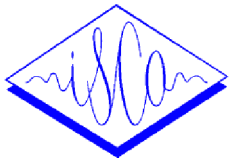
Proceedings of IWSLT 2009, Tokyo - Japan

ISCA Archive
http://www.isca-speech.org/archive



A Unified Framework for Phrase-Based, Hierarchical, and Syntax-Based Statistical Machine Translation

Hieu Hoang, Philipp Koehn, and Adam Lopez

School of Informatics
University of Edinburgh
{hhoang,pkoehn,alopez}@inf.ed.ac.uk

Abstract

Despite many differences between phrase-based, hierarchical, and syntax-based translation models, their training and testing pipelines are strikingly similar. Drawing on this fact, we extend the Moses toolkit to implement hierarchical and syntactic models, making it the first open source toolkit with end-to-end support for all three of these popular models in a single package. This extension substantially lowers the barrier to entry for machine translation research across multiple models.

1. Introduction

Over the last years, statistical machine translation research has produced rapid progress. After phrase-based models succeeded the original word-based approach, new research has focussed on hierarchical and syntax-based models that take the recursive nature of language into account and incorporate varying levels of linguistic annotation.

In this paper, we illustrate the similarities of these systems at all stages of the translation pipeline: modeling (§2), training (§3), and decoding (§4). We describe our implementation of all these models in a common statistical machine translation system, Moses (§5). Finally, we present a comparison of the baseline systems on German-English translation (§6).

2. Models

A naïve view of translation may describe the task as the mapping of words from one language into another, with some reordering. This notion underpins the original statistical machine translation models proposed by the IBM Candide project [1]. However, occasionally words have to be inserted and deleted without clear lexical correspondence on the other side, and words do not always map one-to-one. As a consequence, the word-based models proposed by IBM were burden with additional complexities such as word fertilities and NULL word generation.

2.1. Phrase-Based Models

Over the last decade, word-based models have been all but abandoned (they still live on in word alignment methods), and replaced by an even simpler view of language. Phrase-based models view translation of small text chunks, again with some reordering [2, 3]. The complexities of many-to-many translation, insertion and deletion are hidden within the phrasal translation table.

To give examples, phrase-based models may include rules such as

assumes || *geht davon aus, dass*
with regard to || *bezüglich*
translation system || *Übersetzungssystem*

Implementations of such phrase-based models of translation have been shown to outperform all existing translation systems for some language pairs [4]. Currently most prominent is the online translation service of Google that follows this approach.[1]

2.2. Hierarchical Phrase-Based Models

However, phrase-based methods fail to capture to capture the essence of many language pairs [5]. One of the reasons is that reordering cannot always be reduced to the reordering of atom phrase units.

Consider the mapping of the following sentence pair fragment:

[1] http://translate.google.com/

take the proposal into account
berücksichtigt den Vorschlag

The English phrasal verb *take into account* wraps around its object *the proposal*. Hierarchical phrase-based models [6] extend the notion of phrase mapping to allow rules such as

take X_1 *into account* ‖ *berücksichtigt* X_1
must explain X_1 ‖ *muss* X_1 *erklären*
either X_1 *or* X_2 ‖ *entweder* X_1 *oder* X_2

Such translation rules may be formalized as a synchronous context free grammar, where the non-terminal X matches any constituent, and nonterminals with the same coindexes (e.g. X_1) are recursively translated by a single rule. Such a formalism reflects one of the major insights of linguistics: Language is recursive and all modern theories of language use recursive structures.

2.3. Syntax-Based Models

The move towards grammar formalism to represent translation models allows the extension of such formalism with linguistic annotations. The generic non-terminal X allows for many nonsensical substitutions in translations, so we may instead restrain these with explicit linguistic categories:

take NP *into account* ‖ *berücksichtigt* NP
must explain NP ‖ *muss* NP *erklären*
either S_1 *or* S_2 ‖ *entweder* S_1 *oder* S_2
either NP_1 *or* NP_2 ‖ *entweder* NP_1 *oder* NP_2

There are many ways to add linguistic annotation to translation rules. Different grammar formalism offer different sets of non-terminals. Annotation may be added at the source or the target or both, Head relationships may provide additional assistance in translation. Synchronous context-free grammars may also require purely non-lexical rules that only consist of non-terminals.

