Bayesian Induction of Bracketing Inversion Transduction Grammars

Markus Saers **Dekai Wu**
Human Language Technology Center
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
{masaers|dekai}@cs.ust.hk

Abstract

We present a novel approach to learning phrasal inversion transduction grammars via Bayesian MAP (maximum *a posteriori*) or information-theoretic MDL (minimum description length) model optimization so as to incorporate simultaneously the choices of model structure as well as parameters. In comparison to most current SMT approaches, the model learns phrase translation lexicons that (a) do not require enormous amounts of run-time memory, (b) contain significantly less redundancy, and (c) provide an obvious basis for generalization to abstract translation schemas. Model structure choice is biased by a description length prior, while parameter choice is driven by data likelihood biased by a parameter prior. The search over possible model structures is made feasible by a novel top-down rule segmenting heuristic which efficiently incorporates estimates of the posterior probabilities. Since the priors reward model parsimony, the learned grammar is very concise and still performs significantly better than the maximum likelihood driven bottom-up rule chunking baseline.

1 Introduction

We introduce a minimalist, unsupervised learning model that induces relatively clean, compact phrasal translation lexicons by employing a novel Bayesian approach that attempts to find the maximum *a posteriori* (MAP) or minimum description length (MDL) model. The approach iteratively segments the rules in a top-down fashion which allows for efficient estimation of the model prior and the likelihood of the data given a limited change in the model—it is, in other words, possible to generate a set of possible rule segmentations and compare them using the model posteriors or description length.

Our new approach differs from most other SMT approaches to unsupervised learning of phrasal translations, which (a) require enormous amounts of run-time memory, (b) contain a high degree of redundancy, and (c) do not provide an obvious basis for generalization to abstract translation schemas. The current state-of-the-art in SMT (Koehn *et al.*, 2003; Chiang, 2005) relies on long pipelines of mismatched learning models and heuristics. There is no way for latter stages of the pipeline to recover a mistake of ommission made in an earlier stage, which forces the individual steps to massively overgenerate hypotheses. This typically manifests as massive redundancy in the phrasal lexicon, which causes significant overhead at run-time. 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 these deficiencies well. By staying within a single framework throughout training and testing, we do not have to overgenerate hypotheses—instead, we are able to evaluate their effect on the posterior model probability at the time they are proposed during learning. This cuts down the size of the phrasal lexicon significantly, and consequently saves the decoder a lot of run-time resources. The fact that we learn a phrasal inversion transduction grammar, or ITG (Wu, 1997) also means that the power to generalize and abstract over categories is built into the formalism (although we will not make use of this feature in this work).

A word as to the bigger-picture motivation for this line of inquiry may be necessary. By insisting on the fundamental machine learning principle of matching the training model to the testing model, we accept forfeiting the short term boost

International Joint Conference on Natural Language Processing, pages 1158–1166,
Nagoya, Japan, 14-18 October 2013.

in BLEU that is typically seen when embedding a learned ITG in the midst of the common heuristics employed in statistical machine translation. For example, Cherry and Lin (2007); Zhang *et al.* (2008); Blunsom *et al.* (2008, 2009); Haghighi *et al.* (2009); Saers and Wu (2009, 2011); Blunsom and Cohn (2010); Burkett *et al.* (2010); Riesa and Marcu (2010); Saers *et al.* (2010); Neubig *et al.* (2011, 2012) all plug some aspect of the ITGs they learn into training pipelines for existing, mismatched decoders, typically in the form of the word alignment that an ITG imposes on a parallel corpus as it is biparsed. Although this allows us to tap into the vast engineering efforts that have gone into tweaking existing decoders, it also prevents us from understanding the quality of the learned transduction grammar, whose characteristics become obscured by the many unrelated variables in the subsequent processing pipeline. Our own past work has also taken similar approaches, but it is not necessary to do so—instead, any ITG can be used for decoding by directly parsing with the input sentence as a hard constraint, as we do in this paper. The motivation for our present series of experiments is that as a field we are well served by tackling the fundamental questions as well, and not exclusively focusing on engineering short term incremental BLEU score boosts where the quality of an induced ITG itself is obscured because it is embedded within many other heuristic algorithms.

