Probabilistic Inference for Machine Translation

Phil Blunsom and **Miles Osborne**
School of Informatics, University of Edinburgh
10 Crichton Street, Edinburgh, EH8 9AB, UK
{pblunsom,miles}@inf.ed.ac.uk

Abstract

We advance the state-of-the-art for discriminatively trained machine translation systems by presenting novel probabilistic inference and search methods for synchronous grammars. By approximating the intractable space of all candidate translations produced by intersecting an ngram language model with a synchronous grammar, we are able to train and decode models incorporating millions of sparse, heterogeneous features. Further, we demonstrate the power of the discriminative training paradigm by extracting structured syntactic features, and achieving increases in translation performance.

1 Introduction

The goal of creating statistical machine translation (SMT) systems incorporating rich, sparse, features over syntax and morphology has consumed much recent research attention. Discriminative approaches are widely seen as a promising technique, potentially allowing us to further the state-of-the-art. Most work on discriminative training for SMT has focussed on linear models, often with margin based algorithms (Liang et al., 2006; Watanabe et al., 2006), or rescaling a product of sub-models (Och, 2003; Ittycheriah and Roukos, 2007).

Recent work by Blunsom et al. (2008) has shown how translation can be framed as a probabilistic log-linear model, where the distribution over translations is modelled in terms of a latent variable on derivations. Their approach was globally optimised and discriminative trained. However, a language model, an information source known to be crucial for obtaining good performance in SMT, was notably omitted. This was because adding a language model would mean that the normalising partition function could no longer be exactly calculated, thereby preventing efficient parameter estimation.

Here, we show how language models can be incorporated into large-scale discriminative translation models, without losing the probabilistic interpretation of the model. The key insight is that we can use Monte-Carlo methods to approximate the partition function, thereby allowing us to tackle the extra computational burden associated with adding the language model. This approach is theoretically justified and means that the model continues to be both probabilistic and globally optimised. As expected, using a language model dramatically increases translation performance.

Our second major contribution is an exploitation of syntactic features. By encoding source syntax as features allows the model to use, or ignore, this information as it sees fit, thereby avoiding the problems of coverage and sparsity associated with directly incorporating the syntax into the grammar (Huang et al., 2006; Mi et al., 2008). We report on translation gains using this approach.

We begin by introducing the synchronous grammar approach to SMT in Section 2. In Section 3 we define the parametric form of our model and describe techniques for approximating the intractable space of all translations for a given source sentence. In Section 4 we evaluate the ability of our model to effectively estimate the highly dependent weights for the sparse features and real-valued language model. In addition we describe how

$S \Rightarrow \langle X_1 \text{ 的 。 } , X_1 . \rangle$

$X_1 \Rightarrow \langle X_2 \text{ 是 昨天 深夜 } X_3 , X_2 \ X_3 \text{ late last night} \rangle$

$X_2 \Rightarrow \langle \text{布朗 , Brown} \rangle$

$X_3 \Rightarrow \langle \text{从 } X_4 \text{ 抵达 } X_5 , \text{ arrived in } X_5 \text{ from } X_4 \rangle$

$X_4 \Rightarrow \langle \text{上海 , Shanghai} \rangle$

$X_5 \Rightarrow \langle \text{北京 , Beijing} \rangle$

Figure 1. An example SCFG derivation from a Chinese source sentence which yields the English sentence: *"Brown arrived in Shanghai from Beijing late last night."*

our model can easily integrate rich features over source syntax trees and compare our training methods to a state-of-the-art benchmark.

2 Synchronous context free grammar

A synchronous context free grammar (SCFG, (Lewis II and Stearns, 1968)) describes the generation of pairs of strings. A string pair is generated by applying a series of paired context-free rewrite rules of the form, $X \rightarrow \langle \alpha, \gamma, \sim \rangle$, where X is a non-terminal, α and γ are strings of terminals and non-terminals and $\sim$ specifies a one-to-one alignment between non-terminals in α and γ. In the context of SMT, by assigning the source and target languages to the respective sides of a SCFG it is possible to describe translation as the process of parsing the source sentence, while generating the target translation (Chiang, 2007).

