From morphosyntactic tagging to identification of verbal multiword expressions: a discriminative approach

Jakub Waszczuk

Heinrich Heine University Düsseldorf, Germany

October 29, 2018
ICS PAS, Warsaw

Outline

Outline

Concraft-pl

Basic information

- A morphosyntactic tagger for Polish based on conditional random fields
- Not SOA any more (see PolEval 2017)
- Nevertheless, the set of users $\neq \emptyset$
  - Practical tagging speed
  - Compatible with Morfeusz
  - Some people just don't like deep nets

Concraft-pl

Basic information

- A morphosyntactic tagger for Polish based on conditional random fields
- Not SOA any more (see PolEval 2017)
- Nevertheless, the set of users $\neq \emptyset$
  - Practical tagging speed
  - Compatible with Morfeusz
  - Some people just don't like deep nets

What's new in Concraft-pl 2.0

- It's slower
- Does not rely on Maca
- Concraft-pl 2.0 can:
  - Divide sentences into paragraphs
  - Perform tokenization (segmentation)

Ambiguous segmentation

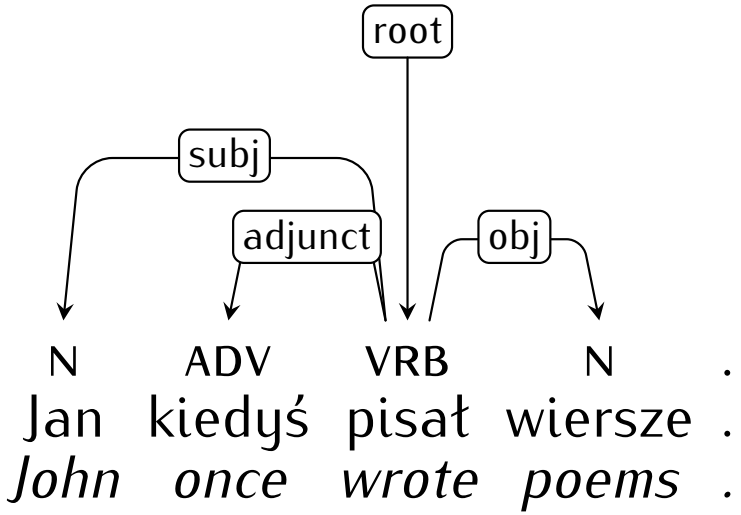
Jan ***kiedyś*** *pisał wiersze.*

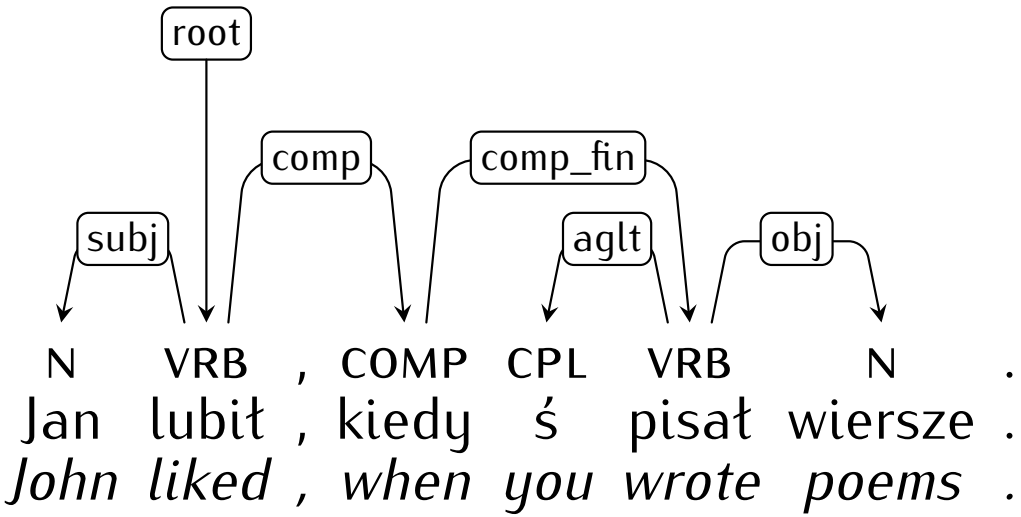
Jan lubił, ***kiedyś*** *pisał wiersze.*

(1) Jan kiedyś pisał wiersze.
John once wrote poems.

(2) Jan lubił, kiedy ś pisał wiersze.
John liked when you wrote poems.

Ambiguous segmentation





Morfeusz

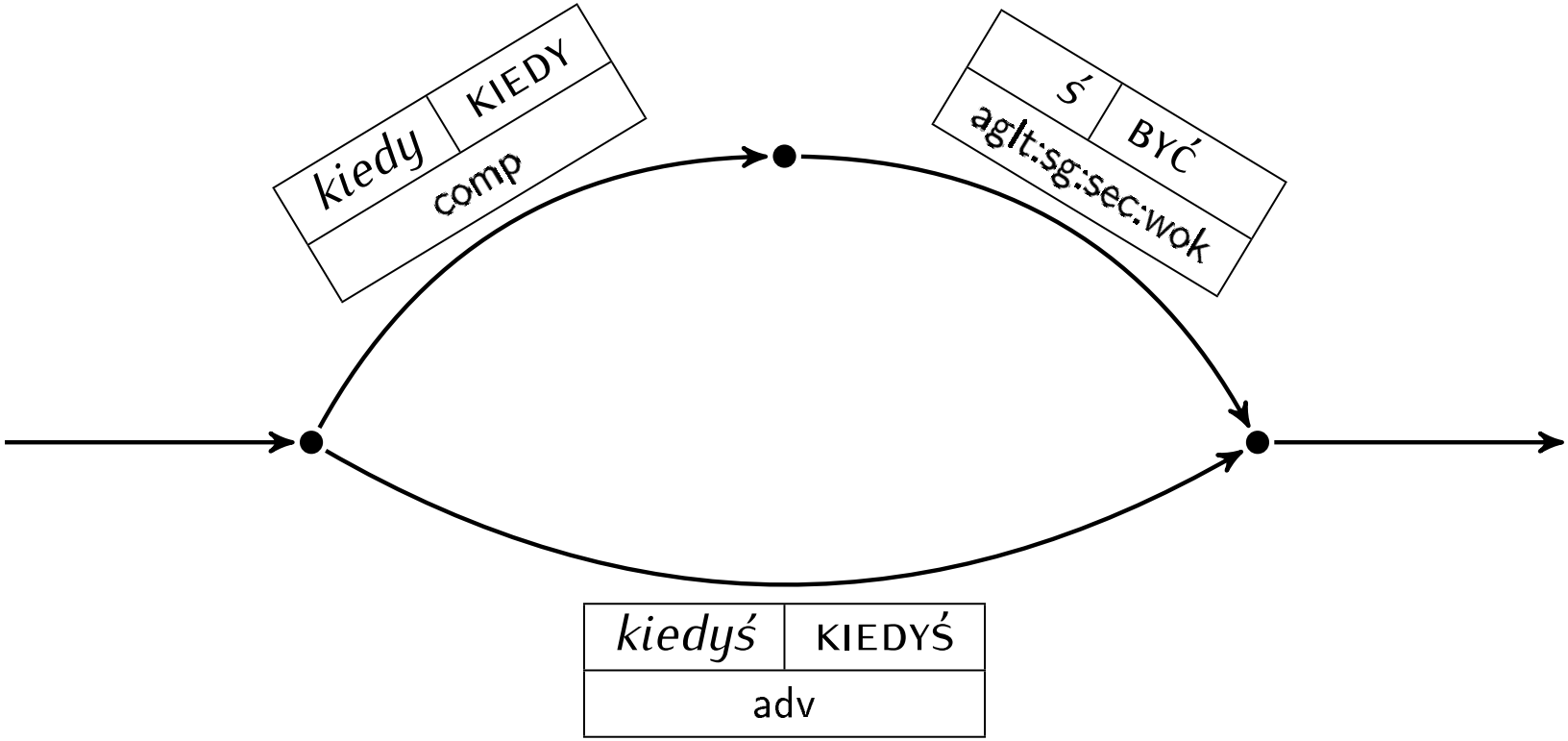


Figure 1: Directed acyclic graph (DAG) analysis of the word *kiedyś*

Existing solutions for Polish word segmentation (1)

