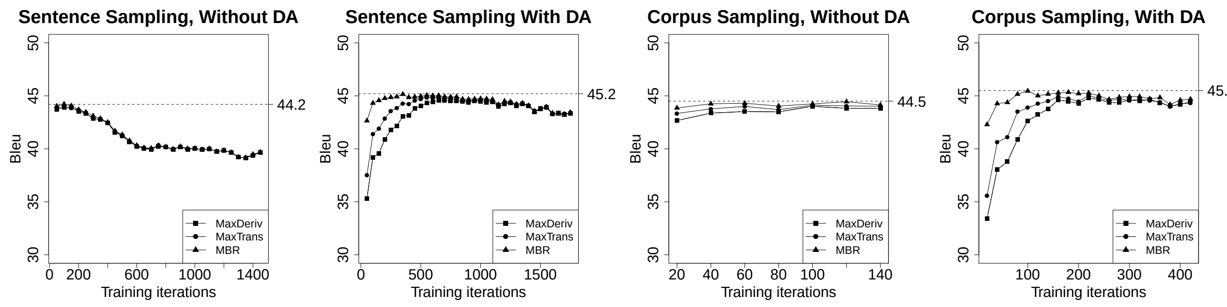


Figure 3: Heldout performance for Arabic-English training averaged across 5 minimum risk training runs. Best scores achieved are indicated by dotted line.

are shown in Table 2.[7]

We observe that on 3 out of 5 datasets, the sampler results are much more stable than MERT and as stable on the other 2 datasets. We attribute the improved stability to the more powerful optimization algorithm used by the sampler which uses gradient information to steer the model towards better weights. MERT, alternatively, optimizes one feature at a time using line search and therefore does not explore the full feature space as thoroughly.

Translation results with the sampler are better than with MERT on 2 datasets, are equal on another 2 and worse in one case. The improvements withe the sampler are obtained in the case of out-of-domain data suggesting that the minimum risk training objective generalizes better than the 1-best objective of MERT.

	MERT/Moses		Sampler	
Test set	Best	σ	MBR	σ
AR-EN MT05	**44.5** (lMBR)	0.12	**44.5**	0.14
FR-EN In	**33.4** (nMBR)	0.12	33.2	0.06
FR-EN Out	19.5 (nMBR)	0.12	**19.8**	0.05
DE-EN In	**27.8** (Viterbi)	0.10	**27.8**	0.11
DE-EN Out	16.0 (lMBR)	0.30	**16.6**	0.12

Table 2: *Final* results comparing MERT/Moses pipeline with unified sampler pipeline. Sampler uses corpus sampling during training and MBR decoding at test time. Moses results are averaged across decoding runs using weights from 10 MERT runs and sampler results are averaged across 10 decoding runs for each of 5 different training runs. We report BLEU scores and standard deviation (σ).

[7]The MBR decoding times, averaged over 10 decoding runs of 50 sentences each, are 10 secs/sent for Moses nbest MBR, 40 secs/sent for Moses lattice MBR and 180 secs/sent for the sampler.

	Viterbi	nMBR	lMBR	Sampler MBR
AR-EN MT05	44.2	44.4	**44.8**	**44.8**
FR-EN In	33.1	33.2	**33.3**	**33.3**
FR-EN Out	19.6	19.8	**19.9**	**19.9**
DE-EN In	27.7	27.9	**28.0**	**28.0**
DE-EN Out	16.0	16.3	**16.6**	**16.6**

Table 3: Comparison of decoding methods using expected BLEU trained weights. We report Viterbi, nbest MBR (nMBR) and lattice MBR (lMBR) decoding scores vs *best* sampler MBR decoding performance. We selected the best weight set based on performance on heldout data.

5.7 Moses with expected BLEU weights

In a final set of experiments, we reran the Moses decoder this time using weights obtained through expected BLEU optimization. Here, for each language pair, we picked the weight set that gave the best results on held-out data. Note that the results which we show in Table 3 are over one run only, so are not strictly comparable to those in Table 2 which are averaged over several training and decoding runs. We also report the best results obtained with the sampler MBR decoder using these weights.

In contrast to Table 1, here we see a consistent improvement across all test-sets when going from Viterbi decoding to n-best then to lattice MBR. Except for in-domain French-English, the translation results are superior to the best scores shown (in bold) in Table 1, confirming that the minimum risk training objective is able to find good weight sets. Interestingly, we also observe that sampler MBR gets the same exact results for all test sets as lattice MBR.

