Dependency Parsing for Low-Resource Languages Using "Comparable" Multilingual Data

Jiaming Luo Sunoo Park Hayk Saribekyan

MIT

1 Introduction

Parsing is an important problem in Natural Language Processing. This work's interest is in dependency parsing, where each word in the sentence is assigned a "head" word, thus, forming a directed rooted tree with words as nodes. State-of-the-art supervized parsers achieve an accuracy of 90% on resource rich languages such as English.

A major impediment to dependency parsing for low-resource languages is the lack of availability of gold tree banks on which to train parsers. In recent years, a number of approaches have been explored in the literature to overcome such issues.

A recent result of [6] considers the setting when parallel multilingual corpora are available (their research is based on Europarl), and proposes generating the "gold" training data for a target-language parser by *projecting* arcs onto the target-language sentences based on the dependency trees for the parallel sentences in other languages (the trees for the languages other than the target language as assumed to be known). Their method achieves an accuracy of 78.8%.

However, the method of [6] is not very applicable to low-resource languages, which by definition, do not tend to have large parallel corpora with high-resource languages for which we have tree banks. The hierarchical tensor model developed by [10] achieves accuracy of 62.5% in an unsupervized settings, and 74.2% in a semi-supervized setting, where only 50 gold trees are available for training in low-resource language. [7] trains a discriminative model, adopting the idea of selective sharing from [5], and achieves 62.5% on German as well.

In this project, we explore techniques to project dependency arcs onto target-language sentences from *comparable* corpora: that is, documents which are about the same topic and which are likely to contain some *similar* sentences, but which are not exact translations of each other. The inspiration for this direction of work is that Wikipedia can serve as an abundant source of comparable corpora across many different languages, so if we can develop effective dependency tree projection techniques for comparable corpora, these techniques could be very applicable to low-resource languages[1].

2 Corpora

We test our methods both on a truly parallel corpus, Europarl [2], and on a corpus of comparable articles from Wikipedia [8]. We use English and French as our source languages and German as our target language. The choice of these languages for testing was due to the availability of high-quality gold data, which facilitated the evaluation of our methods.

[1] Code is available at https://github.mit.edu/hayks/nlp-project

Our alignment algorithms also require a dictionary to translate words between different languages, for which we used data from Wiktionary [9]. The motivation for basing the alignment algorithm on only a dictionary was that dictionaries of reasonable quality tend to be available even for low-resource languages.

Wikipedia has many articles esspecially for the resource-rich languages, which we use as source languages. To speed up the process of data preparation and training as well, we have chosen to download only a subset of them. In particular, we have used all articles that have an Armenian translation. Wikipedia has more than 100,000 articles in Armenian. As a result, our corpora contains about 90,000 articles in English and 70,000 articles in French.

We have used English Wiktionary, which contains more than 80,000 words in English with 50,000 translations to both French and German.

3 Our approach

We begin with an overview of our basic approach. The (low-resource) language which we want to learn to parse by projecting dependencies from other languages will be referred to as the *target language*. In addition, we refer to one or more *source languages*, for which we assume we have gold trees and good parsers, and from which we want to project dependencies to the target language. In our experiements we have used English as a source langage and German as a target language.

From our corpus, we first extract Wikipedia articles which are available in the target language and at least one source language. Then, we follow the high-level steps described below.

```
For each pair of (source, target)-language articles:
    For each sentence in the source-language article:
        Identify all similar sentences in the target-language article
        Align the words in each of these sentence pairs and project trees
Resolve conflicts if projected trees overlap
Train a supervized parser using the projected trees
```

We will elaborate on each step below.

3.1 Challenges

The problem of projecting dependency arcs based on comparable corpora raises some challenges that do not arise in the case of clean parallel corpora. Before the arcs can be projected, parallel (or similar) sentences must first be *aligned*. In the case of parallel corpora, traditional alignment techniques can be used. However, with comparable corpora, we would like an alignment technique that deals well with additions, deletions, and paraphrasing between the two sentences to be aligned. Moreover, in order to get as many projected arcs as possible, we would like not only to align full sentences with each other, but also to align sub-clauses of sentences which may be similar even if the full sentences are not. Some examples are given in Figures 1 and 2.

Another challenge is that we would like to project arcs from as many source sentences and languages as possible, to make maximal use of the all the comparable data that Wikipedia offers. For example, 3 shows how two source sentences which project useful dependencies onto different parts of a target sentence.