Bayesian approaches to grammar induction have a long history in computation linguistics. Starting with monolingual grammar induction (Chen, 1995; Stolcke and Omohundro, 1994), and moving on to transduction grammar induction (Blunsom *et al.*, 2008, 2009; Blunsom and Cohn, 2010; Neubig *et al.*, 2011, 2012). So far, the induced transduction grammar have only been used to derive Viterbi-style word alignments to feed into existing translation system, and there has been no evaluation of the grammars actually learned. In contrast, we directly evaluate the grammars that we induce.

Our algorithm for learning the structure of an ITG relies on segmenting known bilingual segments, starting with the sentence pairs of the training data, and continuing with the segments learned in this way. This is similar to the Recursive Alignment Model, or MAR (Vilar, 2005; Vilar and Vidal, 2005). Our method is, however, learning a full ITG, where MAR only learns a translation lexicon; furthermore, MAR is a discriminative model, whereas ours is a generative.

Transduction grammars can also be induced from treebanks instead of unannotated corpora, which cuts down the vast search space by enforcing additional, external constraints—taking it from the realm of unsupervised induction into the realm of supervised induction. This approach was pioneered by Galley *et al.* (2006) with numerous variants in subsequent research, 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. Our view on this line of research is that it 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.

The work presented in this paper is related to our preliminary work with description length as learning objective (Saers *et al.*, 2013b). A key difference lies in the added parameter training, which facilitates a completely Bayesian interpretation. The present paper aims to be self-contained, by explaining the relationships throughout.

2 Background

In this section we briefly survey essential foundations for inversion transduction grammars and description length together with its Bayesian interpretation—in other words, what we search for, and how.

2.1 Inversion transduction grammars

Inversion transduction grammars, or ITGs (Wu, 1997), are an expressive yet efficient way to model translation. Much like context-free grammars (CFGs), they allow for sentences to be explained through composition of smaller units into larger units, but where CFGs are restricted to generate monolingual sentences, ITGs generate *pairs of sentences*—**transductions** rather than languages. Naturally, the components of different languages may have to be ordered differently, which means that transduction grammars need to handle these differences in order. Rather than allowing arbitrary reordering and pay the price of exponential time complexity, ITGs allow the only monotonically straight or inverted order of the productions,

which cuts the time complexity down to a manageable polynomial.

Formally, an ITG is a tuple $\langle N, \Sigma, \Delta, R, S \rangle$, where N is a finite nonempty set of nonterminal symbols, Σ is a finite set of terminal symbols in L_0, Δ is a finite set of terminal symbols in L_1, R is a finite nonempty set of inversion transduction rules and $S \in N$ is a designated start symbol. An inversion transduction rule is restricted to take one of the following forms:

$$S \rightarrow [A],\ A \rightarrow \left[\Psi^+\right],\ A \rightarrow \langle \Psi^+ \rangle$$

where $S \in N$ is the start symbol, $A \in N$ is a nonterminal symbol, and Ψ^+ is a nonempty sequence of nonterminals and biterminals. A **biterminal** is a pair of symbol strings: $\Sigma^* \times \Delta^*$, where at least one of the strings have to be nonempty. The square and angled brackets signal straight and inverted order respectively. With straight order, both the L_0 and the L_1 productions are generated left-to-right, but with inverted order, the L_1 production is generated right-to-left. The brackets are frequently left out when there is only one element on the right-hand side, which means that $S \rightarrow [A]$ is shortened to $S \rightarrow A$.

Like CFGs, ITGs also have a 2-normal form, analogous to the Chomsky normal form for CFGs, where the rules are further restricted to only the following four forms:

$$S \rightarrow A,\ A \rightarrow [BC],\ A \rightarrow \langle BC \rangle,\ A \rightarrow e/f$$

where $S \in N$ is the start symbol, $A, B, C \in N$ are nonterminal symbols and e/f is a biterminal string.