`Maca` [Radziszewski and Śniatowski, 2011a]

- **Idea**: transform and simplify the segmentation graph to a sequence of tokens based on heuristic rules
- **Taggers**: `WMBT` [Radziszewski and Śniatowski, 2011b], `WCRFT` [Radziszewski, 2013], `Concraft` [Waszczuk, 2012], `KRNNT` [Wróbel, 2017]
- **Disadvantage**: segmentation performed in pre-processing

Example

```
Jan     newline
    Jan     subst:sg:nom:m1
kiedyś     space
    kiedyś     adv
pisał     space
    pisać     praet:sg:m1:imperf
    pisać     praet:sg:m2:imperf
    pisać     praet:sg:m3:imperf
wiersze     space
    wiersz     subst:pl:nom:m3
    wiersz     subst:pl:acc:m3
    wiersz     subst:pl:voc:m3
    wiersza     subst:pl:nom:f
    wiersza     subst:pl:acc:f
    wiersza     subst:pl:voc:f
.     none
    .     interp
```

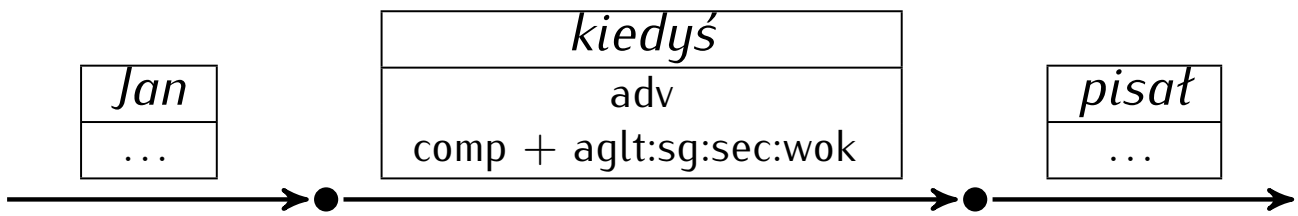
```
Jan     newline
    Jan     subst:sg:nom:m1
lubił     space
    lubić     praet:sg:m1:imperf
    lubić     praet:sg:m2:imperf
    lubić     praet:sg:m3:imperf
,     none
    ,     interp
kiedyś     space
    kiedyś     adv
pisał     space
    pisać     praet:sg:m1:imperf
    pisać     praet:sg:m2:imperf
    pisać     praet:sg:m3:imperf
wiersze     space
    wiersz     subst:pl:nom:m3
    wiersz     subst:pl:acc:m3
    ...
```

Figure 2: Maca (v. 1.0.7) output for the two example sentences

Existing solutions for Polish word segmentation (2)

`MorphoDiTa-pl` [Piasecki and Walentynowicz, 2017]

- **Idea**: encode all segmentation ambiguities as morphosyntactic labeling ambiguities



- **Disadvantage**: hard to generalize (spelling errors, speech processing, multiword units, etc.)

Sequential CRF

Basics

- **Structure**: sequence of labels (morphosyntactic tags)
- **Model**: multiclass logistic regression
  - any sequence of labels y is reduced to a feature vector f_y
  - given a sentence x and a parameter vector θ:

$$p_\theta(y|x) = \frac{\exp(\theta \cdot f_y)}{\sum_{y' \in \mathbf{Y}(x)} \exp(\theta \cdot f_{y'})}$$

  - $\mathbf{Y}(x)$ is the solution space for sentence x
- **Inference**: efficient due to factorization of the model

Factorization in Concraft

Input sequence

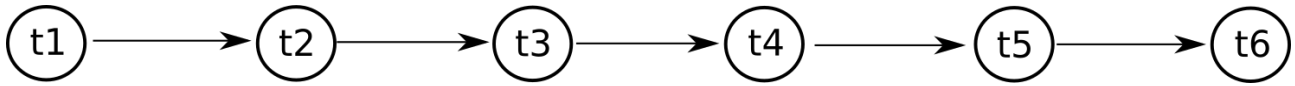


Figure 3: The circles represent the tags assigned to the subsequent words in the sentence.

Factorization in Concraft

Input sequence

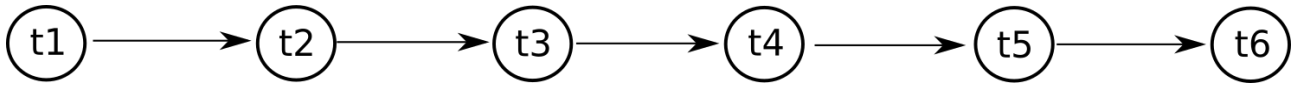


Figure 3: The circles represent the tags assigned to the subsequent words in the sentence.

Factorization in 2-order sequential CRFs

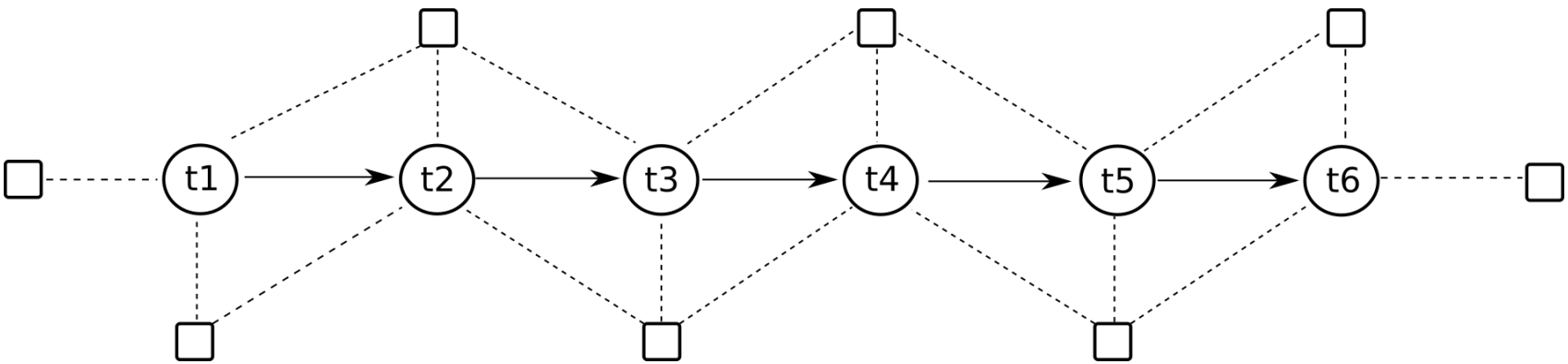


Figure 4: The squares represent the factors, which capture relations between triples of adjacent words.

Factorization in Concraft

Potential in 1-order sequential CRFs

$$\exp(\theta \cdot f_y) = \prod_{i=1}^{|y|} \exp(\theta \cdot f(y_i, y_{i-1}, i)) \tag{1}$$

Factorization in Concraft

Potential in 1-order sequential CRFs

$$\exp(\theta \cdot f_y) = \prod_{i=1}^{|y|} \exp(\theta \cdot f(y_i, y_{i-1}, i)) \tag{1}$$

Viterbi decoding

$$\alpha(y, i) = \max_{y' \in Y} \big(\exp(\theta \cdot f(y, y', i)) \cdot \alpha(y', i-1)\big) \tag{2}$$

Sequential CRF

Jan (*John*)	kiedyś (*once*)	pisał (*wrote*)	wiersze (*poems*)
subst:sg:nom:m1	subst:sg:nom:m1	subst:sg:nom:m	subst:sg:nom:m
...	...	...	...
subst:pl:acc:m3	subst:pl:acc:m3	subst:pl:acc:m3	**subst:pl:acc:m3**
...	...	...	...
adj:sg:nom:m1:pos	adj:sg:nom:m1:pos	adj:sg:nom:m1:pos	adj:sg:nom:m1:pos
...	...	...	...
adv	**adv**	adv	adv
...	...	...	...
praet:sg:m1:imperf	praet:sg:m1:imperf	**praet:sg:m1:imperf**	praet:sg:m1:imperf
...	...	...	...
prep:gen	prep:gen	prep:gen	prep:gen
...	...	...	...
interp	interp	interp	interp