However, we cannot simply take the union of all projected arcs to be the projected dependency tree of the target sentence, because these may not be consistent with each other (in particular, they may not form a tree or a forest). An example of conflicting projections is given in Figure 4, where

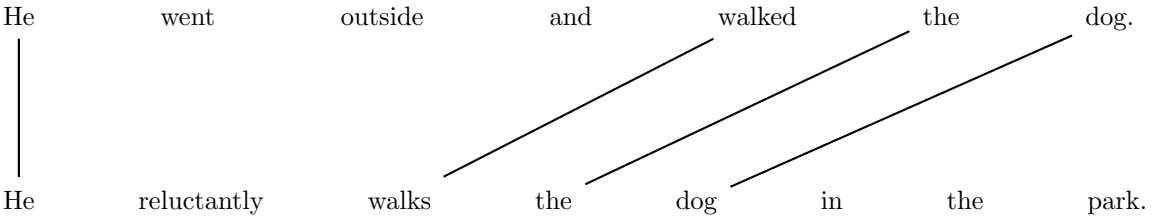

Figure 1: Alignment of similar sentences in the presence of additions/deletions/paraphrasing

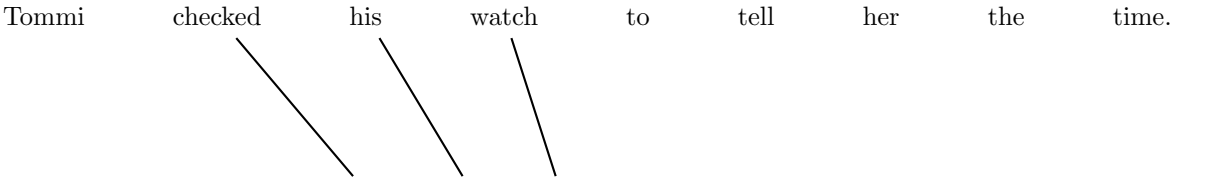

Figure 2: Alignment of dissimilar sentences which have similar sub-clauses

the incorrect projection from the second sentence would induce dependency arcs that contradict the correct first projection. An interesting question is how to decide which arcs to keep, and which to discard, in the case of conflicting projections. We will experiment with a few different methods of choosing which arcs to keep, which are detailed in Section 3.3.

3.2 Identifying and aligning comparable sentence pairs

When considering the problem of identifying comparable sentence pairs across different-language Wikipedia articles, a question that immediately arises is:

How similar are the sentences in a pair of different-language Wikipedia articles on the same topic?

Usually, the articles will be quite different in the details: we probably cannot expect exact translations of sentences in one article to be present in the other article. We confirmed this hypothesis by running an initial test on a small number of article pairs, in which we searched for sentence pairs for which a t fraction of the words are present in both sentences (according to a dictionary translation). We found that for high thresholds (e.g. $t = 0.9$), no sentence pairs were found. For lower thresholds, such as $t = 0.5$, the resulting pairs of sentences are very specific (for example first sentences of the articles), which is not representative of the dataset. If we further decreased t do avoid such issues, we observed that very dissimilar sentence pairs were found.

Given this observation, we took the approach (described in the Introduction) of trying to project any subtree of a source language sentence onto each target sentence. Our method is to try to align

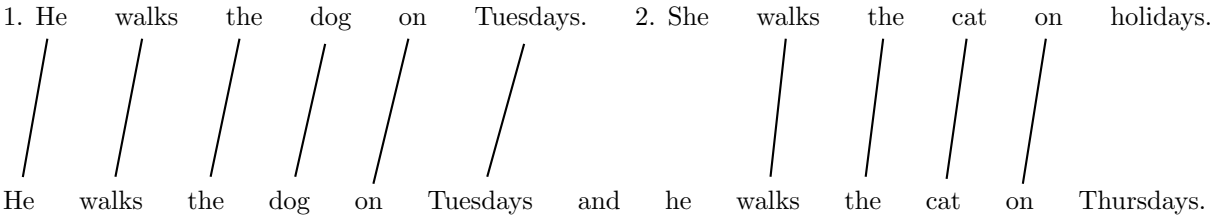

Figure 3: Projecting from multiple source sentences can be useful.

