Slavic Forest, Norwegian Wood

Rudolf Rosa and **Daniel Zeman** and **David Mareček** and **Zdeněk Žabokrtský**
Charles University, Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics
Malostranské náměstí 25, Prague, Czech Republic
{rosa, zeman, marecek, zabokrtsky}@ufal.mff.cuni.cz

Abstract

^{D}We once had a corp,
or should we say,^{C}it once had^{D}us
^{D}They showed us its tags,
isn't it great,^{C}unified^{D}tags
^{Dmi}They asked us to parse
and they told us to use^{G}everything
^{Dmi}So we looked around
and we noticed there was near^{Em}nothing^{AA7}
We took other langs,
bitext aligned: words one-to-one
We played for two weeks,
and then they said, here is the test
The parser kept training till morning,
just until deadline
So we had to wait and hope what we get
would be just fine
And, when we awoke,
the results were done, we saw we'd won
So, we wrote this paper,
isn't it good, Norwegian wood.

1 Introduction

This paper describes the winning submission to the Cross-lingual Dependency Parsing shared task at VarDial 2017 (Zampieri et al., 2017).

The goal was to devise a labeled dependency parser for a target language with no treebank available, utilizing treebanks of other very close source languages, and plaintext sentence-aligned source-target parallel data. The task is simulated on target languages for which treebanks do exists, but are not provided to the participants.

As the focus of the task is on parsing per se, a supervised part-of-speech (*POS*) tagger for each target language is provided. Moreover, all of the treebanks come from the Universal Dependencies (*UD*) collection v 1.4 (Nivre et al., 2016), which means that their syntactic and morphological annotation – tree topology, dependency relation labels (*deprels*), universal POS tags (*UPOS*), and morphological features (*morpho feats*) – follows the universal cross-lingual UD scheme.[1]

Consonantly with the focus of the VarDial workshop on similar languages, the source and target languages are very close to each other, with very similar grammars and a nearly one-to-one correspondence on the level of individual words. Therefore, we decided to mostly disregard systematic structural heterogeneity between the languages, and focus primarily on lexical differences.

Our method relies on a context-independent word-by-word machine translation (*MT*) of the source treebank into the target language, based on a one-to-one word alignment provided by a heuristic aligner for similar languages. This switch from a cross-lingual to a pseudo-monolingual setting allows us to easily apply source-trained taggers and parsers to the target data and vice versa.

We also employ several homogenization techniques, mostly to overcome systematic differences in treebank annotations. Specifically, we normalize the deprels in the source treebanks to better correspond to the target deprels, and we subselect only cross-lingually consistent morpho feats.

2 Related Work

The notorious fact that there are several thousand languages used around the globe makes it necessary to search for NLP methods that could be applicable to a wider range of languages, ideally without too much effort invested into building language-specific resources for new languages again and again. This is by far not specific to dependency parsing, for which—like for most other "traditional" NLP tasks—various approaches have

[1] See http://universaldependencies.org/docsv1 for a description of the UD scheme.

Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects, pages 210–219,
Valencia, Spain, April 3, 2017. ©2017 Association for Computational Linguistics

been developed, ranging from fully unsupervised methods (whose performance seems to be limited) to supervised methods with radically economy-driven annotation management.

We limit the scope of the following overview only to cross-lingual transfer of dependency parsers from a resource-rich source language(s) to a resource-poor target language. In addition, this paper does not touch the discussions whether a tree (and what kind of tree) is a reasonable representation for a sentence structure, and whether all languages do really share their structural properties to such an extent that a single type of representation is viable for all of them. Though such issues deserve intensive attention, and perhaps even more so now when UD have gained such a fascinating momentum, we take the two assumptions simply for granted. Neither do we present the genesis of the current UD collection, preceded by HamleDT treebank collection by Zeman et al. (2014), going back to the CoNLL 2006 and 2007 tasks (Buchholz and Marsi, 2006; Nivre et al., 2007), and to earlier POS standardization efforts. In this overview, we limit ourselves to the scope outlined by the VarDial shared task, whose goal is to develop a parser for a (virtually) under-resourced language closely related to a resource-rich language.[2]

We believe that most of the published approaches could be classified into two broad families which we call *tree-transfer-based* methods and *common-abstraction-based* methods. The former project individual dependency trees across the language boundary prior to training a target parser. The latter methods transfer a parser model trained directly on the source treebank, but limited only to abstract features shared by both languages.