6 Discussion

We have shown that the sampler of Arun et al. (2009) can be used to perform minimum risk training over an unpruned search space. Our proposed corpus sampling technique, like MERT, is able to optimize corpus BLEU directly whereas alternate parameter estimation techniques usually employed in SMT optimize *approximations* of BLEU. Chiang et al. (2008b) accounts for the online nature of the MIRA optimization algorithm by smoothing the sentence-level BLEU precision counts of a translation with a weighted average of the precision counts of previously decoded sentences, thus approximating corpus BLEU. As for minimum risk training, prior implementations have either used sentence-level BLEU (Zens et al., 2007) or a linear approximation to BLEU (Smith and Eisner, 2006; Li and Eisner, 2009).

At test time, the sampler works best as an MBR decoder, but also allows us to verify past claims about the benefits of marginalizing over alignments during decoding. We compare the sampler MBR decoder's performance against MERT-optimized Moses run under three different decoding regimes, finding that the sampler does as well or better on 4 out of 5 datasets.

Our training and testing pipeline has the advantage of being able to handle a large number of both local and global features so we expect in the future to outperform the standard MERT and dynamic programming-based search pipeline further.

As shown in Section 5.2, lattice MBR in some cases leads to a marked drop in performance. (Kumar et al., 2009) mention that the linear approximation to BLEU used in their lattice MBR algorithm is not guaranteed to match corpus BLEU, especially on unseen test sets. To account for these cases, they allow their algorithm to *back-off to the MAP* solution. One possible reason for the drop in performance in our lattice MBR experiments is that the implementation we use does not employ this back-off strategy.

Table 3 provides valuable insights as to the merits of the lattice MBR approach versus our own sampling based pipeline. Firstly, whereas with MERT optimized weights, the benefits of lattice MBR are debatable (Table 1), running Moses with minimum risk trained weights gives results that are in line with what we would expect - lattice MBR does systematically better than competing decoding algorithms. This suggests that the unbiased minimum risk training criterion used by the sampler is a better fit for lattice MBR than the MERT criterion, and also that the mismatch between linear and corpus BLEU mentioned before might not be the reason for the results in Table 1.

Secondly, we find that sampling MBR matches lattice MBR on the minimum risk trained weights. The MBR sampler uses samples drawn from the distribution as hypothesis and evidence sets, typically 1000 samples for the former and 10000 samples for the latter. In the lattice MBR experiments of Tromble et al. (2008), it is shown that this size of hypothesis set is sufficient. Their evidence set, however, is significantly larger than ours.[8]Table 3 suggests that, since it is not biased by heuristic pruning, the sampler's limited evidence set is enough to give a good estimate of the probability distribution whereas beam-search based MBR needs to scale from using n-best lists to lattices to get equivalent results.

Sampling the phrase-based model is expensive, meaning that lattice MBR is still faster (around 4x) to run than sampler MBR. However, due to the *unified* nature of the training and decoding criterion in our approach, the minimum risk trained weights can be plugged *directly* into the sampler MBR decoder, whereas lattice MBR requires an additional expensive step of tuning the model hyper-parameters (Kumar et al., 2009).

In future work, we also intend to look at more efficient ways of generating samples. One possibility is to interleave Gibbs sampling steps using low order ngram language model distributions with Metropolis-Hasting steps that use higher order language model distributions.

7 Related Work

Expected BLEU training for phrase-based models has been successfully attempted by (Smith and Eisner, 2006; Zens et al., 2007), however they both used biased n-best lists to approximate the posterior distribution. Li and Eisner (2009) present work on performing expected BLEU training with deterministic annealing on translation forests generated by Hiero (Chiang, 2007). Since BLEU does not factorize over the search graph, they use the linear approximation of Tromble et al. (2008) instead.

Pauls et al. (2009) present an alternate training criterion over translation forests called CoBLEU,

[8]up to 10^{81} as per Tromble et al. (2008)

similar in spirit to expected BLEU training, but aimed to maximize the *expected counts* of n-grams appearing in reference translations. This training criterion is used in conjunction with consensus decoding (DeNero et al., 2009), a linear-time approximation of MBR.

In contrast to the approaches above, the algorithms presented in this paper are able to explore an *unpruned* search space. By using corpus sampling, we can perform minimum risk training with corpus BLEU rather than any approximations of this metric. Also, since we maintain a probabilistic formulation across training and decoding, our approach does not require a grid-search for a scaling factor as in Tromble et al. (2008).

