

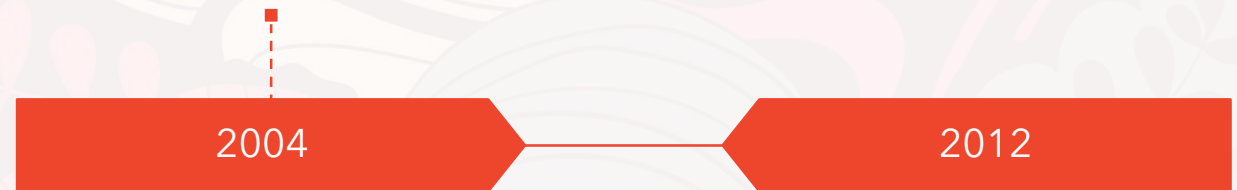


CSC2517

Ming Feng Wan

Topic: Unsupervised Dependency Model

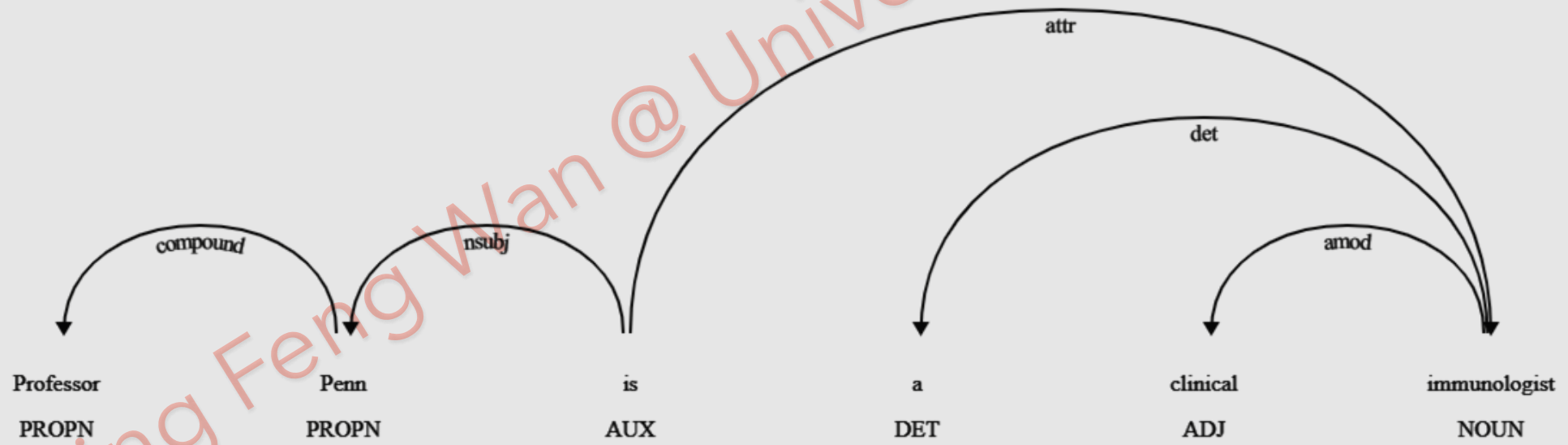
Corpus-Based Induction of Syntactic
Structure: Models of Dependency and
Constituency



Unsupervised dependency parsing
without training

Recap: Dependency Grammar

The notion that linguistic units, e.g. words, are connected to each other by directed links.





CORPUS-BASED INDUCTION OF SYNTACTIC STRUCTURE: MODELS OF DEPENDENCY AND CONSTITUENCY

Dan Klein, Christopher Manning (2004)

1 Introduction

- Induced models of linguistic constituency and dependency
- Recovered linguistically plausible structures
 - Undermined arguments based on “the poverty of the stimulus”* (Chomsky, 1965)
 - * The argument from linguistics that children are not exposed to rich enough data within their linguistic environments to acquire every feature of their language.