2.1 Tree-transfer-based approaches

In the tree-transfer-based approaches, a synthetic pseudo-target treebank is created by some sort of projection of individual source trees into the target language. Then a standard monolingual parser can be trained using the pseudo-target treebank in a more or less standard way. As it is quite unlikely that a manually annotated source treebank with high-quality human-made target translations and high-quality alignment exists, one or more of the necessary components must be approximated. And even if all these data components existed, the task of dependency tree projection would inevitably lead to collisions that have to be resolved heuristically, especially in the case of many-to-one or many-to-many alignments, as investigated e.g. by Hwa et al. (2005) and more recently by Tiedemann (2014) or Ramasamy et al. (2014).

This family embraces the following approaches:

- using a *parallel corpus* and projecting the trees through word-alignment links, with authentic texts in both languages but an automatically parsed source side,
- using a *machine-translated parallel corpus*, with only one side containing authentic texts and the other being created by MT; both translation directions have pros and cons:
  - source-to-target MT allows for using a gold treebank on the source side,
  - target-to-source MT allows the parser to learn to work with real texts in the target language, for which, in addition, a gold POS labeling might be available.

Obviously there are certain trade-offs related to this family of tree transfer approaches. For example, using MT to create a synthetic parallel corpus often results in a considerably lower text quality, but provides more reliable alignment links. In addition, such alignment typically has a higher amount of one-to-one word alignments, which facilitates tree projection; in case of extremely close languages, as in this paper, the MT system can be constrained to produce only 1:1 translations.

There are two additional advantages of the tree-transfer-based approach:

- the feature set used by the target language parser is independent of the features that are applicable to the source language,
- we can easily use only sentence pairs (or tree fragments) with a reasonably high correspondence between source and target structures, as done by Rasooli and Collins (2015).

2.2 Common-abstraction-based approaches

By using a "common abstraction" we mean using features that have the same or very similar "meaning" both in the source and target language. Obviously, word forms cannot be easily used directly, as there are various spelling and morpho-

[2] Crosslingual transfer is not used only in truly under-resourced scenarios, but also in situations in which it is hoped that features explicitly manifested in one language (such as morphological agreement) could boost parsing performance in some other language in which they are less overt. Such bilingually informed parsing scenarios are studied e.g. by Haulrich (2012).

logical differences even between very close languages. Using such shared features allows a parser that was trained on a source treebank to be used directly on target texts; i.e. the source-target "transfer" of the parser is trivial, compared to a source-target transfer of the treebank as described in §2.1.

The common abstraction features used by the parser can be linguistically motivated, or induced by mathematical methods such as clustering and vector space representation:

- Unified POS tags: a POS tagset simplified and unified to the extent that it was usable for both source and target languages was behind one of the first experiments with delexicalized parsing by Zeman and Resnik (2008). The advantage of such approaches lies in their linguistic interpretability. On the other hand, in spite of the substantial progress in tagset harmonization since the work of Zeman (2008), this approach can end up in a very limited intersection of morphological categories in case of more distant languages.
- Word clusters have been successfully applied in many NLP fields, with the clusters of Brown et al. (1992) being probably the most prominent representative. Täckström et al. (2012) showed that cross-lingually induced clusters can serve as the common abstract features for cross-lingual parsing.
- Word embeddings, if induced with some cross-lingual constraints and mapped into a shared low-dimensional space, can also be used, as shown e.g. by Duong et al. (2015).

An obvious trade-off that appears with this family of methods is associated with the specificity/generality of the shared abstract representation of words. For example, in the case of delexicalization by a common POS tagset, the question arises what is the best granularity of shared tags. The more simplified tags, the more language-universal information is captured, but the more information is lost at the same time. Moreover, even if two languages share a particular morphological category, e.g. pronoun reflexivity, it is hard to predict whether adding this distinction into the shared tagset helps the resulting parser or not.

A variation that appers with this family of methods is the usage of "relexicalization". The base parser resulting from the transfer is applied on (unseen) target data, and a new parser is self-trained on this data; a successful application of this approach is documented by Täckström et al. (2013).

