Efficient Extraction of Oracle-best Translations from Hypergraphs

Zhifei Li and **Sanjeev Khudanpur**
Center for Language and Speech Processing and Department of Computer Science
The Johns Hopkins University, Baltimore, MD 21218, USA
zhifei.work@gmail.com and khudanpur@jhu.edu

Abstract

Hypergraphs are used in several syntax-inspired methods of machine translation to compactly encode exponentially many translation hypotheses. The hypotheses closest to given *reference translations* therefore cannot be found via brute force, particularly for popular measures of closeness such as BLEU. We develop a dynamic program for extracting the so called *oracle-best hypothesis* from a hypergraph by viewing it as the problem of finding the most likely hypothesis under an n-gram *language model* trained from only the reference translations. We further identify and remove massive redundancies in the dynamic program state due to the sparsity of n-grams present in the reference translations, resulting in a very efficient program. We present runtime statistics for this program, and demonstrate successful application of the hypotheses thus found as the targets for discriminative training of translation system components.

1 Introduction

A *hypergraph*, as demonstrated by Huang and Chiang (2007), is a compact data-structure that can encode an exponential number of hypotheses generated by a regular phrase-based machine translation (MT) system (e.g., Koehn et al. (2003)) or a syntax-based MT system (e.g., Chiang (2007)). While the hypergraph represents a very large set of translations, it is quite possible that some desired translations (e.g., the reference translations) are not contained in the hypergraph, due to pruning or inherent deficiency of the translation model. In this case, one is often required to find the translation(s) in the hypergraph that are most similar to the desired translations, with similarity computed via some automatic metric such as BLEU (Papineni et al., 2002). Such maximally similar translations will be called *oracle-best* translations, and the process of extracting them *oracle extraction*. Oracle extraction is a nontrivial task because computing the similarity of any one hypothesis requires information scattered over many items in the hypergraph, and the exponentially large number of hypotheses makes a brute-force linear search intractable. Therefore, efficient algorithms that can exploit the structure of the hypergraph are required.

We present an efficient oracle extraction algorithm, which involves two key ideas. Firstly, we view the oracle extraction as a bottom-up *model scoring* process on a hypergraph, where the model is "trained" on the reference translation(s). This is similar to the algorithm proposed for a *lattice* by Dreyer et al. (2007). Their algorithm, however, requires maintaining a separate dynamic programming state for each distinguished sequence of "state" words and the number of such sequences can be huge, making the search very slow. Secondly, therefore, we present a novel look-ahead technique, called *equivalent oracle-state maintenance*, to merge multiple states that are equivalent for similarity computation. Our experiments show that the *equivalent oracle-state maintenance* technique significantly speeds up (more than 40 times) the oracle extraction.

Efficient oracle extraction has at least three important applications in machine translation.

Discriminative Training: In discriminative training, the objective is to tune the model parameters, e.g. weights of a perceptron model or conditional random field, such that the reference translations are preferred over competitors. However, the reference translations may not be reachable by the translation system, in which case the oracle-best hypotheses should be substituted in training.

Proceedings of NAACL HLT 2009: Short Papers, pages 9–12,
Boulder, Colorado, June 2009.

System Combination: In a typical *system combination* task, e.g. Rosti et al. (2007), each component system produces a set of translations, which are then grafted to form a confusion network. The confusion network is then rescored, often employing additional (language) models, to select the final translation. When measuring the goodness of a hypothesis in the confusion network, one requires its score under each component system. However, some translations in the confusion network may not be reachable by some component systems, in which case a system's score for the most similar reachable translation serves as a good approximation.

Multi-source Translation: In a multi-source translation task (Och and Ney, 2001) the input is given in *multiple* source languages. This leads to a situation analogous to system combination, except that each component translation system now corresponds to a specific source language.

2 Oracle Extraction on a Hypergraph

In this section, we present the oracle extraction algorithm: it extracts one or more translations in a hypergraph that have the maximum BLEU score[1] with respect to the corresponding reference translation(s).

The BLEU score of a hypothesis h relative to a reference r may be expressed in the log domain as,

$$\log \text{BLEU}(r,h) = \min\left[1-\frac{|r|}{|h|},0\right] + \sum_{n=1}^{4}\frac{1}{4}\log p_n.$$

The first component is the brevity penalty when $|h|<|r|$, while the second component corresponds to the geometric mean of n-gram precisions p_n (with clipping). While BLEU is normally defined at the corpus level, we use the sentence-level BLEU for the purpose of oracle extraction.

Two key ideas for extracting the oracle-best hypothesis from a hypergraph are presented next.

