A Probabilistic Generative Model for an Intermediate Constituency-Dependency Representation

Federico Sangati

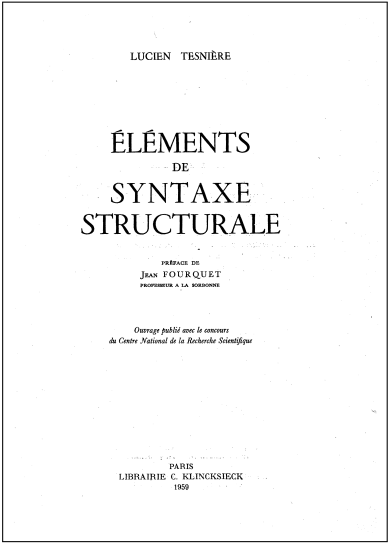
Institute for Logic, Language and Computation
University of Amsterdam

July 13, 2010

Outline

1. *TDS Representation*
   - Lucien Tesnière
   - Penn WSJ Treebank Conversion into TDS
   - Elements of TDS
2. *Parsing*
   - A 3-step generative model
   - Parsing through re-ranking
   - Results
3. *Conclusions*

Lucien Tesnière



LUCIEN TESNIÈRE

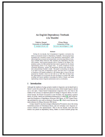
ÉLÉMENTS
DE
SYNTAXE
STRUCTURALE

PRÉFACE DE
JEAN FOURQUET
PROFESSEUR A LA SORBONNE

Ouvrage publié avec le concours
du Centre National de la Recherche Scientifique

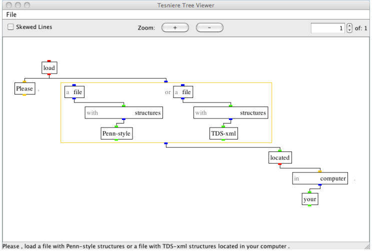
PARIS
LIBRAIRIE C. KLINCKSIECK
1959

Converting the Penn WSJ Treebank into TDS format



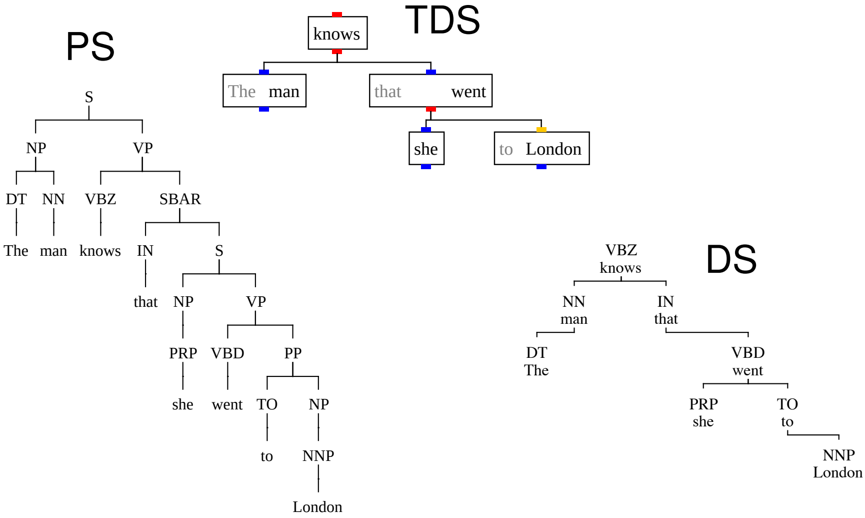
Federico Sangati and Chiara Mazza.
An English Dependency Treebank à la Tesnière.
Proceedings TLT8, December 09.

- Fully automatic (49208 sentences)
- Conversion and visualization publicly available at: `staff.science.uva.nl/~fsangati/TDS`

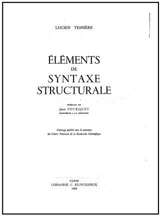


Please , load a file with Penn-style structures or a file with TDS-xml structures located in your computer .