2.3 Other variations

Aufrant et al. (2016) combines both main strategies described above by adapting the word order in source sentences to be more similar to that of the target language, e.g. by swapping the order of an attribute and its nominal head; the information about these configurations was extracted from the WALS World Atlas of Language Structures (Dryer and Haspelmath, 2013). Such processing of source language trees fits to the first family of approaches, as it resembles a (very limited) MT preprocessing; but after this step, a POS-delexicalized parser transfer is used, which fits the second family.

When processing more than a few under-resourced languages, choosing the best source language should be ideally automatized too. One could rely on language phylogenetic trees or on linguistic information available e.g. in WALS, or on more mechanized measures, such as Kullback-Leibler divergence of POS trigram distributions (Rosa and Žabokrtský, 2015).

In addition, we might want to combine information from more source languages, like in the case of multi-source transfer introduced by McDonald et al. (2011). Choosing source language weights to be used as mixing coefficients becomes quite intricate then as we face a trade-off between similarity of the source languages to the target language and the size of resources available for them.

3 Task and Data

The task was to perform labeled dependency parsing of each of the three target languages, Slovak (*SK*), Croatian (*HR*), and Norwegian (*NO*), without using target treebanks. In the constrained track of the task, we were only allowed to use provided source treebanks and source-target parallel data for source languages closely related to the target languages: Czech (*CS*) as a source for SK, Slovenian (*SL*) for HR, and Danish (*DA*) and Swedish (*SV*) for NO. Because of reported good performance in the baselines, we use the DA and SV data concatenated into "Dano-Swedish" (*DS*).

For development testing of our systems, small target *dev treebanks* were provided, with golden syntactic annotation, and morphological annotation (UPOS and morpho feats) predicted by supervised taggers; the taggers were also provided. Final test treebanks were annotated in the same way.

For an exact description of the task, the datasets, models, baselines and upper-bounds, please refer to (Zampieri et al., 2017) and the task webpage.[3]

The task specifies Labeled Attachment Score (LAS) as the primary metric, and Unlabeled Attachment Score (UAS) as a secondary one.

4 Components

In §4.1 we describe the baseline setup, which we further enrich by the components described in the following sections; the final setups used for each of the target languages are specified in §5.

The development and employment of the components was guided by continual evaluation on the dev treebanks. We evaluated several variations of each component, and selected the best performing variant separately for each target language.[4] Hyperparameter tuning was performed neither for the tagger and parser nor for any of the components, as this was forbidden by the shared task rules.[5]

4.1 Baseline

As our starting point, we took the task baseline. It consists of a UDPipe tagger and parser (Straka et al., 2016),[6] trained on the source treebank with the default settings, except:

- the parser is trained without using the morpho feats (i.e. only using word form and UPOS)[7]
- the tagger is trained to only produce UPOS.[8]

We train the tagger and parser together, which means that UDPipe trains the tagger, applies it to the treebank, and trains the parser using morphological annotation predicted by the tagger. We have found this setup to perform better than training on gold annotation by +1.6 LAS on average.

4.2 Annotation Normalization

Unlike some older work in this area, we work with multi-lingual data that is harmonized across languages, i.e. all languages should be syntactically and morphologically annotated according to the same UD guidelines. However, the current level of harmonization is still far from perfect. Certain deprels occur in the source treebanks but not in the target treebank (or vice versa), but not due to differences in the treebank languages or domains – it is just because of differences in annotation, despite the intention of UD to annotate the same things in the same way. We obviously cannot modify the test data in any way, but we can make the source data as similar to the target annotation as possible. By doing so, we simulate a likely real-world scenario: when people want to parse a resource-poor language, they supposedly know what kind of deprels they want in the output.

For example, CS contains a language-specific `nummod:gov` deprel, which never occurs in SK. We do not want the parser to learn to assign that deprel, because we are not going to score on such relations. Hence, we replace all occurrences of `nummod:gov` in the source treebank by the more general `nummod` deprel, which is also used in SK.

Similarly, one may want to modify the UPOSes and morpho feats, which the parser gets as input and can use them to improve syntactic analysis. It seems reasonable to adjust or hide tags unavailable in the target data; e.g., the SK treebank does not distinguish `SCONJ` from `CONJ`, and `DET` from `PRON`; or, the Scandinavian treebanks disagree on when participles are `VERB` and when `ADJ`.

