# Iterative Rule Segmentation under Minimum Description Length for Unsupervised Transduction Grammar Induction

Markus Saers, Karteek Addanki, and Dekai Wu

Human Language Technology Center
Dept. of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{masaers,vskaddanki,dekai}@cs.ust.hk

**Abstract.** We argue that for purely incremental unsupervised learning of phrasal inversion transduction grammars, a minimum description length driven, iterative top-down rule segmentation approach that is the polar opposite of Saers, Addanki, and Wu's previous 2012 bottom-up iterative rule chunking model yields significantly better translation accuracy and grammar parsimony. We still aim for unsupervised bilingual grammar induction such that training and testing are optimized upon the same exact underlying model—a basic principle of machine learning and statistical prediction that has become unduly ignored in statistical machine translation models of late, where most decoders are badly mismatched to the training assumptions. Our novel approach learns phrasal translations by recursively subsegmenting the training corpus, as opposed to our previous model—where we start with a token-based transduction grammar and iteratively build larger chunks. Moreover, the rule segmentation decisions in our approach are driven by a minimum description length objective, whereas the rule chunking decisions were driven by a maximum likelihood objective. We demonstrate empirically how this trades off maximum likelihood against model size, aiming for a more parsimonious grammar that escapes the perfect overfitting to the training data that we start out with, and gradually generalizes to previously unseen sentence translations so long as the model shrinks enough to warrant a looser fit to the training data. Experimental results show that our approach produces a significantly smaller and better model than the chunking-based approach.

## 1 Introduction

In this paper we argue that significantly better phrasal inversion transduction grammars, or ITGs [1], can be obtained through unsupervised learning with a minimum description length, or MDL, driven, iterative top-down rule segmentation approach, than with the polar opposite: a maximum likelihood driven, iterative bottom-up rule chunking model (as described in our previous work [2]). The size of the search space—all possible combinations of phrasal transduction

A.-H. Dediu et al. (Eds.): SLSP 2013, LNAI 7978, pp. 224–235, 2013.

rules—is so huge that any help we can get in navigating it is welcome; choosing a top-down rather than bottom-up search strategy gives a smaller initial set of rules, and generating new candidates by segmenting (linear in the length of the rule) rather than chunking (quadratic in the size of the rule set) is much more efficient. Given this relationship between the size and number of rules on the one hand, and the search complexity on the other, using minimum description length as learning objective makes a lot of sense: it is designed to explicitly factor in the size of the model, and to keep it parsimonious, as opposed to the maximum likelihood objective, which has no mechanism for rewarding parsimony. We show empirically that the proposed search strategy gives better translation accuracy with a smaller model than our previous model [2].

Our approach also represents a new attack on the problem suffered by most current SMT approaches of learning phrase translations: they require enormous amounts of run-time memory, contain a high degree of redundancy, and fails to provide an obvious basis for generalization to abstract translation schemas. In particular, phrasal SMT models such as [3] and [4] often search for candidate translation segments and transduction rules by committing to a word alignment based on very different modelling assumptions [5, 6], and heuristically derive lexical segment translations [7]. In fact, it is possible to improve the performance by tossing away most of the learned segmental translations [8]. In contrast, we adopt a more "pure" methodology for evaluating transduction grammar induction than typical system building papers. Instead of embedding our learned ITG in the midst of many other heuristic components for the sake of a short term boost in BLEU, we focus on scientifically understanding the behavior of pure MDL-based search for phrasal translations, divorced from the effect of other variables, even though BLEU is naturally much lower this way. The common practice of plugging some aspect of a learned ITG into either (a) a long pipeline of training heuristics and/or (b) an existing decoder that has been patched up to compensate for earlier modeling mistakes—as we and others have done before, see for example [9–20]—obscures the specific traits of the induced grammar. Instead, we directly use our learned ITG in translation mode (any transduction grammar also represents a decoder when parsing with the input sentence as a hard constraint) which allows us to see exactly which aspects of correct translation the transduction rules have captured.

When the structure of an ITG is induced without supervision, we and others have so far assumed that smaller rules should get clumped together into larger rules. This is a natural way to search, since maximum likelihood (ML) tends to improve with longer rules, which is typically balanced with Bayesian priors [10]. Bayesian priors are also used in Gibbs sampling [11, 15], as well as other non-parametric learning methods [19, 20]. All of the above evaluate their models by feeding them into mismatched decoders, making it hard to evaluate how accurate the learned models themselves were. In this work we take a radically different approach, and start with the longest rules possible and attempt to segment them into shorter rules iteratively. This makes ML useless, since our initial model maximizes it. Instead, we balance the ML objective with a minimum

description length (MDL) objective, which let us escape the initial ML optimum by rewarding *model parsimony*. The MDL objective is very similar to a Bayesian prior over the structure.