Comparing with PS and DS



Dependency Relations (Connexion)



LUCIEN TESNIÈRE

ÉLÉMENTS
DE
SYNTAXE
STRUCTURALE

PRÉFACE DE
JEAN FOURQUET
PROFESSEUR A LA SORBONNE

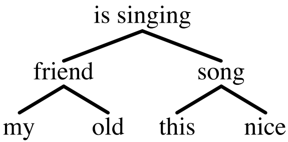
Ouvrage publié avec le concours
du Centre National de la Recherche Scientifique

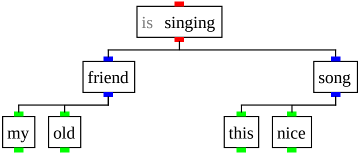
PARIS
LIBRAIRIE C. KLINCKSIECK
1959

1. — Les connexions structurales établissent entre les mots des rapports de **dépendance.** Chaque connexion unit en principe un terme **supérieur** à un terme **inférieur.**

7. — Lá connexion est **indispensable** à l'expression de la pensée. Sans la connexion, nous ne saurions exprimer aucune pensée continue et nous ne pourrions qu'énoncer une succession d'images et d'idées isolées les unes des autres et sans lien entre elles[1].

8. — C'est donc la connexion qui donne à la phrase son caractère **organique** et **vivant,** et qui en est comme le **principe vital.**



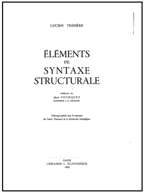


Word types

All words are divided into two classes:

- **Content words**: nouns, verbs, adjectives, etc.
- **Functional words**: aux., determiners, prepositions, etc.

e.g. Snoopy is flying on the doghouse



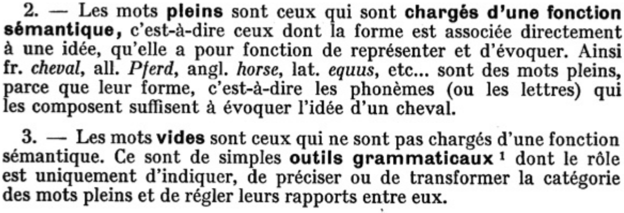
LUCIEN TESNIÈRE

ÉLÉMENTS
DE
SYNTAXE
STRUCTURALE

PRÉFACE DE
JEAN FOURQUET
PROFESSEUR A LA SORBONNE

Ouvrage publié avec le concours
du Centre National de la Recherche Scientifique

PARIS
LIBRAIRIE C. KLINCKSIECK
1959



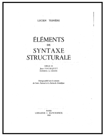
2. — Les mots **pleins** sont ceux qui sont **chargés d'une fonction sémantique**, c'est-à-dire ceux dont la forme est associée directement à une idée, qu'elle a pour fonction de représenter et d'évoquer. Ainsi fr. *cheval*, all. *Pferd*, angl. *horse*, lat. *equus*, etc... sont des mots pleins, parce que leur forme, c'est-à-dire les phonèmes (ou les lettres) qui les composent suffisent à évoquer l'idée d'un cheval.

3. — Les mots **vides** sont ceux qui ne sont pas chargés d'une fonction sémantique. Ce sont de simples **outils grammaticaux** [1] dont le rôle est uniquement d'indiquer, de préciser ou de transformer la catégorie des mots pleins et de régler leurs rapports entre eux.

Block of Words (Nucléus)

A *block* always includes a single content word and any number of functional words (possibly none).

e.g. Snoopy | is flying | on the doghouse



LUCIEN TESNIÈRE

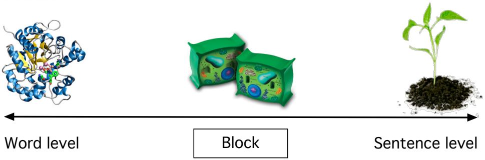
ÉLÉMENTS
DE
SYNTAXE
STRUCTURALE

PRÉFACE DE
Jean FOURQUET
PROFESSEUR À LA SORBONNE

Ouvrage publié avec le concours du Centre National de la Recherche Scientifique

PARIS
LIBRAIRIE C. KLINCKSIECK
1959

14. — Le nucléus est donc en dernière analyse l'entité syntaxique élémentaire, le matériau fondamental de la charpente structurale de la phrase, et en quelque sorte la **cellule** constitutive qui en fait un organisme vivant.



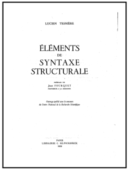
Word level

Block

Sentence level

Categories (Catégories)

Tesnière distinguishes four *block categories*: nouns, adjectives, verbs, adverbs.