Ming Feng Wan @ University of Toronto

2.1 Representation and Evaluation

- The quality of a hypothesized dependency structure can be evaluated by accuracy as compared to a gold-standard dependency structure
- Where possible, report an accuracy figure for both directed and undirected dependencies

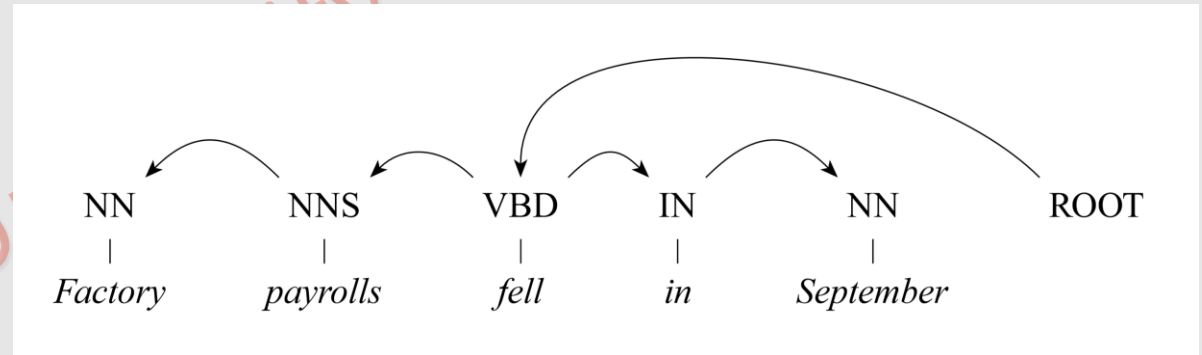
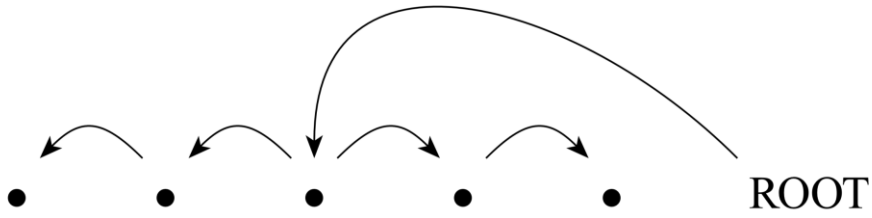


Figure 1(b)

In the dependency structure of figure 1(b), the dependencies are $\{(ROOT, fell), (fell, payrolls), (fell, in), (in, September), (payrolls, Factory)\}$.

2.2 Dependency Model

- Existing unsupervised generative dependency models: first generate a word-free graph G , then populate the sentence s conditioned on G .
- Trained on over 30M words of raw newswire, using EM in an entirely unsupervised fashion
- The resulting parser predicted dependencies at below chance level
- Incapable of encoding even first-order valence facts

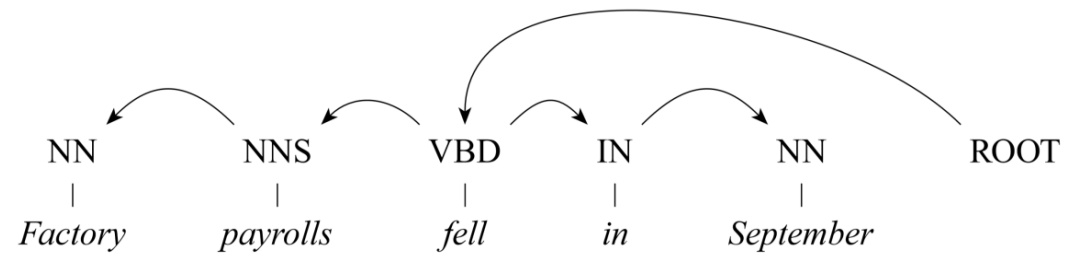


$P(D) = P(s, G)$
 $= P(G)P(s|G)$
 $= P(G) \prod_{(i,j,dir) \in G} P(i_{-1}s_i | j_{-1}s_j, dir)$

Notation: $i_{-1}s_i$: i th word of the sentence, dir : direction of the dependency (left or right)

3 An Improved Dependency Model

- DMV (dependency model with valence)
- Introduces the concept of "STOP" to unsupervised dependency model
- Begin at the ROOT. In the standard way, each head generates a series of non-STOP arguments to one side, then a STOP argument to that side, then non-STOP arguments to the other side, then a second STOP.