In this paper we only consider grammars extracted using the heuristics described for the Hiero SMT system (Chiang, 2007). Note however that our approach is general and could be used with other synchronous grammar transducers (e.g., (Galley et al., 2006)). SCFG productions can specify that the order of the child non-terminals is the same in both languages (a *monotone* production), or is reversed (a *reordering* production). Without loss of generality, here we add the restriction that non-terminals on the source and target sides of the grammar must have the same category. Figure 1 shows an example derivation for Chinese to English translation.

3 Model

We start by defining a log-linear model for the conditional probability distribution over target translations of a given source sentence. A sequence of SCFG rule applications which produce a translation from a source sentence is referred to as a *derivation*, and each translation may be produced by many different derivations. As the training data only provides source and target sentences, the derivations are modelled as a latent variable.

The conditional probability of a derivation, $\mathbf{d}$, for a target translation, $\mathbf{e}$, conditioned on the source, $\mathbf{f}$, is given by:

$$p_\Lambda(\mathbf{d}, \mathbf{e}|\mathbf{f}) = \frac{\exp \sum_k \lambda_k H_k(\mathbf{d}, \mathbf{e}, \mathbf{f})}{Z_\Lambda(\mathbf{f})} \quad (1)$$

$$\text{where} \quad H_k(\mathbf{d}, \mathbf{e}, \mathbf{f}) = \sum_{r \in \mathbf{d}} h_k(\mathbf{f}, r, q(r, \mathbf{d})) \quad (2)$$

Using Equation (1), the conditional probability of a target translation given the source is the sum over all of its derivations:

$$p_\Lambda(\mathbf{e}|\mathbf{f}) = \sum_{\mathbf{d} \in \Delta(\mathbf{e}, \mathbf{f})} p_\Lambda(\mathbf{d}, \mathbf{e}|\mathbf{f})$$

where $\Delta(\mathbf{e}, \mathbf{f})$ is the set of all derivations of the target sentence $\mathbf{e}$ from the source $\mathbf{f}$.

Here k ranges over the model's features, and $\Lambda = \{\lambda_k\}$ are the model parameters (weights for their corresponding features). The function $q(r, \mathbf{d})$ returns the target ngram context, for a language model with order m, of rule r in derivation $\mathbf{d}$. For a rule which spans the target words (i, j) and $target\_yield(\mathbf{d}) = \{t_0, \cdots, t_l\}$:

$$q(r, \mathbf{d}) = \begin{cases} t_i \cdots t_{i+m-2} \star t_{j-m+2} \cdots t_j & \text{if } j - i > m \\ t_i \cdots t_j & \text{otherwise} \end{cases}$$

The feature functions h_k are real-valued functions over the source and target sentences, and can include overlapping and non-independent features of the data. The features must decompose with the derivation and the ngram context defined by the function q, as shown in Equation (2). The features can reference the entire source sentence coupled with each rule, r, and its target context, in a derivation.

By directly incorporating the language model context q into the model formulation, we will not

be able to exactly compute the partition function $Z_\Lambda(\mathbf{f})$, which sums over all possible derivations. Even though a dynamic program over this space would still run in polynomial time, as shown by Chiang (2007), a packed chart representation of the partition function for the binary Hiero grammars used in this work would require $\mathcal{O}(n^3|T|^{4(m-1)})$ space,[1] which is far too large to be practical.

Instead we approximate the partition function using a sum over a large subset of the possible derivations ($\Delta(\mathbf{e}, \mathbf{f})$):

$$Z_\Lambda(\mathbf{f}) \approx \sum_{\mathbf{e}} \sum_{\mathbf{d} \in \{\subset \Delta(\mathbf{e},\mathbf{f})\}} \exp \sum_k \lambda_k H_k(\mathbf{d}, \mathbf{e}, \mathbf{f}) = \tilde{Z}_\Lambda(\mathbf{f})$$

This model formulation raises the questions of what an appropriate *large subset* of derivations for training is, and how to efficiently calculate the sum over all derivations in decoding. In the following sections we elucidate and evaluate our solutions to these problems.