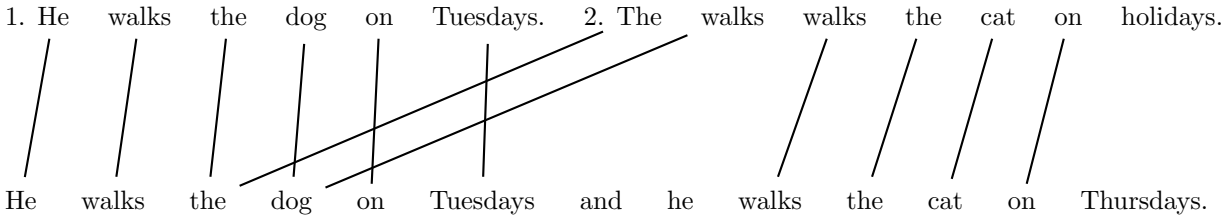

Figure 4: Projecting from multiple source sentences can be problematic.

every pair of (source, target)-language sentences. If sufficiently many aligned words are found, we then project dependencies *among the aligned words* from the source sentence to the target sentence.

3.2.1 Alignment

The alignment process for a sentence pair starts by looking at each word w_s in the source sentence: if there is a unique word w_t in the target sentence such that w_t is a translation of w_s according to the dictionary, then we align w_s and w_t.

After this initial pass, we deal with aligning source words w_s for which there are multiple words $w_{t_1}, \ldots, w_{t_m}$ in the target sentence such that w_{t_i} is a translation of w_s according to the dictionary. We remark that this is a very common scenario: for example, determiners such as the commonly occur many times in a single sentence. Our algorithm aligns each such source word w_s to exactly one target word w_{t_i} such that:

$$i^* = \operatorname*{arg\,min}_{j \in \{1,\ldots,m\}} d(w_s, w_{t_j}),$$

where d is a distance metric that we define below.

The distance metric The basic idea underlying our distance metric is that we want to align w_s to the target word w_{t_i} that has the most similar context (i.e. surrounding words) to the context of w_s. To measure the similarity of the contexts, we take into account the following two aspects.

In the following, window is the number of words we include in the context. In our experiments we have used $\mathsf{window} = 5$. In this case the context of w_s consists of the two preceding words w_s^2, w_s^1, the word w_s itself, and the two following words w_s^1, w_s^2.

- *Distance to neighboring words.* We would like to design a distance metric based on the similarity of words w which are present in the context of both w_s and w_t. Recall that w_s and w_t are in different languages, so to be precise, we consider word pairs (w'_s, w'_t) such that w'_s is in the context of w_s and w'_t is in the context of w_t, and w'_s has already been aligned to w'_t. Let W be the set of all such shared neighbors (w'_s, w'_t). Let $\Delta S(w'_s, w_s)$ denote the distance between w'_s and w_s in the source sentence (i.e. how many words apart they are), and let $\Delta T(w'_t, w_t)$ denote the distance between w'_t and w_t in the target sentence. We define a distance metric $d_{\mathsf{Manhattan}}$ as follows:

$$d_{\mathsf{Manhattan}} = \sum_{(w'_s, w'_t) \in W} \left|\Delta_S(w'_s, w_s) - \Delta_T(w'_t, w_t)\right| + \mathsf{window} - |W|$$

  Note that the additive term $\mathsf{window}|W|$ is designed to increase the distance when w_s and w_t do not have many shared neighbors.

- *Context as a word vectors.* Another metric we tried uses word vectors [4] of the word and its context. Specifically, for a word w_s in the source language, we concatenate the word vectors of its context i.e. $(w_s^{-2}, w_s^{-1}, w_s, w_s^1, w_s^2)$ to get the vector representation of its context $v_{source}(s)$. Similarly, for a word in target language w_t we get its context vector representation $v_{target}(t)$. Although cosine distance is a good measure for semantic similarity, the word vectors are trained separately in source and target languages. Therefore, we have used CCA to embed the word vectors of English and German into a common semantic space.

Note that throughout this section we assumed that we have a dictionary that will map words from source language to the target one. In general, this is a task by itself, because dictionaries have normal forms of the words, whereas many languages have multiple noun cases and verb forms.

We used a stemmer called Snowball from the NLTK package [1] to overcome this issue. Specifically, for each stem work s let W_s be the set of words in the dictionary that have the same stem s. Now, if w is a word in the source language with stem s, then W_s is the set of possible words in their dictionary form that w can match to. We use edit distance to pick the actual match. Our experiments show that this method works well for our purposes.

