Combining Top-down and Bottom-up Search for Unsupervised Induction of Transduction Grammars

Markus SAERS and **Karteek ADDANKI** and **Dekai WU**
Human Language Technology Center
Dept. of Computer Science and Engineering
Hong Kong University of Science and Technology
{masaers|vskaddanki|dekai}@cs.ust.hk

Abstract

We show that combining *both* bottom-up rule chunking and top-down rule segmentation search strategies in purely unsupervised learning of phrasal inversion transduction grammars yields significantly better translation accuracy than either strategy alone. Previous approaches have relied on incrementally building larger rules by chunking smaller rules bottom-up; we introduce a complementary top-down model that incrementally builds shorter rules by segmenting larger rules. Specifically, we combine iteratively chunked rules from Saers *et al.* (2012) with our new iteratively segmented rules. These integrate seamlessly because both stay strictly within a pure transduction grammar framework inducing under matching models during both training and testing—instead of decoding under a completely different model architecture than what is assumed during the training phases, which violates an elementary principle of machine learning and statistics. To be able to drive induction top-down, we introduce a minimum description length objective that trades off maximum likelihood against model size. We show empirically that combining the more liberal rule chunking model with a more conservative rule segmentation model results in significantly better translations than either strategy in isolation.

1 Introduction

In this paper we combine both bottom-up chunking and top-down segmentation as search directions in the unsupervised pursuit of an inversion transduction grammar (ITG); we also show that the combination of the resulting grammars is superior to either of them in isolation. For the bottom-up chunking approach we use the method reported in Saers *et al.* (2012), and for the top-down segmentation approach, we introduce a minimum description length (MDL) learning objective. The new learning objective is similar to the Bayesian maximum a posteriori objective, and makes it possible to learn top-down, which is impossible using maximum likelihood, as the initial grammar that rewrites the start symbol to all sentence pairs in the training data already maximizes the likelihood of the training data. Since both approaches result in stochastic ITGs, they can be easily combined into a single stochastic ITG which allows for seamless combination. The point of our present work is that the two different search strategies result in very different grammars so that the combination of them is superior in terms of translation accuracy to either of them in isolation.

The transduction grammar approach has the advantage that induction, tuning and testing are optimized on the exact same underlying model—this used to be a given in machine learning and statistical prediction, but has been largely ignored in the statistical machine translation (SMT) community, where most current SMT approaches to learning phrase translations that (a) require enormous amounts of run-time memory, and (b) contain a high degree of redundancy. In particular, phrase-based SMT models such as Koehn *et al.* (2003) and Chiang (2007) often search for candidate translation segments and transduction rules by committing to a word alignment that is completely alien to the grammar, as it is learned with very different models (Brown *et al.* (1993), Vogel *et al.* (1996)), whose output is then combined heuristically to form the alignment actually used to extract lexical segment translations (Och

Proceedings of the 7th Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 48–57,
Atlanta, Georgia, 13 June 2013.

and Ney, 2003). The fact that it is even possible to improve the performance of a phrase-based direct translation system by tossing away most of the learned segmental translations (Johnson *et al.*, 2007) illustrates the above points well.

Transduction grammars can also be induced from treebanks instead of unannotated corpora, which cuts down the vast search space by enforcing additional, external constraints. This approach was pioneered by Galley *et al.* (2006), and there has been a lot of research since, usually referred to as **tree-to-tree**, **tree-to-string** and **string-to-tree**, depending on where the analyses are found in the training data. This complicates the learning process by adding external constraints that are bound to match the translation model poorly; grammarians of English should not be expected to care about its relationship to Chinese. It does, however, constitute a way to borrow nonterminal categories that help the translation model.

It is also possible for the word alignments leading to phrase-based SMT models to be learned through transduction grammars (see for example Cherry and Lin (2007), Zhang *et al.* (2008), Blunsom *et al.* (2008), Saers and Wu (2009), Haghighi *et al.* (2009), Blunsom *et al.* (2009), Saers *et al.* (2010), Blunsom and Cohn (2010), Saers and Wu (2011), Neubig *et al.* (2011), Neubig *et al.* (2012)). Even when the SMT model is hierarchical, most of the information encoded in the grammar is tossed away, when the learned model is reduced to a word alignment. A word alignment can only encode the lexical relationships that exist between a sentence pair according to a single parse tree, which means that the rest of the model: the alternative parses and the syntactic structure, is ignored.