3.1 Sampling Derivations

The training and decoding algorithms presented in the following sections rely upon Monte-Carlo techniques, which in turn require the ability to draw derivation samples from the probability distribution defined by our log-linear model. Here we adapt previously presented algorithms for sampling from a PCFG (Goodman, 1998) for use with our synchronous grammar model. Algorithm 1 describes the algorithm for sampling derivations. The sampling algorithm assumes the pre-existance of a packed chart representation of all derivations for a given source sentence. The *inside algorithm* is then used to calculate the scores needed to define a multinomial distribution over all partial derivations associated with expanding a given child rule. These initial steps are performed once and then an unlimited number of samples can be drawn by calling the recursive SAMPLE procedure. MULTI draws a sample from the distribution over rules for a given chart cell, CHILDREN enumerates the chart cells connected to a rule as variables, and DERIVATION is a recursive tree data structure for derivations. The algorithm is first called on a category and chart cell spanning the entire chart, and then proceeds top down by using the function MULTI to draw the next rule to expand from the distribution defined by the inside scores.

[1] where $|T|$ is the size of the terminal alphabet, i.e. the number of unique English words.

Algorithm 1 Top-down recursive derivation sampling algorithm.

```
1: procedure SAMPLE(X, i, k)
2:     rule ← MULTI(inside_chart(X, i, k))
3:     c = φ
4:     for (child_category, x, y) ∈ CHILDREN(rule) do
5:         c ← c ∪ SAMPLE(child_category, x, y)
6:     end for
7:     return DERIVATION(X, children)
8: end procedure
```

3.2 Approximate Inference

Approximating the partition function with $\tilde{Z}_\Lambda(\mathbf{f})$ could introduce biases into inference and in the following discussion we describe measures taken to minimise the effects of the approximation bias.

An obvious approach to approximating the partition function, and the feature expectations required for calculating the derivative in training, is to use the packed chart of derivations produced by running the cube pruning beam search algorithm of Chiang (2007) on the source sentence. In this case $\tilde{Z}_\Lambda(\mathbf{f})$ includes all the derivations that fall within the cube pruning beam, hopefully representing the majority of the probability mass. We denote the partition function estimated with this cube beam approximation as $\tilde{Z}^{cb}_\Lambda(\mathbf{f})$. This approach has the advantage of using the same beam search dynamic program during training as is used for decoding. As the approximated partition function does not contain all derivations, it is possible that some, or all, of the derivations of the reference translation from the parallel corpus may be excluded. We must therefore intersect the packed chart built from the cube beam with that of the reference derivations to ensure consistency.

Although, as would be done using cube-pruning, it would seem intuitively sensible to approximate the partition function using only high probability derivations, it is possible that doing so will bias our model in odd ways. The space of derivations contained within the beam will be tightly clustered about a maximum, and thus a model trained with such an approximation will only see a very small