2.1 Oracle Extraction as Model Scoring

Our *first* key idea is to view the oracle extraction as a bottom-up *model scoring* process on the hypergraph. Specifically, we train a 4-gram language model (LM) on only the *reference translation(s)*, and use this LM as the *only* model to do a Viterbi search on the hypergraph to find the hypothesis that has the maximum (oracle) LM score. Essentially, the LM is simply a table memorizing the counts of n-grams found in the reference translation(s), and the LM score is the log-BLEU value (instead of log-probability, as in a regular LM). During the search, the dynamic programming (DP) states maintained at each item include the left- and right-side LM context, and the length of the partial translation. To compute the n-gram precisions p_n incrementally during the search, the algorithm also memorizes at each item a vector of maximum numbers of n-gram matches between the partial translation and the reference(s). Note however that the *oracle state* of an item (which decides the uniqueness of an item) depends only on the LM contexts and span lengths, not on this vector of n-gram match counts.

The computation of BLEU also requires the brevity penalty, but since there is no explicit alignment between the source and the reference(s), we cannot get the exact reference length $|r|$ at an intermediate item. The exact value of brevity penalty is thus not computable. We approximate the true reference length for an item with a *product* between the *length* of the source string spanned by that item and a *ratio* (which is between the lengths of the whole reference and the whole source sentence). Another approximation is that we do not consider the effect of clipping, since it is a global feature, making the strict computation intractable. This does not significantly affect the quality of the oracle-best hypothesis as shown later. Table 1 shows an example how the BLEU scores are computed in the hypergraph.

The process above may be used either in a first-stage decoding or a hypergraph-rescoring stage. In the latter case, if the hypergraph generated by the first-stage decoding does not have a set of DP states that is a superset of the DP states required for oracle extraction, we need to split the items of the first-stage hypergraph and create new items with sufficiently detailed states.

It is worth mentioning that if the hypergraph items contain the state information necessary for extracting the oracle-best hypothesis, it is straightforward to further extract the k-best hypotheses in the hypergraph (according to BLEU) for any $k \geq 1$ using the algorithm of Huang and Chiang (2005).

[1] We believe our method is general and can be extended to other metrics capturing only n-gram dependency and other compact data structures, e.g. lattices.

Item	$\|\mathbf{h}\|$	$\|\widetilde{\mathbf{r}}\|$	matches	$\log$ **BLEU**
Item A	5	6.2	(3, 2, 2, 1)	-0.82
Item B	10	9.8	(8, 7, 6, 5)	-0.27
Item C	17	18.3	(12, 10, 9, 6)	-0.62

Table 1: Example computation when items `A` and `B` are combined by a rule to produce item `C`. $|\widetilde{\mathbf{r}}|$ is the approximated reference length as described in the text.

2.2 Equivalent Oracle State Maintenance

The process above, while able to extract the oracle-best hypothesis from a hypergraph, is very slow due to the need to maintain a dedicated item for each *oracle state* (i.e., a combination of left-LM state, right-LM state, and hypothesis length). This is especially true if the baseline system uses a LM whose order is smaller than four, since we need to split the items in the original hypergraph into many sub-items during the search. To speed up the extraction, our *second* key idea is to maintain an *equivalent oracle state*.

Roughly speaking, instead of maintaining a different state for different language model words, we collapse them into a single state whenever it does not affect BLEU. For example, if we have two left-side LM states `a b c` and `a b d`, and we know that the reference(s) do not have any n-gram ending with them, then we can reduce them both to `a b` and ignore the last word. This is because the combination of neither left-side LM state (`a b c` or `a b d`) can contribute an n-gram match to the BLEU computation, regardless of which prefix in the hypergraph they combine with. Similarly, if we have two right-side LM states `a b c` and `d b c`, and if we know that the reference(s) do not have any n-gram starting with either, then we can ignore the first word and reduce them both to `b c`. We can continue this reduction recursively as shown in Figures 1 and 2, where IS-A-PREFIX(e_i^m) (or IS-A-SUFFIX(e_1^i)) checks if e_i^m (resp. e_1^i) is a prefix (suffix) of any n-gram in the reference translation(s). For BLEU, $1 \leq n \leq 4$.

This *equivalent oracle state maintenance* technique, in practice, dramatically reduces the number of distinct items preserved in the hypergraph for oracle extraction. To understand this, observe that if all hypotheses in the hypergraph together contain m unique n-grams, for any fixed n, then the total number of equivalent items takes a multiplicative factor that is $O(m^2)$ due to left- and right-side LM state

EQ-L-STATE (e_1^m)

```
els ← e_1^m
for i ← m to 1                    ▷ right to left
    if IS-A-SUFFIX(e_1^i)
        break                     ▷ stop reducing els
    else
        els ← e_1^{i-1}           ▷ reduce state
return els
```

Figure 1: Equivalent Left LM State Computation.

EQ-R-STATE (e_1^m)

```
ers ← e_1^m
for i ← 1 to m                    ▷ left to right
    if IS-A-PREFIX (e_i^m)
        break                     ▷ stop reducing ers
    else
        ers ← e_{i+1}^m           ▷ reduce state
return ers
```

Figure 2: Equivalent Right LM State Computation.

maintenance of Section 2.1. This multiplicative factor under the equivalent state maintenance above is $O(\tilde{m}^2)$, where $\tilde{m}$ is the number of unique n-grams in the reference translations. Clearly, $\tilde{m} \ll m$ by several orders of magnitude, leading to effectively much fewer items to process in the chart.