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 [21], 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.

The MDL objective that we will be using to drive the learning has been used before, but to induce monolingual grammars. [22] uses a method similar to MDL called *Bayesian model merging* to learn the structure of hidden Markov models as well as stochastic context-free grammars (SCFGs). 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. [23] 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 [24]. Our work in this paper is markedly different to even the previous NLP work in that (a) we induce an inversion transduction grammar rather than a monolingual grammar, and (b) we focus on learning the terminal segments rather than the nonterminal categories. We would, of course, like to learn nonterminal categories as well, but will defer that to future work.

We start by taking a closer look at the minimum description length principle (Section 2). Then we describe how the top-down ITG is initialized (Section 3) and generalized (Section 4). After that we outline the empirical experiment (Section 5) and the results (Section 6) before offering some conclusions (Section 7).

## 2 The Minimum Description Length Principle

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 [25, 26]. 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 *less certain* 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:

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

where $\Phi$ is the model and $D$ is the data. Note the clear parallel to probabilities that have been moved into the logarithmic domain, but keep in mind that lengths do not necessarily have a probabilistic interpretation, whereas probabilities always have a length-in-bits interpretation [27].

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—we call this 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 model, and the first would make the description of the data longer. That is: generalization raises the first term and lowers the second term in Equation 1. A good generalization will lower the total MDL, whereas a poor one will raise it.

### 2.1 Measuring the Length of a Corpus

The information-theoretic view of the problem gives a hint at the operationalization of description length of a corpus given a grammar. [27] stipulates that we can get a lower bound on the number of bits required to encode a specific outcome of a random variable. We thus define description length of the corpus given the grammar as:

$$\mathrm{DL}\left(D|\Phi\right) = -\lg P\left(D|\Phi\right)$$

### 2.2 Measuring the Length of a Transduction Grammar

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 measures; this section describes how we do this.

To serialize an ITG, we first need to determine the alphabet that the message will be written in. We obviously 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 rule begins; 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. What we end up with is 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, **biterminals**—pairs of $L_0$-terminals and $L_1$-terminals that model translation equivalences. The serialized form of a grammar will be the serialized form of all rules concatenated.

Consider the following toy grammar:

$$S \to A,\ A \to \langle AA \rangle,\ A \to [AA]\,,\ A \to \text{have/yǒu},\ A \to \text{yes/yǒu},\ A \to \text{yes/shì}$$

Its serialized form would be: $[]SA\langle\rangle AAA[]AAA[]A\text{haveyou}[]A\text{yesyou}[]A\text{yessh}$. Now that we have a message made up of discrete symbols, 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 $-\lg\left(\frac{1}{N}\right)$ bits to encode (where $N$ is the number of different symbols—the type count). The above example grammar 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 the ITG

Rather than starting out with a fairly 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 grammar conforms to the normal form and the start symbol always rewrites to precisely one nonterminal symbol. This does violate the minimum description length principle, as the introduction of new symbols, by definition, makes the description of the model longer, but conforming to the normal form of inversion transduction grammars was deemed more important than strictly minimizing the description length. Our initial grammar thus looks like this:

$$S \to A, \quad A \to e_{0..T_0}/f_{0..V_0}, \quad A \to e_{0..T_1}/f_{0..V_1}, \quad ..., \quad A \to e_{0..T_N}/f_{0..V_N}$$

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 $i^{\text{th}}$ output sentence (making $e_{0..T_i}$ the $i^{\text{th}}$ output sentence), and $V_i$ is the length of the $i^{\text{th}}$ input sentence (making $f_{0..V_i}$ the $i^{\text{th}}$ input sentence).

## 4 Generalizing the ITG

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; when we do this, we move some of the probability mass which was concentrated in the training data out to other data that are still unseen—the very definition of generalization. The over all 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:

$$A \rightarrow \text{five thousand yen is my limit/wǒ zùi dūo chū wǔ qīan rì yúan}$$

This rule can be split into three rules:

$$A \rightarrow \langle AA \rangle$$
$$A \rightarrow \text{five thousand yen/ wǔ qīan rì yúan}$$
$$A \rightarrow \text{is my limit/wǒ zùi dūo chū}$$