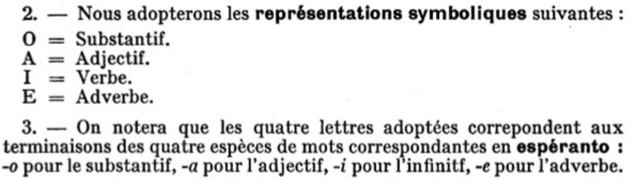
LUCIEN TESNIÈRE

ÉLÉMENTS
DE
SYNTAXE
STRUCTURALE

PRÉFACE DE
Jean FOURQUET
PROFESSEUR À LA SORBONNE

Ouvrage publié avec le concours
du Centre National de la Recherche Scientifique

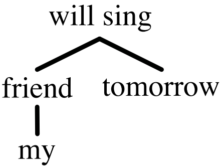
PARIS
LIBRAIRIE C. KLINCKSIECK
1959

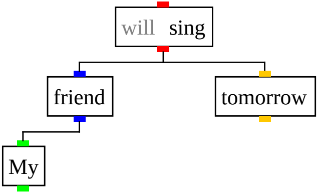


2. — Nous adopterons les **représentations symboliques** suivantes :

O = Substantif.
A = Adjectif.
I = Verbe.
E = Adverbe.

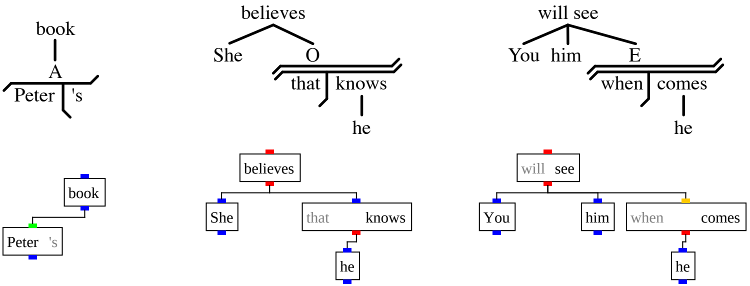
3. — On notera que les quatre lettres adoptées correpondent aux terminaisons des quatre espèces de mots correspondantes en **espéranto** : -*o* pour le substantif, -*a* pour l'adjectif, -*i* pour l'infinitf, -*e* pour l'adverbe.





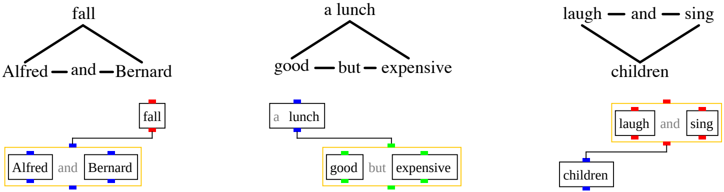
Transference (Translation)

A **shifting process** which makes a block change from the original category of the content word, to another category, by means of zero or more functional words belonging to the same block, called ***transferrers***.

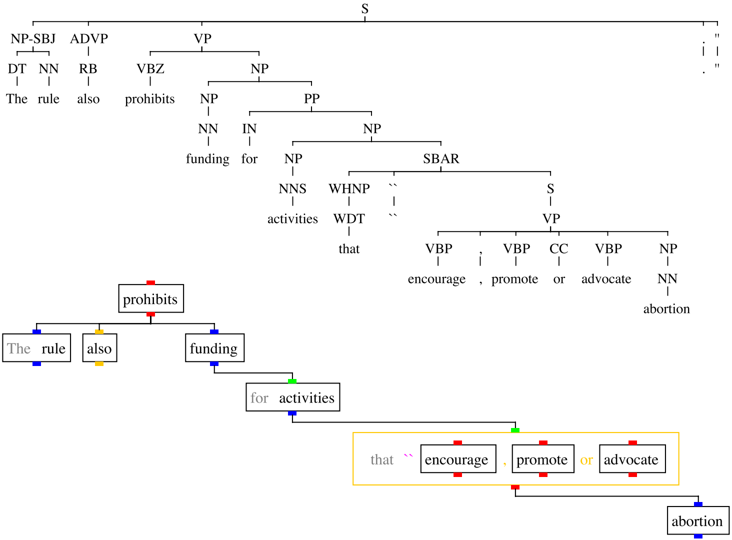


Junction (Jonction)

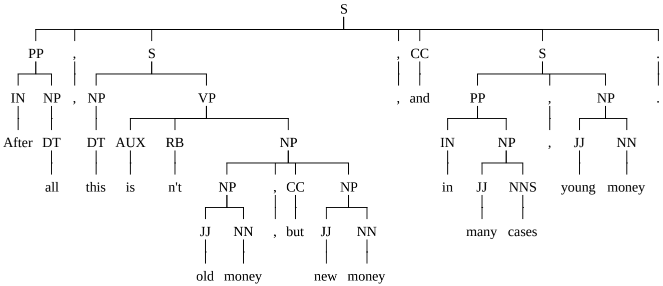
- It groups blocks, the ***conjuncts***, into a unique block entity.
- The conjuncts are connected horizontally by means of ***conjunctions*** (possibly missing).