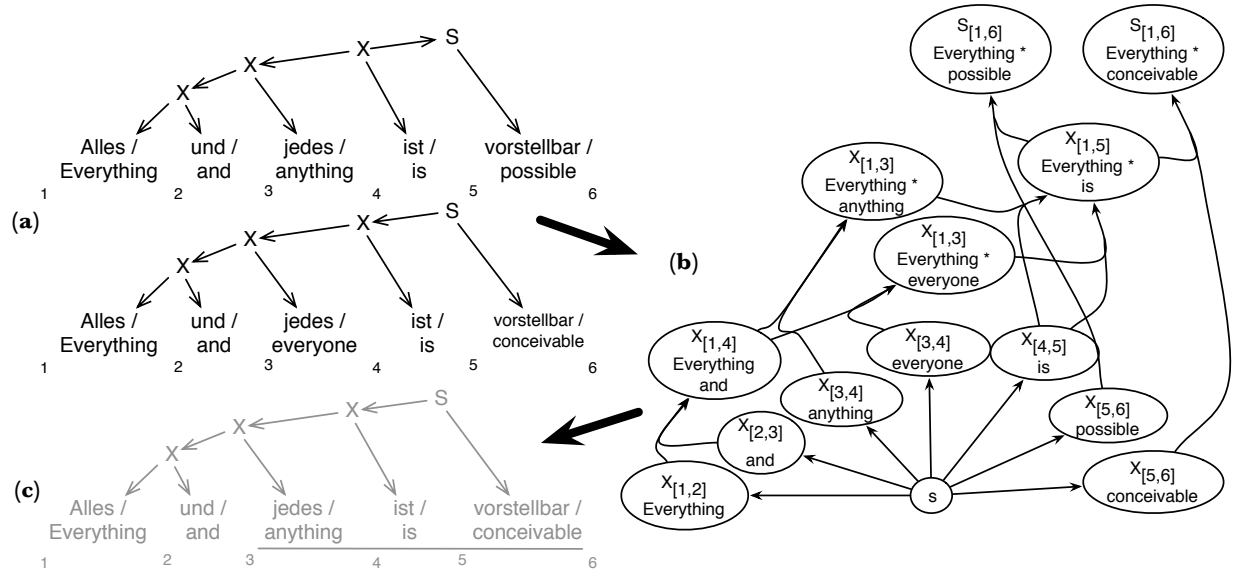

Figure 2. A German-English translation example of building $\tilde{Z}_{\Lambda}^{sam}(\mathbf{f})$ from samples. (a) Two sample derivations are drawn from the model, (b) these samples are then combined into a packed representation, here represented by a hypergraph with target translations elided for a bigram language model. The derivation in (c) is contained within the hypergraph even though it was never explicitly inserted.

part of the overall distribution, possibly leading it astray. Consider the example of a language model feature: as this is a very strong indicator of translation quality, we would expect all derivations within the beam to have a similar (high) language model score, thereby robbing this feature of its discriminating power. However if our model could also see the low probability derivations it would be clear that this feature is indeed very strongly correlated with good translations. Thus a good approximation of the space of derivations is one that includes both good and bad examples, not just a cluster around the maximum.

A principled solution to the problem of approximating the partition function would be to use a Markov Chain Monte Carlo (MCMC) sampler to estimate the sum with a large number of samples. Most of the sampled derivations would be in the high probability region of the distribution, however there would also be a number of samples drawn from the rest of the space, giving the model a more global view of the distribution, avoiding the pitfalls of the narrow view obtained by a beam search. Although effective, the computational cost of such an approach is prohibitive as we would need to draw hundreds of thousands of samples to obtain convergence, for every training iteration.

Here we mediate between the computational advantages of a beam and the broad view of the distribution provided by sampling. Using the algorithm outlined in Section 3.1 we draw samples from the distribution of derivations and then insert these samples into a packed chart representation. This process is illustrated in Figure 2. The packed chart created by intersecting the sample derivations represents a space of derivations much greater than the original samples. In Figure 2 the chart is built from the first two sampled derivations, while the third derivation can be extracted from the chart even though it was never explicitly entered. This approximation of the partition function (denoted $\tilde{Z}_{\Lambda}^{sam}(\mathbf{f})$) allows us to build an efficient packed chart representation of a large number of derivations, centred on those with high probability while still including a significant representation of the low probability space. Derivations corresponding to the reference can be detected during sampling and thus we can build the chart for the reference derivations at the same time as the one approximating the partition function. This could lead to some, or none of, the possible reference derivations being included, as they may not have been sampled. Although we could intersect all of the reference derivations with the sampled chart, this could distort the distribution over derivations,

and we believe it to be advantageous to keep the distributions between the partition function and reference charts consistent.

Both of the approximations proposed above, $\tilde{Z}^{cb}_{\Lambda}(\mathbf{f})$ and $\tilde{Z}^{sam}_{\Lambda}(\mathbf{f})$, rely on the pre-existence of a trained translation model in order to either guide the cube-pruning beam, or define the probability distribution from which we draw samples. We solve this chicken and egg problem by first training an exact translation model *without* a language model, and then use this model to create the partition function approximations for training *with* a language model. We denote the distribution without a language model as $p^{-LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$ and that with as $p^{+LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$.

A final training problem that we need to address is the appropriate initialisation of the model parameters. In theory we could simply randomly initialise Λ for $p^{+LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$, however in practice we found that this resulted in poor performance on the development data. This is due to the complex non-convex optimisation function, and the fact that many features will fall outside the approximated charts resulting in random, or zero, weights in testing. We introduce a novel solution in which we use the Gaussian prior over model weights to tie the exact model trained without a language model, which assigns sensible values to all rule features, with the approximated model. The prior over model parameters for $p^{+LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$ is defined as:

$$p^{+LM}(\lambda_k) \propto e^{-\frac{\|\lambda_k - \lambda_k^{-LM}\|^2}{2\sigma^2}}$$

Here we have set the mean parameters of the Gaussian distribution for the approximated model to those learnt for the exact one. This has the effect that any features that fall outside the approximated model will simply retain the weight assigned by the exact model. While for other feature weights the prior will penalise substantial deviations away from Λ^{-LM}, essentially encoding the intuition that the rule rule parameters should not change substantially with the inclusion of language model features.