The minimum description length (MDL) objective that we will be using to drive the learning will provide a way to escape the maximum-likelihood-of-the-data-given-the-model optimum that we start out with. However, going only by MDL will also lead to a degenerate case, where the size of the grammar is allowed to shrink regardless of how unlikely the corpus becomes. Instead, we will balance the length of the grammar with the probability of the corpus given the grammar. This has a natural Bayesian interpretation where the length of the grammar acts as a prior over the structure of the grammar.

Similar approaches have been used before, but to induce monolingual grammars. Stolcke and Omohundro (1994) use a method similar to MDL called *Bayesian model merging* to learn the structure of hidden Markov models as well as stochastic context-free grammars. The SCFGs are induced by allowing sequences of nonterminals to be replaced with a single nonterminal (chunking) as well as allowing two nonterminals to merge into one. Grünwald (1996) uses it to learn nonterminal categories in a context-free grammar. It has also been used to interpret visual scenes by classifying the activity that goes on in a video sequences (Si *et al.*, 2011). Our work in this paper is markedly different to even the previous NLP work in that (a) we induce an inversion transduction grammar (Wu, 1997) rather than a monolingual grammar, and (b) we focus on learning the terminal segments rather than the nonterminal categories.

The similar Bayesian approaches to finding the model structure of ITGs have been tried before, but only to generate alignments that mismatched translation models are then trained on, rather than using the ITG directly as translation model, which we do. Zhang *et al.* (2008) use variational Bayes with a sparsity prior over the parameters to prevent the size of the grammar to explode when allowing for adjacent terminals in the Viterbi biparses to chunk together. Blunsom *et al.* (2008), Blunsom *et al.* (2009) and Blunsom and Cohn (2010) use Gibbs sampling to find good phrasal translations. Neubig *et al.* (2011) and Neubig *et al.* (2012) use a method more similar to ours, but with a Pitman-Yor process as prior over the structures.

The idea of iteratively segmenting the existing sentence pairs to find good phrasal translations has also been tried before; Vilar and Vidal (2005) introduces the Recursive Alignment Model, which recursively determines whether a bispan is a good enough translation on its own (using IBM model 1), or if it should be split into two bispans (either in straight or inverted order). The model uses length of the input sentence to determine whether to split or not, and uses very limited local information about the split point to determine where to split. Training the parameters is done with a maximum likelihood objective. In contrast, our model is one single generative model (as opposed to an *ad hoc* model), trained with a minimum description length objective (rather than trying to maximize the probability of the train-

ing data).

The rest of the paper is structured so that we first take a closer look at the minimum description length principle that will be used to drive the top-down search (Section 2). We then show how the top-down grammar is learned (Sections 3 and 4), before showing how we combine the new grammar with that of Saers *et al.* (2012) (Section 5). We then detail the experimental setup that will substantiate our claims empirically (Section 6) before interpreting the results of those experiments (Section 7). Finally, we offer some conclusions (Section 8).

2 Minimum description length

The minimum description length principle is about finding the optimal balance between the size of a model and the size of some data given the model (Solomonoff (1959), Rissanen (1983)). Consider the information theoretical problem of encoding some data with a model, and then sending both the encoded data *and* the information needed to decode the data (the model) over a channel; the minimum description length would be the minimum number of bits sent over the channel. The encoded data can be interpreted as carrying the information necessary to disambiguate the ambiguities or uncertainties that the model has about the data. Theoretically, the model can *grow in size* and become *more certain* about the data, and it can *shrink in size* and become *more uncertain* about the data. An intuitive interpretation of this is that the exceptions, which are a part of the encoded data, can be moved into the model itself. By doing so, the size of the model increases, but there is no longer an exception that needs to be conveyed about the data. Some "exceptions" occur frequently enough that it is a good idea to incorporate them into the model, and some do not; finding the optimal balance minimizes the total description length.

Formally, the description length (DL) is:

$$\mathrm{DL}\left(M, D\right) = \mathrm{DL}\left(D|M\right) + \mathrm{DL}\left(M\right) \qquad (1)$$

Where M is the model and D is the data. Note the clear parallel to probabilities that have been moved into the logarithmic domain.

In natural language processing, we never have complete data to train on, so we need our models to generalize to unseen data. A model that is very certain about the training data runs the risk of not being able to generalize to new data: it is over-fitting. It is bad enough when estimating the parameters of a transduction grammar, and catastrophic when inducing the structure of the grammar. The key concept that we want to capture when learning the structure of a transduction grammar is *generalization*. This is the property that allow it to translate new, unseen, input. The challenge is to pin down what generalization actually is, and how to measure it.