Finally, we tried to normalize several rather randomly spotted phenomena whose analysis systematically differs across languages. The most prominent example is the Scandinavian word *både* in *både A och/og B* “both A and B”. In SV, the word is tagged `CONJ` and attached via the `advmod` deprel, in DA it is `ADV/advmod`, and in NO it is `CONJ/cc`. Normalizing instances of *både* alone increased LAS on NO by almost 1 point!

Our normalization is based on manual error analyses of parser outputs on the dev treebanks.

4.3 Word-by-Word Machine Translation

The core of our approach is a move from a cross-lingual to a pseudo-monolingual setting by translating the word forms in the source treebank into the target language. It has three steps: word-alignment of the parallel data, extraction of a translation table from the aligned data, and the treebank translation itself.

[3] `https://bitbucket.org/tiedemann/vardial2017`

[4] We evaluated 114 different setups: 63 were evaluated for all 3 target languages, 12 for 2, and 39 for 1 target language; in total, 252 LAS scores were computed on the dev treebanks.

[5] Following the UDPipe manual, we deviated from the default tagger hyperparameters in case of CS (due to its huge treebank), setting them to: `guesser_suffix_rules=6; guesser_enrich_dictionary=4`

[6] Version `3e65d69` from 3rd Jan 2017, obtained from `https://github.com/ufal/udpipe`

[7] Parameter: `embedding_feats=0`

[8] Parameters: `use_lemma=0;provide_lemma=0; use_xpostag=0;provide_xpostag=0; use_feats=1;provide_feats=1`

We employ a simple word-based MT approach, which we tried as a first attempt but found it good enough for our purpose; we have yet to evaluate how it compares to more sophisticated methods.

4.3.1 Word-alignment

Since the source and target languages in our task are very close to each other, we decided to use the heuristic Monolingual Greedy Aligner (*MGA*) of Rosa et al. (2012),[9] rather than e.g. the usual Giza++ (Och and Ney, 2003) – most standard word aligners ignore word similarity, which we believe to be useful and important in our setting.

MGA utilizes the word, lemma, and tag similarity based on Jaro-Winkler distance (Winkler, 1990), and the similarity of relative positions in the sentences, to devise a score for each potential alignment link as a linear combination of these, weighted by pre-set weights. The iterative alignment process then greedily chooses the currently highest scoring pair of words to align in each step; each word can only be aligned once. The process stops when one of the sides is fully aligned, or when the scores of the remaining potential links fall below a pre-set threshold.

We used MGA as is, with the default values of the hyperparameters and with no adaptation to the UD annotation style or the specific languages of the task. Even though MGA was originally designed for aligning same-language sentences (especially Czech), we found it to perform well enough in our setting, and therefore left potential tuning and adaptations for future work.

Before aligning, we preprocess the parallel data by the Treex tokenizer, the provided target tagger, and a source tagger trained on the source treebank.

4.3.2 Translation table extraction

For our methods to be easily applicable, we require a one-to-one translation, which we can afford due to the high similarity of the languages. Therefore, we extract a translation *word* table rather than the more usual phrase table from the aligned data. Moreover, due to the simplicity of the subsequent translation step, it is sufficient for us to only store the best (most frequent) translation for each word; we use Jaro-Winkler similarity of the source and target word forms as a tie breaker.

Identical source word forms with differing UPOS or morpho feats annotations are treated as distinct words, serving as a basic source-side disambiguation; we rely on these source annotations being available at inference for selecting the translation. To reduce the OOV rate, two backoff layers are also stored, the first disregarding the morpho feats, and the second also disregarding the UPOS.

An option that we leave for future research is to use the alignment scores provided by the MGA when constructing the translation table.

For simplicity, we create only one joint translation table for translating DS into NO.

4.3.3 Treebank translation

We translate each source treebank into the target language word-by-word, independent of any source or target context. We use the golden annotation of UPOS and morpho feats for source-side disambiguation; a backoff layer is used if the translation table does not contain the source word form with the given annotations. OOVs are left untranslated. This results in a pseudo-target treebank, with golden annotations from the source treebank and word forms in the target language.