8 Conclusions

We have presented a unified approach to the task of parameter estimation and decoding for a phrase-based system using the standard translation evaluation metric, BLEU. Using a Gibbs sampler to explore the entire probability distribution allows us to implement two probabilistic sound algorithms, minimum risk training and its equivalent, MBR decoding, in an unbiased way. The probabilistic formulation also allows us to use gradient based optimization techniques which produce stable model parameters. At decoding time, we show the benefits of marginalizing over derivations and that MBR gives better results than other decoding criteria.

Since our optimization algorithm can cope with a large number of features, in future work, we plan to incorporate more expressive features in the model. We use a Gibbs sampler for inference so there is scope for exploring non-local features which might not easily be added to dynamic programming based models.

Acknowledgments

This research was supported in part by the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-2-001; and by the EuroMatrix project funded by the European Commission (6th Framework Programme). The project made use of the resources provided by the Edinburgh Compute and Data Facility (`http://www.ecdf.ed.ac.uk/`). The ECDF is partially supported by the eDIKT initiative (`http://www.edikt.org.uk/`).

References

Abhishek Arun, Chris Dyer, Barry Haddow, Phil Blunsom, Adam Lopez, and Philipp Koehn. 2009. Monte carlo inference and maximization for phrase-based translation. In *Proceedings of CoNLL*, pages 102–110.

Phil Blunsom and Miles Osborne. 2008. Probabilistic inference for machine translation. In *Proc. of EMNLP 2008*.

Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proc. of ACL-HLT*.

Alexandre Bouchard-Côté, Slav Petrov, and Dan Klein. 2009. Randomized pruning: Efficiently calculating expectations in large dynamic programs. In *Advances in Neural Information Processing Systems 22*, pages 144–152.

Chris Callison-Burch, Philipp Koehn, Christoph Monz, and Josh Schroeder, editors. 2009. *Proc. of Workshop on Machine Translations*.

David Chiang, Steve DeNeefe, Yee Seng Chan, and Hwee Tou Ng. 2008a. Decomposability of translation metrics for improved evaluation and efficient algorithms. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 610–619, Honolulu, Hawaii, October. Association for Computational Linguistics.

David Chiang, Yuval Marton, and Philip Resnik. 2008b. Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Honolulu, Hawaii, October. Association for Computational Linguistics.

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

John DeNero, David Chiang, and Kevin Knight. 2009. Fast consensus decoding over translation forests. In *Proceedings of ACL/AFNLP*, pages 567–575.

Stuart Geman and Donald Geman. 1984. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741.

J.H. Johnson, J. Martin, G. Foster, and R. Kuhn. 2007. Improving translation quality by discarding most of the phrasetable. In *Proc. of EMNLP-CoNLL*, Prague.

P. Koehn, F.J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. of HLT-NAACL*, pages 48–54, Morristown, NJ, USA.

P. Koehn, H. Hoang, A. Birch Mayne, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL Demos*, pages 177–180.

S. Kumar and W. Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *Processings of HLT-NAACL*.

Shankar Kumar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of ACL/AFNLP*, pages 163–171.

Zhifei Li and Jason Eisner. 2009. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Proceedings of EMNLP*, pages 40–51.

Zhifei Li, Jason Eisner, and Sanjeev Khudanpur. 2009. Variational decoding for statistical machine translation. In *Proceedings of ACL/AFNLP*, pages 593–601.

F. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*, pages 160–167.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Adam Pauls, John Denero, and Dan Klein. 2009. Consensus training for consensus decoding in machine translation. In *Proceedings of EMNLP*, pages 1418–1427.

Kenneth Rose. 1998. Deterministic annealing for clustering, compression, classification, regression, and related optimization problems. In *Proceedings of the IEEE*, pages 2210–2239.

Nicol N. Schraudolph. 1999. Local gain adaptation in stochastic gradient descent. Technical Report IDSIA-09-99, IDSIA.

David A. Smith and Jason Eisner. 2006. Minimum risk annealing for training log-linear models. In *Proceedings of COLING-ACL*, pages 787–794.

Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of EMNLP*, pages 620–629.

Richard Zens, Sasa Hasan, and Hermann Ney. 2007. A systematic comparison of training criteria for statistical machine translation. In *Proceedings of EMNLP*, pages 524–532.