Note that the original rule consists of 16 symbols (in our encoding scheme), whereas the new tree rules consists of $4 + 9 + 9 = 22$ symbols. Add to that the fact that three rules are likely to be less probable than one rule when parsing, which makes the training data longer as well. It is reasonable to believe that the bracketing inverted rule 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 terminal rule:

$$A \rightarrow \text{the total fare is five thousand yen/}$$
$$\text{zǒng gòng de fèi yòng shì wǔ qīan rì yúan}$$

This rule can also be split into three rules:

$$A \rightarrow [AA]$$
$$A \rightarrow \text{the total fare is/zǒng gòng de fèi yòng shì}$$
$$A \rightarrow \text{five thousand yen/wǔ qīan rì yúan}$$

```
G                   // The ITG
biaffixes_to_rules  // Maps biaffixes to the rules they occur in
do
   biaffixes_delta = []
   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)
   real_delta = 0
   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)
while real_delta < 0
```

**Fig. 1.** Pseudocode for the top-down search algorithm using description length as learning objective

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 "wǔ qīan rì yúan". 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 can be found in Figure 1. The pseudocode uses the methods `eval_dl`, `sort_by_delta` and `make_segmentations`. These methods evaluate the difference in description length, sorts candidates by these differences, and commits to a given set of candidates, respectively. To evaluate the description length of a proposed set of candidate segmentations, we need to calculate the difference in description length between the current model, and the model that would result from committing to the candidate segmentations:

$$\mathrm{DL}\left(\Phi', D\right) - \mathrm{DL}\left(\Phi, D\right) = \mathrm{DL}\left(D|\Phi'\right) - \mathrm{DL}\left(D|\Phi\right) + \mathrm{DL}\left(\Phi'\right) - \mathrm{DL}\left(\Phi\right)$$

The model lengths are trivial, as we merely have to encode the rules that are removed and inserted according to our encoding scheme and plug in the summed lengths in the above equation (making sure to add any one rule once only). This leaves the difference in data length, which is:

$$\mathrm{DL}\left(D|\Phi'\right) - \mathrm{DL}\left(D|\Phi\right) = -\lg \frac{P\left(D|\Phi'\right)}{P\left(D|\Phi\right)}$$

This lets us determine the probability through biparsing with the grammar being induced. Biparsing is, however, a very expensive operation, and we are making relatively small changes to the grammar, so we will further assume that we can estimate the description length difference in closed form based on the grammar parameters. 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 grammar parameters $\theta'$ will be identical to the old grammar parameters $\theta$, except that:

$$\begin{aligned}
\theta'_{r_0} &= 0 \\
\theta'_{r_1} &= \theta_{r_1} + \frac{1}{3}\theta_{r_0} \\
\theta'_{r_2} &= \theta_{r_2} + \frac{1}{3}\theta_{r_0} \\
\theta'_{r_3} &= \theta_{r_3} + \frac{1}{3}\theta_{r_0}
\end{aligned}$$

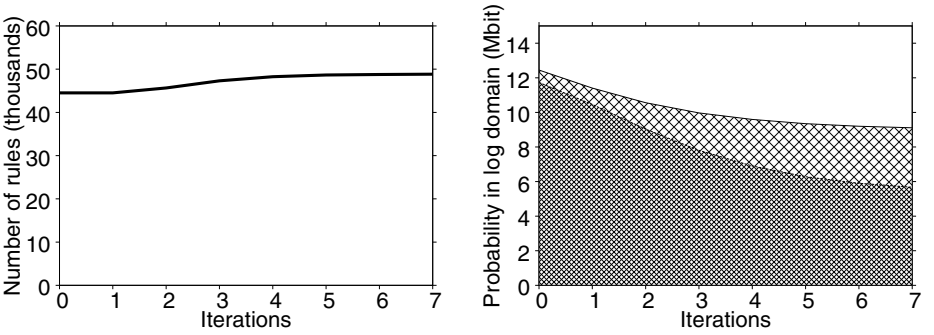
We estimate the probability of the corpus given this new parameters to be:

$$-\lg \frac{P\left(D|\Phi'\right)}{P\left(D|\Phi\right)} \approx -\lg \frac{\theta'_{r_1}\theta'_{r_2}\theta'_{r_3}}{\theta_{r_0}}$$

To generalize this to a set of rule segmentations, we construct the new parameters $\theta'$ to reflect all the changes in the set in a first pass, and then sum the differences in description length for all the rule segmentations with the new parameters in a second pass.

## 5 Experimental Setup

We have made the claim that iterative top-down segmentation guided by the objective of minimizing the description length is superior to iterative bottom-up chunking as a way of learning stochastic inversion transduction grammars in an unsupervised fashion. We have spent the paper so far outlining how this can be done in practice, and we are now about to show that the outlined method is indeed superior. To substantiate our claim, 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 the grammar iteratively a total of seven times (iteration 1–7), after which the changes are negligible. For each iteration we will record




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

**Fig. 2.** 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 size of the model (bottom) and the size of the data given the model (top).