One property of generalization for grammars is that it will lower the probability of the training data. This may seem counterintuitive, but can be understood as moving some of the probability mass away from the training data and putting it in unseen data. A second property is that rules that are specific to the training data can be eliminated from the grammar (or replaced with less specific rules that generate the same thing). The second property would shorten the description of the grammar, and the first would make the description of the corpus given the grammar longer. That is: generalization raises the first term and lowers the second in Equation 1. A good generalization will lower the total MDL, whereas a poor one will raise it; a good generalization will trade a little data *certainty* for more model *parsimony*.

2.1 Measuring the length of a corpus

The information-theoretic view of the problem also gives a hint at the operationalization of *length*. Shannon (1948) stipulates that the number of bits it takes to encode that a probabilistic variable has taken a certain value can be encoded using as little as the negative logarithmic probability of that outcome.

Following this, the parallel corpus given the transduction grammar gives the number of bits required to encode it: $\mathrm{DL}\left(C|G\right) = -\log_2\left(P\left(C|G\right)\right)$, where C is the corpus and G is the grammar.

2.2 Measuring the length of an ITG

Since information theory deals with encoding sequences of symbols, we need some way to serialize an inversion transduction grammar (ITG) into a message whose length can be measured.

To serialize an ITG, we first need to determine the alphabet that the message will be written in. We need one symbol for every nonterminal, L_0-terminal and L_1-terminal. We will also make the assumption that all these symbols are used in at least one

rule, so that it is sufficient to serialize the rules in order to express the entire grammar. To serialize the rules, we need some kind of delimiter to know where one rule starts and the next ends; we will exploit the fact that we also need to specify whether the rule is straight or inverted (unary rules are assumed to be straight), and merge these two functions into one symbol. This gives the union of the symbols of the grammar and the set $\{[], \langle\rangle\}$, where [] signals the beginning of a straight rule, and $\langle\rangle$ signals the beginning of an inverted rule. The serialized format of a rule will be: rule type/start marker, followed by the left-hand side nonterminal, followed by all right-hand side symbols. The symbols on the right-hand sides are either nonterminals or **biterminals**—pairs of L_0-terminals and L_1-terminals that model translation equivalences. The serialized form of a grammar is the serialized form of all rules concatenated.

Consider the following toy grammar:

$$S \rightarrow A, \qquad A \rightarrow \langle AA \rangle, \quad A \rightarrow [AA]\,,$$
$$A \rightarrow \text{have}/\text{有}, \quad A \rightarrow \text{yes}/\text{有}, \quad A \rightarrow \text{yes}/\text{是}$$

Its serialized form would be:

$$[]SA\langle\rangle AAA[]AAA[]A\text{have}\text{有}[]A\text{yes}\text{有}[]A\text{yes}\text{是}$$

Now we can, again turn to information theory to arrive at an encoding for this message. Assuming a uniform distribution over the symbols, each symbol will require $-\log_2\left(\frac{1}{N}\right)$ bits to encode (where N is the number of different symbols—the type count). The above example has 8 symbols, meaning that each symbol requires 3 bits. The entire message is 23 symbols long, which means that we need 69 bits to encode it.

3 Model initialization

Rather than starting out with a general transduction grammar and fitting it to the training data, we do the exact opposite: we start with a transduction grammar that fits the training data as well as possible, and generalize from there. The transduction grammar that fits the training data the best is the one where the start symbol rewrites to the full sentence pairs that it has to generate. It is also possible to add any number of nonterminal symbols in the layer between the start symbol and the bisentences without altering the probability of the training data. We take advantage of this by allowing for one intermediate symbol so that the start symbol conforms to the normal form and always rewrites to precisely one nonterminal symbol. This violate the MDL principle, as the introduction of new symbols, by definition, makes the description of the model longer, but conforming to the normal form of ITGs was deemed more important than strictly minimizing the description length. Our initial grammar thus looks like this:

$$\begin{aligned} S &\rightarrow A, \\ A &\rightarrow e_{0..T_0}/f_{0..V_0}, \\ A &\rightarrow e_{0..T_1}/f_{0..V_1}, \\ &\ldots, \\ A &\rightarrow e_{0..T_N}/f_{0..V_N} \end{aligned}$$

Where S is the start symbol, A is the nonterminal, N is the number of sentence pairs in the training corpus, T_i is the length of the i^{th} output sentence (which makes $e_{0..T_i}$ the i^{th} output sentence), and V_i is the length of the i^{th} input sentence (which makes $f_{0..V_i}$ the i^{th} input sentence).

