Two baselines for unsupervised dependency parsing*

Anders Søgaard
Center for Language Technology
University of Copenhagen
DK-2300 Copenhagen S
soegaard@hum.ku.dk

Abstract

Results in unsupervised dependency parsing are typically compared to branching baselines and the DMV-EM parser of Klein and Manning (2004). State-of-the-art results are now well beyond these baselines. This paper describes two simple, heuristic baselines that are much harder to beat: a simple, heuristic algorithm recently presented in Søgaard (2012) and a heuristic application of the universal rules presented in Naseem et al. (2010). Our first baseline (RANK) outperforms existing baselines, including PR-DVM (Gillenwater et al., 2010), while relying *only on raw text*, but all submitted systems in the Pascal Grammar Induction Challenge score better. Our second baseline (RULES), however, outperforms several submitted systems.

1 RANK: a simple heuristic baseline

Our first baseline RANK is a simple heuristic baseline that does not rely on part of speech. It only assumes raw text. The intuition behind it is that a dependency structure encodes something related to the relatively salience of words in a sentence (Søgaard, 2012). It constructs a word graph of the words in a sentence and applies a random walk algorithm to rank the words by salience. The word ranking is then converted into a dependency tree using a simple heuristic algorithm.

The graph over the words in the input sentence is constructed by adding directed edges between the word nodes. The edges are not weighted, but multiple edges between nodes will make transitions between them more likely.

The edge template was validated on development data from the English Penn-III treebank (Marcus et al., 1993) and first presented in Søgaard (2012):

- *Short edges.* To favor short dependencies, we add links between all words and their neighbors. This makes probability mass flow from central words to their neighboring words.

- *Function words.* We use a keyword extraction algorithm without stop word lists to extract function or non-content words. The algorithm is a crude simplification of TextRank (Mihalcea and Tarau, 2004) that does not rely on linguistic resources, so that we can easily apply it to low-resource languages. Since we do not use stop word lists, highly ranked words will typically be function words. For the 50-most highly ranked words, we add additional links from their neighboring words. This will add additional probability mass to the function words. This is relevant to capture structures such as prepositional phrases where the function words take content words as complements.

- *Morphological inequality.* If two words w_i, w_j have different prefixes or suffixes, i.e. the first two or last three letters, we add an edge between them.

Given the constructed graph we rank the nodes using the algorithm in Page and Brin (1998), also known as PageRank. The input to the PageRank algorithm is any directed graph $G = \langle E, V \rangle$ and the output is an assignment $PR : V \rightarrow \mathbb{R}$ of a score, also referred to as PageRank, to each node in the graph, reflecting the probability of ending up in that node in a random walk.

*

NAACL-HLT Workshop on the Induction of Linguistic Structure, pages 81–83,
Montréal, Canada, June 3-8, 2012.

from/to	The	finger-pointing	has	already	begun	.
The	0	3	2	2	3	2
finger-pointing	3	0	5	2	3	2
has	2	4	0	3	3	2
already	2	2	5	0	3	2
begun	2	3	3	3	0	3
.	2	2	3	2	4	0
PR(%)	13.4	17.4	21.2	15.1	19.3	13.6

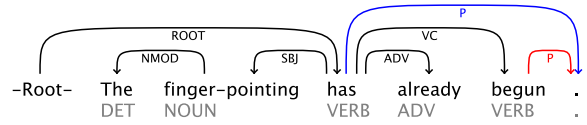

Figure 1: Graph, pagerank (PR) and predicted dependency structure for sentence 7 in PTB-III Sect. 23.

The words are now ranked by their PageRank (Figure 1), and from the word ranking we derive a dependency tree. The derivation is very simple: We introduce a store of potential heads, initialized as a singleton containing the word with the highest PageRank (which is attached to the artificial root note). Each word is now assigned a syntactic head taken from all the words that were already assigned heads. Of these words, we simply select the closest possible head. In case of ties, we select the head with the highest PageRank.

2 RULES: a simple rule-based baseline

Our second baseline is even simpler than our first one, but makes use of input part of speech. In particular it builds on the idea that unsupervised parsing can be informed by universal dependency rules (Naseem et al., 2010). We reformulate the universal dependency rules used in Naseem et al. (2010) in terms of the universal tags provided in the shared task (Figure 2), but unlike them, we do not engage in grammar induction. Instead we simply present a straight-forward heuristic application of the universal dependency rules:

RULES finds the head of each word w by finding the nearest word w' such that POS(w')→POS(w) is a universal dependency rule. In case of ties, we select the left-most head in the candidate set. The head of the sentence is said to be the left-most verb. Note that we are not guaranteed to find a head satisfying a universal dependency rule. In fact when the dependent has part of speech AUX or '.' we will never find such a head. If no head is found, we attach the dependent to the artificial root node.

