A Structured Language Model for Incremental Tree-to-String Translation

Heng Yu
Institute of Computing Tech., CAS

Haitao Mi
T.J. Watson Research Center, IBM

Liang Huang
City University of New York (CUNY)

Qun Liu
ICT, CAS & Dublin City University

PROBLEM

Most MT systems rely on n-gram model to ensure the fluency, thus lacking syntax information on the target side. One solution is to use advanced translation models, like string-to-tree and tree-to-tree models, which unfortunately entail high complexity and difficult integration. A more promising way is to use structured language model, but it’s hard in two aspects:

1. significant growth of search space
2. incomparable hypothesis in original beams

CONTRIBUTIONS

We integrate a structured language model (SLM), a shift-reduce parser [1] in particular, into an incremental tree-to-string system [2] to capture target-side syntax with low computational cost. Our main contributions are:

1. effective SLM integration into MT decoding
2. introduce an efficient grouping and pruning method to handle the growing search space
3. gain an average improvement of 0.7 points in terms of (Ter-Bleu)/2 over the baseline

INTEGRATE SLM INTO INCREMENTAL TREE-TO-STRING DECODING

We use a shift-reduce (SR) parser to build target side dependency trees on the fly and calculate SLM score based on shift-reduce cost. The following figure shows the example of the integrator of the SLM into incremental tree-to-string decoding.

For each translation state (left col.), there is one or several parsing states (right col.). $S_t$ denotes a dependency parsing structure, and the shaded nodes are exposed roots of $S_t$. We can see that $S_t$ and $S'_t$ lead to different dependency trees, thus affecting the prediction of SLM.

INTEGRATION SCHEMES

We introduce three integration schemes:

1. Naive: adding parsing signatures into translation signatures
2. Best-parse [3]: keeping the best dependency structure for each translation
3. Grouping: regrouping hypothesis by reduce steps in each bin

The right figure shows our Grouping scheme: we group the hypothesis according to their reduce steps $(N(S))$, thus making them comparable in terms of dependency structure in each group. Then a two-phase pruning is performed: first keep the n-best hypothesis in each group, then select the top-k groups. This could enhance parsing accuracy by avoiding the risk of greedy search.

EXPERIMENTAL RESULTS

Our baseline is the incremental tree-to-string decoder [2]. Line 3 denotes using SLM to re-rank the output, it achieves small improvement due to limited search space. Line 4-6 show the performance of the integration schemes: naïve scheme hurts the system by restricting translation variance in each beam, whereas grouping scheme performs the best. This shows better parsing leads to better SLM which in turn improves translation.

A FUTURE DIRECTION

The structured language model introduced in our work only takes into account the target string, and ignores the reordering information in the source side. Thus, our future work seeks to incorporate more source side syntax information to guide the parsing of the target side, and tune a structured language model for both Bleu and parsing accuracy. Another potential work lies in the more efficient searching and pruning algorithms for integration.

REFERENCES

[1] Liang Huang, Kenji Sagae. Dynamic programming for linear-time incremental parsing. In ACL ’10

ACKNOWLEDGMENTS

Yu and Liu were supported by CAS Action Plan for the Development of Western China. Liu was partially supported by the Science Foundation Ireland at DCU. Huang was supported by DARPA funding, a Google Faculty Research Award, and a PSC-CUNY Award.