Proposal of CoNLL 2017 Shared Task in Multi-lingual Dependency Parsing

Organizing Committee

Chair

Jan Hajič, Charles University, Prague <hajic@ufal.mff.cuni.cz>

Management Group

Joakim Nivre, Uppsala University <joakim.nivre@lingfil.uu.se>
Daniel Zeman, Charles University, Prague <zeman@ufal.mff.cuni.cz>
Filip Ginter, University of Turku <ginter@cs.utu.fi>
Slav Petrov, Google <slav@google.com>
Milan Straka, Charles University, Prague <straka@ufal.mff.cuni.cz>
Martin Popel, Charles University, Prague <popel@ufal.mff.cuni.cz>

Support Group

Christopher Manning, Stanford University
Marie-Catherine de Marneffe, Ohio State University
Yoav Goldberg, Bar Ilan University
Reut Tsarfaty, Weizmann Institute of Science
Sampo Pyysalo, University of Cambridge
Francis Tyers, UiT / The Arctic University of Norway
Jenna Kanerva, University of Turku
Lilja Øvrelid, University of Oslo
Giuseppe Celano, University of Leipzig
Miguel Ballesteros, Pompeu Fabra University
Çağrı Çöltekin, University of Tübingen
Tommi Pirinen, University of Hamburg
Paola Merlo, University of Geneva
Anssi Yli-Jyrä, University of Helsinki
Özlem Çetinoğlu, University of Stuttgart
Juhani Luotolahti, University of Turku

Description of the Shared Task

Learning dependency parsers is a popular sub-task of natural language learning, useful in downstream applications as well as in linguistic research. Besides parsing proper, this task should address two related problems: working with languages for which little or no data is available, and working in a real-world setting, without manually disambiguated POS tags, lemmas etc.

Participating systems will have to find syntactic dependencies between words, i.e. a *parent* of each word, and the type (label) of the dependency *relation*. There will be multiple test sets in various languages but all the data sets will adhere to one common annotation style: Universal Dependencies (UD, http://universaldependencies.org/).

Participants will be asked to parse raw text where no gold-standard annotation (tokenization, lemmas, morphology) is available. The only exception is sentence boundaries, which will be provided. However, there is an open-source baseline pipeline (https://ufal.mff.cuni.cz/udpipe/) that the participants can run instead of training their own model for e.g. tokenization or tagging. We will even provide variants of the test data that have been automatically tokenized and POS-tagged by UDPipe. We believe that this makes the task reasonably accessible for everyone.

We do not plan on running separate open and closed tracks. Instead, we want to include every system in a single track, which will be formally closed, but we intend to make the list of permitted resources rather broad; see below for more on the data selection process.

Data

The task will only work with resources that are publicly available, royalty-free, under a license that is free at least for non-commercial usage (e.g. CC BY-NC-SA, BY-SA etc.) The right to use the data must not be limited to the shared task, i.e. the data must be available for follow-up research, too.

Treebanks

The main data sets will be taken from the current version of the Universal Dependencies treebanks. Note that this means that the test data will be known in advance and we have to trust the participants not to take advantage of it. We hope to be able to provide for some languages additional test sets that have not been previously released. These sets will not necessarily be from the same domain as the training data for the given language, but they will adhere to the UD annotation guidelines. Gold standard data from these sets will be only made available after the evaluation phase. Finally, there will be one or more **surprise language(s),** which have not been previously released in UD and for which we will not provide training or development data, except a small sample at the beginning of the evaluation phase. Gold-standard data of the surprise language(s) will be also made available after the shared task.

A conservative estimate is that we will be able to evaluate the systems on 15-20 languages. We will only include UD treebanks which pass validation tests for the current UD guidelines, and for which we can obtain a test set of at least 10,000 words. There is no upper limit on the test size (the largest test set is currently ~170K). Participants will receive training+development data with gold-standard tokenization, POS tags and dependency relations; for some languages also lemmas and/or morphological features. The size of this data will vary according to availability. For some languages it may be as small as the test data (or even smaller), for others it will be 9× larger then the test data, for the surprise languages it will be close to zero. One part of this data will be formally designated as the development data, but the participants will be free to use it for training of their systems too.

