A Transition-Based System for Joint Lexical and Syntactic Analysis

Matthieu Constant[1] and Joakim Nivre[2]

(1) Université de Lorraine (France)
(2) Uppsala University (Sweden)

October 4, 2016

Contributions

A new factorized representation of lexical and syntactic analysis

- Dependency analysis
- Inclusion of Multiword Expression analysis

A new transition-based system

- Input: a sequence of tokens
- Output: above representation
- Special mechanisms to handle Multiword Expressions

Work originally presented at ACL 2016

Multiword Expressions (MWE)

Definitional features

- Sequence of several items with a **certain degree of non-compositionality**: *cut the mustard*, *hot dog*
- **Multidimensional idiomaticity**: pragmatic, semantic, syntactic, lexical and morphological
- Wide range of linguistic phenomena

Semi-compositionality

- **Examples:** *white wine*, *make a decision*
- Internal components may be compositionally modified.
  Example: ***make*** *several good* ***decisions***

Multiword Expressions (MWE)

Definitional features

- Sequence of several items with a **certain degree of non-compositionality**: *cut the mustard*, *hot dog*
- **Multidimensional idiomaticity**: pragmatic, semantic, syntactic, lexical and morphological
- Wide range of linguistic phenomena

Semi-compositionality

NEED FOR INTERNAL STRUCTURE

- **Examples:** *white wine*, *make a decision*
- Internal components may be compositionally modified.
  Example: ***make*** *several good* ***decisions***

Multiword Expressions (MWE)

Fixed vs. non-fixed MWEs

- **Fixed**: Fully lexicalized sequence with no variation nor insertion (Sag et al. 2002) with irregular syntax
  Examples: *by and large*
- **Non-fixed**: regular syntax, variations, insertions, ...
  Examples: *make several decisions*, *the decision you made*

Embedding

- **Example 1**: (make (big deal))
- **Example 2**: ((Los Angeles) Lakers)

Multiword Expressions (MWE)

A FIXED MWE = A SYNTACTIC UNIT

Fixed vs. non-fixed MWEs

- **Fixed**: Fully lexicalized sequence with no variation nor insertion (Sag et al. 2002) with irregular syntax
  Examples: *by and large*
- **Non-fixed**: regular syntax, variations, insertions, ...
  Examples: *make several decisions*, *the decision you made*

Embedding

- **Example 1**: (make (big deal))
- **Example 2**: ((Los Angeles) Lakers)

Multiword Expressions (MWE)

A FIXED MWE = A SYNTACTIC UNIT

Fixed vs. non-fixed MWEs

- **Fixed**: Fully lexicalized sequence with no variation nor insertion (Sag et al. 2002) with irregular syntax
  Examples: *by and large*
- **Non-fixed**: regular syntax, variations, insertions, ...
  Examples: *make several decisions*, *the decision you made*

Embedding

A LEXICAL UNIT = A TREE

- **Example 1**: (make (big deal))
- **Example 2**: ((Los Angeles) Lakers)

Related Work

Joint lexical and syntactic parsing in the dependency framework

MWE annotation in syntactic tree

- MWE annotation as flat subtree $\rightarrow$ internal structure is lost (Nivre and Nilsson 2004, Erygit et al 2011, Seddah et al. 2013, Nasr et al 2015)
- Syntactic structure is kept, MWE status is added to the syntactic arc label (Vincze et al. 2013, Candito and Constant 2014)
- Dual representation: regular vs. irregular MWEs (Candito and Constant 2014)

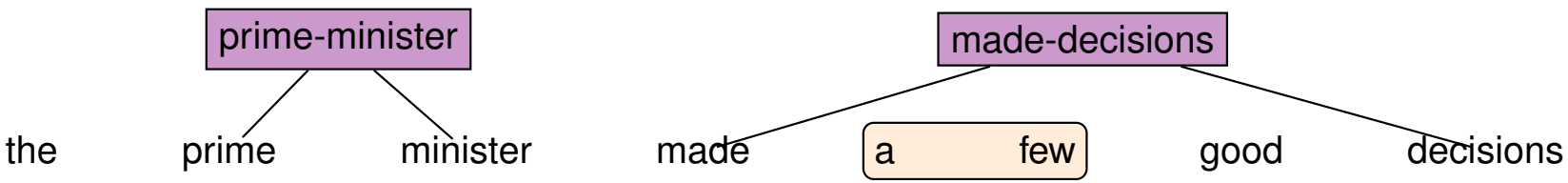
Use of off-the-shelf parsers

The concatenation of MWE status and syntactic function increases parsing complexity

