Benchmarking Joint Lexical and Syntactic Analysis on Multiword-Rich Data

Mathieu Constant, Hector Martinez Alonso

Benchmarking Joint Lexical and Syntactic Analysis on Multiword-Rich Data

Matthieu Constant
Université de Lorraine, ATILF, CNRS
Nancy, France
Mathieu.Constant@univ-lorraine.fr

Héctor Martinez Alonso
Inria (ALMAnaCH)
Paris, France
hector.martinez-alonso@inria.fr

Abstract

This article evaluates the extension of a dependency parser that performs joint syntactic analysis and multiword expression identification. We show that, given sufficient training data, the parser benefits from explicit multiword information and improves overall labeled accuracy score in eight of the ten evaluation cases.

1 Introduction

In this paper, we expand the work of Constant and Nivre (2016) —henceforth CN16— by evaluating their system more extensively, representing Multiword Expressions (MWEs) in different ways that are linguistically motivated. Their transition-based system jointly performs lexical analysis and syntactic dependency parsing, using special transitions for MWE identification. In particular, these special transitions generate new lexical nodes for MWEs, that can also serve as nodes of the syntactic dependency trees. Their system is based on the classical split between fixed and free MWEs. Fixed MWEs defined by Sag et al. (2002) are contiguous. They are considered syntactically non-decomposable and are represented as a single syntactic node that requires a part of speech (POS) tag like all other tokens. Free MWEs are the remaining MWEs, that usually display regular internal structure and variations. The system predicts their internal syntactic structure, and their MWE status. The hypothesis behind this approach is that such a specialized extension of a standard transition-based parser to capture lexical relation for MWEs is a better option than using a regular transition based parser that relies on distributed annotation for MWEs as it is used for example in Universal Dependencies [UD] (Nivre et al., 2016).

In our experiments, we used UD, which we believe is an interesting playground because it provides different lexical-association labels such as MWEs. Nonetheless, we encounter an important drawback regarding MWEs, i.e. they are not provided with an overall POS, which plays a key role in our parsing systems. We have therefore proposed a common filler principle in order to automatically assign a POS to each MWE. Our other hypothesis is that enriching treebanks with explicit annotation of MWE status and MWE POS should help parsing accuracy. In addition, since UD treebanks may be non-projective, we have improved the parsing algorithm to account for non-projective trees, which the original work in CN16 could not provide. In the setup of CN16, only projective sentences could be used for training.

2 Joint lexical and syntactic analysis

The system by Constant and Nivre (2016) is based on a factorized lexical and syntactic representation, that consists of a graph over lexical nodes. Every lexical node corresponds to a lexical unit: either a simple unit or an MWE. It incorporates linguistic attributes (unit form, POS tag). MWE nodes may be of two sorts: fixed and free MWEs.

The representation can be decomposed into a lexical and a syntactic layer. The lexical layer is a forest of trees over lexical nodes. Every MWE is represented as a tree whose root is the lexical node of the MWE and its children are its (potentially non-adjacent) components. For instance, the verb-particle construction (VPC) *gave up* is a verbal lexical node which child nodes are *give* and *up*. The syntactic layer is a dependency tree over syntactic nodes. A syntactic node is either a simple lexical unit or a fixed MWE.

These two layers share the syntactic nodes because these nodes correspond to lexical ones. Con-

Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017), pages 181–186,
Valencia, Spain, April 4.