Nevertheless, let us stress that all the presented models reduce translation to the mapping of small chunks of text.

3. Training

The basic notion of statistical machine translation is to learn translation from the statistics over actual translation as it is manifest in a translation corpus. When translating a new sentence, we would like to construct a translation that has the strongest empirical evidence.

The research questions evolve around how to slice up the evidence into manageable units and how to weight their relative importance. There are obvious differences between the three models that we presented in the previous section, but there are also overwhelming similarities.

Consider Figure 1. The training pipeline for the three models is almost identical. Syntax-based models require the additional step of syntactic annotation of the training data. The main difference is in rule extraction, but even here the same method with some additional steps for some of the models are applies:

- Extract all phrase pairs consistent with the word alignment. In syntax-based models these have to correspond to syntactic constituents.
- In hierarchical and syntax-based models: Find sub-phrases and replace them with non-terminals. In hierarchical models the non-terminal is X, in syntactic models the non-terminal is taken from the syntax tree.
- Store all extracted phrase pairs and rules for scoring.

To provide one empirical fact to support this argument: The adaptation of the originally purely phrase-based training process in Moses to hierarchical and syntax-based models took less than one month of work.

Many syntax-based models relax the requirement that phrases have to correspond to syntactic constituents. For instance, in one of the best-performing models translation units may correspond to syntactic *treelets* (tree fragments), permitting reordering at a scope larger than that of a single constituent and its immediate children [7]. Also, spans that only match a sequence of constituents or incomplete constituents may be labeled with complex tags such as DET+ADJ or NP/N [8]. Note that these are manipulations of the syntax trees that do not change in any way the rule extraction method.

There are many refinements to the the rule extraction method. Limits may be imposed to span sizes as well as number of words and non-terminals. Fractional counts may be for rules extracted from the same spans. Only minimal rules may be extracted to explain a sentence pair. Smoothing counts may be done using Good Turing discounting or other methods.

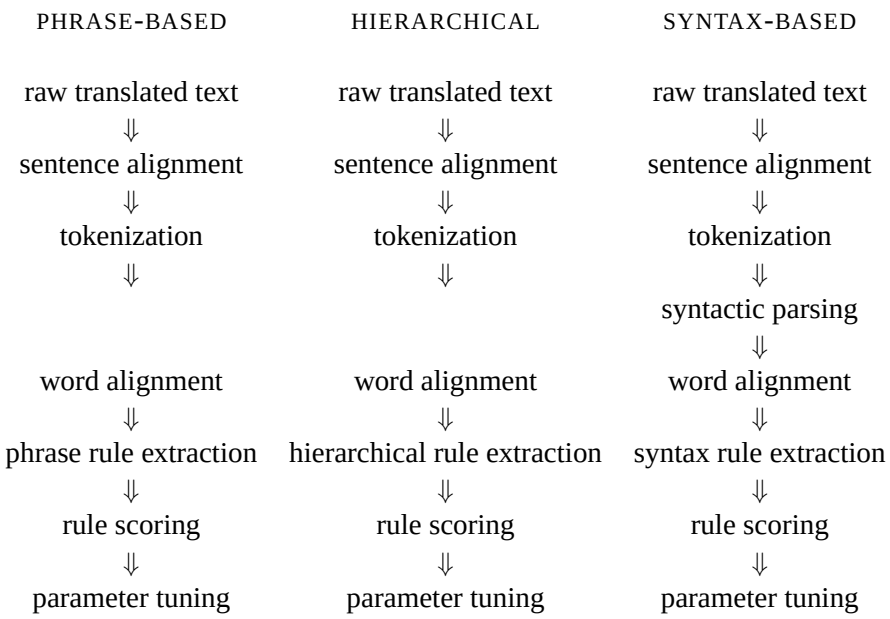


Figure 1: **Training pipelines:** Note that some of the steps are not just very similar, but identical. For instance, the parameter tuning step may the same method that is agnostic about how models generate and score translation candidates.