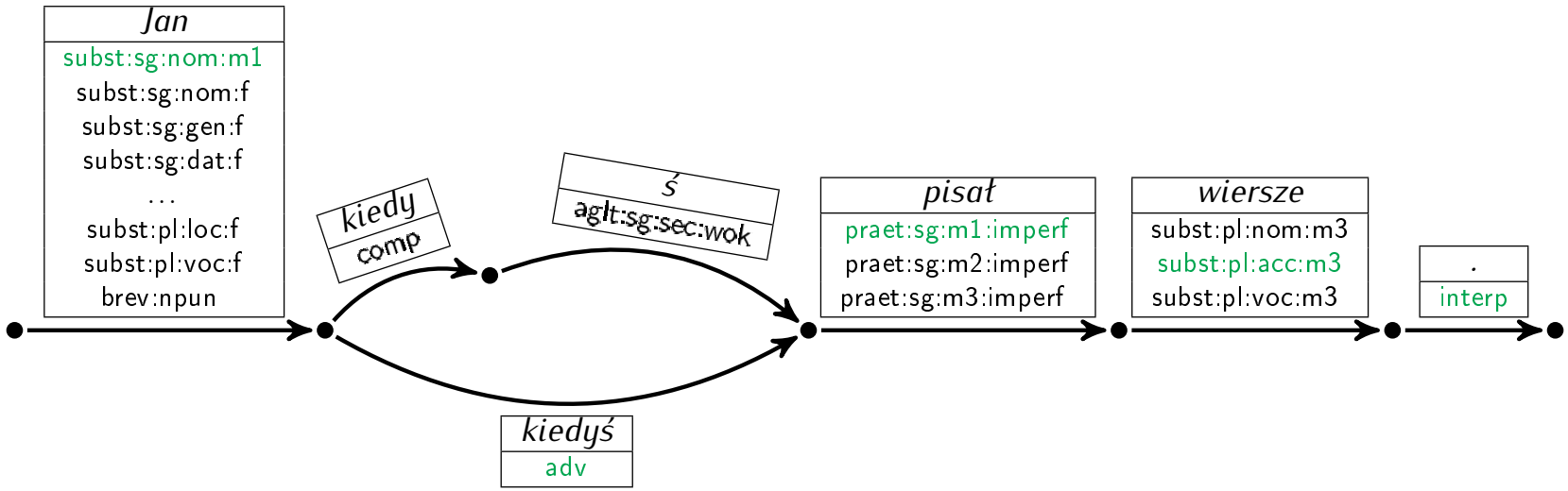
Solution space: $\mathsf{Y}(x) = T^{|x|}$, where T is a tagset

Constrained CRF (Concraft 1.0)

Jan (*John*)	kiedyś (*once*)	pisał (*wrote*)	wiersze (*poems*)	.
subst:sg:nom:m1	**adv**	**praet:sg:m1:imperf**	subst:pl:nom:m3	**interp**
subst:sg:nom:f		praet:sg:m2:imperf	**subst:pl:acc:m3**	
subst:sg:gen:f		praet:sg:m3:imperf	subst:pl:voc:m3	
subst:sg:dat:f				
...				
subst:pl:loc:f				
subst:pl:voc:f				
...				
brev:npun				

Solution space: $Y(x) = \prod_{i=1}^{|x|} r_i$, where r_i is the set of possible labels (tags) assigned to position i ($r_i = T$ for out-of-vocabulary words)

DAG-based CRF (Concraft 2.0)



Solution space: $\mathsf{Y}(x)$ is the set of labeled paths in the input DAG

DAG-based CRF (`Concraft 2.0`)

Inference

The standard algorithms for CRFs [Wainwright et al., 2008]:

- *max-product* – for Viterbi decoding
- *sum-product* – for forward and backward sums

can be straightforwardly adapted to DAG-based CRFs.

DAG-based CRF (Concraft 2.0)

Inference

The standard algorithms for CRFs [Wainwright et al., 2008]:

- *max-product* – for Viterbi decoding
- *sum-product* – for forward and backward sums

can be straightforwardly adapted to DAG-based CRFs.

Viterbi decoding (1-order)

$$\alpha(e) = \max_{e' \in \mathrm{pred}(e)} \left(\exp(\theta \cdot f(e, e')) \cdot \alpha(e') \right) \tag{3}$$

where:

- e – edge in the DAG
- $\mathrm{pred}(e)$ – the set of edges preceding e in the DAG

DAG-based CRF (Concraft 2.0)

Features

The same as in Concraft 1.0:

- 2-order *transition* features (t_{i-2}, t_{i-1}, t_i)
- *observation* features (o_i, t_i), where o_i is an observation (wordform, suffix, prefix, shape, etc.) related to word i

"*the wordform of the token on position* $i - 2$"

Issue

$i - 2$ may not be uniquely defined

Solution

- $i - 1$ is defined as the shortest edge preceding i
- $i + 1$ is defined as the shortest edge following i

Evaluation datasets

Baroque Corpus of Polish [Kieraś et al., 2017]

- 500,000 tokens large manually annotated subcorpus of the Baroque Corpus of Polish (1601–1772)
- 3% of tokens have ambiguous segmentation

Corpus of Polish 1830–1918 [Kieraś and Woliński, 2018]

- 625,000 tokens large manually annotated corpus of Polish texts published between 1830 and 1918
- 1.3% of tokens have ambiguous segmentation

NCP: 0.3% tokens have ambiguous segmentation

Evaluation results

	Baroque	1830-1918
Tagging:		
precision	0.882724	0.903176
recall	0.88303	0.903335
Guessing:		
precision	0.60125	0.610493
recall	0.601214	0.609796
Segmentation:		
precision	0.937455	0.951261
recall	0.948684	0.965946

Segm. baselines	Baroque	1830-1918
shortest path:		
precision	0.712871	0.694111
recall	0.503595	0.517577
longest path:		
precision	0.264848	0.294253
recall	0.41452	0.47628
freq. based:		
precision	0.838571	0.911858
recall	0.724294	0.823025

Table 1: Results (our system on the left, baselines on the right)

Where to find more

Implementation

- **Repository**: `https://github.com/kawu/concraft-pl`
- **Languages**: Haskell + Dhall (configuration – NEW)
- **License**: 2-clause BSD
- **Model**: pre-trained on NKJP1M merged with Morfeusz
- **„Bindings“**: Python client for the client/server mode
- **Concraft-pl 1.0**: avaliable in branch `maca`

Conclusions and future work

Polish word segmentation

- Potentially hard to resolve in pre-processing
- Neglected so far due to relatively low frequency

Concraft-pl 2.0's approach

- More principled than what was proposed before
- Segmentation as a by-product of disambiguation, or
- No segmentation at all! (just output marginal probabilities)

Conclusions and future work

Polish word segmentation

- Potentially hard to resolve in pre-processing
- Neglected so far due to relatively low frequency

Concraft-pl 2.0's approach

- More principled than what was proposed before
- Segmentation as a by-product of disambiguation, or
- No segmentation at all! (just output marginal probabilities)

Future work

?

Outline

Multiword expressions

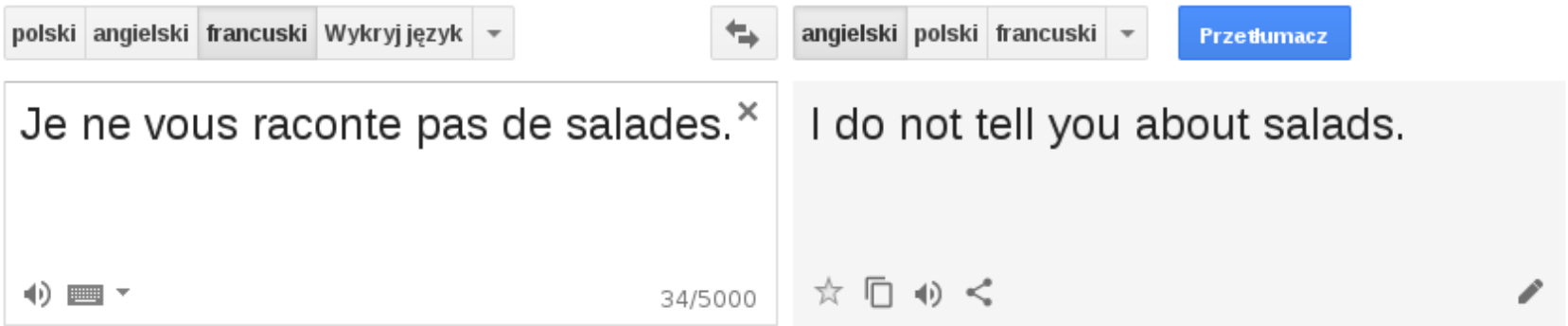
Properties

- Contain two or more words and show **idiosyncratic** behavior at different levels

  by and large
  to pull one's socks up
  white wine

- Non-compositional meaning

MWEs and NLP


polski angielski francuski Wykryj język
angielski polski francuski
Przetłumacz
Je ne vous raconte pas de salades.
34/5000
I do not tell you about salads.