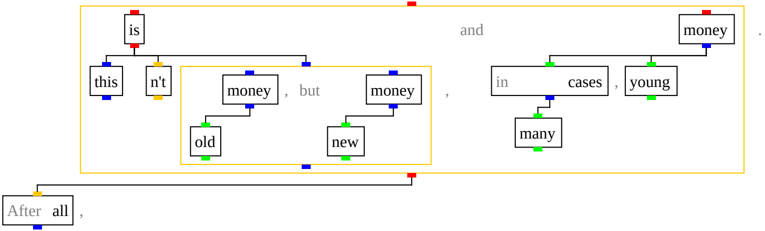


Coordination : TDS vs. PS

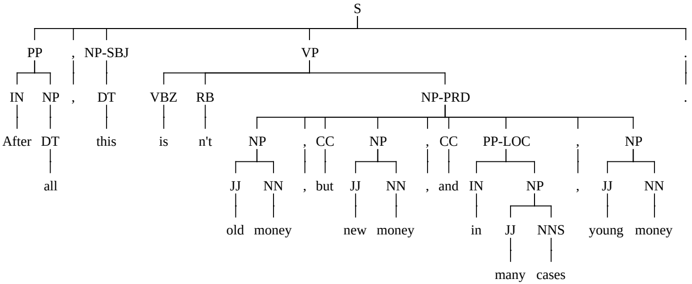


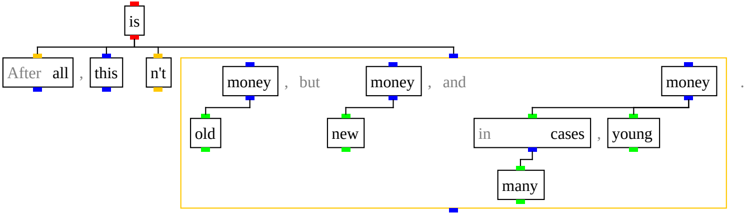
TDS vs. PS: Charniak's parser





TDS vs. PS: Charniak's parser





A 3-steps generative model

A sentence structure is generated in three phases:

A 3-steps generative model

A sentence structure is generated in three phases:

1. **Generate** generic blocks (top-dows, left-right) specifying categories, and functional words.

$$P_G(S_G) = \prod_{B \in dependentBlocks(S)} P(B|parent(B), direction(B), leftSibling(B))$$

A 3-steps generative model

A sentence structure is generated in three phases:

1. **Generate** generic blocks (top-dows, left-right) specifying categories, and functional words.

$$P_G(S_G) = \prod_{B \in dependentBlocks(S)} P(B|parent(B), direction(B), leftSibling(B))$$

2. **Expand** generic blocks to either one standard block or several conjunct blocks and conjunctions.

$$P_E(S_E) = \prod_{B \in blocks(S)} P(elements(B)|derivedCat(B))$$

A 3-steps generative model

A sentence structure is generated in three phases:

1. **Generate** generic blocks (top-dows, left-right) specifying categories, and functional words.

$$P_G(S_G) = \prod_{B \in dependentBlocks(S)} P(B|parent(B), direction(B), leftSibling(B))$$

2. **Expand** generic blocks to either one standard block or several conjunct blocks and conjunctions.

$$P_E(S_E) = \prod_{B \in blocks(S)} P(elements(B)|derivedCat(B))$$

3. **Fill** standard blocks with words.

$$P_F(S_F) = \prod_{B \in standardBlocks(S)} P(cw(B)|cw(parent(B)), cats(B), fw(B), context(B))$$

A 3-steps generative model

A sentence structure is generated in three phases:

1. **Generate** generic blocks (top-dows, left-right) specifying categories, and functional words.

$$P_G(S_G) = \prod_{B \in dependentBlocks(S)} P(B|parent(B), direction(B), leftSibling(B))$$

2. **Expand** generic blocks to either one standard block or several conjunct blocks and conjunctions.