Lexical and Syntactic representation

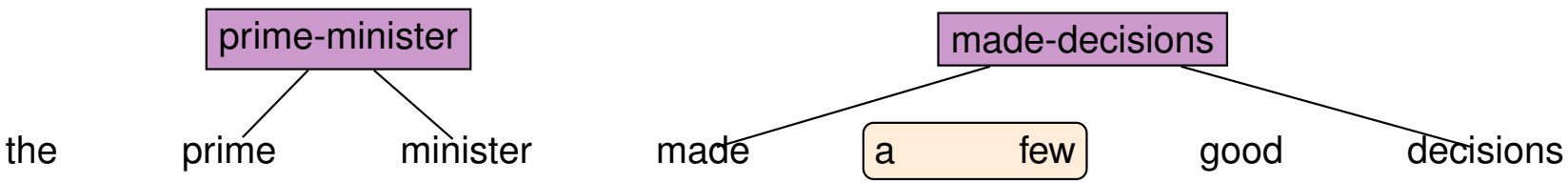
the prime minister made a few good decisions

Lexical and Syntactic representation

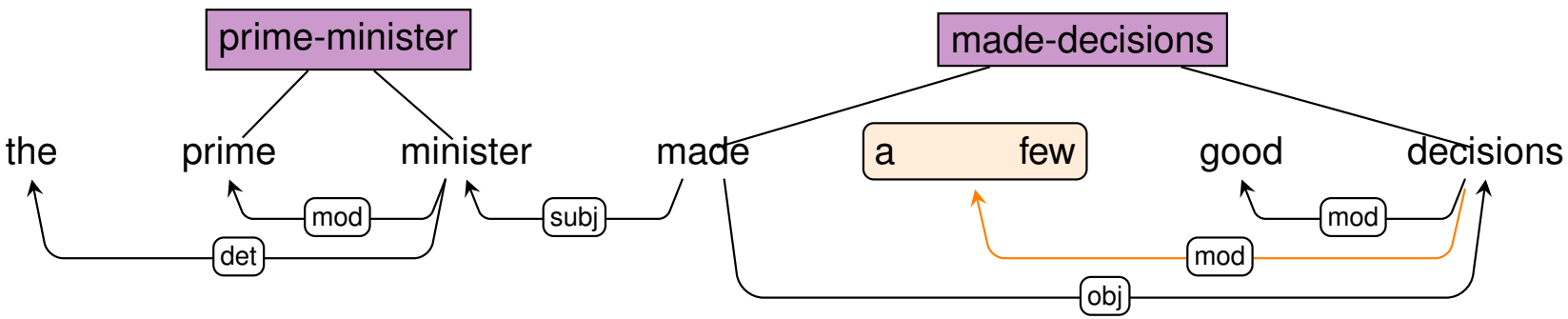


Lexical and Syntactic representation

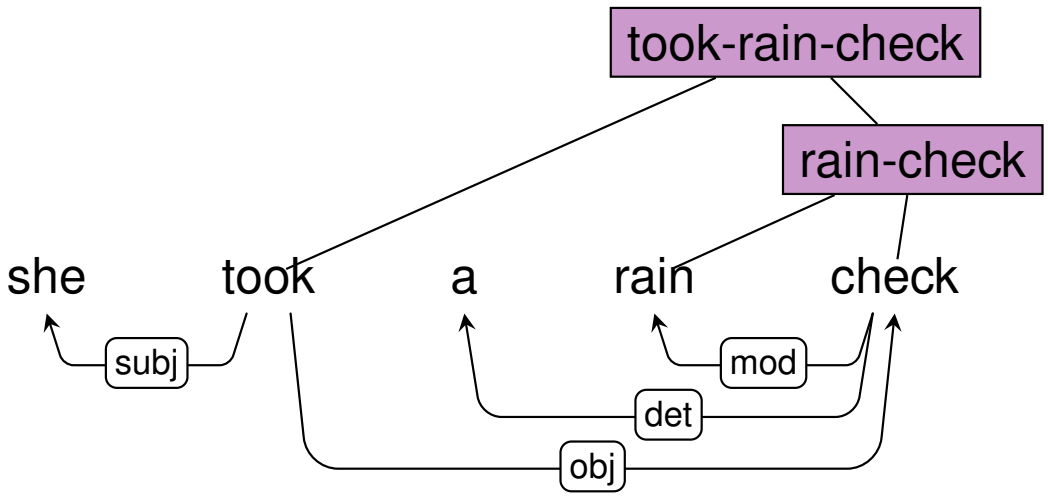
Form: made decisions
Lemma: make decision
POS: V
...