Example

3 An Improved Dependency Model

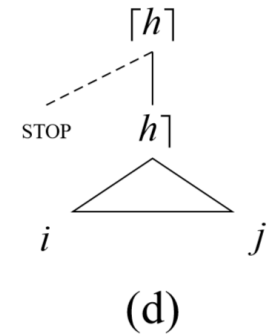
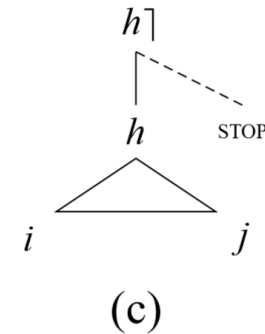
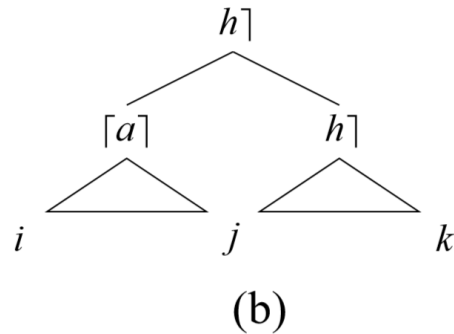
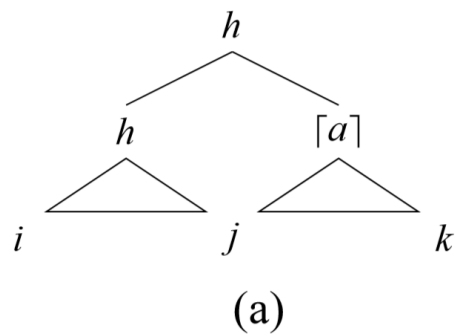
- Two kinds of derivation events
- First, the decision to terminate (generate STOP) or not:
 $P_{\text{STOP}}(\text{STOP}|h, dir, adj)$. If a stop is generated, no more arguments are generated for current head to current side.
- If stop is not generated, another argument is chosen using:
 $P_{\text{CHOOSE}}(a|h, dir)$.

- Formally, for a dependency structure D
 - $D(h)$: fragment of the dependency tree rooted at h
 - $deps_D(h, l)$: Left dependents
 - $deps_D(h, r)$: Right dependents.

$$P(D(h)) = \prod_{dir \in \{l, r\}} \prod_{a \in deps_D(h, dir)} P_{\text{STOP}}(\neg \text{STOP}|h, dir, adj) \\ P_{\text{CHOOSE}}(a|h, dir) P(D(a)) \\ P_{\text{STOP}}(\text{STOP}|h, dir, adj)$$