MWEs and parsing

Syntactic parsing

- Syntactic structure: describes syntactic relations between the words of a sentence
- The goal of parsing: find the correct syntactic structure for a given input sentence

Interactions between MWEs and parsing

- MWE identification and syntactic parsing can mutually enhance each other [Nivre and Nilsson, 2004, Finkel and Manning, 2009, Wehrli et al., 2010, Constant et al., 2013, Nasr et al., 2015, Constant and Nivre, 2016]

Syntax and MWEs

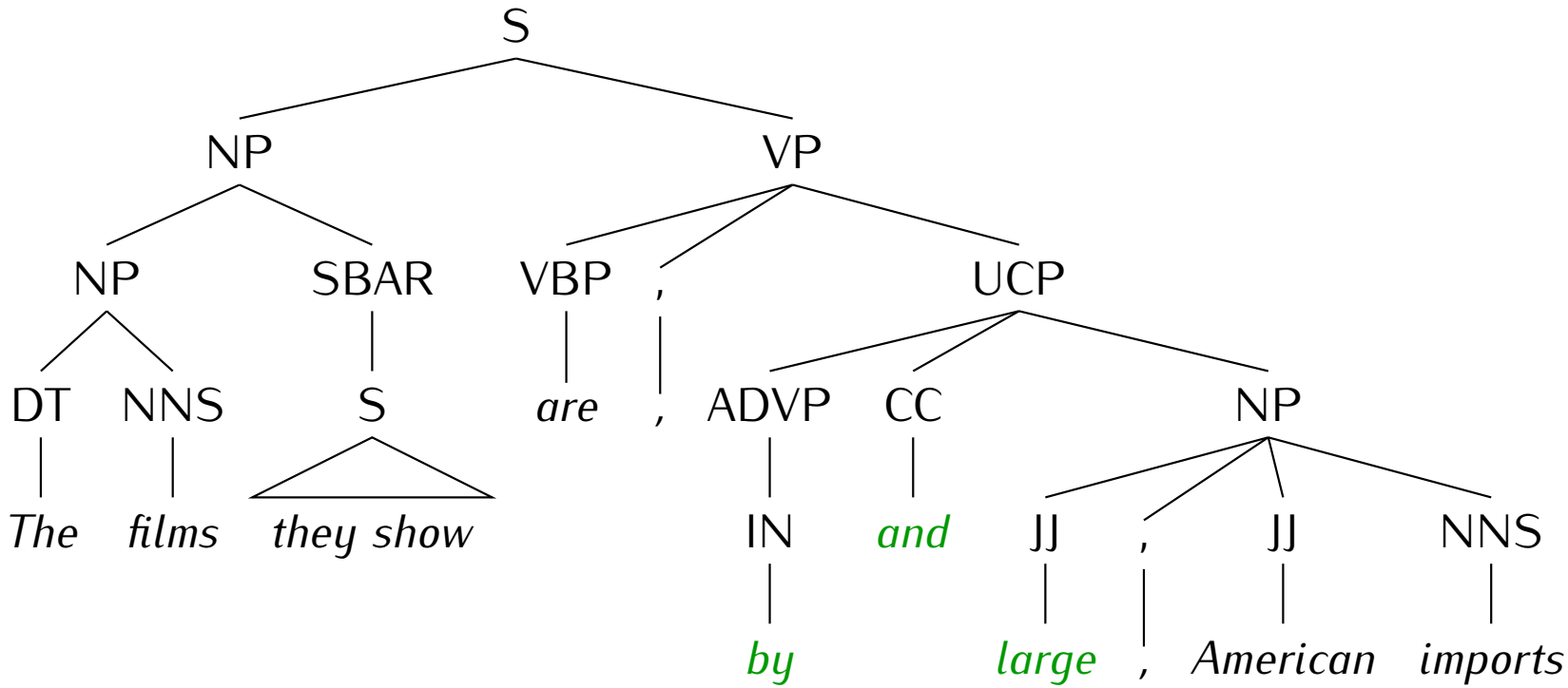


Figure 5: An example of a tree obtained with the *Berkeley Parser* (`http://tomato.banatao.berkeley.edu:8080/parser/parser.html`)

Syntax and MWEs

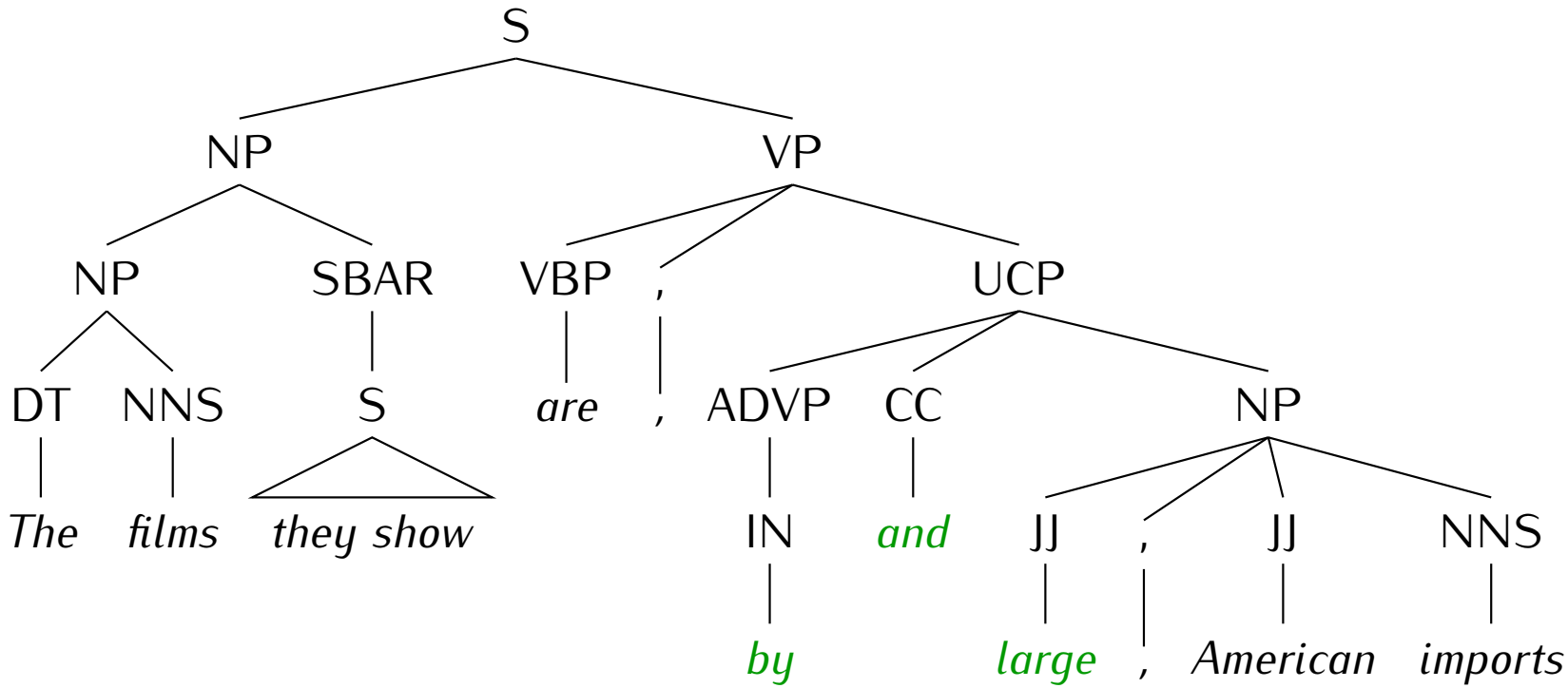


Figure 5: An example of a tree obtained with the *Berkeley Parser* (`http://tomato.banatao.berkeley.edu:8080/parser/parser.html`)