This results in the following log-likelihood objective and corresponding gradient:

$$\mathcal{L} = \sum_{(\mathbf{e}_i,\mathbf{f}_i)\in\mathcal{D}} \log p^{+LM}_{\Lambda}(\mathbf{e}_i|\mathbf{f}_i) + \sum_k \log p^{+LM}_0(\lambda_k)$$

$$\frac{\partial\mathcal{L}}{\partial\lambda_k} = E_{p^{+LM}_{\Lambda}(\mathbf{d}|\mathbf{e}_i,\mathbf{f}_i)}[h_k] - E_{p^{+LM}_{\Lambda}(\mathbf{e}|\mathbf{f}_i)}[h_k] - \frac{\lambda_k^{+LM} - \lambda_k^{-LM}}{\sigma^2}$$

3.3 Decoding

As stated in Equation 3 the probability of a given translation string is calculated as the sum of the probabilities of all the derivations that yield that string. In decoding, where the reference translation is not known, the exact calculation of this summation is NP-Hard. This problem also arises in monolingual parsing with probabilistic tree substitution grammars and has been tackled in the literature using Monte-Carlo sampling methods (Chappelier and Rajman, 2000). Their approach is directly applicable to our SCFG decoding problem and we can use Algorithm 1 to draw sample translation derivations for the source sentence. The probability of a translation can be calculated simply from the number of times a derivation that yields it was sampled, divided by the total number of samples. For the $p^{-LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$ model we can build the full chart of all possible derivations and thus sample from the true distribution over derivations. For the $p^{+LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$ model we suffer the same problem as in training and cannot build the full chart. Instead a chart is built using the cube-pruning algorithm with a wide beam and we then draw samples from this chart. Although sampling from a reduced chart will result in biased samples, in Section 4 we show this approach to be effective in practice.[2] In Section 4 we compare our sampling approach to the heuristic beam search proposed by Blunsom et al. (2008).

It is of interest to compare our proposed decoding algorithms to minimum Bayes risk (MBR) decoding (Kumar and Byrne, 2004), a commonly used decoding method. From a theoretical standpoint, the summing of derivations for a given translation is exactly

[2] We have experimented with using a Metropolis Hastings sampler, with $p^{-LM}_{\Lambda}(\mathbf{e}|\mathbf{f})$ as the proposal distribution, to sample from the true distribution with the language model. Unfortunately the sample rejection rate was very high such that this method proved infeasibly slow.

equivalent to performing MBR with a 0/1 loss function over derivations. From a practical perspective, MBR is normally performed with B_{LEU} as the loss and approximated using n-best lists. These n-best lists are produced using algorithms tuned to remove multiple derivations of the same translation (which have previously been seen as undesirable). However, it would be simple to extend our sampling based decoding algorithm to calculate the MBR estimate using B_{LEU}, in theory providing a lower variance estimate than attained with n-best lists.

4 Evaluation

We evaluate our model on the IWSLT 2005 Chinese to English translation task (Eck and Hori, 2005), using the 2004 test set as development data for tuning the hyperparameters and MERT training the benchmark systems. The statistics for this data are presented in Table 1.[3] The training data made available for this task consisted of 40k pairs of transcribed utterances, drawn from the travel domain. The development and test data for this task are somewhat unusual in that each sentence has a single human translated reference, and fifteen paraphrases of this reference, provided by monolingual annotators. Model performance is evaluated using the standard B_{LEU} metric (Papineni et al., 2002) which measures average n-gram precision, $n \leq 4$, and we use the NIST definition of the brevity penalty for multiple reference test sets. We provide evaluation against both the entire multi-reference sets, and the single human translation.

Our translation grammar is induced using the standard alignment and rule extraction heuristics used in hierarchical translation models (Chiang, 2007).[4] As these heuristics aren't based on a generative model, and don't guarantee that the target translation will be reachable from the source, we discard those sentence pairs for which we cannot produce a derivation, leaving 38,405 sentences for training.

Our base model contains a single feature for each rule which counts the number of times it appeared in a particular derivation. For models which include a language model, we train a standard Kneser-Ney trigram model on the target side of the training corpus. We also include a word penalty feature to compensate for the shortening effect of the language model. In total our model contains 2.9M features.

The aims of our evaluation are: (1) to determine that our proposed training regimes are able to realise performance increase when training sparse rule features and a real valued language model feature together, (2) that the model is able to effectively use rich features over the source sentence, and (3) to confirm that our model obtains performance competitive with the current state-of-the-art.