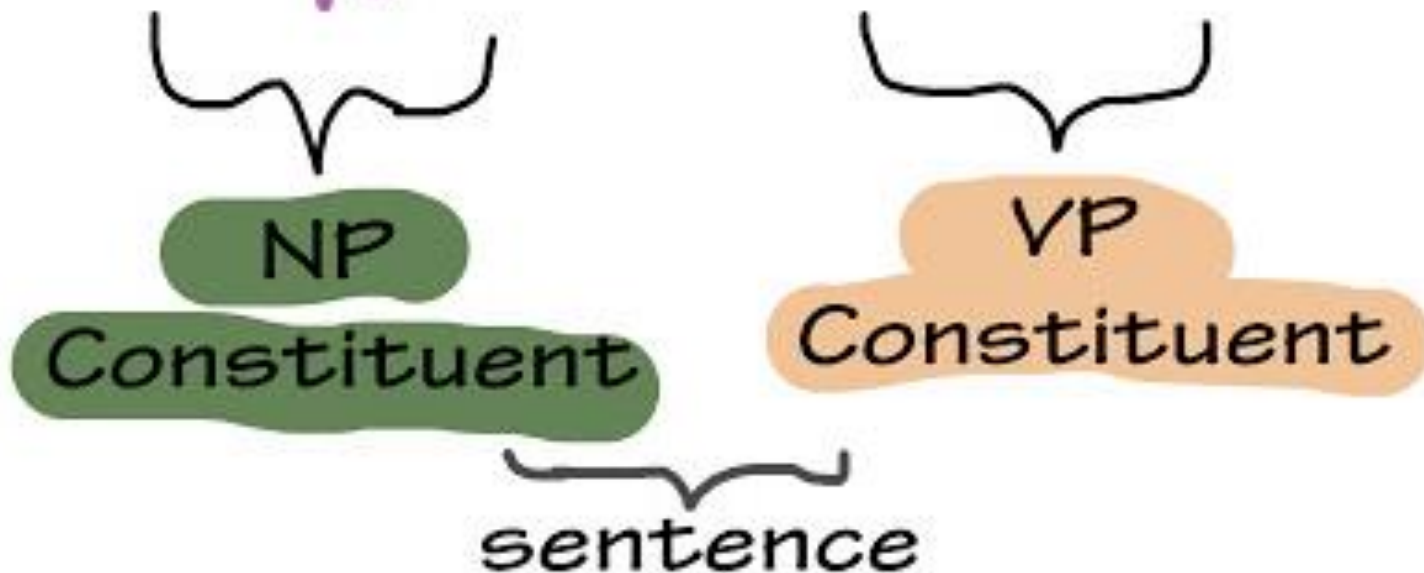
3 An Improved Dependency Model

- One can view a structure generated by this derivational process as a “lexicalized”* tree
- First published result to break the adjacent-word heuristic (at 33.6% for this corpus)
- This dependency induction model is reasonably successful. However, the model can be improved by paying more attention to syntactic constituency





The cat slept quickly.



Recap: Constituent

In syntactic analysis, a constituent is a word or a group of words that functions as a single unit within a hierarchical structure.

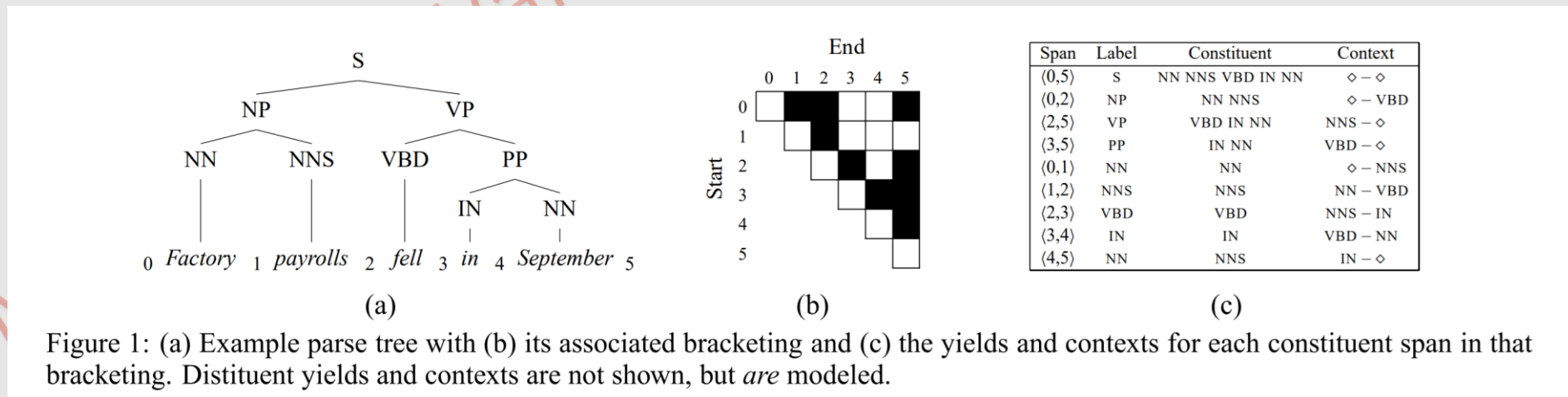
Example: NP, VP, PP

4 Distributional Constituency Induction

- Simple distributions over adjacent words can induce quite high-quality word classes, largely corresponding to traditional parts of speech (Finch, 1993; Schütze, 1995; Clark, 2000).
- Clark (2001) and Klein and Manning (2002) show that this approach can be successfully used for discovering syntactic constituents as well.
- Problem faced in Klein and Manning (2002) : it is easier to cluster word sequences (or word class sequences) than to tell how to put them together into trees.
- A constituent-context model (CCM) solves this problem by building constituency decisions directly into the distributional model

4 Distributional Constituency Induction

- Sentences are given as sequences S of word classes (parts-of-speech)
- $O(n^2)$ index pairs $\langle i, j \rangle$
- A bracketing of a sentence, Boolean matrix B
- The first stage is to choose a bracketing B for the sentence, which is a maximal non-crossing subset of the spans (equivalent to a binary tree).



