Merged bilingual trees based on Universal Dependencies in Machine Translation

David Mareček
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics, Charles University in Prague
Malostranské náměstí 25, 118 00 Prague, Czech Republic
marecek@ufal.mff.cuni.cz

Abstract

In this paper, we present our new experimental system of merging dependency representations of two parallel sentences into one dependency tree. All the inner nodes in dependency tree represent source-target pairs of words, the extra words are in form of leaf nodes. We use Universal Dependencies annotation style, in which the function words, whose usage often differs between languages, are annotated as leaves. The parallel treebank is parsed in minimally supervised way. Unaligned words are there automatically pushed to leaves. We present a simple translation system trained on such merged trees and evaluate it in WMT 2016 English-to-Czech and Czech-to-English translation task. Even though the model is so far very simple and no language model and word-reordering model were used, the Czech-to-English variant reached similar BLEU score as another established tree-based system.

1 Introduction

Tree-based machine translation systems (Chiang et al., 2005; Dušek et al., 2012; Sennrich and Haddow, 2015) are alternatives to the leading phrase-based MT systems (Koehn et al., 2007) and newly very progressive neural MT systems (Bahdanau et al., 2015). Our approach aims to produce bilingual dependency trees, in which both source and target sentences are encoded together. We adapt the Universal Dependencies annotation style (Nivre et al., 2016), in which the functional words[1] are in the leaf nodes and therefore the grammatical differences between the languages does not much affect the common dependency structure. We were partially inspired by the Stochastic inversion transduction grammars (Wu, 1997).

Our *merged dependency trees* are defined in Section 2. The data we use and necessary preprocessing is in Section 3. The merging algorithm itself, which merges two parallel sentences into one, is described in Section 4. Section 5 presents the minimally supervised parsing of the merged sentences. The experimental translation system using the merged trees is described in Section 6. Finally, we present our results (Section 7) and conclusions (Section 8).

2 Merged trees

We introduce "merged trees", where parallel sentences from two languages are represented by a single dependency tree. Each node of the tree consists of two word-forms and two POS tags. An example of such merged dependency tree is in Figure 3. If two words are translations of each other (1-1 alignment), they share one node labeled by both of them. Words that do not have their counterparts in the other sentence (1-0 or 0-1 alignment) are also represented by nodes and the missing counterpart is marked by label <empty>. All such nodes representing a single word without any counterpart are leaf nodes. This ensures that the merged tree structure can be simply divided into two monolingual trees, not including empty nodes. The two separated trees are also "internally" isomorfic, the only differences are in leaves.

The annotation style of Universal Dependencies is suitable for the merged tree structures, since majority of function words are annotated as leaves there. Function words are the ones which often cannot be translated as one-to-one. For example,

[1] Functional words are determiners, prepositions, conjunctions, auxiliary verbs, particles, etc.

ADJ	adjective	PART	particle
ADP	adposition	PRON	pronoun
ADV	adverb	PROPN	proper noun
AUX	auxiliary verb	PUNCT	punctuation
CONJ	coord. conj.	SCONJ	subord. conj.
DET	determiner	SYM	symbol
INTJ	interjection	VERB	verb
NOUN	noun	X	other
NUM	numeral		

Table 1: List of part-of-speech tags used in Universal Dependencies annotation style.

prepositions in one languages can be translated as different noun suffixes in another one. Some languages use determiners, some not. Auxiliary verbs are also used differently across languages.

3 Data

The parallel data we use in the experiments is the training part of the Czech-English parallel corpus CzEng 1.0 (Bojar et al., 2012). It consists of more than 15 million sentences, 206 million tokens on the Czech side and 232 million tokens on the English side. We extract the parallel sentences with original tokenization from the CzEng export-format together with the part-of-speech (POS) tags and the word alignment.

The original CzEng POS tags, Prague Dependency Treebank tags (Hajič et al., 2006) for Czech and Penn Treebank tags (Marcus et al., 1993) for English, are mapped to the universal POS tagset developed for Universal Dependencies (Nivre et al., 2016). The simple 1-to-1 mapping was taken from the GitHub repository.[2] The POS tags used in Universal Dependencies POS are listed in Table 1.

4 Merging algorithm

Parallel sentences tagged by the universal POS tags are then merged together using the algorithm in Figure 1. We describe the algorithm for the English-to-Czech translation, even though the procedure is generally language universal.

The algorithm uses two unidirectional alignments, which we call *en2csAlign* and *cs2enAlign*. For each English word, the *en2csAlign* defines its counterpart in the Czech sentence. The *cs2enAlign* defined the English counterpart for each Czech word.[3] These alignment links are direct outputs from GIZA++ word-alignment tool (Och and Ney, 2003) before symmetrization.