$$P_E(S_E) = \prod_{B \in blocks(S)} P(elements(B)|derivedCat(B))$$

3. **Fill** standard blocks with words.

$$P_F(S_F) = \prod_{B \in standardBlocks(S)} P(cw(B)|cw(parent(B)), cats(B), fw(B), context(B))$$

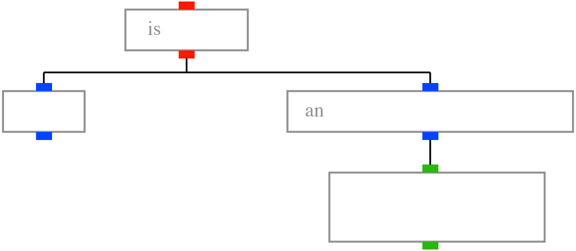
$$\boxed{P(S) = \quad P_G(S_G) \cdot P_E(S_E) \cdot P_F(S_F)}$$

Generating a sentence

Mary is singing an old and beautiful song

Generating a sentence

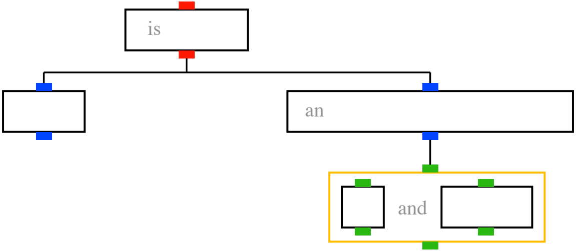
Mary is singing an old and beautiful song



(after 4 applications of GENERATE)

Generating a sentence

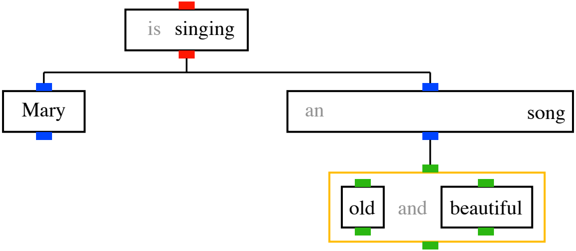
Mary is singing an old and beautiful song



(after 6 applications of EXPAND)

Generating a sentence

Mary is singing an old and beautiful song



(after 5 applications of FILL)

Parsing through Re-ranking

The Idea

- An other parser provides k-best PS candidates.
- Convert them into TDS representation.
- Compute the prob. of each candidate.
- Select the one with max. probability (re-ranking).

Parsing through Re-ranking

The Idea

- An other parser provides k-best PS candidates.
- Convert them into TDS representation.
- Compute the prob. of each candidate.
- Select the one with max. probability (re-ranking).

Motivation

- Implement and compare different settings / models.
- Without implementing different full parsers.

A generative re-ranking model for dependency parsing

Federico Sangati, Willem Zuidema and Rens Bod
Institute for Logic, Language and Computation
University of Amsterdam
Science Park 904, 1098 XH Amsterdam, The Netherlands

Federico Sangati, Willem Zuidema, and Rens Bod.
A generative re-ranking model for dependency parsing.
Proceedings IWPT 09.

Evaluation metrics

Standard Evaluations

- **F-Score (F1)**
- **Unlabeled Word Attachment Score (UAS)**

Evaluation metrics

Standard Evaluations

- **F-Score (F1)**
- **Unlabeled Word Attachment Score (UAS)**

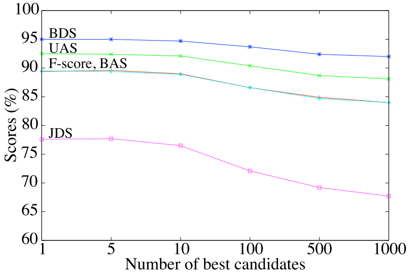
New Proposed Evaluations

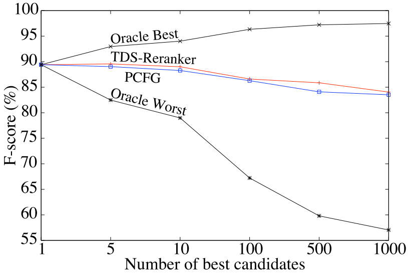
- **Block Detection Score (BDS)**: accuracy of detecting the correct boundaries of the blocks in the structure.
- **Block Attachment Score (BAS)**: accuracy of detecting the correct governing block of each block in the structure.
- **Junction Detection Score (JDS)** : accuracy of detecting the correct list of conjuncts composing each junction block in the structure.