2.2 MAP and MDL

Our approach to transduction grammar induction can be equivalently interpreted either from a Bayesian perspective as finding the grammar model Φ with maximum *a posteriori* probability (MAP) given training data corpus D, or from a compression perspective as finding the model Φ with minimum description length (MDL) needed to encode data D so we can transmit both the encoded data and the model needed to decode it using as few bits as possible.

In the MAP case, the goal is to find the model Φ with the maximum posterior probability, given the data D and assuming a prior $P(\Phi)$ over the space of models:

$$P(\Phi|D) = \frac{P(\Phi)\,P(D|\Phi)}{P(D)}$$

which gives the following search problem:

$$\underset{\Phi}{\text{argmax}} P(\Phi|D) = \underset{\Phi}{\text{argmax}} P(\Phi)\,P(D|\Phi)$$

since the data is fixed.

In the MDL case, the minimum description length principle is about compressing a corpus by finding the optimal balance between the size of a model, $\text{DL}(\Phi)$, and the size of some data given the model, $\text{DL}(D|\Phi)$(Solomonoff, 1959; Rissanen, 1983). In information theoretic terms, we encode the data with a model, and then transmit both the encoded data *and* the information needed to decode the data (the model) over a channel; the minimum description length is the minimum number of bits we can get away with sending over the channel. The encoded data can be interpreted as carrying the information necessary to disambiguate the uncertainties that the model has about the data. The model can *grow in size* and become *more certain* about the data, or it can *shrink in size* and become *more uncertain* about the data. Formally, description length (DL) is:

$$\text{DL}(\Phi, D) = \text{DL}(\Phi) + \text{DL}(D|\Phi)$$

which gives the following search problem:

$$\underset{\Phi}{\text{argmin}} \text{DL}(\Phi, D) = \underset{\Phi}{\text{argmin}} \text{DL}(\Phi) + \text{DL}(D|\Phi)$$

which is equivalent to the MAP search problem if we follow the Shannon (1948) lower bound on the number of bits required to encode a specific outcome of a random variable such that $\text{DL}(\cdot) = -\lg P(\cdot)$ and conversely $P(\cdot) = 2^{-\text{DL}(\cdot)}$, since in that case the description length of the model $\text{DL}(\Phi) = -\lg P(\Phi)$ and the description length of the data given the model $\text{DL}(D|\Phi) = -\lg P(D|\Phi)$. These two interchangeable views of the problem are complimentary; in our previous work on minimizing description length (Saers *et al.*, 2013a,b), we have used this equality to model the description length of the data given the model in terms of the probability of biparsing a parallel corpus with an ITG.

In Bayesian modeling, it is frequently useful to break out different aspects of the prior. For transduction grammars, we will break the prior into three aspects: the type of transduction grammar (Φ_G), the structure of the grammar (the specific set of rules conforming to the type, Φ_S), and the parameters of the grammar (θ_Φ). The prior is thus

broken down such that:

$$P(\Phi) = P(\Phi_G)\, P(\Phi_S|\Phi_G)\, P(\theta_\Phi|\Phi_S, \Phi_G)$$

The prior over grammar formalisms $P(\Phi_G)$ will be kept fixed at *bracketing inversion transduction grammar* in this paper. In our previous work on minimizing description length, the model length depended purely on the grammar structure, which was what we were trying to induce. Reusing that in the Bayesian interpretation gives:

$$P(\Phi_S|\Phi_G) = 2^{-\mathrm{DL}(\Phi_S|\Phi_G)}$$

The next section contains details about how the description length of ITGs is calculated. For the parameter prior $P(\theta_\Phi|\Phi_S, \Phi_G)$, we choose a symmetric Dirichlet distribution over rule right-hand sides given rule left-hand sides, with a concentration parameter of two ($\alpha_0 = \alpha_1 = \cdots = \alpha_{R_i-1} = 2$ for all i).