Input: enF, enT, csF, csT: arrays of forms and tags of the English and Czech sentence
Input: $en2csAlign, cs2enAlign$: unidirectional alignment links between English and Czech
Output: $mrgF, mrgT$: arrays of form and tags of the merged sentence

```
k = 0;
foreach i ∈ {1, ..., |enF|} do
    used = 0;
    foreach j ∈ {1, ..., |csF|} do
        if cs2enAlign[i] ≠ j then continue;
        k++;
        if en2csAlign[j] = i then
            mrgF[k] = enF[i] + '_' + csF[j];
            mrgT[k] = enT[i] + '_' + csT[j];
            used = 1;
        else
            mrgF[k] = '<empty>_' + csF[j];
            mrgT[k] = '<empty>_' + csT[j];
        end
    end
    if used = 0 then
        k++;
        mrgF[k] = enF[i] + '_<empty>';
        mrgT[k] = enT[i] + '_<empty>';
    end
end
return mrgF, mrgT;
```

Figure 1: Merging algorithm pseudocode.

The algorithm traverses through the source sentence and for each word, it collects all its target counterparts using the *cs2enAlign*[4] The Czech word, where the *cs2enAlign* and *en2csAlign* intersect, creates the word pair with the English one. The other Czech words stay alone and are completed with the $<empty>$ label. If there is no intersection counterpart for the English word, it is also completed with the $<empty>$ label.

Figure 2 shows one example of merging. The pairs of words connected by both *cs2enAlign*

[2] https://github.com/UniversalDependencies

[3] In CzEng corpus export format, these alignments are called *ali_there* and *ali_back*, sometimes they are also called *left* and *right* alignments.

[4] Since we search for all the Czech words that are aligned to the English one, we need the *cs2enAlign*.

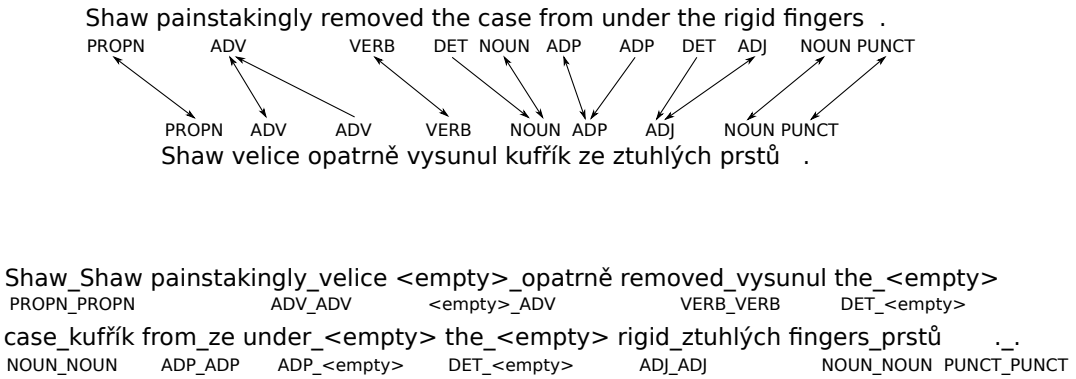

Figure 2: Example of merging English and Czech sentence together with the Universal Dependencies POS tags. Alignment links are depicted by arrows. Bidirectional arrows represent the intersection connections.

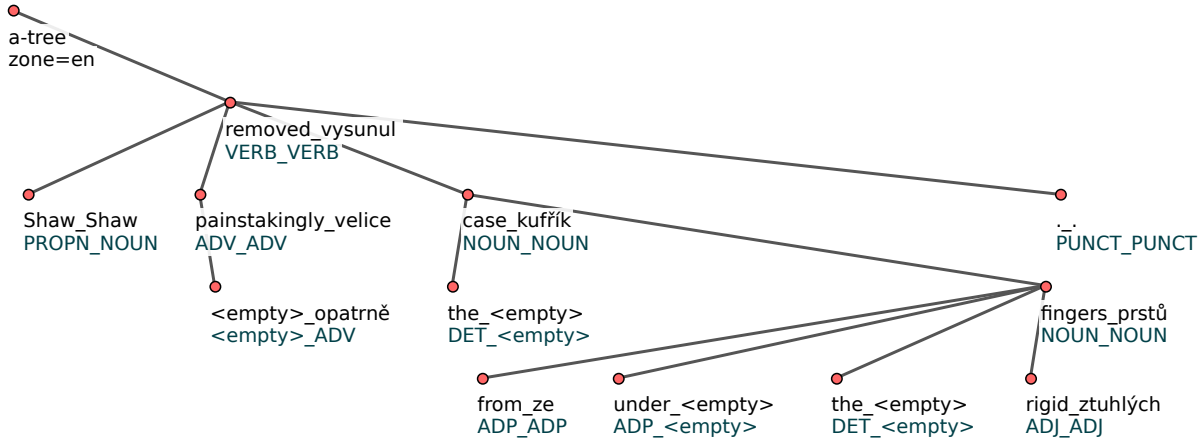

Figure 3: Example of English-Czech merged tree. The same sentence as in Figure 2 is shown.

and *en2csAlign* links are paired together into one word, their POS tags are paired in the same way. The words without intersection counterparts are paired with <empty> words or <empty> POS tags respectively. The tokens in the merged sentence are ordered primarily according to the English sentence. The Czech words with <empty> counterparts are together with Czech words aligned with the same English word. Globally, the Czech word order cannot be preserved due to crossing intersection alignment links, which is a quite common phenomenon.