4 Model generalization

To generalize the initial inversion transduction grammar we need to identify parts of the existing biterminals that could be validly used in isolation, and allow them to combine with other segments. This is the very feature that allows a finite transduction grammar to generate an infinite set of sentence pairs. Doing this moves some of the probability mass, which was concentrated in the training data, to unseen data—the very definition of generalization. Our general strategy is to propose a number of sets of biterminal rules and a place to segment them, evaluate how the description length would change if we were to apply one of these sets of segmentations to the grammar, and commit to the best set. That is: we do a greedy search over the power set of possible segmentations of the rule set. As we will see, this intractable problem can be reasonable efficiently approximated, which is what we have implemented and tested.

The key component in the approach is the ability to evaluate how the description length would change if a specific segmentation was made in the grammar.

This can then be extended to a set of segmentations, which only leaves the problem of generating suitable sets of segmentations.

The key to a successful segmentation is to maximize the potential for reuse. Any segment that can be reused saves model size. Consider the terminal rule:

$$A \rightarrow \text{five thousand yen is my limit/}\\ \text{我最多出五千日元}$$

(Chinese gloss: 'wǒ zùi dūo chū wǔ qīan rì yúan'). This rule can be split into three rules:

$$A \rightarrow \langle AA \rangle,$$
$$A \rightarrow \text{five thousand yen/五千日元},$$
$$A \rightarrow \text{is my limit/我最多出}$$

Note that the original rule consists of 16 symbols (in our encoding scheme), whereas the new three rules consists of $4 + 9 + 9 = 22$ symbols. It is reasonable to believe that the bracketing inverted rule is in the grammar already, but this still leaves 18 symbols, which is decidedly longer than 16 symbols—and we need to get the length to be shorter if we want to see a net gain, since the length of the corpus given the grammar is likely to be longer with the segmented rules. What we really need to do is find a way to reuse the lexical rules that came out of the segmentation. Now suppose the grammar also contained this terminal rule:

$$A \rightarrow \text{the total fare is five thousand yen/}\\ \text{总共的费用是五千日元}$$

(Chinese gloss: 'zǒng gòng de fèi yòng shì wǔ qīan rì yúan'). This rule can also be split into three rules:

$$A \rightarrow [AA],$$
$$A \rightarrow \text{the total fare is/总共的费用是},$$
$$A \rightarrow \text{five thousand yen/五千日元}$$

Again, we will assume that the structural rule is already present in the grammar, the old rule was 19 symbols long, and the two new terminal rules are $12+9 = 21$ symbols long. Again we are out of luck, as the new rules are longer than the old one, and three rules are likely to be less probable than one rule during parsing. The way to make this work is to realize that the two existing rules share a bilingual affix—a **biaffix**: "five thousand dollars" translating into "五千日元". If we make the two changes at the same time, we get rid of $16 + 19 = 35$ symbols worth of rules, and introduce a mere $9 + 9 + 12 = 30$ symbols worth of rules (assuming the structural rules are already in the grammar). Making these two changes at the same time is essential, as the length of the five saved symbols can be used to offset the likely increase in the length of the corpus given the data. And of course: the more rules we can find with shared biaffixes, the more likely we are to find a good set of segmentations.

Our algorithm takes advantage of the above observation by focusing on the biaffixes found in the training data. Each biaffix defines a set of lexical rules paired up with a possible segmentation. We evaluate the biaffixes by estimating the change in description length associated with committing to all the segmentations defined by a biaffix. This allows us to find the best set of segmentations, but rather than committing only to the one best set of segmentations, we will collect all sets which would improve description length, and try to commit to as many of them as possible. The pseudocode for our algorithm is as follows:

```
G                     // The grammar
biaffixes_to_rules    // Maps biaffixes to the
                      //    rules they occur in
biaffixes_delta = []  // A list of biaffixes and
                      //    their DL impact on G
for each biaffix b :
   delta = eval_dl(b, biaffixes_to_rules[b], G)
   if (delta < 0)
      biaffixes_delta.push(b, delta)
sort_by_delta(biaffixes_delta)
for each b:delta pair in biaffixes_delta :
   real_delta = eval_dl(b, biaffixes_to_rules[b], G)
   if (real_delta < 0)
      G = make_segmentations(b, biaffixes_to_rules[b], G)
```

The methods `eval_dl`, `sort_by_delta` and `make_segmentations` evaluates the impact on description length that committing to a biaffix would cause, sorts a list of biaffixes according to this delta, and applies all the changes associated with a biaffix to the grammar, respectively.