The full search problem we are trying to solve is thus:

$$\begin{aligned}&\operatorname*{argmax}_{\Phi_G,\Phi_S,\theta_\Phi} P(\Phi_G) \times P(\Phi_S|\Phi_G) \\ &\qquad \times P(\theta_\Phi|\Phi_S, \Phi_G) \times P(D|\Phi_G, \Phi_S, \theta_\Phi)\end{aligned}$$

or conversely:

$$\begin{aligned}&\operatorname*{argmin}_{\Phi_G,\Phi_S,\theta_\Phi} \mathrm{DL}(\Phi_G) + \mathrm{DL}(\Phi_S|\Phi_G) \\ &\quad + \mathrm{DL}(\theta_\Phi|\Phi_S, \Phi_G) + \mathrm{DL}(D|\theta_\Phi, \Phi_S, \Phi_G)\end{aligned}$$

As stated earlier, we will keep Φ_G fixed so that we are only considering bracketing inversion transduction grammars.

2.3 Description length of ITGs

As mentioned, the structural prior of an ITG is based on its description length. To compute the description length of an ITG, we will turn to information theory, which can be used to compute the space requirements for encoding a sequence of symbols. This requires the serialization of ITGs into sequences of symbols. 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 ends and the next starts; 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. The serialized form of a grammar is the serialized form of all rules concatenated.

Consider the following toy grammar:

$$\begin{array}{lll} S \to A & A \to \langle AA \rangle & A \to [AA] \\ A \to \text{have}/\text{有} & A \to \text{yes}/\text{有} & A \to \text{yes}/\text{是} \end{array}$$

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 $-\lg\frac{1}{N}$ 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 Initializing model structure: Initial ITG rules

To tackle the pitfalls of premature pruning in our earlier rule-chunking approaches of starting out with a fairly general transduction grammar and fitting it to the training data (Saers *et al.*, 2011, 2012), we do the exact opposite here: 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 ITG conforms to the normal form and always rewrites

the start symbol to precisely one nonterminal symbol. Our initial ITG thus contains long rules that look like this:

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

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, V_i is the length of the i^{th} input sentence, $e_{0..T_i}$ is the sequence $e_0 e_1 \dots e_{T_i-1}$ of output tokens (that is: the i^{th} output sentence), and $f_{0..V_i}$ is the sequence $f_0 f_1 \dots f_{V_i-1}$ of input tokens (that is: the i^{th} input sentence).

4 Generalizing model structure: Shortening long ITG rules

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 out to other data that are still unseen—the very notion of generalizing beyond the training data.

In practice, we will segment the existing lexical rules into smaller lexical rules and the structural rules needed to compose them into the original, unsegmented, lexical unit. This preserves the capability to generate the original transduction, but also allows for novel combinations of the newly introduced lexical building blocks into novel sentence pairs, which extends the set of sentence pairs that the grammar can generate. Although it is possible to segment one rule at a time, as we did in Saers *et al.* (2013c), it is better to collect several rules with something in common and exploit this commonality, as we did in Saers *et al.* (2013a,b). Compared to these previous works, the objective criterion that we use to drive structural generalization, as defined in Section 2.2, is also a further improvement; our shift here from an MDL to a MAP interpretation naturally suggests the enhanced formulation of the Bayesian priors.

The general strategy is to propose a number of sets of biterminal rules and a place to segment them, estimate the posterior probability given these sets and commit to the best. 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 reasonably efficiently approximated.

The key component in the approach is the ability to evaluate the change in a posteriori probability 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.

In this work, we are only considering segmentation of lexical rules, which keeps the ITG in normal form, greatly simplifying processing without altering the expressivity. A lexical ITG rule has the form $A \rightarrow e_{0..T}/f_{0..V}$, where A is the left-hand side nonterminal—the category, $e_{0..T}$ is a sequence of T (from position 0 up to but not including position T) L_0 tokens and $f_{0..V}$ is a sequence of V (from position 0 up to but not including position V) L_1 tokens. When segmenting this rule, three new rules are produced which take one of the following forms depending on whether the segmentation is inverted or not:

$$\begin{aligned} &A \rightarrow [BC] && && A \rightarrow \langle BC \rangle \\ &B \rightarrow e_{0..S}/f_{0..U} && \text{or} && B \rightarrow e_{0..S}/f_{U..V} \\ &C \rightarrow e_{S..T}/f_{U..V} && && C \rightarrow e_{S..T}/f_{0..U} \end{aligned}$$

All possible splits of the terminal rule can be accounted for by choosing the identities of B, C, S and U, as well as whether the split it straight or inverted.