5 Minimally Supervised Parallel Parsing

For parsing the merged sentences, we use the Unsupervised Dependency Parser (UDP) implemented by Mareček and Straka (2013). The source code is freely available,[5] and it includes a mechanism how to import external probabilities. The UDP is based on Dependency Model with Valence, a generative model which consists of two sub-models:

- Stop model $p_{stop}(\cdot|t_g, dir)$ represents probability of not generating another dependent in direction dir to a node with POS tag t_g. The direction dir can be left or right. If $p_{stop} = 1$, the node with the tag t_g cannot have any dependent in direction dir. If it is 1 in both directions, the node is a leaf.

- Attach model $p_{attach}(t_d|t_g, dir)$ represents probability that the dependent of the node with POS tag t_g in direction dir is labeled with POS tag t_d.

In other words, the $stop$ model generates edges, while the $attach$ model generates POS tags for the

[5] http://ufal.mff.cuni.cz/udp

ADP, ADV, AUX, CONJ, DET, PART, PRON, PUNCT, SCONJ, <empty>	1.0
ADJ, INTJ, SYM	0.7
NOUN, PROPN, NUM, X	0.4
else	0.1

Table 2: Stop probabilities priors set for individual POS tags.

new nodes. The inference is done using blocked Gibbs sampling (Gilks et al., 1996).

During the inference, the $attach$ and the $stop$ probabilities can be combined linearly with external prior probabilities p^{ext}:

$$p^{final}_{stop} = (1 - \lambda_{stop}) \cdot p_{stop} + \lambda_{stop} \cdot p^{ext}_{stop},$$

$$p^{final}_{attach} = (1 - \lambda_{atach}) \cdot p_{attach} + \lambda_{attach} \cdot p^{ext}_{attach},$$

where the parameters λ define their weights. In the original paper (Mareček and Straka, 2013), the external priors p^{ext}_{stop} were computed based on the reducibility principle on big raw corpora.

We use the external prior probabilities to define grammatical rules for POS tags based on UD annotation style. Our rules simply describe how likely a node with a particular POS is a leaf. In case of the merged trees there is a pair of POS tags in each node. We manually set the p^{ext}_{stop} for the POS tags pairs as listed in Table 2. In case the two POS tags in one node have different p^{ext}_{stop}, we take the higher one. For example, for the pair ADP_VERB, we set its prior stop probability to 1.0 (as to the tag ADP), even though the tag VERB should get 0.1.

It is possible to define different left and right p^{ext}_{stop} priors, however, we decided to set it equally for both the directions, since it is linguistically more language independent.

Example of a merged dependency tree is shown in Figure 3.

6 Our Simple Machine Translation System

Our simple translation system based on the merged-trees has the following 3 steps:

- **training:** We go through the training merged trees and compute so called *tree-n-gram* counts. The *tree-n-grams* are n-grams with added parent and children words into context.
- **parsing:** We parse the input data using a parser trained on the source parts of the merged-trees.
- **decoding:** We use the tree-n-gram counts to predict the most probable translation of each source tree.

In the training phase, we traverse the training merged-trees and collect the tree-n-gram counts. Besides looking on the previous and following words, we look also on the parent words. In the training and decoding phase, we work only with word forms, not with the POS tags. We denote w_i the i-th node in the merged tree and p_i its parent. The previous and following word are w_{i-1} and w_{i+1} respectively. We collect only the source words tree-n-gram counts. Their types are listed in Table 3.

1.	$count(w_{i-1}, w_i, w_{i+1})$
2.	$count(w_i, p_i, w_i + 1)$
3.	$count(w_i, p_i, w_{i-1})$
4.	$count(w_i, w_{i+1})$
5.	$count(w_i, w_{i-1})$
6.	$count(w_i, p_i)$
7.	$count(w_i)$

Table 3: Tree-n-gram types collected.

For each node, we also define full target translation, which consist of the target (in our case Czech) form of the node together with target forms of all child nodes with <empty> source (English) form. For example, in Figure 3, the full Czech translation of the node "*painstakingly_velice*" is not only the word "*velice*", but two words "*velice opatrně*".