Initial:	$([\,], [\,], [0, \ldots, n], \{\,\}, \{\,\})$		
Terminal:	$([x], [\,], [\,], A, L)$		
Shift:	$(\sigma_l, \sigma_s, i\|\beta, A, L)$	$\Rightarrow$	$(\sigma_l\|i, \sigma_s\|i, \beta, A, L)$
Right-Arc(k):	$(\sigma_l\|x\|y, \sigma_s, \beta, A, L)$	$\Rightarrow$	$(\sigma_l\|x, \sigma_s, \beta, A \cup \{(x, k, y)\}, L)$
Left-Arc(k):	$(\sigma_l\|x\|y, \sigma_s, \beta, A, L)$	$\Rightarrow$	$(\sigma_l\|y, \sigma_s, \beta, A \cup \{(y, k, x)\}, L)$
Merge$_F$(t):	$(\sigma_l\|x\|y, \sigma_s\|x\|y, \beta, A, L)$	$\Rightarrow$	$(\sigma_l\|t(x, y), \sigma_s\|t(x, y), \beta, A, L)$
Merge$_N$(t):	$(\sigma_l, \sigma_s\|x\|y, \beta, A, L)$	$\Rightarrow$	$(\sigma_l, \sigma_s\|t(x, y), \beta, A, L)$
Complete:	$(\sigma_l, \sigma_s\|x, \beta, A, L)$	$\Rightarrow$	$(\sigma_l, \sigma_s, \beta, A, L \cup \{x\})$
Swap:	$(\sigma_l, \sigma_s\|x\|y, \beta, A, L)$	$\Rightarrow$	$(\sigma_l, \sigma_s\|y, x\|\beta, A, L)$

Figure 1: Transition system for joint syntactic and lexical analysis handling non-projectivity. This schema simply extends (Constant and Nivre, 2016) system by adding a **Swap** transition.

versely, lexical nodes are not necessarily syntactic ones: the light verb construction *make decision* is an MWE node, but is not part of the syntactic tree; only its components *make* and *decision* are. The fixed MWE *at least* is a syntactic node, but its child nodes *at* and *least* are not.

2.1 A transition-based system

The proposed transition system is a mild extension of an arc-standard system (Nivre, 2004), as schematized in Figure 1. It iteratively builds a graph over lexical nodes by applying a sequence of actions (namely transitions) from an initial parser state (namely **Initial** configuration) to a terminal state (namely **Terminal** configuration). Every parsing state is a 5-uple made of two stacks (a lexical stack σ_l and a syntactic stack σ_s), one buffer (β), a set of already predicted syntactic arcs (A) and a set of already predicted lexical trees (L). We use only one buffer in order to synchronize the prediction of the two layers, as they share elements, namely syntactic nodes. Each element popped from the buffer via the **Shift** transition is put on top of the two stacks.

The set of transitions also includes standard transitions devoted to create syntactic arcs (**Right-Arc** and **Left-Arc**) from the syntactic stack. MWE lexical trees are constructed by applying either **Merge**$_F$ for fixed MWEs or **Merge**$_N$ for free MWEs. Both transitions take the two top elements x and y on the lexical stack (and on the syntactic stack for fixed MWEs since they are shared by the two layers) and creates a new lexical node which children are x and y on top of the stack(s). The **Complete** transition is applied to complete the lexical unit on top of the lexical stack, i.e. it is moved to the set of predicted lexical trees.

The system proposed by Constant and Nivre (2016) only works for predicting projective syntactic trees. A classical way to deal with non-projectivity is to add a **Swap** transition, permuting the two top elements on the syntactic stack, the second element being pushed back to the buffer (Nivre, 2009). This simple integration in our joint system is not as straightforward as in a classical arc-standard system. One needs to add more conditions to apply the **Shift** transition. If the buffer is not empty, the first element x is moved onto the syntactic stack. It is also pushed onto the lexical stack if the following condition holds: x must not have been already pushed on the lexical stack in order to avoid it being processed multiple times during lexical analysis.

In our experiments, we also used a partial system that predicts the syntactic layer only. For this, the lexical stack is deactivated as well as the **Merge**$_N$ and **Complete** transitions. This partial system is equivalent to the one in Nivre (2014).

3 Multiword-aware treebanks

We use the Universal Dependencies treebanks or UD (Nivre et al., 2016) to obtain data for our experiments. UD has different labels that indicate different kinds of lexical associations, some of them more apt for a treatment as fixed or free.

For fixed MWE labeled as *mwe*, UD proposes a flat, first-headed analysis. While this simplifies our task of choosing fixed MWEs for our experiments, we do not have explicit information on the part of speech that a given fixed mwe would have if it were treated as a single lexical unit.