In preliminary experiments, the opposite target-to-source translation led to worse results (by -1.3 LAS on average), possibly because the parser relies more on the correctness of the source, making it less robust when applied to the machine-translated target. Moreover, in case of DS-NO, the target-to-source translation is not straightforward.

4.4 Pre-training Word Embeddings

Because UDPipe uses a neural network parser, all input features have to be converted to vectors. By default, it trains embeddings of each input feature on the pseudo-target treebank jointly with training the parser. As larger data can provide better embeddings, we pretrain word form embeddings on the target side of the parallel data, pretokenized by the Treex tokenizer (Popel and Žabokrtský, 2010),[10] and provide them to UDPipe. We use word2vec (Mikolov et al., 2013), with the parameters suggested in the UDPipe manual.[11]

[9] `https://github.com/ufal/treex/blob/master/lib/Treex/Tool/Align/MonolingualGreedy.pm`

[10] `https://github.com/ufal/treex/blob/master/lib/Treex/Block/W2A/Tokenize.pm`

[11] `-cbow 0 -size 50 -window 10 -negative 5 -hs 0 -sample 1e-1 -binary 0 -iter 15 -min-count 2 -threads 12`

4.5 Morphological Features Subselection

We found out that in the default setting, not using the morphological features leads to better LAS than using them. This is probably caused by the fact that UDPipe treats the morpho feats string as a single unit and is not able to split it and assign different importance to individual features. We therefore try to find an effective subsection of the morphological features.

4.5.1 Keep useful

Collins et al. (1999) showed that *Case* was the most valuable feature for parsing Czech; indeed, when we discard all features but *Case*, we observe better accuracy for all target languages.

One other feature they use with words that do not have *Case* is called *SubPOS* and is specific to the tagset of their corpus. In UD, there are several features with similar function, e.g. *PronType* subcategorizing pronouns or *NumType* subcategorizing numerals. Unfortunately, we found neither of them to help in our setting.

4.5.2 Keep shared

Another possibility is to keep only those features that are highly consistent cross-lingually. For each feature-value pair in the tagged and aligned parallel data, we count the number of times it appears on both sides of an alignment pair. The consistency c of feature-value pair f is computed as:

$$c(f) = \frac{1}{2}\left(\frac{\#(f \in s, f \in t)}{\#(f \in s)} + \frac{\#(f \in s, f \in t)}{\#(f \in t)}\right)$$

where $\#()$ indicates the number of times the feature is present in the source (s), target (t), or both aligned words. We only keep feature-value pairs with consistency higher than a threshold, which we set to 0.7 after having evaluated the values of 0.6, 0.7, and 0.8. We also tried to condition the consistency scores on UPOS, which did not improve LAS.

The two described feature selection mechanisms can also be combined, e.g. by providing the *Case* feature in the morhpo feats field, and the other shared features in the XPOS field, thus enabling the parser to treat them separately.

4.6 Cross-Tagging

There is a considerable body of work on projecting POS taggers across aligned corpora, dating back to (Yarowsky and Ngai, 2001). In combination with cross-language parsing, such techniques are used to provide the parser with target-side POS tags. Our task is specific in that a supervised target POS tagger is available; however, there are still several possibilities of combining tagger and parser models in order to make the parsed data as similar as possible to what the parser was trained on.

- **Baseline.** Train a parser on the source treebank. Tag the target data by a supervised target tagger and parse it by the trained parser, hoping that the tags produced by the target tagger are similar enough to the source tags.
- **Source data cross-tagging (*source-xtag*).** Translate source treebank into the target language, tag it by a supervised target tagger and train a parser on it. Tag the target data by the supervised target tagger and parse it by the trained parser.
- **Target data cross-tagging (*target-xtag*).** Translate the source treebank into the target language and train a tagger and parser on in. Tag the target data by the trained tagger and then parse it by the trained parser.

In addition, we always train the parser jointly with a tagger, so that the parser is trained on monolingually predicted tags, as explained in §4.1.

We have found source-xtag to work well for heterogeneous source data, such as the DS mixture.

Conversely, target-xtag proved useful for SK, where the source treebank is much larger than the target data used to train the target tagger. A tagger trained on the large source treebank provides much better tags, which in turn boosts the parsing accuracy, despite the noise from MT and xtag.