The parsing phase is necessary to get monolingual tree for sentences we need to translate. Since the merged trees preserves the word ordering of the source sentences (English), we can be simply separate single English dependency trees from the merged trees. We train the MST parser (McDonald et al., 2005) on the separated source (English) trees. The parser is then used to parse the input sentences for translation.

In the decoding step, we translate the parsed source (English) tree into target (Czech) sentence. For each the source node, we go through the tree-n-gram list, from the largest n-gram to the single unigram (according to Table 3) and see, whether it

language pair	BLEU	BLEU cased
English-to-Czech	9.5	8.3
Czech-to-English	15.6	13.2

Table 4: Our system BLEU scores.

appears in the training data more than once[6]. If it is there, we translate the node by the most frequent Czech translation found in the training data. If not, we continue to the next type of n-gram until we find n-gram which appear in the training data more than once. If we end up with single unigram, it is enough if it appears once in the training data and we use its translation. In case the English word is out-of-vocabulary, we do not translate it and use the same form for Czech translation.

Note, that the Czech translation of a node can be more words or no word (in case the English word appears most frequently with the <empty> label).

When the whole dependency tree is translated, we simply project the tree into linear sentence by depth-first algorithm.

7 Results

We tested our translation system in WMT2016 News translation task on English-to-Czech and Czech-to-English language pairs. The BLEU scores (Papineni et al., 2002) are shown in Table 4. Both scores are quite low compared to the best translation systems reaching more 25 or 30 BLEU points respectively. However, for the Czech-to-English direction, the results are comparable with the established tree-based system TectoMT (Mareček et al., 2010; Dušek et al., 2012), which has 14.6 BLEU points and 13.6 BLEU points for the cased variant.

Our system is still under development. This is the first attempt to employ the merged trees in machine translation. So far, it does not use any language modelling or word reordering. The fact that not-aligned words are treated as function words can cause shorter translations with missing content words. All such shortcomings are planned to be solved in future work.

8 Conclusions

We presented the *merged trees*, bilingual dependency trees in Universal Dependencies style parsed by minimally supervised way. The main purpose of such trees is to help in machine translation. We showed very simple translation system and evaluated it WMT 2016 News translation task.

In future work, we will work on improving the system. We plan to employ machine learning, beam-search and language modelling to approach the better MT systems.

[6]We do not use n-grams whose appear only once in the training data. We translate the node using a smaller n-gram instead

Acknowledgments

This work was supported by the grant 14-06548P of the Czech Science Foundation and the 7th Framework Programme of the EU grant QTLeap (No. 610516).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Ondřej Bojar, Zdeněk Žabokrtský, Ondřej Dušek, Petra Galuščáková, Martin Majliš, David Mareček, Jiří Maršík, Michal Novák, Martin Popel, and Aleš Tamchyna. 2012. The Joy of Parallelism with CzEng 1.0. In *Proceedings of LREC2012*, Istanbul, Turkey, May. ELRA, European Language Resources Association.

David Chiang, Adam Lopez, Nitin Madnani, Christof Monz, Philip Resnik, and Michael Subotin. 2005. The Hiero Machine Translation System: Extensions, Evaluation, and Analysis. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 779–786, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ondřej Dušek, Zdeněk Žabokrtský, Martin Popel, Martin Majliš, Michal Novák, and David Mareček. 2012. Formemes in English-Czech Deep Syntactic MT. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, WMT '12, pages 267–274, Stroudsburg, PA, USA. Association for Computational Linguistics.

Walter R. Gilks, S. Richardson, and David J. Spiegelhalter. 1996. *Markov chain Monte Carlo in practice*. Interdisciplinary statistics. Chapman & Hall.

Jan Hajič, Eva Hajičová, Jarmila Panevová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, and Marie Mikulová. 2006. Prague Dependency Treebank 2.0. CD-ROM, Linguistic Data Consortium, LDC Catalog No.: LDC2006T01, Philadelphia.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran,

Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June. Association for Computational Linguistics.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Comput. Linguist.*, 19(2):313–330.

D. Mareček, M. Popel, and Z. Žabokrtský. 2010. Maximum entropy translation model in dependency-based MT framework. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and Metrics (MATR)*, pages 201–206. Association for Computational Linguistics.

David Mareček and Milan Straka. 2013. Stop-probability estimates computed on a large corpus improve Unsupervised Dependency Parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 281–290, Sofia, Bulgaria, August. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-Projective Dependency Parsing using Spanning Tree Algorithms. In *Proceedings of Human Langauge Technology Conference and Conference on Empirical Methods in Natural Language Processing (HTL/EMNLP)*, pages 523–530, Vancouver, BC, Canada.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, Portorož, Slovenia. European Language Resources Association.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rico Sennrich and Barry Haddow. 2015. A Joint Dependency Model of Morphological and Syntactic Structure for Statistical Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2081–2087, Lisbon, Portugal, September. Association for Computational Linguistics.

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