4 Distributional Constituency Induction

1. Choose a bracketing B according to some distribution $P(B)$ and then generate the sentence given that bracketing:

$$P(S, B) = P(B)P(S|B)$$

2. All spans guess their sequences and contexts given only a constituency decision B_{ij} of that span.

$$P(s, B) = P(B) \prod_{\langle i, j \rangle} P(i s_j | b_{ij}) P(i-1 s_i \sim j s_{j+1} | b_{ij})$$

- This is a model $P(S, B)$ over hidden bracketings and observed sentences, and it is estimated via EM to maximize the sentence likelihoods $P(S)$ over the training corpus
- Basic CCM outperforms other recent systems on the ATIS corpus (which many other constituency induction systems have reported on)

5 A Combined Model

“Attachment” rewrite for span $\langle h, k \rangle$

$$\frac{P(i s_k | true) P(i-1 s_i \sim k s_{k+1} | true)}{P(i s_k | false) P(i-1 s_i \sim k s_{k+1} | false)}$$

If we multiply all trees’ attachment scores by

$$\prod_{\langle i, j \rangle} P(i s_j | false) P(i-1 s_i \sim j s_{j+1} | false)$$

we are left with each tree being assigned the probability it would have received under the CCM.

- CCM: good at recovering constituency, DMV: good at recovering dependency
- In the combined model, score each tree with the product of the probabilities from the individual models above

Model	UP	UR	UF ₁	Dir	Undir
English (WSJ10 – 7422 Sentences)					
LBRANCH/RHEAD	25.6	32.6	28.7	33.6	56.7
RANDOM	31.0	39.4	34.7	30.1	45.6
RBRANCH/LHEAD	55.1	70.0	61.7	24.0	55.9
DMV	46.6	59.2	52.1	43.2	62.7
CCM	64.2	81.6	71.9	23.8	43.3
DMV+CCM (POS)	69.3	88.0	77.6	47.5	64.5
DMV+CCM (DISTR.)	65.2	82.8	72.9	42.3	60.4
UBOUND	78.8	100.0	88.1	100.0	100.0

6 Conclusion

- Presented a successful new dependency-based model for the unsupervised induction of syntactic structure
- Then demonstrated how this model could be combined with the previous best constituent-induction model
- A key reason that these models are capable of recovering structure more accurately than previous work is that they minimize the amount of hidden structure that must be induced.
- It demonstrates that the broad constituent and dependency structure of a language can be recovered quite successfully (individually or, more effectively, jointly) from a very modest amount of training data.



UNSUPERVISED DEPENDENCY PARSING WITHOUT TRAINING

Anders Søgaard

1 Introduction

- The standard method in unsupervised dependency parsing is to optimize the overall probability of the corpus by assigning trees to its sentences that capture general patterns in the distribution of part-of-speech (POS)
- This paper presents a new and very different approach to unsupervised dependency parsing.
- The parser does not induce a model from a big corpus, but only considers the sentence in question.
- Has obvious advantage for not relying on training data

2 Ranking dependency tree nodes

- Main intuition: the nodes near the root in a dependency structure are in some sense the most important or the most central ones.
- The parser assigns a dependency structure to a sequence of words in two stages.
 - Stage 1
 - Decorates the n nodes of what will become our dependency structure with word forms
 - Constructs a directed acyclic graph from the nodes
 - Ranks them using iterative graph-based ranking (Brin and Page 1998).
 - Stage 2
 - Subsequently, it constructs a tree from the ranked list of words using a simple $O(n^2)$ parsing algorithm.

2.1 Edges

- The graph over the words in the input sentence is constructed by adding directed edges between the word nodes.
 - *Short edges*. Add links between all words and their neighbors.
 - *Function words*. Use a keyword extraction algorithm without stop word lists to extract function or non-content words. For the fifty most highly ranked words, add additional links from their neighboring words.
 - *Morphological inequality*. If two words have different prefixes or suffixes, i.e. the first two or last three letters, add an edge between them.
 - *Verb edges*. All words are attached to all words with a POS tag beginning with 'V. . .'
 - Add links from all verbs to all nouns.
 - The verb edges are the only edges that rely on POS.