The key to a successful segmentation is to maximize the potential for reuse. Any segment that can be reused maximizes the model prior. Consider the lexical rule:

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

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

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

Note that the original rule consists of 16 symbols (in our encoding scheme), whereas the three new

rules consists of $4 + 9 + 9 = 22$ symbols. Add to that that three rules are likely to be less probable than one rule when parsing, which makes the training data less likely as well. It is reasonable to believe that the bracketing inverted rule $A \rightarrow \langle AA \rangle$ is present 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. 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 lexical rule:

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

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

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

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. 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 decrease in the probability of the data given the grammar. 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 posterior probability 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 the posterior probability, and try to commit to as many of them as possible. This minimizes the parsing efforts, which are very expensive.

Algorithm 1 Bayesian learning of ITG structure through iterative rule segmentation.

```
Φ                                   ▷ The ITG being induced
repeat
  δ ← 1
  bs ← collect_biaffixes(Φ)
  bδ ← []
  for all b ∈ bs do
    δ_b ← eval_map(b, Φ)
    if δ_b > 1 then
      bδ ← [bδ, ⟨b, δ_b⟩]
    end if
  end for
  sort_by_delta(bδ)
  for all ⟨b, δ_b⟩ ∈ bδ do
    δ'_b ← eval_map(b, Φ)
    if δ'_b > 1 then
      Φ ← make_segmentations(b, Φ)
      δ ← δ δ'_b
    end if
  end for
until δ ≤ 1
return Φ
```

The pseudocode for the search algorithm can be found in Algorithm 1. It uses the methods `collect_biaffixes`, `eval_map`, `sort_by_delta` and `make_segmentations`, which collects all biaffixes found in the rules of an ITG, evaluates the change in posterior probability caused by segmenting an ITG according to a biaffix, sorts biaffix–change-in-posterior pairs according to the latter, and commits to a set of segmentations, respectively.

To evaluate the change in posterior probability caused by a proposed set of candidate segmentations, we need to calculate the ratio between the posterior of the current model structure and the model structure that would result from committing to the candidate segmentations:

$$\frac{P\left(\Phi'|D\right)}{P\left(\Phi|D\right)} \propto \frac{P\left(\Phi'_S|\Phi_G\right)}{P\left(\Phi_S|\Phi_G\right)} \frac{P\left(D|\Phi'_S, \Phi_G, \theta_{\Phi'}\right)}{P\left(D|\Phi_S, \Phi_G, \theta_{\Phi}\right)}$$

The proportionality holds because we are keeping the model formalism (Φ_G) and the model parameters (θ_Φ and $\theta_{\Phi'}$) fixed. We arrive at the ratio between the structural priors by using description

length:

$$\frac{P\left(\Phi'_S|\Phi_G\right)}{P\left(\Phi_S|\Phi_G\right)} = 2^{-(\mathrm{DL}(\Phi'_S)-\mathrm{DL}(\Phi_S))}$$

Rather than biparsing the entire training data with the two models, we approximate the change as the ratio between the probabilities of the rules that differ. We thus assume that:

$$\frac{P\left(D|\Phi'_S,\Phi_G,\theta_{\Phi'}\right)}{P\left(D|\Phi_S,\Phi_G,\theta_{\Phi}\right)} = \frac{\hat{p}'\left(r_1\right)\hat{p}'\left(r_2\right)\hat{p}'\left(r_3\right)}{\hat{p}\left(r_0\right)}$$

where $\hat{p}$ and $\hat{p}'$ are the estimated rule probability functions within θ_Φ and $\theta_{\Phi'}$ respectively. They differ only with respect to the changed rules, such that:

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

When more than one rule is segmented, we first aggregate the changes in rule probabilities for the entire set of rules, and then aggregate the changes in data probability.

We have now approximated the change in posterior probability to the point that we can efficiently calculate it in closed form for an arbitrary set of rule segmentations.