4. Decoding

Decoding is the process of finding for an input sentence the most probable translation according to out models. Since we will in almost all cases have not seen the sentence before in our training data, we have to break it up into smaller units, for which we have sufficient statistical evidence. Each of the units corresponds to a grammar rule, and the task of the decoding algorithm is to piece together these rules for the optimal sentence translation.

The decoding process is complicated by how the units interact with each other. Reordering models in phrase-based decoding consider the input position of neighboring output phrases. But more severely, n-gram language models tie together translated words that were generated by several rules, so it is not possible to view sentence translation simply as the independent combination of the applied translation rules. In other words, we cannot simply search for the most probable rules that apply to a sentence, but we have to take a number of different scoring functions into account.

There is a fundamental difference when decoding phrase-based models on the one hand, and hierarchical or syntax-based models on the other hand. Phrase-based decoding may proceed sequentially, by building the translation from left to right. This is not easily possible with the other models, since hierarchical rules require the insertion of phrases within other (gapped) phrases. Instead, typically, a chart parsing algorithm is used, which builds up the translation bottom-up by parsing with the source side of the synchronous grammar, covering ever larger spans of the input sentence (that do not have to start at the beginning of the sentence).

Nevertheless, the principles and major components of sequential and chart decoding are the same. The sentence is constructed step-by-step and all component scores are computed immediately. Due to the constraints imposed by the language model, the translation is build in contiguous sequences of words. Each partial translation (or hypothesis) is built by applying a translation rule to one or more already constructed hypotheses. Since the number of hypotheses is exploding, they are organized in stacks which are pruned.

4.1. Hypotheses

We build a sentence translation step-by-step, by applying a translation rule at a time. Each such step results in a partial translation, which we call hypothesis. A new hypothesis is formed by combining one or more hypotheses with a translation rule.

In phrase-based decoding, hypotheses are expanded

by covering additional source words and adding an output phrase.

In hierarchical decoding, more than one hypotheses may be combined by a translation rule. For instance, when appying the rule

$$\textit{either } \mathrm{X}_1 \textit{ or } \mathrm{X}_2 \parallel \textit{entweder } \mathrm{X}_1 \textit{ oder } \mathrm{X}_2$$

we combine the two hypotheses that are matched by X_1 and X_2.

Note that we can only apply this rule, if we have already translations for X_1 and X_2. In other words, we first find translations for smaller spans, and then use these translations in hierarchical rules to cover larger spans.

4.2. Incremental Scoring

When building a new hypothesis, we compute all component scores as much as possible. Some of the component scores are partial estimates, since they require information about future expansions.

Consider the case of the language model. In phrase-based decoding, we compute the language model score for a partial translation from the start of the sentence to the last added word. This last word may lead to very bad language model scores further on (consider a period in the middle of a sentence), but we do not know this at this point.

Even worse, a hypothesis in hierarchical decoding often covers a span that does not start at the beginning of the sentence, so the language model cost for the initial words has to be a estimate, which will be revised once more words are added before it.

However, we do require that each hypothesis represents a contiguous sequence of output words and disallow the insertion of words in the middle later on. This requirement allows us to compute relatively realistic partial language model scores.

4.3. Dynamic Programming

While each application of a translation rule leads to a new hypothesis, we may also reduce the number of hypotheses by combining two hypotheses that are identical in their future search behavior.

In the simplest case, consider two hypotheses that cover the same input words and produced the same output words. They differ only in the translations rules that were applied, i.e. their derivation. For instance, they may have been constructed using shorter or longer phrase pairs. Any subsequent application of translation rules for one of the hypotheses may also be applied to the other, with identical subsequent scores. This is what we mean by identical future search behavior.

Since there is no gain in carrying out identical subsequent searches, we combine these two hypotheses. We may simply drop the worse-scoring hypotheses, but for some applications (e.g., k-best list generation) it is useful to keep a back-pointer from the surviving hypotheses to the path that led to its competitor.

Note that the matching criterion for combining hypotheses is future search behavior. This does not require identical output. Only some aspects of the output matter for future search. For instance an n-gram language model only looks back at the last $n-1$ words in future rule applications. So, in phrase based models, the language model only requires that two hypotheses match in their last $n-1$ words, or even less, if these $n-1$ words are a unknown history to the language model.