Since the parser in Section 2 treats MWEs as single units for attachment purposes after a **Merge** transition, it is desirable to have access to the factual part of speech of that MWE, given that POS

information is the most important feature to determine a word's attachment.

Incorporating POS information of MWEs in our parsers requires enriching the labels of treebanks with the POS of the overall function of the MWE, and indicating whether it is free or fixed.[1]

Fixed MWEs have a factual part of speech when treated as a unit, which need not be the part of speech of any of its parts, e.g. 'by and large' is an adverb, unlike any of its parts. However, factual part of speech of a MWE can be approximated from the MWE's syntactic label, e.g. if the overall label for 'by and large' is *advmod*, its most plausible part of speech is adverb. We refer to the most frequent POS that satisfies a certain label as **most-common filler**.

This heuristic is geared towards completing fixed-MWE information, but we also apply it to free MWEs to give account for the potential particularities in MWEs of the different treebanks.

3.1 Multiword definition scope

The *mwe* label is not the only MWE indicator. For instance, the *name* relation also encodes MWEs, often treated as named entity spans. Different languages have different extensions of the *compound* relation. In English a regular *compound* is something like 'phone **book**' while a *compound:prt* would be '**fall** off'. We use five different variants of the definition of MWE, both free and fixed, for our experiments. These variants aim at capturing semantic and syntactic variation.[2]

Variant A: *mwe* labels are fixed; *compound* (and its extensions), *cc:preconj*, *auxpass:reflex*, and *name* are free. This variant represents a fairly standard view of multiword expressions, where fixed multiwords are only the ones that are grammaticalized, and the other relations that represent multiword-related lexical association like compounds or reflexive pronouns of verbs are treated as free multiwords.

Variant B is similar to A, but also includes all verb auxiliaries as members of free multiwords, namely *aux*, *auxpass* and *cop*. This variant aims at giving account for the pragmatic preference of certain verbs to appear in specific periphrastic tenses or their lexical preference for one kind of auxiliary or modal verb.

Variant C uses the inventory of variant A, but all labels correspond to free multiwords. This variant intends to relax the hard constrain for fixed multiwords to form an uninterrupted span.

Variant D only contemplates the *mwe* label as fixed, and no free expressions. This span aims at measuring the contribution of only focusing on grammaticalized multiwords.

Variant E Same as variant D, except it is a strict variant where discontinuous spans with the *mwe* label are ignored during training.

4 Experiments

Each of the competing systems in our experiments is a combination of one of the five data variants (Section 3.1), and one of the three parsers: the full (FULL) or partial (PART) parsers in Section 2.1, or a standard transition-based parser (STD) without a lexical stack or **Merge/Complete** transitions. For instance, the FULL_C system uses the FULL parser on Variant C of the multiword inventory.

We compare these systems to a baseline without special data transformations for MWEs, and that depends on the standard transition-based parser. The aim of our experiments is not to optimize a parser for UD, but to benefit from the amount and variety of data offered by it to benchmark the possibilities of joint lexical and syntactic prediction.

Data: We have chosen treebanks where the *mwe* label constitutes at least 1% of the labels in the development section, and where the support is of at least 100 instances, cf. Table 1. Note that two of these treebanks are not the canonical treebank for their respective language.

Language	Treebank	Train	Test	Dev
Catalan	*UD_Catalan*	429.2k	59.5k	58.0k
Dutch	*UD_Dutch-LassySmall*	88.9k	4.5k	4.6k
Persian	*UD_Persian*	121.0k	16.0k	15.8k
Spanish	*UD_Spanish-Ancora*	453.2k	53.6k	53.4k
Swedish	*UD_Swedish*	66.6k	20.4k	9.8k

Table 1: Treebank properties.

The only variable parameter in our experiments is the choice of system, namely of parser and data variant. We replicate the remaining parameters with the choices in (Constant and Nivre, 2016), namely 6 training iterations, static oracle, greedy perceptron learning and the same feature templates, which we do not tune for any language. Using no language tuning allows us to evaluate equally on the test and development data.