For practical purposes, we perform the search in two phases: one that focuses on the structure of the ITG, and one that focuses on the probabilities. The former performs top-down rule segmentation as described in Saers *et al.* (2013b), adjusting Φ_S to optimize the posterior (thus affecting the prior over the structure of the ITG $P\left(\Phi_S|\Phi_G\right)$ and the conditional probability of the data given the complete model $P\left(D|\Phi_G,\Phi_S,\theta_\Phi\right)$). The latter adjusts the model parameters θ_Φ to optimize the posterior (thus affecting the prior over the parameters $P\left(\theta_\Phi|\Phi_S,\Phi_G\right)$ and again $P\left(D|\Phi_G,\Phi_S,\theta_\Phi\right)$), assuming the model structure Φ_S to be fixed, as well as Φ_G which remains fixed as bracketing inversion transduction grammars.

The prior is a symmetric Dirichlet distribution over rule right-hand sides given rule left-hand sides. To get the conditional, we have to biparse the training data, and to maximize it, we perform expectation maximization (Dempster *et al.*, 1977), as specified for ITGs by (Wu, 1995) with the caveat that we increase all the fractional counts by one before normalizing. The biparsing is done with our in-house implementation of the cubic time biparsing algorithm described in Saers *et al.* (2009), with a beam width of 100.

5 Experimental setup

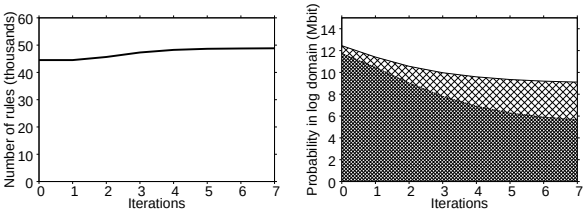
To test the viability of the idea of starting with a very specific ITG consisting of long rules, and iteratively segmenting the rules to induce a more general ITG under a MAP or MDL objective, we have implemented the steps detailed in Sections 3 and 4; in this section we will describe in greater detail the exact experimental conditions of our empirical study.

The initial BITG is set to have the relative frequency of the unique sentences as the probability of the corresponding rules. This parametrization is identical to what we would have arrived at with any other initialization that was subsequently optimized with expectation maximization; in this case it is possible to jump straight to the optimum. The generalization step requires biparsing in order to estimate the posterior—we use the cubic time biparsing algorithm described in Saers *et al.* (2009), with a beam width of 100. In the parameter optimization step, we use the exact same biparser.

As training data, we use the IWSLT07 Chinese–English data set (Fordyce, 2007), which contains 46,867 sentence pairs of training data, 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 which tries to clump characters together into more "word-like" sequences (Wu, 1999).

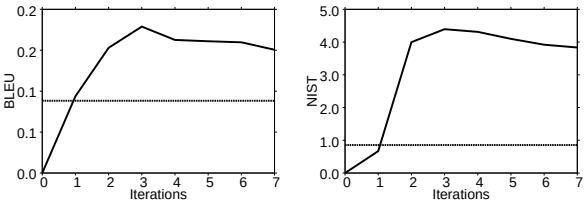
After each induction iteration there is a fully-functional grammar that we can test as a translation system. For this, we use our in-house ITG decoder, which uses a CKY-style parsing algorithm (Cocke, 1969; Kasami, 1965; Younger, 1967) with cube pruning (Chiang, 2007) to integrate the language model scores. We use SRILM (Stolcke, 2002) to train a trigram language model on the English side of the training data.

To evaluate the resulting translations, we use BLEU (Papineni *et al.*, 2002), and NIST (Doddington, 2002), and compare the results against our bottom-up oriented chunking ITG induction approach (Saers *et al.*, 2012).



(a) Rule count (b) Impact of model structure

Figure 1: Number of rules (a), and the impact of changes in the model structure (b) during the structure induction phase. The change in model structure is broken down into the model prior (bottom) and data given model (top).



(a) BLEU scores (b) NIST scores

Figure 2: Variations in translation quality over different iterations. The dotted line represents the baseline (Saers *et al.*, 2012).