3.3 Projection of dependency arcs

Once alignments have been made (according to the algorithm above) for every pair of source and target sentences, any dependency arcs between aligned words in the source sentence are projected onto the corresponding words in the target sentence. In this way, we build up a list of projected arcs for each target sentence.

Then, for each target sentence, we resolve any conflicts between the projected arcs (recall Figure 4). First notice that taking a directed maximum spanning tree of the resulting set of projected arcs is not optimal, because such approach will result in trees where arcs come from different languages. Therefore, we decided to heuristically remove complete trees until we reach a conflict-free dependency tree.

```
k := 1, where k is the number of trees to be removed.
While there are conflicts
    Let S be a subset of size k of the set of projected subtrees
    If removing subtrees in S results in a conflict-free forest
        Return that forest
    Increase k after trying all possible subsets S
```

Notice that trying all possible subsets may be too slow. If there are too many such subsets, in each iteration we try to remove a random subset of size k.

3.4 Training a parser

After heuristically projecting dependency trees onto the target language, we train an MST parser described in [3] using the projected subtrees. The MST parser assigns a score to each possible edge and then uses an MST algorithm for directed graphs to choose the best dependency tree.

We use RBG parser for English and German to generate a "gold" tree bank for the source language to be projected, and for evaluation. RBG parser is used in this case but not before, because it is slower but more accurate. The accuracies of the parser for both English and German are about 87% and 85% respectively.

Table 1: The main results our parser achieves.

	#total	#train	precision	accuracy
Europarl	∼2M	50K	∼75%	∼61%
Wikipedia	∼80K	5K	∼64%	∼41%

Table 2: Some examples of precision break-down for each type of arc.

Arc type	$(verb \rightarrow noun)$	$(noun \rightarrow conj)$	$(noun \rightarrow adj)$	$(noun \rightarrow det)$
Precision	0.838	0.800	0.943	0.846

4 Evaluation

Table 1 summarizes the main results we achieved. As noted earlier, we have run the experiements on smaller dataset than available.

Notice that on Europarl data set our method achieves an accuracy of 61%. This is close to the accuracy of 62.5% gained in state-of-the-art unsupervized methods in [10, 7]. Therefore, we believe that our approach has potential.

Table 2 shows that for some types of arcs the precision is very high (more than 90%).

Although on Europarl dataset our method achieves a reasonable results, our initial goal was to use Wikipedia articles. The next section shows a number of observations we made, and for the sake of speed and simplicity, we have used 1K Europarl sentences for the following experiments:

Higher precision is higher accuracy? We want to examine to what extent precision is translated to the final parsing accuracy, and this question is specially relevant on noisy corpora such as Wikipedia. Interestingly, we found out the existence of overfitting by accident.

We did two experiments: one with the normal settting, and the one with the training data (projections) swapped with golden annotations (provided by the German parser). So on the one hand, we have the projected arcs with precision 75%, and on the other hand, we have the perfectly labeled arcs[2]. Surprisingly, our projections led to 55% accuracy, but golden ones only 47%. Since we are using different datasets for training (Europarl) and testing (Universal Dependencies), this is a strong sign of overfitting on the training corpus.

To test our hypothesis, we split the projections from these 1K sentences into two, and use half for training, half for testing (automatically annotated by our golden parser as well). The accuracy is higher for the golden annotations. This shows that higher precision does lead to higher accuracy, but only on the same corpus, which suggests that more caution and care should be applied if we should build a parser that transfers well across corpora.

More projections is better? Again, this is related to overfitting. We found that the majority of projections only form two-node subtrees consisting essentially of one arc. Specifically we get from 1K Europarl sentences a distribution of arc counts from projected subtrees as in Table 2.

Table 3: Number of projections with respect to the number of arcs in Europral 1K sentences.

#Arcs	1	2	3	4	5	6	7	8	9	10	11	12+
#Projections	620	239	98	56	24	16	7	4	3	2	1	0

[2]The actual accuracy of our gold German parser is 85%.