MWE-related issues: irregular syntax, syntactic ambiguity

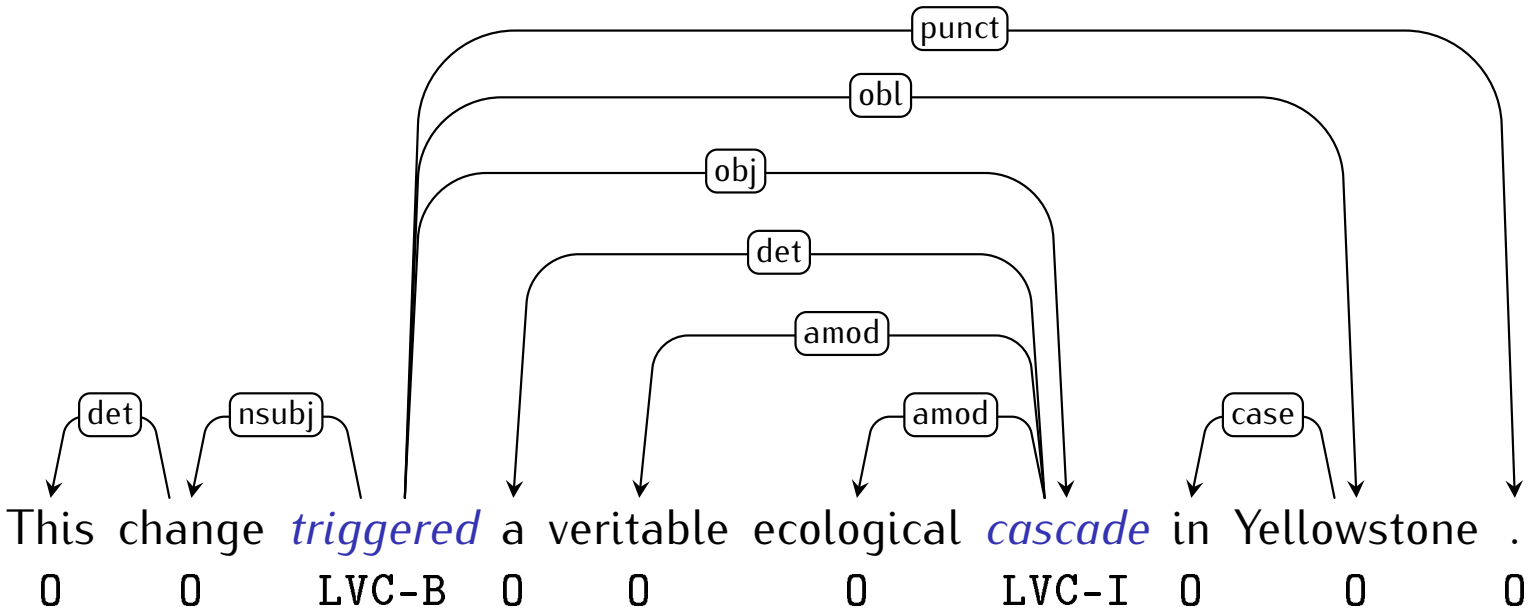
(example: *. . . and it was a fantastic protest – lots of interest from people passing by and large numbers of conversations which were uplifting and positive in the main.*)

Outline

PARSEME Shared Task 2018

- **Task**: automatic identification of verbal MWEs (VMWEs)
- **Data**: manually annotated with VMWEs for 19 languages (including PL)
- **Syntax**: UD trees provided on input (some manually annotad, some obtained using UDPipe)
- **Tracks**: *open* and *closed*

VMWEs, (dis)continuity, and sequential models



Sequential models:

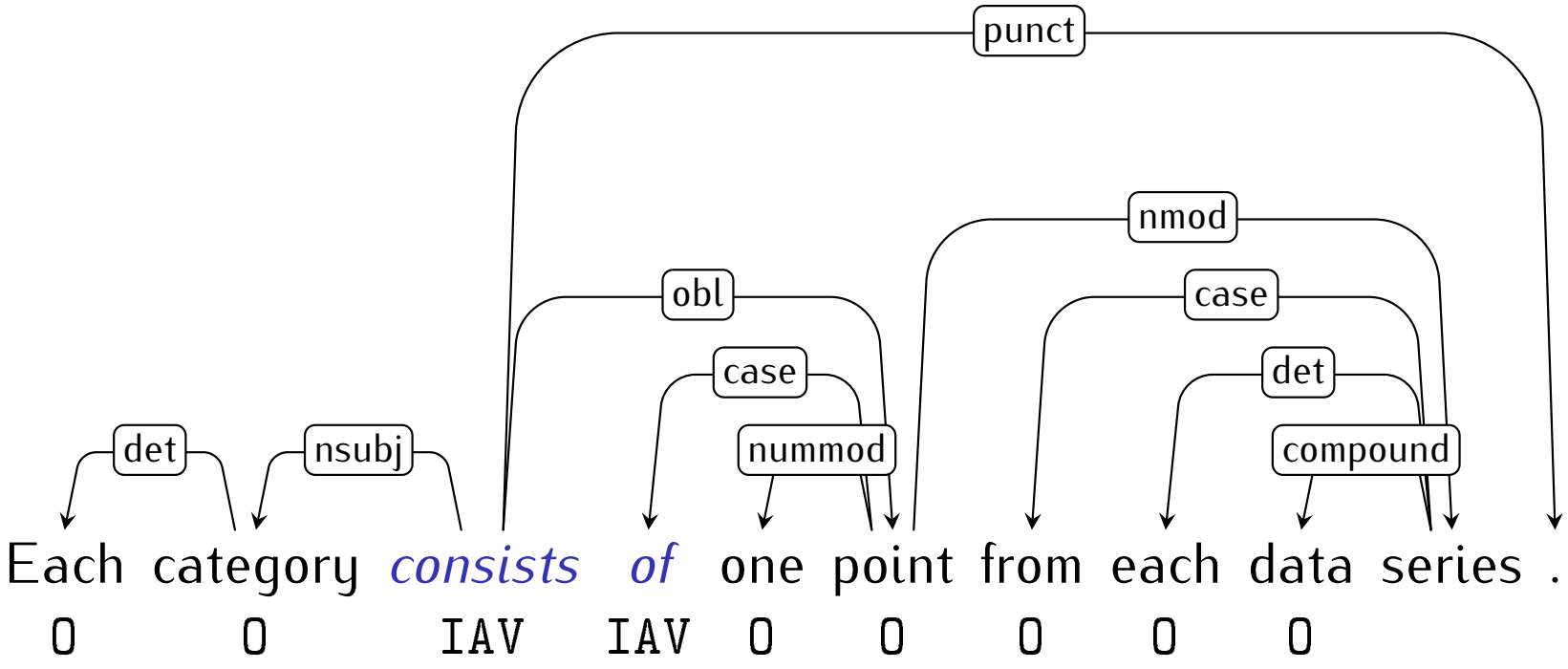
- Don't directly capture the relation between `LVC-B` and `LVC-I`.
- **CRF**: labeling *triggered* with `LVC-B` is independent from labeling *cascade* with `LVC-I` (provided that the material in between is labeled with `O`s).