6 Results

We need to evaluate (a) how well the introduced induction works, and (b) how well the resulting model works. How well the induction works can be seen in Figure 1, which shows how the model changes over iterations. Although the number of rules rises, the model structure prior becomes more probable, indicating that smaller rules are being learned. The improvements in the prior fully makes up for the loss in the probability of the data given the model, which indicates that we are indeed generalizing successfully. The translation quality of the resulting model is found in Table 1 and Figure 2, which show how the translation quality changes as measured by two automatic quality metrics (Papineni *et al.*, 2002; Doddington, 2002). It is clear that the maximum *a posteriori* probability objective pushes the top-down learning approach far past the maximum likelihood objective of the bottom-up chunking learning approach, which is the baseline we are comparing against.

7 Conclusions

We have introduced a minimalist model for unsupervised Bayesian induction of parsimonious phrasal ITGs, and shown that iteratively splitting existing rules into smaller rules driven by a maximum a posteriori probability objective is superior to iteratively chunking atomic rules into longer rules driven by a maximum likelihood objective. A novel top-down segmenting search strategy allows for efficient prediction of changes in posterior probability of the data given changes in the model—a key ingredient for MAP training. Decoding is done directly with induced transduction grammars, a more "pure" evaluation methodology than embedding them within many other heuristic components that obscure induction characteristics. This provides an obvious foundation for generalization to more general transduction grammars.

Table 1: Translation results of the baseline, the initial model and the model after n interations.

System	NIST	BLEU
baseline	0.8554	8.83
initial	0.0000	0.00
iteration 1	0.6686	9.38
iteration 2	3.9976	15.30
iteration 3	**4.3928**	**17.89**
iteration 4	4.3122	16.26
iteration 5	4.0981	16.10
iteration 6	3.9191	15.97
iteration 7	3.8338	15.06

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

Phil Blunsom and Trevor Cohn. "Inducing synchronous grammars with slice sampling." *NAACL HLT 2010*, 238–241. Los Angeles, CA, Jun 2010.

Phil Blunsom, Trevor Cohn, Chris Dyer, and Miles Osborne. "A gibbs sampler for phrasal synchronous grammar induction." *ACL-IJCNLP 2009*, 782–790. Suntec, Singapore, Aug 2009.

Phil Blunsom, Trevor Cohn, and Miles Osborne. "Bayesian synchronous grammar induction." *NIPS 21*. Vancouver, Canada, Dec 2008.

David Burkett, John Blitzer, and Dan Klein. "Joint parsing and alignment with weakly synchronized grammars." *NAACL HLT 2010*, 127–135. Los Angeles, CA, Jun 2010.

Stanley F. Chen. "Bayesian grammar induction for language modeling." *ACL 95*, 228–235. Cambridge, MA, Jun 1995.

Colin Cherry and Dekang Lin. "Inversion transduction grammar for joint phrasal translation modeling." *SSST*, 17–24. Rochester, NY, Apr 2007.

David Chiang. "A hierarchical phrase-based model for statistical machine translation." *ACL-05*, 263–270. Ann Arbor, MI, Jun 2005.

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

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

Arthur Pentland Dempster, Nan M. Laird, and Donald Bruce Rubin. "Maximum likelihood from incomplete data via the em algorithm." *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

George Doddington. "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics." *HLT '02*, 138–145. San Diego, CA, 2002.

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

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. "Scalable inference and training of context-rich syntactic translation models." *COLING/ACL 2006*, 961–968. Sydney, Australia, Jul 2006.

Aria Haghighi, John Blitzer, John DeNero, and Dan Klein. "Better word alignments with supervised itg models." *ACL-IJCNLP 2009*, 923–931. Suntec, Singapore, Aug 2009.

Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. "Improving translation quality by discarding most of the phrasetable." *EMNLP-CoNLL 2007*, 967–975. Prague, Czech Republic, Jun 2007.

Tadao Kasami. "An efficient recognition and syntax analysis algorithm for context-free languages." *Tech. Rep. AFCRL-65-00143*, Air Force Cambridge Research Laboratory, 1965.

Philipp Koehn, Franz Joseph Och, and Daniel Marcu. "Statistical Phrase-Based Translation." *HLT-NAACL 2003*, vol. 1, 48–54. Edmonton, Canada, May/Jun 2003.