[1] Parser is freely available at `https://github.com/MathieuConstant/lgtools`

[2] Conversion code is available at `https://github.com/hectormartinez/mweparse`

Language	LAS_{BL}	LAS_{sys}	System	fixed LAS	fixed head	#O
Catalan	86.09	86.38	FULL_C	94.95 (+1.56)	73.60 (0)	11
Dutch	78.05	78.22	FULL_C	61.82 (+6.47)	52.17 (-0.37)	7
Persian	79.01	—	—	—	—	0
Spanish	85.56	85.77	STD_B	90.13 (-0.15)	70.95 (+7.44)	7
Swedish	81.93	82.25	STD_B	55.59 (-2.67)	50.55 (+4)	8
Catalan	86.16	86.48	PART_F	93.76 (+1.16)	78.13 (+8.11)	12
Dutch	78.83	—	—	—	—	0
Persian	78.79	79.41	FULL_C	82.26 (+2.15)	71.53 (+11.87)	6
Spanish	85.88	85.93	PART_F	87.71 (+0.89)	69.27 (+5.39)	2
Swedish	78.32	79.14	FULL_E	56.92 (+5.43)	50.28 (+12.11)	9

Table 2: Language-wise best system scores for the test (above) and development (below) sections.

5 Results

After prediction, we evaluate each parser-variant combination. Table 2 shows the results for each language for the test (above) and development (below) sections, for which we provide the Labeled Attachment Score of the baseline (LAS_{BL}), the Labeled Attachment Score of the best system above the baseline (LAS_{sys}) as well as the system's descriptor. We also provide the attachment score for the *mwe* label, which corresponds to the accuracy of identifying fixed MWEs, and the labeled accuracy of the first token of fixed MWEs, which corresponds to correctly finding the head of MWEs. For these two metrics, we provide the difference with regard to the baseline. Moreover, we provide the amount of systems out of 12 which outperform the baseline for a given language.

While using any of the variants A–D which contemplate MWEs aids syntactic prediction in general, there is no general preference. We attribute this variation to differences in corpus and linguistic properties, but also in how the UD principles are annotated on each treebank. We do observe, however, that certain parsers lend themselves better for certain data variants. For instance, variant C is best combined with the FULL parser.

The FULL parser yields an average test-section improvement of 4.7% on MWE accuracy across all treebanks and data variants with respect to the baseline, while it gives 3.6% with respect to PART system. This improvement is not only local to MWE labels. We observe a small but consistent improvement of about 0.20 both in root labeled accuracy and in the accuracy of the nominal roles for subject, direct object and nominal modifier.

Larger treebanks have more stable results and aid the learning system. Indeed, the FULL transition system, which has more operations will need more data to converge. This argument is supported by the irregularity of behavior of Dutch and Persian, which only yield improvements in one of the evaluation sections. We attribute this instability to the size of the training set and the sensitivity to sampling bias of the small evaluation sections.

We have also assessed the usefulness of the most common filler heuristic. If during the label-enriching operation (Sec. 3) we mark the label of each part of a MWE with its original POS—instead of giving all parts of a MWE the most-common filler POS—, the LAS drops on 1-2% for all treebanks and variants. Moreover, even for the E variant using the standard parser, the system presents improvements, which is a consequence of the multiword labels, both free and fixed, containing also POS information. We consider this improvement as support evidence for our initial hypothesis.

6 Related Work

MWE processing is an ever growing research topic since Sag et al. (2002), as it has been shown in (Ramisch, 2015). On the side of MWE-aware dependency parsing, the main line of research for joint approaches is to use standard dependency parsers using special arc labels and flat structures for MWEs (Nivre and Nilsson, 2004; Eryiğit et al., 2011; Seddah et al., 2013; Nasr et al., 2015). Vincze et al. (2013) and Candito and Constant (2014) integrate richer arc labels and non-flat structures to predict internal MWE structure. Truly joint approaches incorporating special parsing mechanisms to handle MWE recognition is a recent line of research (Constant and Nivre, 2016).

