TRAVERSAL at PARSEME Shared Task 2018: Identification of Verbal Multiword Expressions Using a Discriminative Tree-Structured Model

Jakub Waszczuk

TRAVERSAL at PARSEME Shared Task 2018: Identification of Verbal Multiword Expressions Using a Discriminative Tree-Structured Model

Jakub Waszczuk
Heinrich Heine University
Düsseldorf, Germany
waszczuk@phil.hhu.de

Abstract

This paper describes a system submitted to the closed track of the PARSEME shared task (edition 1.1) on automatic identification of verbal multiword expressions (VMWEs). The system represents VMWE identification as a labeling task where one of two labels (MWE or not-MWE) must be predicted for each node in the dependency tree based on local context, including adjacent nodes and their labels. The system relies on multiclass logistic regression to determine the globally optimal labeling of a tree. The system ranked 1st in the general cross-lingual ranking of the closed track systems, according to both official evaluation measures: MWE-based F_1 and token-based F_1.

1 Introduction

In this paper we give a description of TRAVERSAL,[1] a system submitted to the edition 1.1 of the PARSEME shared task on automatic identification of VMWEs (Ramisch et al., 2018). The task was multilingual (treebanks annotated with VMWEs were provided for 20 languages) and its aim was to automatically identify VMWEs of several categories: light-verb constructions, idioms, inherently reflexive verbs, verb-particle constructions, multi-verb constructions, and inherently adpositional verbs.

TRAVERSAL relies on the assumption that MWEs form connected syntactic components, i.e., that lexical elements of a single MWE occurrence should be adjacent in the dependency analysis of the underlying sentence. Based on this assumption, TRAVERSAL represents the task of MWE identification as a labeling task where one of two labels (`MWE` or `not-MWE`) must be predicted for each node in the dependency tree based on local contextual information: dependency labels, word forms, lemmas, POS tags, etc., as well as the `MWE`/`not-MWE` labels assigned to adjacent nodes.

In order to capture such properties, our system encodes labelings of dependency trees as tree traversals such that each traversal corresponds to a distinct labeling of the input dependency tree. The task of MWE labeling is then reduced to finding the best traversal of the dependency tree. TRAVERSAL relies on multiclass logistic regression to discriminate between plausible and implausible traversals.

Labeling alone is not sufficient to predict MWEs since it doesn't tell us where the individual MWE occurrences start and end, a sub-task which we refer to as MWE segmentation. TRAVERSAL relies on a rather rudimentary solution to this problem – by default, all adjacent dependency nodes marked as `MWE`s of the same category are assumed to form a single MWE occurrence.

This paper is structured as follows. In Sec. 2 we give information about related work. In Sec. 3 we describe our system. In Sec. 4 we describe the experiments performed to determine an optimal setup for the shared task, and in Sec. 5 we summarize the results obtained by our system with this setup. Finally, we conclude in Sec. 6 and outline the possible directions for future work.

2 Related work

A notable example of a dependency treebank in which MWEs are annotated as connected syntactic components (subtrees) is the Prague Dependency Treebank (Bejček et al., 2012), in which named entities

[1] Available at https://github.com/kawu/traversal under the 2-clause BSD license.

and VMWEs are both annotated on top of the tectogrammatical (deep syntactic/shallow semantic) layer. Treatment of MWEs as connected syntactic components is also adopted by Abeillé and Schabes (1989) within the framework of tree-adjoining grammars (Joshi and Schabes, 1997). In their approach, MWEs are modeled as families of multi-anchored elementary trees.

Two broad strategies of identifying verbal MWEs can be distinguished depending on whether MWE identification takes place during (Vincze et al., 2013; Green et al., 2013; Candito and Constant, 2014; Le Roux et al., 2014; Nasr et al., 2015; Waszczuk et al., 2016; Constant and Nivre, 2016) or after (Constant et al., 2012; Nagy T. and Vincze, 2014) syntactic parsing.[2] The former approach is based on the intuition that both tasks can benefit from each other and, therefore, should be performed jointly. The latter approach is simpler conceptually and can benefit from the significantly restricted solution space in comparison with the joint methods. TRAVERSAL requires that dependency trees are already constructed and, thus, adopts the latter strategy.

The method implemented in TRAVERSAL can be seen as an extension of sequential conditional random fields (CRFs) to tree structures. CRFs were applied to the task of MWE identification by Constant et al. (2012), Scholivet and Ramisch (2017), and Maldonado et al. (2017), among others. However, sequential CRFs are best applied to identification of continuous entities, while verbal MWEs are often discontinuous. The issue of discontinuity is handled in TRAVERSAL transparently since it assumes that input takes the form of a (dependency) tree and not a sequence. Otherwise, it relies on the same statistical backbone as CRFs – multiclass logistic regression (Sutton et al., 2012).

Another family of approaches related to the method used in TRAVERSAL is that of the graph-based approaches to dependency parsing (Kübler et al., 2009). The 2-order extension of Eisner's algorithm, for instance, adopts similar factorization as TRAVERSAL in that it captures relations between the nodes, their parents, and their siblings (McDonald and Pereira, 2006). This algorithm, however, is restricted to projective dependency trees only. Exact non-projective parsing with such a 2-order model is intractable (McDonald and Satta, 2007). The method implemented in TRAVERSAL (which is already 2-order) can be extended to capture higher-order relations without becoming intractable and can be used in combination with non-projective trees. This is possible because TRAVERSAL labels the nodes but considers the input dependency structure as fixed.