Graham Neubig, Taro Watanabe, Shinsuke Mori, and Tatsuya Kawahara. "Machine translation without words through substring alignment." *ACL 2012*, 165–174. Jeju Island, Korea, Jul 2012.

Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. "An unsupervised model for joint phrase alignment and extraction." *ACL HLT 2011*, 632–641. Portland, OR, Jun 2011.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. "BLEU: a method for automatic evaluation of machine translation." *ACL-02*, 311–318. Philadelphia, PA, Jul 2002.

Jason Riesa and Daniel Marcu. "Hierarchical search for word alignment." *ACL 2010*, 157–166. Uppsala, Sweden, Jul 2010.

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

Markus Saers, Karteek Addanki, and Dekai Wu. "From finite-state to inversion transductions: Toward unsupervised bilingual grammar induction." *COLING 2012*, 2325–2340. Mumbai, India, Dec 2012.

Markus Saers, Karteek Addanki, and Dekai Wu. "Combining top-down and bottom-up search for unsupervised induction of transduction grammars." *SSST-7*, 48–57. Atlanta, GA, Jun 2013a.

Markus Saers, Karteek Addanki, and Dekai Wu. "Iterative rule segmentation under minimum description length for unsupervised transduction grammar induction." Adrian-Horia Dediu, Carlos Martín-Vide, Ruslan Mitkov, and Bianca Truthe (eds.), *Statistical Language and Speech Processing, First International Conference, SLSP 2013*, Lecture Notes in Artificial Intelligence (LNAI). Tarragona, Spain: Springer, Jul 2013b.

Markus Saers, Karteek Addanki, and Dekai Wu. "Unsupervised transduction grammar induction via minimum description length." *HyTra*, 67–73. Sofia, Bulgaria, Aug 2013c.

Markus Saers, Joakim Nivre, and Dekai Wu. "Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm." *IWPT'09*, 29–32. Paris, France, Oct 2009.

Markus Saers, Joakim Nivre, and Dekai Wu. "Word alignment with stochastic bracketing linear inversion transduction grammar." *NAACL HLT 2010*, 341–344. Los Angeles, CA, Jun 2010.

Markus Saers and Dekai Wu. "Improving phrase-based translation via word alignments from Stochastic Inversion Transduction Grammars." *SSST-3*, 28–36. Boulder, CO, Jun 2009.

Markus Saers and Dekai Wu. "Principled induction of phrasal bilexica." *EAMT-2011*, 313–320. Leuven, Belgium, May 2011.

Markus Saers, Dekai Wu, Chi kiu Lo, and Karteek Addanki. "Speech translation with grammar driven probabilistic phrasal bilexica extraction." *Interspeech 2011*, 2089–2092. 2011.

Claude Elwood Shannon. "A mathematical theory of communication." *The Bell System Technical Journal*, 27:379–423, 623–656, Jul, Oct 1948.

Ray J. Solomonoff. "A new method for discovering the grammars of phrase structure languages." *IFIP*, 285–289. 1959.

Andreas Stolcke. "SRILM – an extensible language modeling toolkit." *ICSLP2002 - INTERSPEECH 2002*, 901–904. Denver, CO, Sep 2002.

Andreas Stolcke and Stephen Omohundro. "Inducing probabilistic grammars by bayesian model merging." R. C. Carrasco and J. Oncina (eds.), *ICGI-94*, 106–118. Springer, 1994.

Juan Miguel Vilar. "Experiments using mar for aligning corpora." *ACL 2005 Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, 95–98. Ann Arbor, Jun 2005.

Juan Miguel Vilar and Enrique Vidal. "A recursive statistical translation model." *ACL 2005 Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*, 199–207. Ann Arbor, Jun 2005.

Dekai Wu. "Trainable coarse bilingual grammars for parallel text bracketing." *WVLC-3*, 69–81. Cambridge, MA, Jun 1995.

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

Zhibiao Wu. "LDC Chinese segmenter." 1999.

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

Hao Zhang, Chris Quirk, Robert C. Moore, and Daniel Gildea. "Bayesian learning of non-compositional phrases with synchronous parsing." *ACL-08: HLT*, 97–105. Columbus, OH, Jun 2008.