Note that if no target tagger is available, we must either use target-xtag, or we may project a tagger across the parallel data in the style of Yarowsky and Ngai (2001) and use the resulting tagger in our baseline or source-xtag scenarios.[12]

We also experimented with cross-tagging of only the UPOS or only the morpho feats, with different setups being useful for different languages.

Although the UDPipe tagger can also be trained to perform lemmatization, we have not found any way to obtain and utilize lemmas that would improve the cross-lingual parsing.[13]

[12] Our approach still needs a target tagger to perform the word alignment, but we believe that for very close languages, the word forms alone might be sufficient to obtain a good-enough alignment; or, a different word aligner could be used.

[13] We tried to translate the lemmas, as well as to perform simple stemming, such as cropping or devowelling.

	Component	SK	HR	NO
§4.2	Normalize source annotations	✓	✓	✓
§4.3	Translate word forms	✓	✓	✓
§4.4	Pre-train form embeddings	✓	✓	✓
§4.6	Source-xtag of UPOS	×	×	✓
§4.5	Add *Case* morpho feat	✓	✓	✓
§4.5	Add shared morpho feats	✓	×	×
§4.6	Target-xtag of morpho feats	✓	×	✓
§4.6	Target-xtag of UPOS	✓	×	×

Table 1: Components used for the various languages, listed in the order in which they are applied. The *Case* feature was used in the best (after-deadline) SK setup, but not in the submitted setup.

5 Individual Language Setups

In our final setup, we enrich the baseline (§4.1) by various components (§4), as listed in Table 1:

1. Normalize source treebank annotations
2. Translate source treebank to target language
3. Pre-train target word form embeddings
4. *For NO:* Cross-tag UPOS in source treebank
5. Prune source treebank morphological features, keeping only *Case*
6. *For SK:* Put frequently shared morpho feats into the "XPOS" field in source treebank
7. Train a tagger on source treebank, tagging UPOS and *Case* (for SK also "XPOS")
8. Retag source treebank by the trained tagger
9. Train a parser on source treebank, using the pre-trained word form embeddings, UPOS, and *Case* (for SK also "XPOS")
10. *For HR:* Prune target morphological features, keeping only *Case*
11. *For NO and SK:* Cross-tag *Case* in target
12. *For SK:* Cross-tag UPOS and morphological features in target
13. Parse target corpus by the trained parser

We believe that the utility of the language-specific components owes to the following:

- For NO, there are two different source languages. Translating them both to NO and retagging them with the NO tagger makes the training data more homogeneous.[14]
- SK and CS seem to be the closest languages in the shared task, both being morphologically very rich, which explains the usefulness of employing additional shared morpho feats.
- The CS treebank is extremely large, leading to the fact that a pseudo-SK tagger, trained on the CS treebank translated to SK, performs far better than the original SK tagger.

	SK	HR	NO
Setup	LAS on test		
Baseline	53.72	53.35	59.95
Our	**78.12**	60.70	70.21
Supervised	69.14	**68.51**	**78.23**
Reaching supervised	158%	48%	56%
Setup	LAS on dev		
Baseline	55.97	55.88	59.31
Our	77.49	64.32	69.99
Supervised	70.27	74.27	78.10
Reaching supervised	145%	48%	56%

Table 2: Evaluation using LAS. *Reaching supervised* is how far we got on the scale between the baseline and the supervised setup.

6 Results

The results we achieved on the dev and test treebanks are listed in Table 2. For SK, we got an even better result of 79.37% LAS (78.63% on dev) just 6 minutes after the deadline by combining shared morphological features and *Case*, while the submitted setup only contained the shared features without *Case*. The baseline and supervised LAS are shown as reported by organizers.

We can see that for both HR and NO, we achieve a score that is approximately half the way from the baseline to the supervised setup. The fact the CS and SK are very close, and that the CS treebank is huge, leads to amazing results for SK, leaving the supervised "upper-bound" far behind.