Evaluating the impact on description length breaks down into two parts: the difference in description length of the grammar $\mathrm{DL}(G') - \mathrm{DL}(G)$ (where G' is the grammar that results from applying all the changes that committing to a biaffix dictates),

and the difference in description length of the corpus given the grammar $\mathrm{DL}(C|G') - \mathrm{DL}(C|G)$. These two quantities are simply added up to get the total change in description length.

The difference in grammar length is calculated as described in Section 2.2. The difference in description length of the corpus given the grammar can be calculated by biparsing the corpus, since $\mathrm{DL}(C|G') = -\log_2(P(C|p'))$ and $\mathrm{DL}(C|G) = -\log_2(P(C|p))$ where p' and p are the rule probability functions of G' and G respectively. Biparsing is, however, a very costly process that we do not want to have inside a loop. Instead, we assume that we have the original corpus probability (through biparsing *outside* the loop), and estimate the new corpus probability from it (in closed form). Given that we are splitting the rule r_0 into the three rules r_1, r_2 and r_3, and that the probability mass of r_0 is distributed uniformly over the new rules, the new rule probability function p' will be identical to p, except that:

$$\begin{aligned} p'(r_0) &= 0, \\ p'(r_1) &= p(r_1) + \frac{1}{3}p(r_0), \\ p'(r_2) &= p(r_2) + \frac{1}{3}p(r_0), \\ p'(r_3) &= p(r_3) + \frac{1}{3}p(r_0) \end{aligned}$$

Since we have eliminated all the occurrences of r_0 and replaced them with combinations of r_1, r_2 and r_3, the probability of the corpus given this new rule probability function will be:

$$P(C|p') = P(C|p)\frac{p'(r_1)\,p'(r_2)\,p'(r_3)}{p(r_0)}$$

To make this into a description length, we need to take the negative logarithm of the above, which results in:

$$\mathrm{DL}(C|G') = \mathrm{DL}(C|G) - \log_2\left(\frac{p'(r_1)\,p'(r_2)\,p'(r_3)}{p(r_0)}\right)$$

The difference in description length of the corpus given the grammar can now be expressed as:

$$\mathrm{DL}(C|G') - \mathrm{DL}(C|G) = -\log_2\left(\frac{p'(r_1)p'(r_2)p'(r_3)}{p(r_0)}\right)$$

To calculate the impact of a set of segmentations, we need to take all the changes into account in one go. We do this in a two-pass fashion, first calculating the new probability function (p') and the change in grammar description length (taking care not to count the same rule twice), and then, in the second pass, calculating the change in corpus description length.

5 Model combination

The model we learn by iteratively subsegmenting the training data is guaranteed to be parsimonious while retaining a decent fit to the training data; these are desirable qualities, but there is a real risk that we failed to make some generalization that we should have made; to counter this risk, we can use a model trained under more liberal conditions. We chose the approach taken by Saers *et al.* (2012) for two reasons: (a) the model has the same form as our model, which means that we can integrate it seamlessly, and (b) their aims are similar to ours but their method differs significantly; specifically, they let the model grow in size as long as the data reduces in size. Both these qualities make it a suitable complement for our model.

Assuming we have two grammars (G_a and G_b) that we want to combine, the interpolation parameter α will determine the probability function of the combined grammar such that:

$$p_{a+b}(r) = \alpha p_a(r) + (1-\alpha)p_b(r)$$

for all rules r in the union of the two rule sets, and where p_{a+b} is the rule probability function of the combined grammar and p_a and p_b are the rule probability functions of G_a and G_b respectively. Some initial experiments indicated that an α value of about 0.4 was reasonable (when G_a was the grammar obtained through the training scheme outlined above, and G_b was the grammar obtained through the training scheme outlined in Saers *et al.* (2012)), so we used 0.4 in this paper.

6 Experimental setup

We have made the claim that iterative top-down segmentation guided by the objective of minimizing the description length gives a better precision grammar than iterative bottom-up chunking, and that the combination of the two gives superior results to either

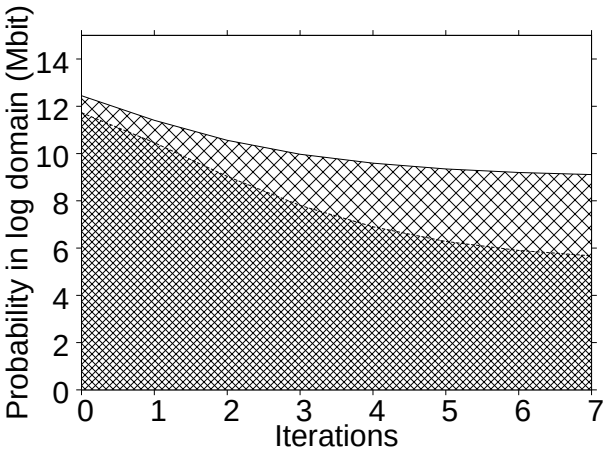

Figure 1: Description length in bits over the different iterations of top-down search. The lower portion represents DL (G) and the upper portion represents DL $(C|G)$.