TRAVERSAL

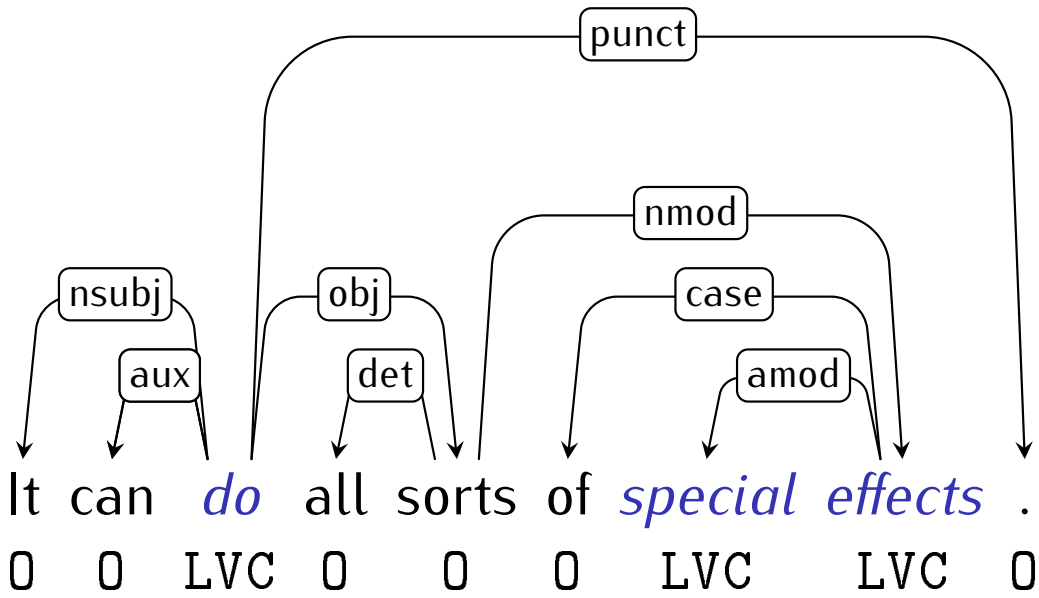
Principles

- **Assumption**: VMWEs continuous in dependency structures
- **Division**: VMWE identification = labeling + segmentation
- **Labeling**: multiclass logistic regression
- **Segmentation**: two simple heuristics

Breaking cases (attested)



Breaking cases (attested)



Dataset statistics

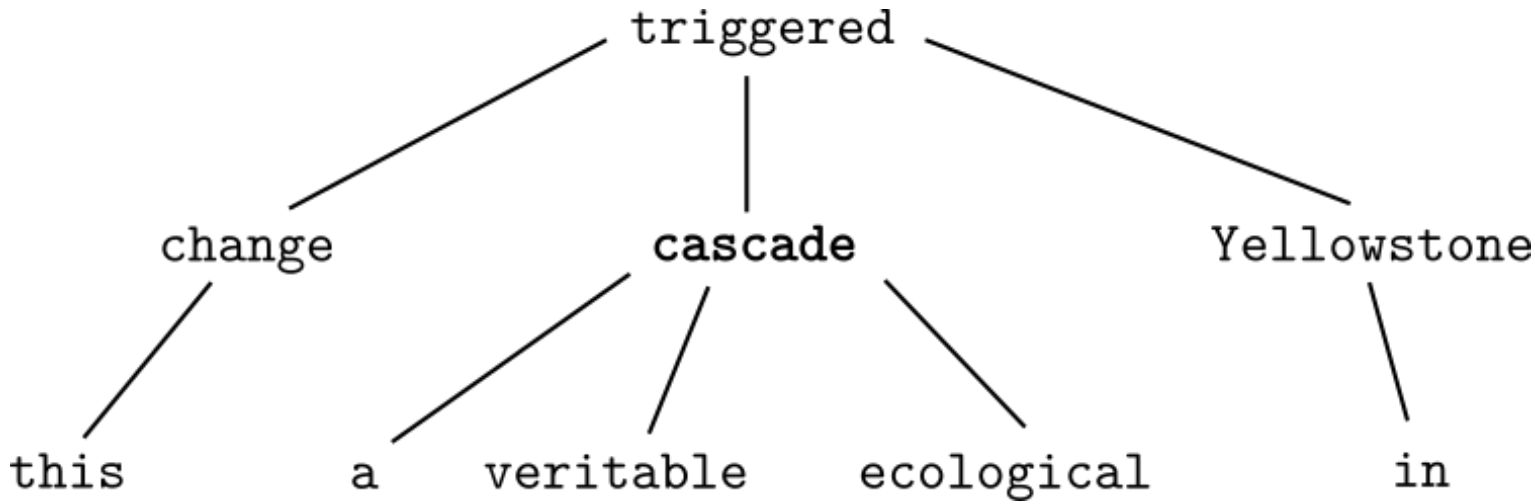
	% Continuous	
	Seq	Dep
BG	80.39	95.0
DE*	57.39	90.76
EL	48.58	92.86
EN**	68.28	91.84
ES	74.05	76.55
EU*	80.62	95.28
FA*	85.09	94.48
FR**	58.1	98.17
HE	75.82	76.45
HI	91.2	98.5

	% Continuous	
	Seq	Dep
HR	57.23	63.38
HU**	92.78	99.53
IT	64.78	94.62
LT†	58.33	n/a
PL*	70.26	80.87
PT*	53.5	95.54
RO	68.96	98.64
SL*	44.51	71.61
TR	49.5	55.54

Table 2: The % of continuous VMWEs in terms of sequences and dependency trees in the Train+Dev datasets for the individual languages. The datasets with dependencies annotated manually, partially manually, or not at all, are marked with **, *, or †, respectively. For the other datasets/sentences, dependencies were obtained automatically.

Factorization

Goal: capture the relations between `MWE` labels of adjacent nodes.

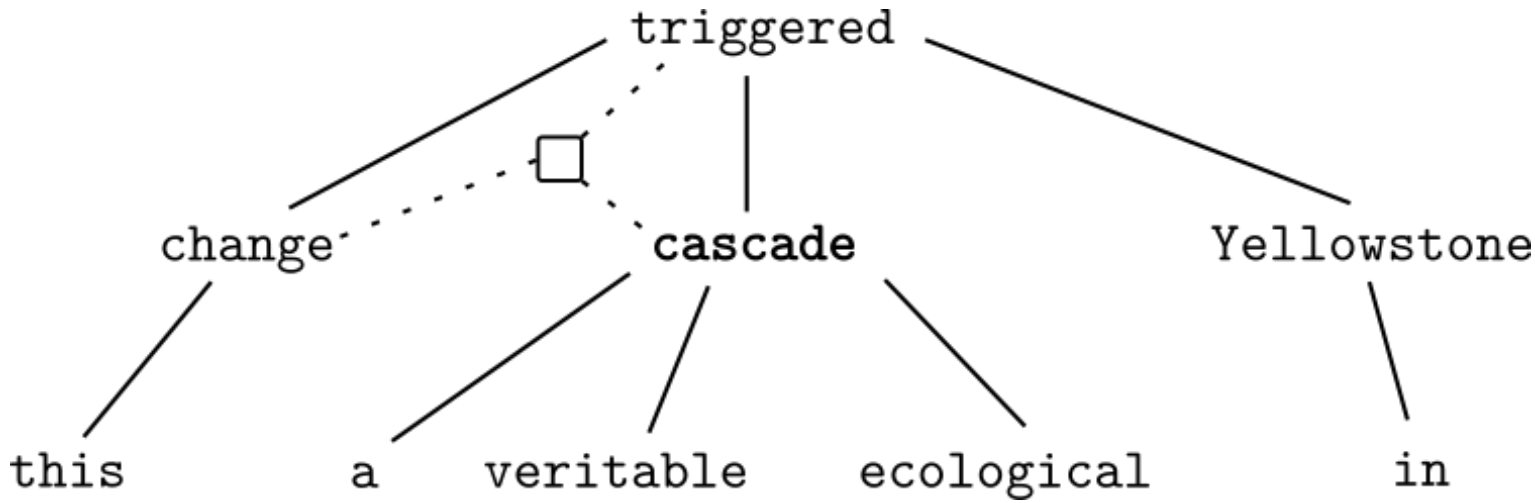


The labeling model can be seen as:

- Extension of sequential CRFs to tree structures
- Variant of the 2-order Eisner's graph-based parser
- Multiclass logistic regression model with special factorization allowing efficient computation

Factorization

Goal: capture the relations between `MWE` labels of adjacent nodes.