4.1 Inference and Decoding

We have described a number of modelling choices which aim to compensate for the training biases introduced by incorporating a language model feature through approximate inference. Our a priori knowledge from other SMT systems suggests that incorporating a language model should lead to large increases in B_{LEU} score. In this evaluation we aim to determine whether our training regimes are able to realises these expected gains.

Table 2 shows a matrix of development B_{LEU} scores achieved by varying the approximation of the partition function in training, and varying the decoding algorithm. If we consider the vertical axis we can see that the sampling method for approximating the partition function has a small but consistent advantage over using the cube-pruning beam. The charts produced by the sampling approach occupy roughly half the disc space as those produced by the beam search, so in subsequent experiments we present results using the $\tilde{Z}^{sam}_{\Lambda}(\mathbf{f})$ approximation.

Comparing the decoding algorithms on the horizontal axis we can reconfirm the findings of Blunsom et al. (2008) that the max-translation decoding outperforms the Viterbi max-derivation approximation. It is also of note that this B_{LEU} increase is robust to the introduction of the language model feature, assuaging fears that the max-translation approach may have been doing the job of the language model. We also compare using Monte-Carlo sampling for decoding with the previously proposed heuristic beam search algorithm. The difference between the two algorithms is small, however

[3] Development and test set statistics are for the single human translation reference.

[4] With the exception that we allow unaligned words at the boundary of rules. This improves training set coverage.

	Training		Development		Test	
	Chinese	English	Chinese	English	Chinese	English
Utterances	38405		500		506	
Segments/Words	317621	353116	3464	3752	3784	3823
Av. Utterances Length	8	9	6	7	7	7
Longest Utterance	55	68	58	62	61	56

Table 1. IWSLT Chinese to English translation corpus statistics.

Model	Max-derivation	Max-translation(Beam)	Max-translation(Sampling)
$p_\Lambda^{-LM}(\mathbf{e}\|\mathbf{f})$	31.0	32.5	32.6
$p_\Lambda^{+LM}(\mathbf{e}\|\mathbf{f})$ $(\tilde{Z}_\Lambda^{cb}(\mathbf{f}))$	39.1	39.8	39.8
$p_\Lambda^{+LM}(\mathbf{e}\|\mathbf{f})$ $(\tilde{Z}_\Lambda^{sam}(\mathbf{f}))$	39.9	40.5	40.6

Table 2. Development set results for varying the approximation of the partition function in training, $\tilde{Z}_\Lambda^{cb}(\mathbf{f})$ vs. $\tilde{Z}_\Lambda^{sam}(\mathbf{f})$, and decoding using the Viterbi max-derivation algorithm, or the max-translation algorithm with either a beam approximation or Monte-Carlo sampling.

we feeling the sampling approach is more theoretically justified and adopt it for our later experiments.

The most important result from this evaluation is that both our training regimes realise substantial gains from the introduction of the language model feature. Thus we can be confident that our model is capable of modelling the distribution over translations even when the space over all derivations is intractable to dynamically program exactly.

4.2 A Discriminative Syntactic Translation Model

In the previous sections we've described and evaluated a statistical model of translation that is able to estimate a probability distribution over translations using millions of sparse features. A prime motivation for such a model is the ability to define complex features over more than just the surface forms of the source and target strings. There are limitless options for such features, and previous work has focused on defining token based features such as part-of-speech and morphology (Ittycheriah and Roukos, 2007). Although such features are applicable to our model, here we attempt to test the model's ability to incorporate complex features over source-side syntax trees, essentially subsuming and extending previous work on tree-to-string translation models (Huang et al., 2006; Mi et al., 2008).

We first parse the source side of our training, development and test corpora using the Stanford parser.[5] Next, while building the synchronous charts required for training, whenever a rule is used in a derivation a feature is activated which captures: (1) the constituent spanning the rule's source side in the syntax tree (if any) (2) constituents spanning any variables in the rule, and (3) the rule's target side surface form. Figure 3 illustrates this process.

These syntactic features are equivalent to the grammar rules extracted for tree-to-string translation systems. The key difference in our model is that the source syntax tree is treated as conditioning context and it's information encoded as features, thus this information can be used or ignored as the model sees fit. This avoids the problems associated with explicitly encoding the source syntax in the grammar, such as sparsity and overly constraining the model. In addition we could easily incorporate features over multiple source trees, for example mixing labelled syntax trees with dependency graphs.