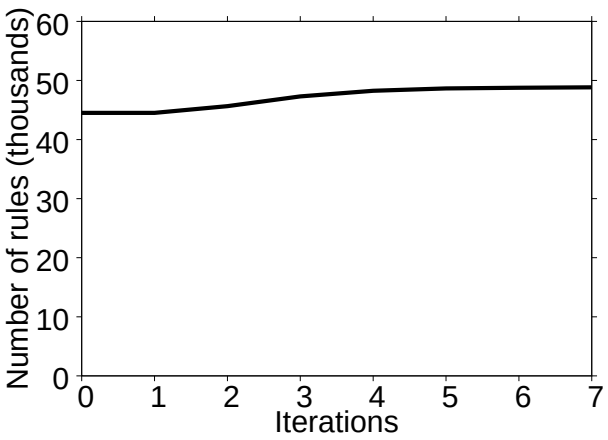

Figure 2: Number of rules learned during top-down search over the different iterations.

approach in isolation. We have outlined how this can be done in practice, and we now substantiate that claim empirically.

We will initialize a stochastic bracketing inversion transduction grammar (BITG) to rewrite it's one nonterminal symbol directly into all the sentence pairs of the training data (iteration 0). We will then segment the grammar iteratively a total of seven times (iterations 1–7). For each iteration we will record the change in description length and test the grammar. Each iteration requires us to biparse the training data, which we do with the cubic time algorithm described in Saers *et al.* (2009), with a beam width of 100.

As training data, we use the IWSLT07 Chinese–English data set (Fordyce, 2007), which contains 46,867 sentence pairs of training data, 506 Chinese sentences of development data with 16 English reference translations, and 489 Chinese sentences with 6 English reference translations each as test data; all the sentences are taken from the traveling domain. Since the Chinese is written without whitespace, we use a tool that tries to clump characters together into more "word like" sequences (Wu, 1999).

As the bottom-up grammar, we will reuse the grammar learned in Saers *et al.* (2012), specifically, we will use the BITG that was bootstrapped from a bracketing finite-state transduction grammar (BFSTG) that has been chunked twice, giving biterminals where the monolingual segments are 0–4 tokens long. The bottom-up grammar is trained on the same data as our model.

To test the learned grammars as translation models, we first tune the grammar parameters to the training data using expectation maximization (Dempster *et al.*, 1977) and parse forests acquired with the above mentioned biparser, again with a beam width of 100. To do the actual decoding, we use our in-house ITG decoder. The decoder uses a CKY-style parsing algorithm (Cocke, 1969; Kasami, 1965; Younger, 1967) and cube pruning (Chiang, 2007) to integrate the language model scores. The decoder builds an efficient hypergraph structure which is then scored using both the induced grammar and the language model. The weights for the language model and the grammar, are tuned towards BLEU (Papineni *et al.*, 2002) using MERT (Och, 2003). We use the ZMERT (Zaidan, 2009) implementation of MERT as it is a robust and flexible implementation of MERT, while being loosely coupled with the decoder. We use SRILM (Stolcke, 2002) for training a trigram language model on the English side of the training data. To evaluate the quality of the resulting translations, we use BLEU, and NIST (Doddington, 2002).

7 Experimental results

The results from running the experiments detailed in the previous section can be summarized in four graphs. Figures 1 and 2 show the size of our new, segmenting model during induction, in terms of description length and in terms of rule count. The initial ITG is at iteration 0, where the vast majority

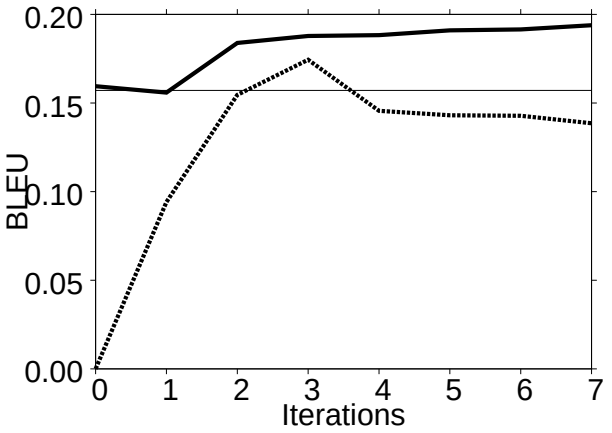

Figure 3: Variations in BLEU score over different iterations. The thin line represents the baseline bottom-up search (Saers *et al.*, 2012), the dotted line represents the top-down search, and the thick line represents the combined results.