Table 3 shows our results in comparison to the second-best system of (Tiedemann, 2017). When evaluating with LAS, our system clearly outperforms them by a large margin for all three languages; however, the score difference practically disappears for NO and HR and is greatly diminished for SK when UAS is used for evaluation instead. We hypothesize that most of these additional gains in LAS are due to the deprel normalization, which (Tiedemann, 2017) might not have employed, and which is bound to have negligible effect on UAS. This belief is also strongly supported by the estimated improvement brought by the normalization component according to the ablation analysis (see next paragraph), which very tightly corresponds to the amount of lead we lose when going from LAS to UAS evaluation.

Table 4 reports the ablation analysis performed on the dev treebanks to estimate the effect of individual components. We report the deterioration

[14] However, it is better to use the original morphological features in source treebank and cross-tag them on target treebank, presumably because annotation of *Case* in SV is much richer than in NO.

System	SK	HR	NO
LAS			
Our	78.12	60.70	70.21
Tiedemann	73.14	57.98	68.60
UAS			
Our	84.92	69.73	77.13
Tiedemann	82.87	69.57	76.77

Table 3: Comparison of LAS and UAS scores of our system and the second-best system.

Component	SK	HR	NO
Normalize source annotations	2.50	3.11	1.67
Translate word forms	7.04	**5.02**	**6.66**
Pre-train form embeddings	2.83	3.88	5.28
Cross-tag	**11.36**	—	2.92
Add morphological features	2.09	1.70	1.43

Table 4: Ablation analysis: reduction of LAS score when removing various components.

in LAS versus our best setup[15] that occurs when a given component is removed.[16] This serves as an indication of the improvement brought by the component; it is not exact due to some interplay of the components and overlapping of their effects. The "Cross-tag" component refers to the joint effect of any cross-tagging steps used for the respective languages. Similarly, "Add morphological features" refers to adding only the *Case* feature for HR and NO, but adding both *Case* and shared morphological features for SK.

Overall, the most important component seems to be the translation of word forms, leading to improvements of +5 to +7 LAS. This seems to confirm our initial hypothesis that for very close languages, much of the gap between the baseline and the supervised parser can be bridged by appropriate lexicalization. However, the single largest improvement (+11.36 LAS) is achieved by target-xtag of SK, probably because the CS treebank is enormous and because CS and SK are extremely close languages. Other components also brought very nice improvements, amounting to +2.7 LAS on average per component and language.

7 Discussion and Future Work

Overall, our setup has achieved very good results. It surpassed all other submissions to the shared task on each language in both LAS and UAS, halving the gap between the baseline and the supervised parser for two of the languages and even far exceeding supervised for the third. The result for CS-SK shows that for pairs of very similar languages, the usefulness of cross-lingual methods can go beyond the realm of under-resourced languages, improving even upon respectable supervised setups; even better results could probably be obtained by a combination of both.

As we use many of the components in the same way for all of the languages with no need of manual adaptation or evaluation on target data, our approach could also be easily applied to other languages; we plan to do that in the near future.

Other components are unfortunately not applicable to new data in a straightforward manner. We employed cross-tagging in a different way for each of the languages, and although we offered possible explanations of why particular setups work best for particular languages, it is an open question whether these explanations can also be used to guide setting up a system for a new language pair. Furthermore, the annotation normalization has to be devised manually for each of the source and target languages.[17]

Although we found that the translation is the most important component of our pipeline, we have yet to evaluate it properly and identify potential ways to improve its performance.

We also believe that further increases in accuracy may be obtained by substituting UDPipe with a brand new tagger and/or parser that would feature current improvements in the field.

To allow other researchers to examine and/or apply our approach, we have freely released the source codes[18] and models.[19]

Acknowledgments

The work was supported by the grant 15-10472S of the Czech Science Foundation, SVV grant of Charles University, and by the EU project H2020-ICT-2014-1-644402. This work has been using language resources and tools developed, stored and distributed by the LINDAT/CLARIN project of the Ministry of Education, Youth and Sports of the Czech Republic (project LM2015071).

[15] For SK, we use the post-deadline setup which combines *Case* and shared morphological features.

[16] For MT, we take the best setup without cross-tagging as the basis, since the performance of the cross-tagger without MT is low and would obscure the effect of the MT itself.

[17] Or rather for the respective treebanks than languages, since the annotation differences do not necessarily correspond to real differences between the languages.