We limit the extraction of syntactic features to those that appear in at least two training derivations, giving a total of 4.2M syntactic features, for an overall total of 7.1M features.

4.3 Discussion

Table 3 shows the results from applying our described models to the test set. We benchmark our results against a model (Hiero) which was directly trained to optimise $BLEU^{NIST}$ using the standard MERT algorithm (Och, 2003) and the full set of translation and lexical weight features described for the Hiero model (Chiang, 2007). As well as

[5] http://nlp.stanford.edu/software/lex-parser.shtml

Model	$BLEU^{NIST}$	$BLEU^{IBM}$	$BLEU^{HumanRef}$
$p_\Lambda^{-LM}(\mathbf{e}\|\mathbf{f})$	33.5	35.2	25.2
$p_\Lambda^{+LM}(\mathbf{e}\|\mathbf{f})$	44.6	44.6	31.2
$p_\Lambda^{+LM}(\mathbf{e}\|\mathbf{f})$ + syntax	45.3	45.2	31.8
MERT ($BLEU^{NIST}$)	46.2	44.5	30.2

Table 3. Test set results.

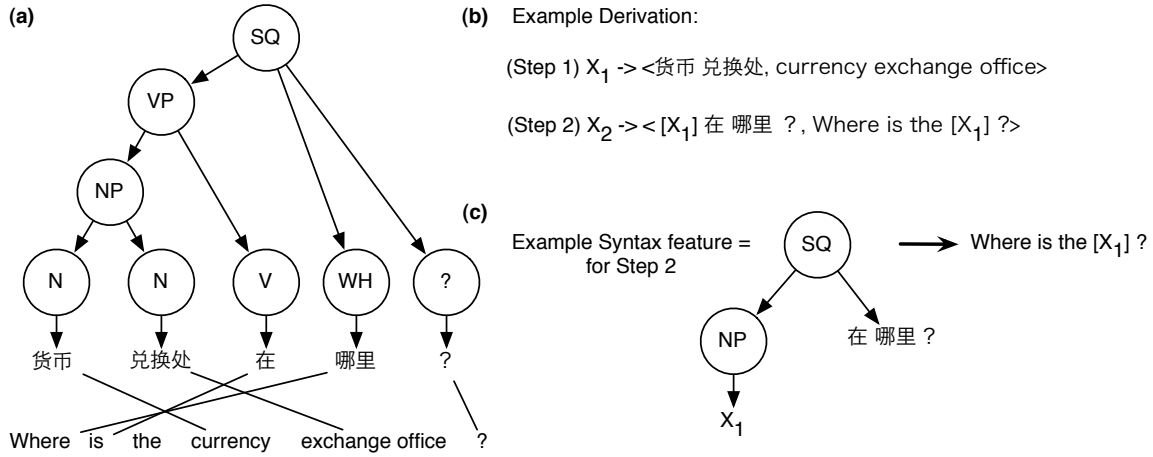

Figure 3. Syntax feature example: For the parsed source and candidate translation (a), with the derivation (b), we extract the syntax feature in (c) by combining the grammar rule with the source syntax of the constituents contained within that rule.

Source	我 身 上 没有 足够 的 钱 去 买 一 张 新 的 飞机 票 。
$p_\Lambda^{-LM}(\mathbf{e}\|\mathbf{f})$	don 't have enough bag on me change please go purchase a new by plane .
$p_\Lambda^{+LM}(\mathbf{e}\|\mathbf{f})$	i have enough money to buy a new one by air .
$p_\Lambda^{+LM}(\mathbf{e}\|\mathbf{f})$ + syntax	i don 't have enough money to buy a new airline ticket .
MERT ($BLEU^{NIST}$)	i don 't have enough money to buy a new ticket .
Reference	i do n't have enough money with me to buy a new airplane ticket .

Table 4. Example test set output produced when: not using a language model, using a language model, also using syntax, output optimised using MERT and finally the reference

$BLEU^{NIST}$ (brevity penalty uses the shortest reference), we also include results from the IBM ($BLEU^{IBM}$) metric (brevity penalty uses the closest reference), and using only the actual human translation in the test set, not the monolingual paraphrase multiple references ($BLEU^{HumanRef}$).