This extremely unbalanced distribution has two downsides. First of all, from the optimization's perspective, small trees do not provide enough signals to drive the algorithm to find the correct optima in the objective landscape. For instance, to parse a tree consisting of two nodes, the parser only has to make the decision about the direction of the single possible edge, but as the tree size grows, it has to choose from an exponentially increasing number of well-formed spanning trees. Secondly, also related to the first point, too many arcs with not so strong signals as supervision can be prone to overfitting. To justify this, we filtered out all subtrees with only one arc, and ended up with a much smaller number of projections, but with more arcs per subtree. Normally we would suspect that the accuracy will go down, because the model has definitely not reached the 'overfitting' turning point, with 1K sentences at only 55% accuracy. However, even with a substantially fewer number of arcs, the model still manages to achieve 56% accuracy, 1% higher than without filtering. We also trained the model with the golden projections, similar to what we did in the last paragraph, which already showed signs of overfitting on Europarl corpus, and also got an increase in accuracy, from 47% to 52%. This suggests that even though we generally need more arcs (or just training data) to improve the parser, it is nonetheless harmful to just bloat up the training dataset without much actual signal.

Does delexicalization help? All the previous results are produced by lexicalized parsers, which uses lexical information[3]. Since we are using different corpora for training and testing, delexicalization might help alleviate part of the overfitting problem. Inspired by this idea, we trained a delexicalized parser with all other settings unchanged, and the accuracy only dropped 1 percent. Note that delexicalization results in a much smaller number of features[4], and the accuracy is still comparable, suggesting we might actually gain by delexicalization or partial lexicalization[5] when building cross-lingual parsers.

Are word vectors useful at all? When finding the best match, word-vector-based method produces fewer (about half) arcs, but 1 or 2 precent higher precision. The final accuracy, however, is 51%, 4 percent lower than using Manhattan distance. In addition, preliminary results show that a bigger context window (say 7, instead of 5) performs noticeably worse. One explanation is that while cosine distance is good measurement of semantic closeness, it might not be so when a concatenation of words are concerned. The reason is that concatenation does not deal with compositionality properly, and a more sophisticated method might consider using a transform matrix[6] to further model the interaction between words in context.

5 Conclusion

We have developed methods for projecting dependency arcs across comparable sentences. Our preliminary experiments demonstrate that these method achieve an accuracy of 61% with precision of 75% when run on truly parallel (Europarl) data, and achieve an accuracy of 41% and precision of 64% on articles from Wikipedia.

In both cases, due to training time and time constrains we were facing, we didn't run on all available data. It would be interesting to see who much performance improves when running on

[3]Namely one feature for each word in the vocab, and any combination that involves any word

[4]16K features compared to 80K.

[5]by limiting the size of vocabulary

[6]Which however, needs to be trained.

larger datasets, both in terms of the size of the source-language corpora and also the number of source languages.

Also, our experiments so far were performed with quite closely linguistically related languages. We believe an interesting future direction would be to evaluate the performance of these techniques when the target language is linguistically less close to the source languages, which will be an important further indicator of the suitability of these techniques for parsing low-resource languages.

References

[1] Steven Bird. "NLTK: the natural language toolkit". In: *Proceedings of the COLING/ACL on Interactive presentation sessions*. Association for Computational Linguistics. 2006, pp. 69–72.

[2] Philipp Koehn. "Europarl: A Parallel Corpus for Statistical Machine Translation". In: *Conference Proceedings: the tenth Machine Translation Summit*. AAMT. Phuket, Thailand: AAMT, 2005, pp. 79–86. URL: `http://mt-archive.info/MTS-2005-Koehn.pdf`.

[3] Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. "Non-projective dependency parsing using spanning tree algorithms". In: *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. 2005, pp. 523–530.

[4] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*. 2013, pp. 3111–3119.

[5] Tahira Naseem, Regina Barzilay, and Amir Globerson. "Selective sharing for multilingual dependency parsing". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics. 2012, pp. 629–637.

[6] Mohammad Sadegh Rasooli and Michael Collins. "Density-Driven Cross-Lingual Transfer of Dependency Parsers". In: ().

[7] Oscar Täckström, Ryan McDonald, and Joakim Nivre. "Target language adaptation of discriminative transfer parsers". In: (2013).

[8] *Wikipedia*. `https://www.wikipedia.org`.

[9] *Wiktionary*. `https://www.wiktionary.org`.

[10] Yuan Zhang and Regina Barzilay. "Hierarchical Low-Rank Tensors for Multilingual Transfer Parsing". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, 2015, pp. 1857–1867. URL: `https://aclweb.org/anthology/D/D15/D15-1213`.