Lexical and Syntactic representation



MWE embedding



Background: a standard transition-based parser

Input/Output

- **Input**: a sequence of tokens
- **Output**: a set of syntactic arcs

Internal mechanism

- predict a **sequence of actions** (namely *transitions*)
- A transition goes from one parsing state (namely *configuration*) to another one
- **Configuration**: a stack, a buffer and a set of arcs

Background: a standard transition-based parser (Cont'd)

Configurations

- **Initial configuration**: buffer filled with input tokens, empty stack and set of arcs
- **Terminal configuration**: buffer is empty, stack has one items left

Transitions

- **Shift**: push the next token of the buffer on top of stack
- **Left-arc**$_k$: creates a left arc labeled k between the two top tokens of the stack; only head item is kept in stack. The created arc is added to the set of arcs
- **Right-Arc**$_k$: same as Left-arc, but creates a right arc

Example

John likes linguistics

Transition

–

Buffer

[John likes linguistics]

Stack

[ ]

Arcs

–

Example

John likes linguistics

Transition

Shift

Buffer

[likes linguistics]

Stack

[John]

Arcs

–

Example

John likes linguistics

Transition

Shift

Buffer

[linguistics]

Stack

[John likes]

Arcs

–

Example

John likes linguistics

Transition

Left-Arc(subj)

Buffer

[linguistics]

Stack

[likes]

Arcs

subj(likes, John)

Example

John likes linguistics

Transition

Shift

Buffer

[ ]

Stack

[likes linguistics]

Arcs

subj(likes, John)

Example

John likes linguistics

Transition

Right-Arc(obj)

Buffer

[ ]

Stack

[likes]

Arcs

subj(likes, John)

obj(likes,linguistics)

Our new transition-based system

Handling two linguistic dimensions

- Two stacks: a syntactic stack and a lexical stack
- One buffer to synchronize the two dimensions
- Processed items: a set of syntactic arcs and a set of lexical trees

Handling MWEs

- Mild extension of arc-standard parser
- Specific transitions to deal with MWE identification

Transition system

Configuration

(Buffer, SynStack, SynArcs, LexStack, LexTrees)

Initial

([w_1, . . . , w_n], [ ], { }, [ ], { })

Input: w_1, . . . , w_n

Terminal

([ ], [x], SynArcs, [ ], LexTrees)

Output: SynArcs, LexTrees

Transition system

Shift

Moves next token from Buffer to both stacks

Right-Arc(k), Left-Arc(k)

Adds syntactic arc between top items on syntactic stack

Merge$_F$(t)

Creates lexical tree from top items on both stacks – fixed MWE

Merge$_N$(t)

Creates lexical tree from top items on lexical stack – non-fixed MWE

Complete

Adds lexical tree from lexical stack

Example parse

Transition

–

Buffer

[he made a few decisions]

SynStack

[ ]

SynArcs

–

LexStack

[ ]

LexTrees

–

Example parse

Transition
Shift

Buffer
[made a few decisions]

SynStack
[he]

SynArcs
–

LexStack
[he]

LexTrees
–

Example parse

Transition
Complete

Buffer
[made a few decisions]

SynStack
[he]

SynArcs
–

LexStack
[ ]

LexTrees
he

Example parse

Transition
Shift

Buffer
[a few decisions]

SynStack
[he made]

SynArcs
–

LexStack
[made]

LexTrees
he

Example parse

Transition
Left-Arc(subj)

Buffer
[a few decisions]

SynStack
[made]

SynArcs
subj(made, he)

LexStack
[made]

LexTrees
he

Example parse

Transition
Shift

Buffer
[few decisions]

SynStack
[made a]

SynArcs
subj(made, he)

LexStack
[made a]

LexTrees
he

Example parse

Transition
Shift

Buffer
[decisions]

SynStack
[made a few]

SynArcs
subj(made, he)

LexStack
[made a few]