In chart decoding the partial output does not necessarily start at the beginning of the sentence, so we also need to consider the first $n-1$ words (or less). The reordering model in phrase-based decoding may introduce additional constraints, and so does any other scoring function that does not solely depend on a single translation rule application.

4.4. Search Graphs and Hypergraphs

A good way to visualize decoding is as search for the best path in a graph: the nodes of the graph are hypotheses (§4.1) and the edges of the graph are rule applications that extend a hypothesis to produce a new hypothesis. From each node, several transitions fan out, due to different translation rules. But several transitions may also fan in to a node due to dynamic programming.

A hypothesis, or state in the search graph, points back to its highest-probable path, but also alternative paths with lower probability. In practice, we store with each state information such as which foreign words have been covered so far, the partial translation constructed so far, and the model scores along with all underlying component scores. But this information may also be obtained by walking back the best possible path.

In chart decoding the transitions may originate from multiple hypotheses. This can visualized as a *hypergraph* [9, 10], a generalization of a graph in which an edge (called a hyperedge) may originate from multiple nodes (called tail nodes). The nodes of the hypergraph

correspond to hypotheses, while the hyperedges correspond to rule applications. Just as in the graph case, we can extract a best *hyperpath* that corresponds to a single set of rule applications.

Note that this is simply an extension of the case for phrase-based models, and indeed the graph generated by a phrase-based model is simply the special case of a hypergraph in which each hyperedge has only one tail node. The virtue of the hypergraph view is that, even though our models have superficially quite different structures, their search spaces can all be represented in the same way, making them amenable to a variety of hypergraph algorithms [11]. These algorithms generalize familiar graph algorithms, which are simply special cases of their hypergraph generalizations. With this in mind, most statistical translation systems can be viewed as implementations of a very small number of generic algorithms, in which the main difference is a model-specific logic [12].

4.5. Stacks

Viewing decoding as the task of finding the most probable path in a search graph or hypergraph, is one visualization of the problem. However, this graph is too large to efficiently construct even for relatively short sentences. We need to focus on the most promising part of the graph. To this end, we first group together comparable hypotheses in stacks, and then prune out the weaker ones.

There are many ways to define the stacks. In sequential decoding, we group together hypotheses that cover the same number of input words. In chart decoding, we group together hypotheses that cover the same input span.

More fine-grained groupings are possible: in sequential decoding we could distinguish between hypotheses that cover different input words [13], and in chart decoding for models with target side syntax, we may keep different stacks for different target-side labels. However, we want to avoid having too many stacks, and such additional distinctions may also be enforced by diversity requirements during pruning [14].

We prune bad hypotheses based on their incremental score so far. When comparing hypotheses that cover different input words, we also include a future cost estimate for the remaining words.

4.6. Search Strategy

The final decision of the decoding algorithm is: In which order do we generate the hypotheses?

The incremental scoring allows us to already compute fairly indicative scores for partial translation, so we broadly pursue a bottom-up decoding strategy, where we generate hypotheses of increasing input word coverage. This also allows efficient dynamic programming, since we generate all hypotheses for a particular span at one time, thus making it easy to find and process matching hypotheses.

We may process all hypotheses *in* one stack or *for* one stack at one time. In other words, either we may go through all hypotheses of one stack, for each find all applicable translation rules, and generate the resulting new hypotheses. Or, we look at a new empty stack, find all sets of hypotheses and translation rules that generate hypotheses in this stack, and then proceed to populate the stack.

The second strategy allows for a nice integration with pruning. If we sort the original hypotheses and the translation rules by their score, then we can focus on first generating the most promising new hypotheses. We may even stop this process early to avoid generating hypotheses. This latter strategy has become known as cube pruning [6], and it has been shown to be a generical algorithm applicable to both phrase-based and hierarchical models [10].

4.7. Decision Rule

Finally, we have to pick one of the hypotheses that cover the entire input sentence to output a translation. Most commonly, this is the hypothesis with the best score, but that is not the only choice.