Results

- Corpus: **Penn WSJ** converted to TDS.
- Training/Test: sec 02-21 / sec 22
- k-best candidates: Charniak's Max-Ent parser.

	Beam	**F1**	**UAS**	**Blocks Detection**	**Blocks Attach.**	**Junctions Detection**
Charniak	$k = 1$	89.4	92.5	95.0	89.5	77.6
PCFG-reranker	$k = 5$	89.0	92.4	**95.1**	89.2	77.5
PCFG-reranker	$k = 1000$	83.5	88.4	92.9	83.6	71.8
TDS-reranker	$k = 5$	**89.6**	92.4	95.0	89.4	**77.7**
TDS-reranker	$k = 10$	89.0	92.1	94.7	88.9	76.5
TDS-reranker	$k = 100$	86.6	90.4	93.7	86.6	72.1
TDS-reranker	$k = 1000$	84.0	88.1	92.0	84.0	67.7



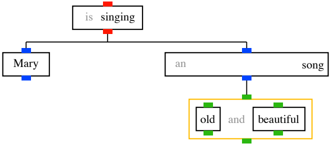


PoS-Tagging & Chunking

- Previous 3 models do not take into account the linear order of words.
- Improve robustness: define **PoS tagging** and **Chunking** model.
- Both implemented as a tagging task with n-gram models (Buchholz 1999, Veenstra 1999).

PoS-Tagging & Chunking

- Previous 3 models do not take into account the linear order of words.
- Improve robustness: define **PoS tagging** and **Chunking** model.
- Both implemented as a tagging task with n-gram models (Buchholz 1999, Veenstra 1999).



Words	Mary	is	singing	an	old	and	beautiful	song
PoS-tags	NNP	AUX	VBG	DT	JJ	CC	JJ	NN
Chunk tags	N	N	I	N	-N	C	N	+I

$$P_{PoS}(S_P) = \prod_{i=1}^{|sentenceLength|} P(word(i), pos(i) | word_{i-1}, pos_{i-1}, pos_{i-2})$$

$$P_{Chunk}(S_C) = \prod_{i=1}^{|sentenceLength|} P(chunk(i) | word_i, pos_i, pos_{i-1}, pos_{i-2}, pos_{i+1})$$

$$P(S) = P_G(S_G) \cdot P_E(S_E) \cdot P_F(S_F) \cdot P_{PoS}(S_P) \cdot P_{Chunk}(S_C)$$

Results

- Corpus: **Penn WSJ** converted to TDS.
- Training/Test: sec 02-21 / sec 22
- k-best candidates: Charniak's Max-Ent parser.

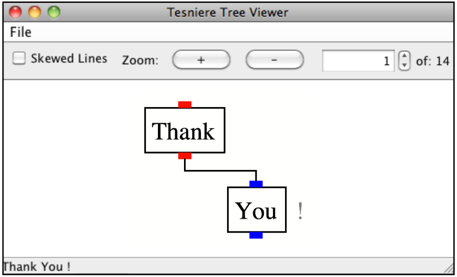
	Beam	F1	UAS	B. Detection	B. Attach.	Junct. Detection
Charniak	$k = 1$	89.4	92.5	95.0	89.5	77.6
PCFG-reranker	$k = 5$	89.0	92.4	**95.1**	89.2	77.5
PCFG-reranker	$k = 1000$	83.5	88.4	92.9	83.6	71.8
TDS-reranker	$k = 5$	**89.6**	92.4	95.0	89.4	**77.7**
TDS-reranker + pos&chunk	$k = 5$	89.6	**92.5**	**95.2**	**89.5**	77.6
TDS-reranker	$k = 10$	89.0	92.1	94.7	88.9	76.5
TDS-reranker	$k = 100$	86.6	90.4	93.7	86.6	72.1
TDS-reranker	$k = 1000$	84.0	88.1	92.0	84.0	67.7
TDS-reranker + pos&chunk	$k = 1000$	**84.8**	**89.3**	**93.5**	**84.9**	**69.7**

Conclusions

- Conversion of the Penn WSJ treebank into TDS.
- Probabilistic model to parse TDS structures.
- 3 New evaluation metrics.
- Improved robustness of the system after adding PoS-tagger & Chunker models.

Conversion and visualization tool available at:

`staff.science.uva.nl/~fsangati/TDS`



Thank You !

`f.sangati@uva.nl`