[18] `http://hdl.handle.net/11234/1-1970`

[19] `http://hdl.handle.net/11234/1-1971`

References

Lauriane Aufrant, Guillaume Wisniewski, and François Yvon. 2016. Zero-resource dependency parsing: Boosting delexicalized cross-lingual transfer with linguistic knowledge. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 119–130, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Peter F. Brown, Peter V. Desouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, pages 149–164, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Collins, Lance Ramshaw, Jan Hajič, and Christoph Tillmann. 1999. A statistical parser for Czech. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 505–512, Stroudsburg, PA, USA. Association for Computational Linguistics.

Matthew S. Dryer and Martin Haspelmath, editors. 2013. *WALS Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Cross-lingual transfer for unsupervised dependency parsing without parallel data. In *CoNLL*, pages 113–122.

Martin Wittorff Haulrich. 2012. *Data-driven bitext dependency parsing and alignment*. Copenhagen Business School, Department of International Business Communication.

Rebecca Hwa, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11:11–311.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 3111–3119.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June. Association for Computational Linguistics.

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)*, pages 1659–1666, Portorož, Slovenia.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Martin Popel and Zdeněk Žabokrtský. 2010. TectoMT: Modular NLP framework. In *Proceedings of IceTAL, 7th International Conference on Natural Language Processing, Reykjavík, Iceland, August 17, 2010*, pages 293–304. Springer.

Loganathan Ramasamy, David Mareček, and Zdeněk Žabokrtský. 2014. Multilingual dependency parsing: Using machine translated texts instead of parallel corpora. *The Prague Bulletin of Mathematical Linguistics*, 102:93–104.

Mohammad Sadegh Rasooli and Michael Collins. 2015. Density-driven cross-lingual transfer of dependency parsers. In *EMNLP*, pages 328–338.

Rudolf Rosa and Zdeněk Žabokrtský. 2015. KL_{cpos^3} – a language similarity measure for delexicalized parser transfer. In *ACL (2)*, pages 243–249.

Rudolf Rosa, Ondřej Dušek, David Mareček, and Martin Popel. 2012. Using parallel features in parsing of machine-translated sentences for correction of grammatical errors. In *Proceedings of the Sixth Workshop on Syntax, Semantics and Structure in Statistical Translation*, SSST-6 '12, pages 39–48, Stroudsburg, PA, USA. Association for Computational Linguistics.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, Paris, France, May. European Language Resources Association (ELRA).

Oscar Täckström, Ryan McDonald, and Jakob Uszkoreit. 2012. Cross-lingual word clusters for direct transfer of linguistic structure. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics:*

Human Language Technologies, NAACL HLT '12, pages 477–487, Stroudsburg, PA, USA. Association for Computational Linguistics.

Oscar Täckström, Ryan T. McDonald, and Joakim Nivre. 2013. Target language adaptation of discriminative transfer parsers. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 1061–1071. The Association for Computational Linguistics.

Jörg Tiedemann. 2014. Rediscovering annotation projection for cross-lingual parser induction. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1854–1864, August.

Jörg Tiedemann. 2017. Cross-lingual dependency parsing for closely related languages – Helsinki's submission to VarDial 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, Valencia, Spain.

William E. Winkler. 1990. String comparator metrics and enhanced decision rules in the Fellegi-Sunter model of record linkage. In *Proceedings of the Section on Survey Research*, pages 354–359.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*, Pittsburgh, Pennsylvania. Association for Computational Linguistics.

Marcos Zampieri, Shervin Malmasi, Nikola Ljubešić, Preslav Nakov, Ahmed Ali, Jörg Tiedemann, Yves Scherrer, and Noëmi Aepli. 2017. Findings of the VarDial evaluation campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, Valencia, Spain.

Daniel Zeman and Philip Resnik. 2008. Cross-language parser adaptation between related languages. In *Workshop on NLP for Less-Privileged Languages, IJCNLP*, Hyderabad, India.

Daniel Zeman, Ondřej Dušek, David Mareček, Martin Popel, Loganathan Ramasamy, Jan Štěpánek, Zdeněk Žabokrtský, and Jan Hajič. 2014. HamleDT: Harmonized multi-language dependency treebank. *Language Resources and Evaluation*, 48(4):601–637.

Daniel Zeman. 2008. Reusable tagset conversion using tagset drivers. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, pages 213–218, Marrakech, Morocco. European Language Resources Association.