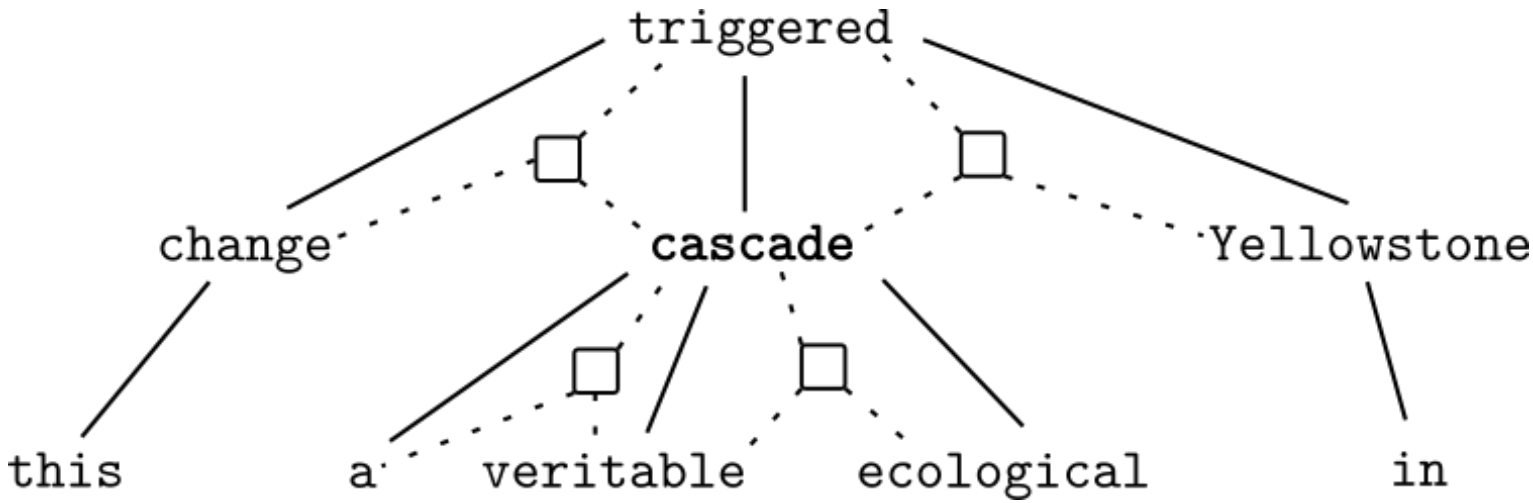


The labeling model can be seen as:

- Extension of sequential CRFs to tree structures
- Variant of the 2-order Eisner's graph-based parser
- Multiclass logistic regression model with special factorization allowing efficient computation

Factorization

Goal: capture the relations between `MWE` labels of adjacent nodes.



The labeling model can be seen as:

- Extension of sequential CRFs to tree structures
- Variant of the 2-order Eisner's graph-based parser
- Multiclass logistic regression model with special factorization allowing efficient computation

Encoding

Goal: given a dependency tree d, construct a *hypergraph* $\mathcal{H}(d)$ encoding all labelings of d. The individual *hyperarcs* should locally capture relations between adjacent nodes and their labels.

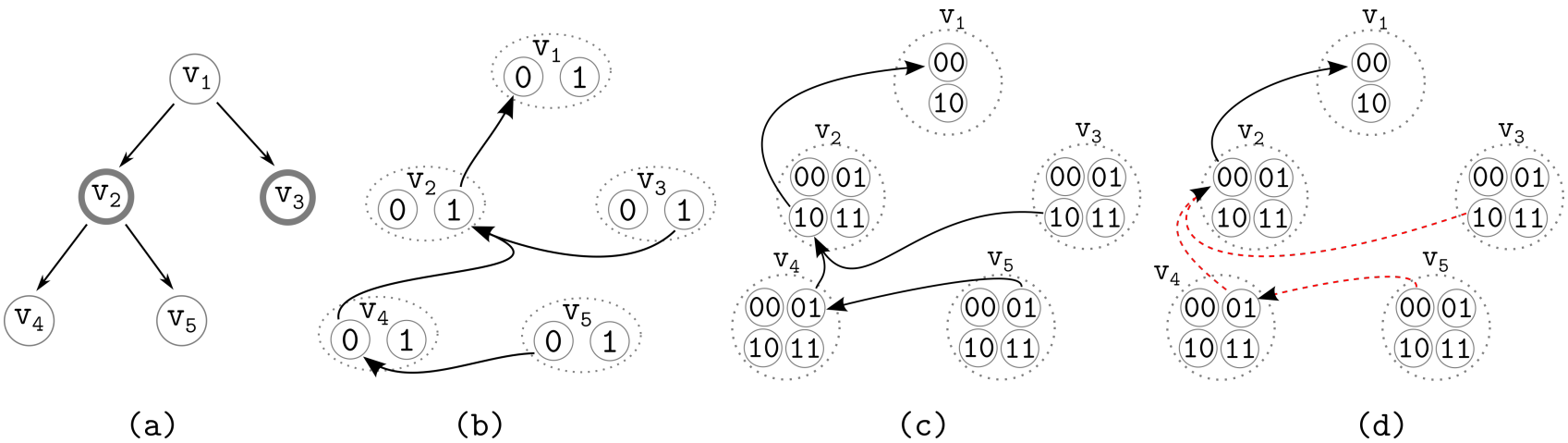


Figure 6: Encoding example. (a) A dependency tree with v_2 and v_3 marked as `MWE`s. (b) The corresponding hyperpath. (c) The corresponding hyperpath with copying of parent labels. (d) An invalid hyperpath, not encoded in the hypergraph.

Labeling model

- **Reduction**. Labeling boils down to finding the best-score hyperpath $h \in \mathcal{H}(d)$ among the set of hyperpaths $\mathcal{H}(d)$.
- **Representation**. Each hyperpath h is reduced to a feature vector f^h of a fixed length n, where n is the number of features, with each hyperarc $a \in h$ represented as a binary feature vector f^a such that $f^a_k = 1$ iff the k-th feature holds within the context of a:

$$f^h = \sum_{a \in h} f^a \tag{4}$$

- **Probability**. Given a vector of parameters θ of length n, the probability of a hyperpath $h \in \mathcal{H}(d)$ given a dependency tree d is defined as:

$$p_\theta(h|d) = \exp(\theta \cdot f^h) \; / \; \sum_{h' \in \mathcal{H}(d)} \exp(\theta \cdot f^{h'}) \tag{5}$$

Labeling model

- **Training**. Training consists in finding the *maximum likelihood estimates* (w.r.t. training data T) given the following (log-)likelihood function (with normal priors over θ):

$$\ell(\theta) = \sum_{(d,h)\in T} \log p_\theta(h|d) - \sum_{k=1}^{n} \frac{\theta_k^2}{2\sigma^2} \tag{6}$$

- **Optimization**. The maximum likelihood estimates are approximated using *stochastic gradient descent* with momentum.
- **Complexity**. The necessary calculations can be performed efficiently using the *inside-outside* algorithm.

Segmentation

Problem

- It's not enough to label nodes as `MWEs` or `not-MWEs`
- The **boundaries** of VMWEs need to be determined

Solutions

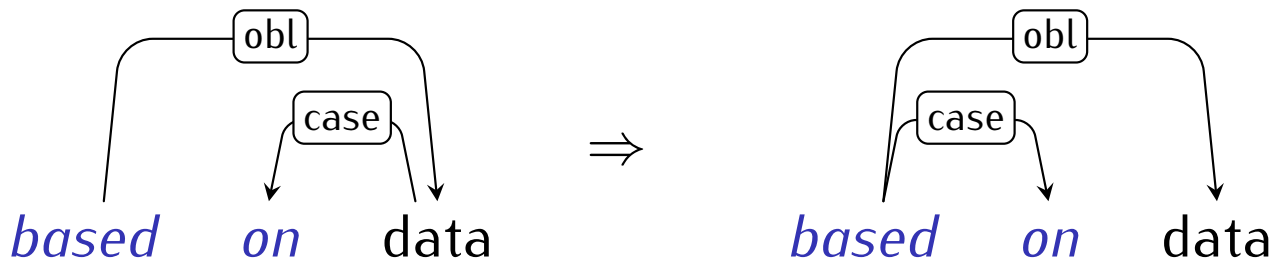
- Consider all adjacent nodes marked as `MWEs` of the same category as a single MWE occurrence (default heuristic)
- If a group of adjacent nodes is marked as `MWEs` but it contains two (or more) verbs, the group is divided into two (or more) distinct MWEs (heuristic applied only to `FA`)
- Variant of IOB encoding adapted to trees (not in the shared task)

Implementation

- **Repository**: `https://github.com/kawu/traversal`
- **Languages**: Haskell + Dhall (configuration)
- **License**: 2-clause BSD

Setup

- **Pre-processing**: *case lifting* (among others)



- **Feature engineering**: `PL` and `FR`
- **Backoff model**: 2-order sequential CRF (`LT`)

Feature templates

Unary	(l_v, m_v)
Binary	$\{(l_v, m_v), (l_w, m_w)\}$ (*unordered pair*)
Sibling Binary	(m_v, m_w), (l_v, l_w, m_v, m_w), (m_v, m_w, d_v, d_w), (l_v, p_w, m_v, m_w), (p_v, l_w, m_v, m_w)
Parent Binary	(m_v, m_w), (l_v, l_w, m_v, m_w), (m_v, m_w, d_v), $(l_v, p_w, m_v, m_w, d_v)$, $(p_v, l_w, m_v, m_w, d_v)$
Ternary	(*none, even though possible to capture*)

Table 3: v is the current node, w is its sibling or parent, l_u is the u's lemma, m_u is the u's MWE label, p_u is the u's universal POS tag, d_u is the dependency label of the arc incoming to u.