LexTrees
he

Example parse

Transition

Merge$_F$(A)

Buffer

[decisions]

SynStack

[made A(a, few)]

SynArcs

subj(made, he)

LexStack

[made A(a, few)]

LexTrees

he

Example parse

Transition
Complete

Buffer
[decisions]

SynStack
[made A(a, few)]

SynArcs
subj(made, he)

LexStack
[made]

LexTrees
he, A(a, few)

Example parse

Transition
Shift

Buffer
[ ]

SynStack
[made A(a, few) decisions]

SynArcs
subj(made, he)

LexStack
[made decisions]

LexTrees
he, A(a, few)

Example parse

Transition
Left-Arc(mod)

Buffer
[ ]

SynStack
[made decisions]

SynArcs
subj(made, he)
mod(decisions, A(a, few))

LexStack
[made decisions]

LexTrees
he, A(a, few)

Example parse

Transition

$\text{Merge}_N(V)$

Buffer

[ ]

SynStack

[made decisions]

SynArcs

subj(made, he)

mod(decisions, A(a, few))

LexStack

[V(made, decisions)]

LexTrees

he, A(a, few)

Example parse

Transition
Complete

Buffer
[ ]

SynStack
[made decisions]

SynArcs
subj(made, he)
mod(decisions, A(a, few))

LexStack
[ ]

LexTrees
he, A(a, few), V(made, decisions)

Example parse

Transition
Right-Arc(obj)

Buffer
[ ]

SynStack
[made]

SynArcs
subj(made, he)
mod(decisions, A(a, few))
obj(made, decisions)

LexStack
[ ]

LexTrees
he, A(a, few), V(made, decisions)

Implementation and Evaluation

Implementation

- **Greedy parser** trained with averaged **perceptron**
- **Hard constraints**: Complete transitions are made implicit, i.e. only activated when arc transitions are selected by classifier

Evaluation

- **Two datasets**: English Web Treebank (+ Streusle) and French Treebank
- **Comparisons** with
  1. standard parser with extended labels including the MWE status
  2. partial systems where some transitions are deactivated
  3. pipeline systems: fixed MWE identification + parsing

Main experimental findings

More detailed results in the paper!

- Joint system outperforms standard parser for MWE analysis
- Hard constraints can be helpful for syntactic analysis
- Lexical layer helps syntactic layer prediction (not reverse)
- Preidentifying fixed MWE seems to be helpful

Results on French Treebank

	DEV				TEST			
System	UAS	LAS	MWE	FMWE	UAS	LAS	MWE	FMWE
Extended Labels	86.28	83.67	77.2	83.2	84.85	82.67	75.5	81.9
Ours (explicit)	86.36	83.77	79.7	86.0	84.98	82.79	**79.3**	**84.8**
Ours (implicit)	**86.61**	**84.10**	**80.0**	**86.2**	**85.04**	**82.93**	78.4	84.3
Syntactic only	86.39	83.77	-	85.0	85.02	82.84	-	83.8
Lexical only	-	-	80.0	-	-	-	79.5	-
Fixed only	-	-	-	85.7	-	-	-	85.7
Pipeline	85.49	83.50	**81.8**	85.7	84.84	82.89	**81.1**	**85.7**

Results on English Web Treebank

	TRAIN Cross-validation			TEST		
System	UAS	LAS	MWE	UAS	LAS	MWE
Extended labels	86.16	81.76	49.6	86.31	82.02	46.8
Ours (explicit)	86.25	82.09	52.9	86.05	81.68	**53.4**
Ours (implicit)	**86.81**	**82.68**	**55.0**	**87.05**	**83.14**	51.6
Syntactic only	86.35	82.23	-	86.41	82.20	-
Lexical only	-	-	54.5	-	-	53.6

Conclusion

Contributions

- A new representation integrating MWE analysis and syntactic analysis
- A new transition-based system predicting such representation including special transitions for handling MWEs

Future work

- Implementing more advanced features: beam-search, dynamic oracles, deep learning, distributional semantics
- Evaluating on more datasets

Thanks!

Questions/Comments?

Statistics on datasets

Corpus	EWT		FTB		
	Train	Test	Train	Dev	Test
# sent.	3,312	500	14,759	1,235	2,541
# tokens	48,408	7,171	443,113	38,820	75,216
# MWEs	2,996	401	23,556	2,119	4,043
# fixed	-	-	10,987	925	1,992