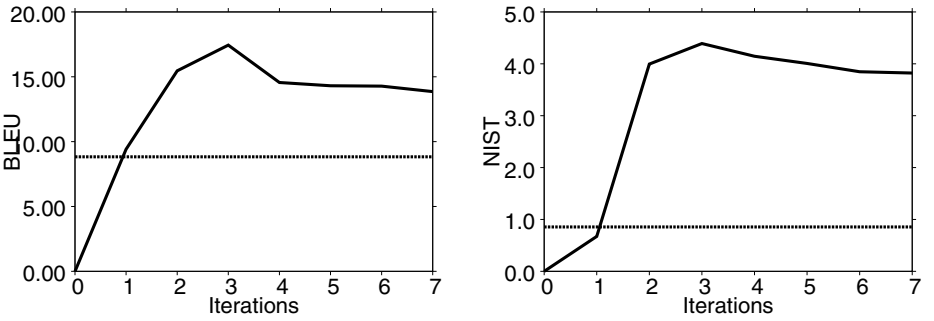
the change in description length and test the learned grammar. Each iteration requires us to biparse the training data to get the data probability of the data given the grammar the iteration starts with. We do this with our in-house implementation of the cubic time algorithm described in [28], with a beam width of 100.

As training data, we use the IWSLT07 Chinese–English data set [29], 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.

To test the learned grammar as a translation model, we first tune the grammar parameters to the training data using expectation maximization [30] and parse forests acquired with the above mentioned in-house 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 [31–33] and cube pruning [34] 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. We use SRILM [35] 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 [36], and NIST [37].

## 6 Results

We claimed that our iterative top-down segmentation guided by the minimum description length objective is superior to iterative bottom-up guided by likelihood for unsupervised induction of inversion transduction grammars; by superior we mean that it produces a smaller model which gives better translation quality, and in the previous section we outlined an experiment to verify this claim.




(a) BLEU scores (b) NIST scores

**Fig. 3.** Variations in translation quality over different iterations. The dotted line represents the baseline [2].

Figure 2 shows the size of our model during induction, both in terms of rule count and in terms of description length. The initial ITG is at iteration 0, where the vast majority of the size is taken up by the model (DL $(\Phi)$, bottom), and very little by the data (DL $(D|\Phi)$, top)—just as we predicted. The trend over the induction phase is a sharp decline 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 by [2] is close to 30 Mbits, and far off the chart.

Figure 3 shows the translation quality of our model as it learns. Here we see a sharp early rise, and then levelling off and even some decline. The main point to focus on is, however, that the second iteration puts us firmly past the results published in [2].

## 7 Conclusions

We have presented an unsupervised learning method for inversion transduction grammars that iteratively segments the training data in a top-down fashion driven by the objective to minimize description length. This contrasts to our previous work where the learning is conducted bottom-up through chunking towards a maximum likelihood objective. Our experiments show that our new approach is superior.

**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, FSGRF13EG28, 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