Raw Data

We will provide additional raw data for the languages of the shared task, useful e.g. for computation of word embeddings. This data will be taken from CommonCrawl and automatically sorted by a language recognizer. It may not be available for all languages (note that UD contains also some classical languages such as Ancient Greek) but we are confident that we will be able to provide more than 100M words for most languages. For convenience, we will provide a variant of this data pre-processed by UDPipe, and also pre-computed word embedding vectors for those participants who want to use them but do not want to tweak their own setting of the word-to-vector software.

Parallel Data

To support multi-lingual and cross-lingual approaches and model transfers, participants are allowed to use data from the OPUS parallel corpus (http://opus.lingfil.uu.se/). We will not redistribute this data; participants are simply referred to the OPUS website.

Call for Additional Data

Instead of organizing a separate open track we will encourage the participants to report (by the end of December) additional data they want to use. If the data is relevant to the task and it meets the public availability condition, it will be added to the list of resources available to participants.

Evaluation of Participating Systems

All systems will be required to generate valid output in the CoNLL-U format for all test sets. They will know the language of the test set, but they must respond even to unknown language codes (for which there are no training data). The systems will be able to select either raw text as input, or one of the alternatives pre-processed by UDPipe (system description papers should indicate what kind of input data the system uses). Participants may choose to test their system on multiple input alternatives, for example to have their own tokenization compared with the baseline tokenization. On each test set, one system may produce up to one output per input alternative, but it must produce valid output for at least one input alternative. We will publish results of all runs of a system, but only the best run will be used in the main system score.

The evaluation will focus on dependency relations, i.e. index of the parent node and the label of the relation. POS tags, lemmas and morphological features are not part of the main evaluation metric, although the systems are free to predict them too. On the other hand, tokenization must be included in the metric because the systems cannot access gold-standard tokenization, and tokens are necessary for dependency evaluation.

The evaluation starts by aligning the system-produced tokens to the gold standard ones; longest common subsequence will be used as matching strategy (see Annex for details). We completely ignore sentence breaks during tokenizer evaluation.

Once tokens (words) are aligned, we will compute two metrics: LAS *(labeled attachment score)* and CNC (*core-non-core score*, a draft is described in http://stp.lingfil.uu.se/~nivre/docs/udeval-cl.pdf but it will be further refined for the shared task). TBD is the main metric that will be used in the main system ranking: the best score of the system for each test set will be taken, and the arithmetic mean of these scores will be the overall score of the system.

LAS *(labeled attachment score)* is a standard evaluation metric in dependency parsing: the number of correct relations is divided by the number of nodes, where each node has just one incoming relation, and the relation is correct if 1. the parent node is identified correctly, and 2. the dependency relation type (label) is assigned correctly. (Note that UD uses the notion of language-specific relation subtypes but we will evaluate only the universal part of the label. For instance, if either the system output or the gold standard data have the label “acl:relcl”, we will disregard the “:relcl” part and only check whether “acl” matches.) In our configuration, the LAS will be modified to take tokenization mismatches into account. A relation is correct only if both nodes of the relation match existing gold-standard nodes. Precision P is the number of correct relations divided by the number of system-produced nodes; recall R is the number of correct relations divided by the number of gold-standard nodes. We define LAS as the F_1-score = 2PR / (P+R).

CNC reflects the fact that Universal Dependencies annotation is centered around relations between content words. Relation attaching function words can be seen as a substitute for what is achieved by morphology in other languages. Since we do not evaluate morphological analysis in this task, it makes sense to also look at results without function words and punctuation. Similarly to our extended LAS metric, CNC is computed as F_1-score of precision and recall. The difference to LAS is that there is a pre-defined list of *core* dependency relations, which will be evaluated. Precision P is the number of correct core relations divided by the number of system produced core relations; recall R is the number of correct core relations divided by the number of gold-standard core relations. Again, a relation is not correct if one of the word forms does not match the gold standard, i.e. if there are tokenization errors.

The evaluation script is not ready at the time of writing this proposal but it will be publicly available by the end of December at the latest.

We plan to use the Tira platform (http://www.tira.io/) as suggested in the Call for proposals. Therefore, the participants will submit the systems, not parsed data, allowing us not to give the testing data beforehand (only after the Shared Task is evaluated).

Timeline

We agree to follow the timeline suggested in the Call. Trial data (not necessarily for all languages) will be available in February 2017, train+dev in March and test data in May. No copyright-related issues have to be solved; the time between now and March will be mostly needed to improve existing datasets and their compliance with the UD guidelines.