Note that like RANK, RULES would give us the correct analysis of the sentence in Figure 1 (excl. punctuation). Surprisingly, RULES turns out to be a *very* competitive baseline.

VERB⟶VERB	NOUN⟶ADJ
VERB⟶NOUN	NOUN⟶DET
VERB⟶ADV	NOUN⟶NOUN
VERB⟶ADP	NOUN⟶NUM
VERB⟶CONJ	
VERB⟶DET	
VERB⟶NUM	
VERB⟶ADJ	
VERB⟶X	
ADP⟶NOUN	ADJ⟶ADV
ADP⟶ADV	

Figure 2: Universal dependency rules (Naseem et al., 2010) wrt. universal tags.

	RANK	RULES	DMV	win	best
Arabic	0.340	0.465	0.274	0.541	0.573
Basque	0.255	0.137	0.321	0.440	0.459
Czech	0.329	0.409	0.276	0.488	0.491
Danish	0.424	0.451	0.395	0.502	0.502
Dutch	0.313	0.405	0.284	0.437	0.492
En-Childes	0.481	0.519	0.498	0.538	0.594
En-WSJ	0.328	0.425	0.335	0.555	0.560
Portuguese	0.371	0.546	0.240	0.418	0.652
Slovene	0.284	0.377	0.242	0.580	0.580
Swedish	0.375	0.551	0.290	0.573	0.573

3 Results

Shared task results were evaluated by the organizers in terms of directed accuracy (DA), also known as unlabeled attachment score, undirected accuracy (UA) and NED (Schwartz et al., 2011), both for short and full length sentences. We will focus on DA for full length sentences here, arguable the most widely accepted metric. Table 1 presents results for all 10 datasets, with DMV based on fine-grained native POS (which performs best on average compared to DMV-CPOS and DMV-UPOS),[1] and *Tu, standard* as the winning system ('win'). The 'best' result cherry-picks the best system for each dataset.

The first thing we note is that our two baselines

[1] In a way it would be fairer to *exclude* native POS and CPOS information, since native tag sets reflect language-specific syntax. Moreover, the validity of relying on manually labeled input is questionable.

are much better than the usual structural baselines. The macro-averages for the branching baselines are 0.252 (left) and 0.295 (right), but if we allow ourselves to cherry-pick the best branching baseline for each language the macro-average of that baseline is 0.352. This corresponds to the macro-average of RANK which is 0.350. The macro-average of RULES is 0.429.

Interestingly, RANK achieves better full length sentence DA than at least one of the submitted systems for each language, except English. The same holds for full length sentence NED. RULES is an even stronger baseline.

Most interestingly the two baselines are significantly better on average than *all* the baselines proposed by the organizers, including DMV-EM and DMV-PR. This is surprising in itself, since our two baselines are completely heuristic and require no training. It seems none of the baseline systems necessarily learn anything apart from simple, universal properties of linguistic trees that we could easily have spelled out in the first place.

More than half of the submitted systems are worse than RULES in terms of DA, but three systems also outperform our baselines by some margin (Bisk, Blunsom and Tu). Since our baselines are better than harmonic initialization, the obvious next step would be to try to initialize EM-based unsupervised parsers by the structures predicted by our baselines.

References

Jennifer Gillenwater, Kuzman Ganchev, Joao Graca, Fernando Pereira, and Ben Taskar. 2010. Sparsity in dependency grammar induction. In *ACL*.

Mitchell Marcus, Mary Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Rada Mihalcea and Paul Tarau. 2004. Textrank: bringing order into texts. In *EMNLP*.

Tahira Naseem, Harr Chen, Regina Barzilay, and Mark Johnson. 2010. Using universal linguistic knowledge to guide grammar induction. In *EMNLP*.

Larry Page and Sergey Brin. 1998. The anatomy of a large-scale hypertextual web search engine. In *International Web Conference*.

Roy Schwartz, , Omri Abend, Roi Reichart, and Ari Rappoport. 2011. Neutralizing linguistically problematic annotations in unsupervised dependency parsing evaluation. In *ACL*.

Anders Søgaard. 2012. Unsupervised dependency parsing without training. *Natural Language Engineering*, 18(1):187–203.