There may be multiple ways to produce the same output. If our goal is to find the most probable translation given the input, then we should find all possible paths through the search graph that result in the same output and sum up their scores. Then, we output the translation with the highest score over all derivation. This is called max-translation decoding vs. max-derivation decoding [15].

But what if the best translation is an outlier? Given the uncertainty in all our models, we may prefer instead a different high-scoring translation that is most similar to the other high-scoring translations. This is the motivation for minimum Bayes risk decoding [16], which has been shown to often lead to better results.

5. Implementation

Based on our observations about the deep similarities between many popular translation models, we have substantially extended the functionality of the Moses toolkit [17], which previously supported only phrase-based models. In particular, our implementation includes a chart decoder that can handle general synchronous context-free grammars, including both hierarchical and syntax-based grammars.

Both phrase-based and hierarchical decoders implement cube pruning [6, 10] and minimum Bayes risk decoding [16].

Our training implementation also includes rule extraction for hierarchical [6] and syntax-based translation. The syntax-based rule extractor produces rules similar to the "composed rules" of [18]. The source code is freely available.[2]

This allows us to take advantage of the mature Moses infrastructure by retaining much of the existing components. Also, the development of a hierarchical system alongside a phrase-based system allows us to more easily and fairly compare and contrast the models.

Re-using and extending the existing Moses decoder reduces the amount of development required. As an illustration, the phrase-based decoder 24,000 lines of code. The more complex hierarchical and syntax extension added 10,000 lines to the codebase.

Some components in a phrase-based and hierarchical decoder are identical, for example, the purpose and application of language models do not change. The linear scoring model is also unchanged. Many of the peripheral modules needed by the decoder also remain unchanged.

Other components required straight-forward extension. This includes the vocabulary and phrase components which are extended to allow non-terminal symbols. The phrase model is also extended to allow non-terminal symbols which can cover multi-word spans.

Because the search spaces of phrase-based and hierarchical models differ, the implementations for search differ. Stack organization, partial translations (hypotheses) and search logic are separate for each translation model. However, we note the many similarity between each implementation which can be abstracted at a later date, following [12].

Consistent with the Moses heritage, the hierarchical decoder supports the factored representation of words. Also, as a generalization of word factors, non-terminals labels on both source and target multi-word spans are permitted, non-terminal words in translation rules are labelled with both the source and target labels, and the left-hand side of all rules have both source and target labels. Using this representation, input sentences to the decoder can be annotated with its corresponding parsed tree, dependency tree, or other span labels.

Also inherited from the original phrase-based Moses decoder is the ability to use multiple language models and alternative translation models.

Model	Rule Count	BLEU
phrase-based	6,246,548	13.0
hierarchical	59,079,493	12.9
target-syntax	2,291,400	12.5

Table 1: Comparison of English-German models using the WMT 2009 News Commentary training set

6. Experiments

Having a uniform framework for a wide range of models allows the comparison of different methods, while keeping most of the secondary conditions equal (e.g. language model, training data preparation, tuning, etc.). We report baseline results for three models: phrase-based, hierarchical, and a syntax-based model that uses syntax on the target side.

We trained systems using the News Commentary training set that was released by WMT 2009[3] for English to German translation. See Table 1 for statistics on rule table sizes and BLEU scores for the news-dev2009b test set.

Decoding for all the three models took about the same time, roughly 0.3 seconds per word. Decoding for hierarchical and syntax-based models is more complex, and we expect to achieve better results by tuning the search algorithm and using larger beam sizes.

The syntax-based model uses the BitPar parser for German [19]. Note that recent work has shown that state-of-the-art performance requires improvements to word alignment [20] and data preparation, which were not done for these experiments.

[2] `http://mosesdecoder.svn.sourceforge.net`

[3] `http://www.statmt.org/wmt09/`

7. Conclusions and Outlook

Our experiments illustrate that the hierarchical and syntactic models in Moses achieve similar quality to the phrase-based model, even though their implementation is less mature. We expect that their performance will continue to be improved by drawing on the substantial body of research in syntactic translation modeling over the last several years. In particular, we plan to extend the rule extraction to the produce syntax-augmented grammars [8], which have been shown to improve on both phrase-based and hierarchical models in some settings [21]. We also plan to implement optimizations for decoding with syntactic grammars, such as tree binarization [22].