On the side of UD, Silveira and Manning (2015) explore whether the UD treebank formalism needs an additional representation to improve parsing. Salehi et al. (2016) identify MWE in a surprise target language with no prior knowledge of MWE

patterns, using training on a UD MWE-aware treebank of a source language.

7 Conclusions and Further Work

We have expanded the CN16 system to a more thorough evaluation, using different variants of multiword inventories. We show that, given sufficient data, the parser benefits from explicit multiword information and improves overall labeled accuracy score in eight of the ten evaluation cases.

Further work includes devising more refined strategies to generate full lexical entries for joined MWE tokens. The most-common filler can also be used to calculate their prototypical morphological features. Moreover, if the treebank is lemmatized we can create pseudolemmas for the MWEs by concatenating the lemmas of the formants. While these pseudolemmas might differ from the actual reference forms, they would have the same distribution as the overall MWEs, thereby contributing to parsing to the same extend than the other lemmas in the treebank.

We also intend to perform a more thorough evaluation of the improvements of non-projective parsing against the increased complexity of the parser, and how it relates to the effect on projectivity of the flat, projective subtrees enforced by fixed MWEs.

The overall system should also be evaluated on the, as per February 2017, upcoming version 2.0 of Universal Dependencies, where the treatment of MWEs has been redefined and the new label inventory provides, besides the *fixed* label for grammaticalized MWEs, a *flat* label for named entities. Ideally, further versions of UD will present a more homogeneous and streamlined treatment of both fixed and free multiwords.

References

Marie Candito and Matthieu Constant. 2014. Strategies for contiguous multiword expression analysis and dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 743–753, Baltimore, Maryland, June. Association for Computational Linguistics.

Matthieu Constant and Joakim Nivre. 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, August. Association for Computational Linguistics.

Gülşen Eryiğit, Tugay Ilbay, and Ozan Arkan Can. 2011. Multiword expressions in statistical dependency parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 45–55, Dublin, Ireland, October. Association for Computational Linguistics.

Alexis Nasr, Carlos Ramisch, José Deulofeu, and André Valli. 2015. Joint dependency parsing and multiword expression tokenization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1116–1126, Beijing, China, July. Association for Computational Linguistics.

Joakim Nivre and Jens Nilsson. 2004. Multiword units in syntactic parsing. In *Proceedings of Methodologies and Evaluation of Multiword Units in Real-World Applications (MEMURA)*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Dan Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In Frank Keller, Stephen Clark, Matthew Crocker, and Mark Steedman, editors, *Proceedings of the ACL Workshop Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57, Barcelona, Spain, July. Association for Computational Linguistics.

Joakim Nivre. 2009. Non-projective dependency parsing in expected linear time. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 351–359, Suntec, Singapore, August. Association for Computational Linguistics.

Joakim Nivre. 2014. Transition-Based Parsing with Multiword Expressions. In *2^{nd} PARSEME General Meeting, Athens, Greece*.

Carlos Ramisch. 2015. *Multiword Expressions Acquisition: A Generic and Open Framework*, volume XIV of *Theory and Applications of Natural Language Processing*. Springer.

Ivan A. Sag, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword Expressions: A Pain in the Neck for NLP. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, volume 2276 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin Heidelberg.

Bahar Salehi, Paul Cook, and Timothy Baldwin. 2016. Determining the multiword expression inventory of a surprise language. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 471–481, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola Galletebeitia, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiórkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clergerie. 2013. Overview of the SPMRL 2013 shared task: A cross-framework evaluation of parsing morphologically rich languages. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, pages 146–182, Seattle, Washington, USA, October. Association for Computational Linguistics.

Natalia Silveira and Christopher Manning. 2015. Does universal dependencies need a parsing representation? an investigation of english. In *Proceedings of the Third International Conference on Dependency Linguistics (Depling 2015)*, pages 310–319, Uppsala, Sweden, August. Uppsala University, Uppsala, Sweden.

Veronika Vincze, János Zsibrita, and István Nagy T. 2013. Dependency parsing for identifying hungarian light verb constructions. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 207–215, Nagoya, Japan, October. Asian Federation of Natural Language Processing.