Other Relevant Information

There have been two CoNLL shared tasks in dependency parsing in 2006 and 2007. The current proposal differs from the previous tasks in several respects. We have treebanks in many languages that share the same annotation style, which makes multi-lingual approaches possible. We propose a new evaluation metric tailored to the content-word-centric UD style. And finally, we evaluate end-to-end parsing with no gold-standard information available to the parsers.

Annex: Data format and evaluation details

TBA

The CoNLL-U data format is described in more detail at http://universaldependencies.org/format.html. It is deliberately similar to the CoNLL-X format that was used in the CoNLL 2006 Shared Task and has become a de-facto standard since then. However, there are a few important extensions. Perhaps most important is the notion of *syntactic words* vs. *multi-word tokens*. It makes the tokenization step in UD harder than the relatively simple procedure called tokenization in other areas of NLP. For instance, German *zum* is a contraction of the preposition *zu* “to”, and the article *dem* “the”. In UD it is a multi-word token consisting of two syntactic words, *zu* and *dem*. These syntactic words are nodes in dependency relations. Learning this is harder than separating punctuation from words, because a contraction is not a pure concatenation of the participating words. The CoNLL-U format uses two different mechanisms here: punctuation that is conventionally written adjacent to a word is a separate single-“word” token, and an attribute in the last column tells that there was no whitespace character between the punctuation symbol and the word. On the other hand, the contraction is a multi-word token which has a separate line starting with range of following syntactic words that belong to it. Consider a German phrase *zur Stadt, zum Haus* “to the city, to the house”. The corresponding CoNLL-U section could look like this:

```
1-2   zur    _   _      _   _   _   _      _   _
1     zu     _   ADP    _   _   3   case   _   _
2     der    _   DET    _   _   3   det    _   _
3     Stadt  _   NOUN_      _   0   root   _   SpaceAfter=No
4     ,      _   PUNCT _    _   3   punct  _   _
5-6   zum    _   _      _   _   _   _      _   _
5     zu     _   ADP    _   _   7   case   _   _
6     dem    _   DET    _   _   7   det    _   _
7     Haus   _   NOUN_      _   3   conj   _   _
```

We will not evaluate whether the system correctly generated the range lines (1-2 zur and 5-6 zum, respectively), nor whether it generated the SpaceAfter=No attribute. But we will have to align the nodes (syntactic words) output by the system to those in the gold standard data. Thus if the system fails to recognize *zur* as a contraction and outputs

```
1     zur
2     Stadt
3     ,
```

we will treat any relations going to or from the node *zur* as incorrect. The same will happen with the node “Stadt,”, should the system fail to separate punctuation from the word *Stadt*.

If the system wrongly splits the word *Haus* and outputs

7-8	Haus
7	Hau
8	das

relations involving either *Hau* or *das* will be considered incorrect.

Even if the system recognizes *zur* as contraction but outputs wrong syntactic word forms, the tokens will be considered incorrect:

1-2	zur
1	zur
2	der

Relations involving node 1 are incorrect but relations involving node 2 may be correct.

In general, it may not be clear what words should be aligned to each other. Consider the following two sequences of word forms:

Gold: zu der Stadt , zu dem Haus

System: xx der zu Stadt , dem dem Haus

We will find the alignment as the longest common subsequence (LCS) of the two sequences; if more than one element of a sequence matches the next element of the LCS, the leftmost candidate will be selected. Thus in the above example, there are two possible LCS of the length 5:

LCS_1: zu Stadt , dem Haus

LCS_2: der Stadt , dem Haus

The algorithm will prefer the LCS that starts earlier in the Gold sequence, i.e. LCS_1. In the System sequence, the first *dem* will be aligned and the second will be discarded. Thus the entire alignment, expressed as Gold-System index pairs, will be

1-3 3-4 4-5 6-6 7-8

TBD: What about the sentence breaks? Milan proposed that we will align tokens without observing sentence boundaries. That would allow us to hide the gold-standard sentence breaks from the systems (and I should remove from the proposal the sentence that we will make them available). A dependency relation that goes across a gold sentence boundary would be automatically incorrect. If on the other hand the system generates a false sentence break, it will not be penalized directly, but there must be at least one gold relation that the system did not find; not getting points for such relations will be indirect penalization for wrong sentence segmentation.