1. Wu, D.: Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. Computational Linguistics 23(3), 377–403 (1997)
2. Saers, M., Addanki, K., Wu, D.: From finite-state to inversion transductions: Toward unsupervised bilingual grammar induction. In: COLING 2012: Technical Papers, Mumbai, India, pp. 2325–2340 (December 2012)
3. Koehn, P., Och, F.J., Marcu, D.: Statistical Phrase-Based Translation. In: HLT/NAACL 2003, Edmonton, Canada, vol. 1 (May/June 2003)
4. Chiang, D.: A hierarchical phrase-based model for statistical machine translation. In: ACL 2005, Ann Arbor, Michigan, pp. 263–270 (June 2005)
5. Brown, P.F., Della Pietra, S.A., Della Pietra, V.J., Mercer, R.L.: The Mathematics of Machine Translation: Parameter estimation. CL 19(2) (1993)
6. Vogel, S., Ney, H., Tillmann, C.: HMM-based Word Alignment in Statistical Translation. In: COLING 1996, vol. 2, pp. 836–841 (1996)
7. Och, F.J., Ney, H.: A Systematic Comparison of Various Statistical Alignment Models. Computational Linguistics 29(1), 19–51 (2003)
8. Johnson, H., Martin, J., Foster, G., Kuhn, R.: Improving translation quality by discarding most of the phrasetable. In: EMNLP/CoNLL 2007, Prague, Czech Republic, pp. 967–975 (June 2007)
9. Cherry, C., Lin, D.: Inversion transduction grammar for joint phrasal translation modeling. In: SSST-1, Rochester, New York, pp. 17–24 (April 2007)
10. Zhang, H., Quirk, C., Moore, R.C., Gildea, D.: Bayesian learning of non-compositional phrases with synchronous parsing. In: ACL/HLT 2008, Columbus, Ohio, pp. 97–105 (June 2008)
11. Blunsom, P., Cohn, T., Dyer, C., Osborne, M.: A Gibbs sampler for phrasal synchronous grammar induction. In: ACL/IJCNLP 2009, Singapore (August 2009)
12. Haghighi, A., Blitzer, J., DeNero, J., Klein, D.: Better word alignments with supervised itg models. In: ACL/IJCNLP'09, Suntec, Singapore (August 2009)
13. Saers, M., Wu, D.: Improving phrase-based translation via word alignments from stochastic inversion transduction grammars. In: SSST-3, Boulder, CO (June 2009)
14. Saers, M., Wu, D.: Principled induction of phrasal bilexica. In: EAMT 2011, Leuven, Belgium, pp. 313–320 (May 2011)
15. Blunsom, P., Cohn, T.: Inducing synchronous grammars with slice sampling. In: HLT/NAACL 2010, Los Angeles, California, pp. 238–241 (June 2010)
16. Burkett, D., Blitzer, J., Klein, D.: Joint parsing and alignment with weakly synchronized grammars. In: HLT/NAACL 2010, Los Angeles, CA (June 2010)
17. Riesa, J., Marcu, D.: Hierarchical search for word alignment. In: ACL 2010, Uppsala, Sweden, pp. 157–166 (July 2010)
18. Saers, M., Nivre, J., Wu, D.: Word alignment with stochastic bracketing linear inversion transduction grammar. In: HLT/NAACL 2010, Los Angeles, California, pp. 341–344 (June 2010)
19. Neubig, G., Watanabe, T., Sumita, E., Mori, S., Kawahara, T.: An unsupervised model for joint phrase alignment and extraction. In: ACL/HLT 2011, Portland, Oregon (June 2011)

20. Neubig, G., Watanabe, T., Mori, S., Kawahara, T.: Machine translation without words through substring alignment. In: ACL 2012, Jeju, Korea (July 2012)
21. Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., Thayer, I.: Scalable inference and training of context-rich syntactic translation models. In: COLING/ACL 2006, Sydney, Australia (July 2006)
22. Stolcke, A., Omohundro, S.: Inducing probabilistic grammars by bayesian model merging. In: Carrasco, R.C., Oncina, J. (eds.) ICGI 1994. LNCS, vol. 862, pp. 106–118. Springer, Heidelberg (1994)
23. Grünwald, P.: A minimum description length approach to grammar inference in symbolic. In: Wermter, S., Scheler, G., Riloff, E. (eds.) IJCAI-WS 1995. LNCS (LNAI), vol. 1040, pp. 203–216. Springer, Heidelberg (1996)
24. Si, Z., Pei, M., Yao, B., Zhu, S.-C.: Unsupervised learning of event and-or grammar and semantics from video. In: IEEE ICCV 2011 (November 2011)
25. Solomonoff, R.J.: A new method for discovering the grammars of phrase structure languages. In: IFIP Congress, pp. 285–289 (1959)
26. Rissanen, J.: A universal prior for integers and estimation by minimum description length. The Annals of Statistics 11(2), 416–431 (1983)
27. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27, 379–423, 623–656 (1948)
28. Saers, M., Nivre, J., Wu, D.: Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In: IWPT 2009, Paris, France, pp. 29–32 (October 2009)
29. Fordyce, C.S.: Overview of the IWSLT 2007 evaluation campaign. In: IWSLT 2007 (2007)
30. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) 39(1), 1–38 (1977)
31. Cocke, J.: Programming languages and their compilers: Preliminary notes. Courant Institute of Mathematical Sciences, New York University (1969)
32. Kasami, T.: An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-00143, Air Force Cambridge Research Laboratory (1965)
33. Younger, D.H.: Recognition and parsing of context-free languages in time $n^3$. Information and Control 10(2), 189–208 (1967)
34. Chiang, D.: Hierarchical phrase-based translation. CL 33(2) (2007)
35. Stolcke, A.: SRILM – an extensible language modeling toolkit. In: ICSLP 2002, Denver, Colorado, pp. 901–904 (September 2002)
36. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: ACL 2002, Philadelphia, PA (July 2002)
37. Doddington, G.: Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In: HLT 2002, San Diego, California, pp. 138–145 (2002)