Dataset statistics after pre-processing

	%VMWE$_{(train+dev)}$		
	Con	**Con$_p$**	**Iso$_p$**
BG	95.0	**97.34**	**93.42**
DE	90.76	**93.17**	**87.69**
EL		**92.86**	**84.66**
EN	91.84	**98.19**	**95.17**
ES	76.55	**90.13**	**84.59**
EU		**95.28**	**84.38**
FA		**94.48**	**57.83**
FR		**98.17**	**92.62**
HE		**76.45**	**73.69**
HI	98.5	**98.88**	**85.58**

	%VMWE$_{(train+dev)}$		
	Con	**Con$_p$**	**Iso$_p$**
HR	63.38	**96.56**	**87.03**
HU		**99.53**	**90.35**
IT		**94.62**	**88.12**
LT		**58.33**	**58.33**
PL	80.87	**86.09**	**82.42**
PT		**95.54**	**90.01**
RO		**98.64**	**95.38**
SL	71.61	**88.12**	**86.14**
TR	55.54	**98.21**	**91.08**

- `Con` – % of connected VMWEs (no value $\Longrightarrow$ `Con`=`Con`$_p$)
- `Con`$_p$ – % of connected VMWEs after pre-processing
- `Iso`$_p$ – % of connected and isolated (with no adjacent VMWEs of the same category) VMWEs after pre-processing

Results

	MWE-based					Token-based				
	P	R	F1	Rank	**Delta**	P	R	F1	Rank	Delta
SL	79.41	54	64.29	1/10	**21.95**	83.61	54.54	66.01	1/10	14.04
HR	68.04	46.59	55.3	1/10	**11.03**	78.14	50.73	61.52	1/10	11.55
IT	63.09	40.32	49.2	1/12	**10.68**	74.42	42.11	53.78	1/12	7.27
...	...	...	...	...	**...**	...	...	...	...	...
PL	77.02	59.22	66.96	1/11	**6.42**	81.85	59.03	68.59	1/11	3.67
FR	77.19	44.18	56.19	1/13	**5.65**	84.72	48.76	61.9	1/13	6.18
...	...	...	...	...	**...**	...	...	...	...	...
DE	62.93	32.73	43.06	3/11	**-2.21**	76.17	38.36	51.02	2/11	-0.64
...	...	...	...	...	**...**	...	...	...	...	...
FA	73.8	58.48	65.26	7/10	**-12.57**	90.19	65.23	75.7	6/10	-5.58
LT	29.61	13.8	18.83	3/10	**-13.34**	55.56	16.92	25.94	3/10	-8.49
AVG	*67.58*	*44.97*	*54*	*1/13*	***4.26***	*77.41*	*48.55*	*59.67*	*1/13*	*5.04*

Table 4: Results for selected languages ordered by the difference between TRAVERSAL's `F1` and `F1` of the other best-performing system (`Delta`)

Results

	Cont.	Discont.	Multi-tok.	Single-tok.	Seen	Unseen	Variant	Identical
F1	57.55	44.36	55.83	25.96	72.92	17.35	63.1	81.88
Delta	2.17	6.96	6.45	-6.86	0.85	-2.36	-1.92	-1.85

Table 5: Macro-average MWE-based F_1-scores for specialized phenomena

	IAV	IRV	LVC.cause	LVC.full	MVC	VID	VPC.full	VPC.semi	LS.ICV
F1	44.31	68.07	23.81	46.03	17.65	34.45	34.84	42.70	30.77
Delta	8.89	8.51	-8.34	6.30	-11.39	8.01	2.07	2.2	10.77

Table 6: Macro-average MWE-based F_1-scores for MWE categories

Conclusions

- State-of-the-art results (at least for some languages)
  - Last-year's winner [Al Saied et al., 2017] still better on average
- Tree-structured input pays off (especially for discontinuous MWEs)
- It's not clear if deep nets outperform the more traditional ML methods in this task
  - Since VMWEs are local in dependency trees, tree-structured CRFs might be good enough
  - Word embeddings can possibly help with identification of more productive categories (VPC), but what about idioms?

Ideas for future work

- Experiment with ternary features
- Improve the configuration language
- Allow input from lexical reasources
- Higher-order model (code optimization)
- Joint labeling and segmentation (IOB schemes for trees)
- Integrate sequential with tree-based models (dual decomposition, loopy belief propagation)

Dziękuję!

References I

Al Saied, H., Constant, M., and Candito, M. (2017).

The atilf-llf system for parseme shared task: a transition-based verbal multiword expression tagger.
In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 127–132.

Constant, M. and Nivre, J. (2016).

A Transition-Based System for Joint Lexical and Syntactic Analysis.
In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 161–171, Berlin, Germany. Association for Computational Linguistics.

Constant, M., Roux, J. L., and Sigogne, A. (2013).

Combining compound recognition and PCFG-LA parsing with word lattices and conditional random fields.
ACM Trans. Speech Lang. Process., 10(3):8:1–8:24.

Finkel, J. R. and Manning, C. D. (2009).

Joint Parsing and Named Entity Recognition.
In *HLT-NAACL*, pages 326–334.

Kieraś, W., Komosińska, D., Modrzejewski, E., and Woliński, M. (2017).

Morphosyntactic Annotation of Historical Texts. The Making of the Baroque Corpus of Polish.
In Ekštein, K. and Matoušek, V., editors, *Text, Speech, and Dialogue 20th International Conference, TSD 2017, Prague, Czech Republic, August 27-31, 2017, Proceedings*, volume 10415 of *LNCS*, pages 308–316. Springer International Publishing.

Kieraś, W. and Woliński, M. (2018).

Manually Annotated Corpus of Polish Texts Published between 1830 and 1918.
In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2018*, Miyazaki, Japan. ELRA.

Nasr, A., Ramisch, C., Deulofeu, J., and André, V. (2015).

Joint Dependency Parsing and Multiword Expression Tokenisation.
In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL'15)*.

References II

Nivre, J. and Nilsson, J. (2004).

Multiword Units in Syntactic Parsing.
In *Proceedings of MEMURA 2004 – Methodologies and Evaluation of Multiword Units in Real-World Applications, Workshop at LREC 2004, May 25, 2004, Lisbon, Portugal*, pages 39–46, Lisbon, Portugal.

Piasecki, M. and Walentynowicz, W. (2017).

MorphoDiTa-based Tagger Addapted to the Polish Language Technology.
pages 377–381.

Radziszewski, A. (2013).

A tiered CRF tagger for Polish.
In *Intelligent tools for building a scientific information platform*, pages 215–230. Springer.

Radziszewski, A. and Śniatowski, T. (2011a).

Maca – a configurable tool to integrate Polish morphological data.

Radziszewski, A. and Śniatowski, T. (2011b).

A memory-based tagger for Polish.
In *Proceedings of TSD 2012*.

Wainwright, M. J., Jordan, M. I., et al. (2008).

Graphical models, exponential families, and variational inference.
Foundations and Trends® in Machine Learning, 1(1–2):1–305.

Waszczuk, J. (2012).

Harnessing the CRF Complexity with Domain-Specific Constraints. The Case of Morphosyntactic Tagging of a Highly Inflected Language.
In *Proceedings of COLING 2012*, pages 2789–2804.

References III

Wehrli, E., Seretan, V., and Nerima, L. (2010).

Sentence analysis and collocation identification.

In *Proceedings of the Workshop on Multiword Expressions: from Theory to Applications (MWE 2010)*, pages 27–35, Beijing, China. Association for Computational Linguistics.

Wróbel, K. (2017).

KRNNT : Polish recurrent neural network tagger.

pages 386–391.