The first result of interest is that we see an increase in performance through the incorporation of the source syntax features. This is an encouraging result as the transcribed speech source sentences are well out of the domain of the data on which the parser was trained, suggesting that our model is able to sift the good information from the noisy in the unreliable source syntax trees. Table 4 shows illustrative system output on the test set.

On the $BLEU^{NIST}$ metric we see that our models under-perform the MERT trained system. We hypothesise that this is predominately due to the interaction of the brevity penalty with the unusual nature of the multiple paraphrase reference training and development data. Their performance is however quite consistent across the different interpretations of the brevity penalty (NIST vs. IBM). This contrasts with the MERT trained model, which clearly over-fits to the NIST metric that it was trained on and underperforms our models when evaluated on the single human test translations. If we directly compare the brevity penalties of the MERT model (0.868) and our discriminative model incorporating source syntax (0.942), on the these single

references, we can see that the MERT training has optimised to the shortest paraphrase reference.

From these results it is difficult to draw any hard conclusions on the relative performance of the different training regimes. However we feel confident in claiming that we have achieved our goal of training a probabilistic model on millions of sparse features which obtains performance competitive with the current state-of-the-art training algorithm.

5 Conclusion

In this paper we have shown that statistical machine translation can be effectively modelled as a well posed machine learning task. In doing so we have described a model capable of estimating a probability distribution over translations using sparse complex features, and achieving performance comparable to the state-of-the-art on standard metrics. With further work on scaling these models to large data sets, and engineering high performance features, we believe this research has the potential to provide significant increases in translation quality.

Acknowledgements

The authors acknowledge the support of the EPSRC grant EP/D074959/1.

References

Phil Blunsom, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proc. of the 46th Annual Conference of the Association for Computational Linguistics: Human Language Technologies (ACL-08:HLT)*, pages 200–208, Columbus, Ohio, June.

Jean-Cédric Chappelier and Martin Rajman. 2000. Monte-carlo sampling for np-hard maximization problems in the framework of weighted parsing. In *NLP '00: Proceedings of the Second International Conference on Natural Language Processing*, pages 106–117, London, UK.

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

Matthias Eck and Chiori Hori. 2005. Overview of the IWSLT 2005 evaluation campaign. In *Proc. of the International Workshop on Spoken Language Translation*, Pittsburgh, October.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. of the 44th Annual Meeting of the ACL and 21st International Conference on Computational Linguistics (COLING/ACL-2006)*, pages 961–968, Sydney, Australia, July.

Joshua T. Goodman. 1998. *Parsing inside-out*. Ph.D. thesis, Cambridge, MA, USA. Adviser-Stuart Shieber.

Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *In Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, Boston, MA.

Abraham Ittycheriah and Salim Roukos. 2007. Direct translation model 2. In *Proc. of the 7th International Conference on Human Language Technology Research and 8th Annual Meeting of the NAACL (HLT-NAACL 2007)*, pages 57–64, Rochester, USA.

Shankar Kumar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proc. of the 4th International Conference on Human Language Technology Research and 5th Annual Meeting of the NAACL (HLT-NAACL 2004)*, pages 169–176.

Philip M. Lewis II and Richard E. Stearns. 1968. Syntax-directed transduction. *J. ACM*, 15(3):465–488.

Percy Liang, Alexandre Bouchard-Côté, Dan Klein, and Ben Taskar. 2006. An end-to-end discriminative approach to machine translation. In *Proc. of the 44th Annual Meeting of the ACL and 21st International Conference on Computational Linguistics (COLING/ACL-2006)*, pages 761–768, Sydney, Australia, July.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *Proc. of the 46th Annual Conference of the Association for Computational Linguistics: Human Language Technologies (ACL-08:HLT)*, pages 192–199, Columbus, Ohio, June.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. of the 41st Annual Meeting of the ACL (ACL-2003)*, pages 160–167, Sapporo, Japan.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Meeting of the ACL and 3rd Annual Meeting of the NAACL (ACL-2002)*, pages 311–318, Philadelphia, Pennsylvania.

Taro Watanabe, Hajime Tsukada, and Hideki Isozaki. 2006. Left-to-right target generation for hierarchical phrase-based translation. In *Proc. of the 44th Annual Meeting of the ACL and 21st International Conference on Computational Linguistics (COLING/ACL-2006)*, pages 777–784, Sydney, Australia.