8. Acknowledgements

This work was supported by the EuroMatrixPlus project funded by the European Commission (7th Framework Programme) and made use of the resources provided by the Edinburgh Compute and Data Facility.[4] The ECDF is partially supported by the eDIKT initiative.[5] This work was also supported in part under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022.

9. References

[1] P. F. Brown, S. A. D. Pietra, V. J. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Computational Linguistics*, vol. 19, no. 2, pp. 263–311, Jun 1993.

[2] F. J. Och and H. Ney, "The alignment template approach to machine translation," *Computational Linguistics*, vol. 30, no. 4, pp. 417–449, Jun 2004.

[3] P. Koehn, F. J. Och, and D. Marcu, "Statistical phrase-based translation," in *Proc. of HLT-NAACL*, May 2003, pp. 127–133.

[4] C. Callison-Burch, P. Koehn, C. Monz, and J. Schroeder, "Findings of the 2009 workshop on statistical machine translation," in *Proc. of WMT*, 2009.

[5] A. Birch, M. Osborne, and P. Koehn, "Predicting success in machine translation," in *Proc. of EMNLP*, 2008.

[6] D. Chiang, "Hierarchical phrase-based translation," *Computational Linguistics*, vol. 33, no. 2, pp. 201–228, 2007.

[7] 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 *Proc. of ACL-Coling*, 2006, pp. 961–968.

[8] A. Zollmann and A. Venugopal, "Syntax augmented machine translation via chart parsing," in *Proc. of WMT*, 2006.

[9] D. Klein and C. Manning, "Parsing and hypergraphs," in *Proc. of IWPT*, 2001.

[10] L. Huang and D. Chiang, "Forest rescoring: Faster decoding with integrated language models," in *Proc. of ACL*, Jun 2007, pp. 144–151.

[11] G. Gallo, G. Longo, and S. Pallottino, "Directed hypergraphs and applications," *Discrete Applied Mathematics*, vol. 42, no. 2, Apr 1993.

[12] A. Lopez, "Translation as weighted deduction," in *Proc. of EACL*, 2009.

[13] C. Tillman and H. Ney, "Word reordering and a dynamic programming beam search algorithm for statistical machine translation," *Computational Linguistics*, vol. 29, no. 1, pp. 98–133, Mar 2003.

[14] R. Zens and H. Ney, "Improvements in dynamic programming beam search for phrase-based statistical machine translation," in *Proc. of IWSLT*, 2008.

[15] A. Arun, C. Dyer, B. Haddow, P. Blunsom, A. Lopez, and P. Koehn, "Monte carlo inference and maximization for phrase-based translation," in *Proc. of CoNLL*, 2009.

[16] S. Kumar and W. Byrne, "Minimum bayes-risk decoding for statistical machine translation," in *Proc. of HLT-NAACL*, 2004.

[17] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open source

[4] `http://www.ecdf.ed.ac.uk/`
[5] `http://www.edikt.org.uk/`

toolkit for statistical machine translation," in *Proc. of ACL Demo and Poster Sessions*, Jun 2007, pp. 177–180.

[18] S. DeNeefe, K. Knight, W. Wang, and D. Marcu, "What can syntax-based MT learn from phrase-based MT?" in *Proc. of EMNLP-CoNLL*, Jun 2007, pp. 755–763.

[19] H. Schmid, "Efficient parsing of highly ambiguous context-free grammars with bit vectors," in *Proc. of COLING*, 2004.

[20] V. L. Fossum, K. Knight, and S. Abney, "Using syntax to improve word alignment precision for syntax-based machine translation," in *Proc. of WMT*, June 2008, pp. 44–52.

[21] A. Zollmann, A. Venugopal, F. Och, and J. Ponte, "A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT," in *Proc. of Coling*, 2008.

[22] W. Wang, K. Knight, and D. Marcu, "Binarizing syntax trees to improve syntax-based machine translation accuracy," in *Proc. of EMNLP-CoNLL*, 2007, pp. 746–754.