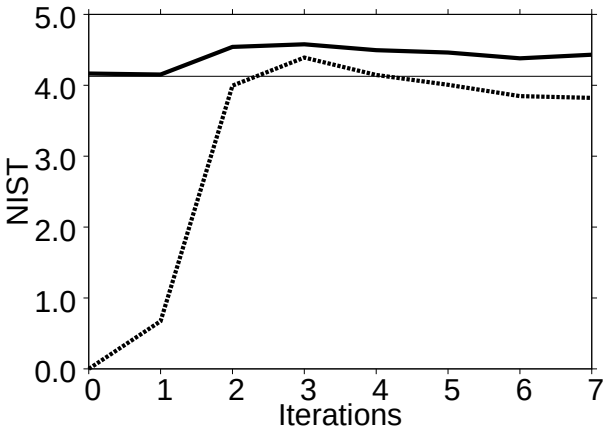

Figure 4: Variations in NIST score over different iterations. The thin line represents the baseline bottom-up search (Saers *et al.*, 2012), the dotted line represents the top-down search, and the thick line represents the combined results.

of the size is taken up by the model (DL (G)), and very little by the data (DL $(C|G)$)—just as we predicted. The trend over the induction phase is a sharp decrease in model size, and a moderate increase in data size, with the overall size constantly decreasing. Note that, although the number of rules rises, the total description length decreases. Again, this is precisely what we expected. The size of the model learned according to Saers *et al.* (2012) is close to 30 Mbits—far off the chart. This shows that our new top-down approach is indeed learning a more parsimonious grammar than the bottom-up approach.

Figures 3 and 4 shows the translation quality of the learned model. The thin flat lines show the quality of the bottom-up approach (Saers *et al.*, 2012), whereas the thick curves shows the quality of the new, top-down model presented in this paper without (dotted line), and without the bottom-up model (solid line). Although the MDL-based model is better than the old model, the combination of the two is still superior. It is particularly encouraging to see that the over-fitting that seems to take place after iteration 3 with the MDL-based approach is ameliorated with the bottom-up model.

8 Conclusions

We have introduced a purely unsupervised learning scheme for phrasal stochastic inversion transduction grammars that is the first to combine two opposing ways of searching for the phrasal translations: a bottom-up rule chunking approach driven by a maximum likelihood (ML) objective and a top-down rule segmenting approach driven by a minimum description length (MDL) objective. The combination approach takes advantage of the fact that the conservative top-down MDL-driven rule segmenting approach learns a very parsimonious, yet competitive, model when compared to a liberal bottom-up ML-driven approach. Results show that the combination of the two opposing approaches is significantly superior to either of them in isolation.

9 Acknowledgements

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under BOLT contract no. HR0011-12-C-0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; by the European Union under the FP7 grant agreement no. 287658; and by the Hong Kong Research Grants Council (RGC) research grants GRF620811, GRF621008, and GRF612806. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, the EU, or RGC.

References

P. Blunsom and T. Cohn. Inducing synchronous grammars with slice sampling. In *HLT/NAACL2010*, pages 238–241, Los Angeles, California, June 2010.

P. Blunsom, T. Cohn, and M. Osborne. Bayesian synchronous grammar induction. In *Proceedings of NIPS 21*, Vancouver, Canada, December 2008.

P. Blunsom, T. Cohn, C. Dyer, and M. Osborne. A gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of ACL/IJCNLP*, pages 782–790, Suntec, Singapore, August 2009.

P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The Mathematics of Machine Translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

C. Cherry and D. Lin. Inversion transduction grammar for joint phrasal translation modeling. In *Proceedings of SSST*, pages 17–24, Rochester, New York, April 2007.

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

J. Cocke. *Programming languages and their compilers: Preliminary notes*. Courant Institute of Mathematical Sciences, New York University, 1969.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the 2nd International Conference on Human Language Technology Research*, pages 138–145, San Diego, California, 2002.