2.2 Ranking - PageRank Explanation

- Trump needs a ranked list of "fake news" media (news medias that criticizes him) to wail at on twitter
- He first furiously read a "fake news" article
- The news article will link other "fake news" articles written by other news medias, and he clicks on one of them at random
- Every time he reads a "fake news" article, he records the news media behind it
- He will eventually have a ranked list of news medias that criticizes him the most

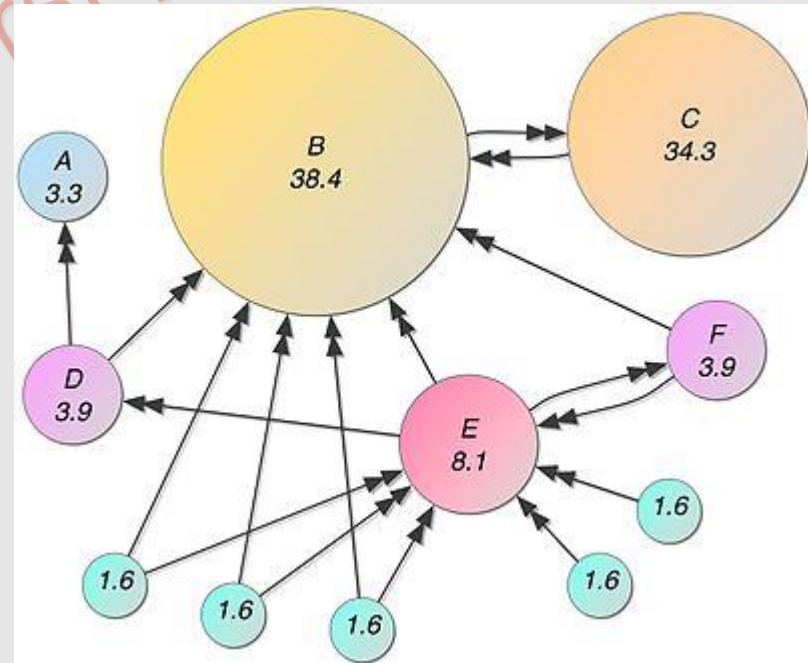


2.2 Ranking

More formally, 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 vertex in the graph such that all scores sum to 1.

$$PR(v) = \sum_{w \in B_v} \frac{PR(w)}{L(w)}$$

where B_v is the set of vertices such that $(w, v) \in E$, and $L(w)$ is the number of outgoing links from w , i.e. $|\{(u, u') | (u, u') \in E, u = w\}|$