One may view this idea of maintaining equivalent states more generally as an *outside* look-ahead during bottom-up *inside* parsing. The look-ahead uses some external information, e.g. IS-A-SUFFIX$(\cdot)$, to *anticipate* whether maintaining a detailed state now will be of consequence later; if not then the inside parsing eliminates or collapses the state into a coarser state. The technique proposed by Li and Khudanpur (2008a) for decoding with large LMs is a special case of this general theme.

3 Experimental Results

We report experimental results on a Chinese to English task, for a system that is trained using a similar pipeline and data resource as in Chiang (2007).

3.1 Goodness of the Oracle-Best Translations

Table 2 reports the average speed (seconds/sentence) for oracle extraction. Hypergraphs were generated with a trigram LM and expanded on the fly for 4-gram BLEU computation.

Basic DP	Collapse equiv. states	speed-up
25.4 sec/sent	0.6 sec/sent	$\times$ 42

Table 2: Speed of oracle extraction from hypergraphs. The basic dynamic program (Sec. 2.1) improves significantly by collapsing *equivalent oracle states* (Sec. 2.2).

Table 3 reports the goodness of the oracle-best hypotheses on three standard data sets. The highest achievable BLEU score in a hypergraph is clearly much higher than in the 500-best *unique* strings. This shows that a hypergraph provides a much better basis, e.g., for reranking than an n-best list.

As mentioned in Section 2.1, we use several approximations in computing BLEU (e.g., no clipping and approximate reference length). To justify these approximations, we first extract 500-best unique *oracles* from the hypergraph, and then rerank the oracles based on the true sentence-level BLEU. The last row of Table 3 reports the reranked one-best oracle BLEU scores. Clearly, the approximations do not hurt the oracle BLEU very much.

Hypothesis space	MT'04	MT'05	MT'06
1-best (Baseline)	35.7	32.6	28.3
500-unique-best	44.0	41.2	35.1
Hypergraph	52.8	51.8	37.8
500-best oracles	**53.2**	**52.2**	**38.0**

Table 3: Baseline and oracle-best 4-gram BLEU scores with 4 references for NIST Chinese-English MT datasets.

3.2 Discriminative Hypergraph-Reranking

Oracle extraction is a critical component for hypergraph-based discriminative reranking, where millions of model parameters are discriminatively tuned to prefer the oracle-best hypotheses over others. Hypergraph-reranking in MT is similar to the forest-reranking for *monolingual parsing* (Huang, 2008). Moreover, once the oracle-best hypothesis is identified, discriminative models may be trained on *hypergraphs* in the same way as on n-*best* lists (cf e.g. Li and Khudanpur (2008b)). The results in Table 4 demonstrate that hypergraph-reranking with a discriminative LM or TM improves upon the baseline models on all three test sets. Jointly training both the LM and TM likely suffers from over-fitting.

Test Set	MT'04	MT'05	MT'06
Baseline	35.7	32.6	28.3
Discrim. LM	35.9	33.0	28.2
Discrim. TM	**36.1**	**33.2**	**28.7**
Discrim. TM+LM	36.0	33.1	28.6

Table 4: BLEU scores after discriminative hypergraph-reranking. Only the language model (LM) or the translation model (TM) or both (LM+TM) may be discriminatively trained to prefer the oracle-best hypotheses.

4 Conclusions

We have presented an efficient algorithm to extract the oracle-best translation hypothesis from a hypergraph. To this end, we introduced a novel technique for *equivalent oracle state maintenance*, which significantly speeds up the oracle extraction process. Our algorithm has clear applications in diverse tasks such as discriminative training, system combination and multi-source translation.

References

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

M. Dreyer, K. Hall, and S. Khudanpur. 2007. Comparing Reordering Constraints for SMT Using Efficient BLEU Oracle Computation. *In Proc. of SSST.*

L. Huang. 2008. Forest Reranking: Discriminative Parsing with Non-Local Features. *In Proc. of ACL.*

L. Huang and D. Chiang. 2005. Better k-best parsing. *In Proc. of IWPT*.

L. Huang and D. Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. *In Proc. of ACL.*

P. Koehn, F. J. Och, and D. Marcu.2003. Statistical phrase-based translation. *In Proc. of NAACL.*

Z. Li and S. Khudanpur. 2008a. A Scalable Decoder for Parsing-based Machine Translation with Equivalent Language Model State Maintenance. *In Proc. SSST*.

Z. Li and S. Khudanpur. 2008b. Large-scale Discriminative n-gram Language Models for Statistical Machine Translation. *In Proc. of AMTA*.

F. Och and H. Ney. 2001. Statistical multisource translation. *In Proc. MT Summit VIII*.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. *In Proc. of ACL.*

A.I. Rosti, S. Matsoukas, and R. Schwartz. 2007. Improved word-level system combination for machine translation. *In Proc. of ACL.*