C. S. Fordyce. Overview of the IWSLT 2007 evaluation campaign. In *Proceedings of IWSLT*, pages 1–12, 2007.

M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, and I. Thayer. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of COLING/ACL-2006*, pages 961–968, Sydney, Australia, July 2006.

Peter Grünwald. A minimum description length approach to grammar inference in symbolic. *Lecture Notes in Artificial Intelligence*, (1040):203–216, 1996.

A. Haghighi, J. Blitzer, J. DeNero, and D. Klein. Better word alignments with supervised itg models. In *Proceedings of ACL/IJCNLP-2009*, pages 923–931, Suntec, Singapore, August 2009.

H. Johnson, J. Martin, G. Foster, and R. Kuhn. Improving translation quality by discarding most of the phrasetable. In *Proceedings of EMNLP-CoNLL-2007*, pages 967–975, Prague, Czech Republic, June 2007.

T. Kasami. An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-00143, Air Force Cambridge Research Laboratory, 1965.

P. Koehn, F. J. Och, and D. Marcu. Statistical Phrase-Based Translation. In *Proceedings of HLT/NAACL-2003*, volume 1, pages 48–54, Edmonton, Canada, May/June 2003.

G. Neubig, T. Watanabe, E. Sumita, S. Mori, and T. Kawahara. An unsupervised model for joint phrase alignment and extraction. In *Proceedings of ACL/HLT-2011*, pages 632–641, Portland, Oregon, June 2011.

G. Neubig, T. Watanabe, S. Mori, and T. Kawahara. Machine translation without words through substring alignment. In *Proceedings of ACL-2012*, pages 165–174, Jeju Island, Korea, July 2012.

F. J. Och and H. Ney. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, 2003.

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

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

J. Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of Statistics*, 11(2):416–431, June 1983.

M. Saers and D. Wu. Improving phrase-based translation via word alignments from Stochastic Inversion Transduction Grammars. In *Proceedings of SSST-3*, pages 28–36, Boulder, Colorado, June 2009.

M. Saers and D. Wu. Principled induction of phrasal bilexica. In *Proceedings of EAMT-2011*, pages 313–320, Leuven, Belgium, May 2011.

M. Saers, J. Nivre, and D. Wu. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *Proceedings of IWPT'09*, pages 29–32, Paris, France, October 2009.

M. Saers, J. Nivre, and D. Wu. Word alignment with stochastic bracketing linear inversion transduction grammar. In *Proceedings of HLT/NAACL-2010*, pages 341–344, Los Angeles, California, June 2010.

M. Saers, K. Addanki, and D. Wu. From finite-state to inversion transductions: Toward unsupervised bilingual grammar induction. In *Proceedings of COLING 2012: Technical Papers*, pages 2325–2340, Mumbai, India, December 2012.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–, July, October 1948.

Z. Si, M. Pei, B. Yao, and S. Zhu. Unsupervised learning of event and-or grammar and semantics from video. In *Proceedings of the 2011 IEEE International Conference on Computer Vision (ICCV)*, pages 41–48, November 2011.

R. J. Solomonoff. A new method for discovering the grammars of phrase structure languages. In *IFIP Congress*, pages 285–289, 1959.

A. Stolcke and S. Omohundro. Inducing probabilistic grammars by bayesian model merging. In R. C. Carrasco and J. Oncina, editors, *Grammatical Inference and Applications*, pages 106–118. Springer, 1994.

A. Stolcke. SRILM – an extensible language modeling toolkit. In *Proceedings of ICSLP-2002*, pages 901–904, Denver, Colorado, September 2002.

J. M. Vilar and E. Vidal. A recursive statistical translation model. In *ACL-2005 Workshop on Building and Using Parallel Texts*, pages 199–207, Ann Arbor, Jun 2005.

S. Vogel, H. Ney, and C. Tillmann. HMM-based Word Alignment in Statistical Translation. In *Proceedings of COLING-96*, volume 2, pages 836–841, 1996.

D. Wu. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377–403, 1997.

Z. Wu. LDC Chinese segmenter, 1999.

D. H. Younger. Recognition and parsing of context-free languages in time n^3. *Information and Control*, 10(2):189–208, 1967.

O. F. Zaidan. Z-MERT: A Fully Configurable Open Source Tool for Minimum Error Rate Training of Machine Translation Systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88, 2009.

H. Zhang, C. Quirk, R. C. Moore, and D. Gildea. Bayesian learning of non-compositional phrases with synchronous parsing. In *Proceedings of ACL-08: HLT*, pages 97–105, Columbus, Ohio, June 2008.