2.3 Example

of Toronto

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

Ming

Need to transform this to dependency trees

Recap: Projectivity

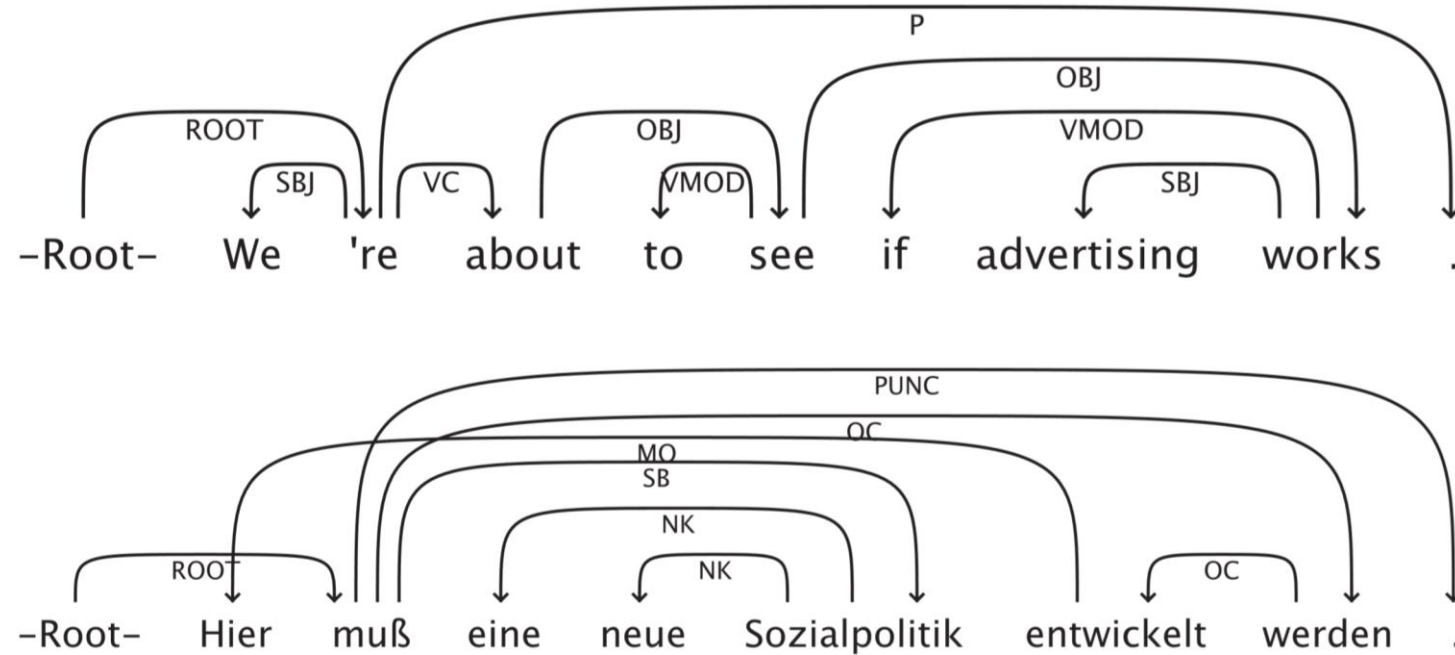


Fig. 1. Derived dependency structures from the Penn-III treebank of English and the TIGER treebank of German.


```

1:  $\pi = \langle n_1, \dots, n_m \rangle$  # the ranking of nodes
2:  $H = \{n_0\}$  # possible heads
3:  $D = \emptyset$  # dependency structure
4:  $pr2ind : \{1, \dots, m\} \rightarrow \{1, \dots, m\}$  # a mapping from rank to sentence position
5: for  $1 \leq i \leq m$  do
6:   if  $|H|=1$  then
7:      $c = 0$  # used to ensure single-rootedness
8:   else
9:      $c = 1$ 
10:  end if
11:   $n_{j'} = \arg \min_{n_j \in H[c:]} |pr2ind(i) - pr2ind(j)|$  # select head of  $w_{pr2ind(i)}$ 
12:   $H = n_i \cup H$  # make  $n_i$  a possible head
13:   $D = \{(w_{pr2ind(i)} \leftarrow w_{pr2ind(j')})\} \cup D$  # add new edge to  $D$ 
14: end for
15: return  $D$ 

```

Fig. 3. Parsing algorithm without universal dependency rules.

3.1 Without universal rules

* Note that $H[c :]$ is list slice notation used to indicate that when $c = 1$, the first element on the list H is ignored.

Runs in $O(n^2)$

Guarantees projectivity, closest possible head is always selected

3.1 Without universal rules - Example

The derivation of a simple Penn-III sentence 'The market crumbled !'

π	H	D
\langle crumbled, market, the, . \rangle \langle market, the, . \rangle \langle the, . \rangle	\langle ROOT \rangle \langle crumbled \rangle \langle crumbled, market \rangle	\emptyset $\{$ ROOT \rightarrow crumbled $\}$ $\{$ ROOT \rightarrow crumbled, crumbled \rightarrow market $\}$
\langle . \rangle	\langle crumbled, market, the \rangle	$\{$ ROOT \rightarrow crumbled, crumbled \rightarrow market, market \rightarrow the $\}$
\langle \rangle	\langle crumbled, market, the, . \rangle	$\{$ ROOT \rightarrow crumbled, crumbled \rightarrow market, market \rightarrow the, crumbled \rightarrow . $\}$

Fig. 4. Example derivation.

3.2 With universal rules

- Naseem et al. (2010) introduced universal dependency rules as linguistic priors or soft constraints in unsupervised dependency parsing.
- Modify the above parsing algorithm in the following simple way: When selecting a head for any word on the ranked list, first see if there exists a head such that the head-dependent pair is an instantiation of one of our universal dependency rules.
- More precisely, select the closest head that enables us to instantiate a rule. If not possible, select the nearest possible
- Can produce non-projective dependency structures

ROOT → VERB	
VERB → VERB	NOUN → ADJ
VERB → NOUN	NOUN → DET
VERB → ADV	NOUN → NOUN
VERB → ADP [new]	NOUN → NUM
VERB → CONJ [new]	
VERB → DET [new]	
VERB → NUM [new]	
VERB → ADJ [new]	
ADP → NOUN	ADP → ADV

	DMV PR-AS 140	E-DMV PR-AS 140	us	usUR	Brod10	Nase10
Bulgarian	54.0	59.8	46.4	51.4	-	-
Czech	32.0	54.6	44.0	45.7	-	-
Danish	42.4	47.2	50.8	51.4	41.9	51.9
Dutch	37.9	46.6	39.7	38.3	35.3	-
English	61.9	64.4	52.6	59.9	39.3	71.9
German	39.6	35.7	48.7	57.6	-	-
Japanese	60.2	59.4	62.3	60.9	-	-
Portuguese	47.8	49.5	47.0	54.6	-	71.5
Slovene	50.3	51.2	38.3	39.7	-	50.9
Spanish	62.4	57.9	47.0	52.6	-	67.2
Swedish	38.7	41.4	52.3	60.5	-	62.1
Turkish	53.4	56.9	50.3	54.0	-	-
AV	48.4	52.2	48.3	52.2	-	-

Fig. 7. Unlabeled attachment scores (in %) on sentences of length of at most 10 ignoring punctuation.

4 Experiments

On average considerably better than DMV PR-AS 140 and as good as E-DMV PR-AS 140.

Consistently worse than Naseem et al. (2010)

Results are very promising for a radically new model

5 Error analysis – German data

- The universal dependency rules do not seem to help in attaching verbs but the parser without rules is already very good at attaching verbs.
- The rules considerably improve attachment of adpositions, but improvements are also observed with nouns, determiners and adjectives.
- The rules allow us to attach dependents to candidate heads further away, so we predict more long dependencies.
- This also leads to a considerable absolute improvement of 15% in f-score for long dependencies (to words at least seven positions away).

acc	us	usUR
NOUN	28%	34%
VERB	49%	49%
DET	44%	48%
ADP	17%	45%
ADJ	41%	48%

f-score	us	usUR
to_root	57.3%	57.3%
1	65.7%	69.4%
2	30.3%	45.6%
3–6	24.0%	45.9%
7–	25.3%	40.6%

	rand	randUR	us	usUR	usnP	Spit11
Bulgarian	21.3	27.6	37.1	39.3	33.3	40.5
Czech	20.9	24.0	35.8	36.9	32.2	37.8
Danish	22.2	25.1	36.6	37.8	41.7	37.1
Dutch	32.5	31.7	37.6	37.9	29.9	14.0
German	21.8	30.9	34.6	41.2	31.2	28.6
Japanese	36.2	39.9	41.4	39.5	43.3	27.5
Portuguese	21.2	30.1	45.0	46.7	33.5	33.5
Slovene	16.9	23.4	33.0	33.6	31.3	31.2
Spanish	18.6	26.3	38.0	39.0	29.4	32.3
Swedish	25.9	34.1	43.8	48.5	38.7	46.4
Turkish	24.4	27.5	41.9	42.6	38.5	40.9
AV	23.8	29.1	38.6	40.2	34.8	33.6

6 Unsupervised dependency parsing without POS

- usnP : an unsupervised dependency parser that does not rely on information about POS at all
- Considerably better than baselines with random order (rand, randUR)

7 Conclusion

- Presented a new approach to unsupervised dependency parsing.
- The key idea is that a dependency structure also expresses centrality or saliency
- Thus, unsupervised dependency parser works in two stages: it first uses iterative graph-based ranking to rank words in terms of centrality, and then constructs a dependency tree from the ranking.
- Parser was shown to be competitive to state-of-the-art unsupervised dependency parsers.
- Obtains promising results in the absence of information about POS.