3 System description

In this section, we describe the details of the implemented system. In Sec. 3.1, we summarize the types of features the system is able to incorporate. In Sec. 3.2, we explain how to encode the possible tree labelings as tree traversals, and we formalize the notion of traversals themselves. In Sec. 3.3, we describe the multiclass logistic regression model used to score the traversals.

3.1 Feature types

In order to discriminate the "good" from the "bad" labelings, TRAVERSAL relies on the following types of features, restricted in their scope to adjacent nodes and their labels:

- Unary features, restricted to the context of a single node
- Binary parent-child features, restricted to the context of a node and its parent
- Binary sibling features, restricted to the context of a node and its closest sibling[3] on the left
- Ternary features, giving access to the node, its parent, and its closest left sibling node

In our experiments we only relied on unary and binary features, assuming that ternary features might lead to overfitting. Using ternary features is nevertheless a feasible option as it does not change the computational cost of labeling in this model.

[2] Another approach would be to identify MWEs before parsing, but we posit it is not very well adapted to verbal MWEs.

[3] We say that node v is a sibling of node w if both have the same parent in the dependency tree.

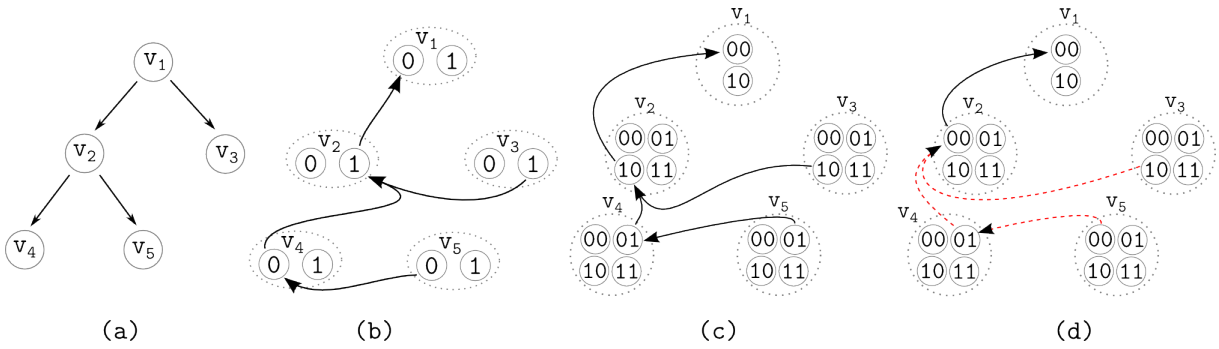

Figure 1: (a) Dependency tree with five lexical nodes. (b) A hyperpath corresponding to the labeling where only v_2 and v_3 are considered as MWEs. (c) A hyperpath with copying of parent labels, corresponding the same labeling as (b). (d) An invalid hyperpath, not encoded in the hypergraph (because of the red dashed hyperarcs – for instance, the hyperarc between v_5 and v_4 connects two hypernodes with different parent labels, even though v_5 and v_4 are siblings).

3.2 Encoding

Encoding consists in constructing a compact representation of all traversals of the given dependency tree. Formally, the resulting structure is a hypergraph (Gallo et al., 1993; Klein and Manning, 2001), and each traversal of the dependency tree is represented by a distinct hyperpath in this hypergraph.

Consider the dependency tree depicted in Fig. 1 (a). Its hypergraph encoding contains two hypernodes for each node in the dependency tree, one labeled with 0 (not-MWE), the other labeled with 1 (MWE). Furthermore, the hypergraph is populated with hyperarcs in such a way that it contains one hyperpath per each possible labeling of the nodes. For instance, the hyperpath shown in Fig. 1 (b) represents the labeling where v_2 and v_3 are both considered as MWEs (the hyperpath traverses these nodes via hypernodes labeled with 1), while the other nodes are not-MWEs.

Hyperacs determine the scope of the features we can define. However, certain relations we want to capture are not available in this hypergraph representation yet. For instance, there is no hyperarc between v_3 and v_1 in Fig. 1 (b), which prevents us from using binary parent/child features. To solve this issue, we copy the labels of parents to their children, as shown in Fig. 1 (c). Namely, each node is now represented by 4 hypernodes encoding the node's label as well as the label of its parent: 00 means that both nodes are not-MWEs, 10 that the node is a MWE but the parent is not, 01 that the parent is a MWE but the node is not, and 11 that both are MWEs. When populating the hypergraph with hyperarcs, we make sure that only valid combinations of hypernodes are connected – invalid hyperpaths such as the one shown in Fig. 1 (d) are thus not encoded.

3.3 Model

Let $\mathcal{H}(d)$ be the set of hyperpaths encoded in the hypergraph for a given dependency tree d. The task of labeling is reduced to the task of finding the best-score hyperpath $h \in \mathcal{H}(d)$. To solve the latter problem, we rely on a multiclass logistic regression model in which each hyperpath h is reduced to a vector of features f^h of a fixed length n, where n is the number of features. More precisely, (i) each hyperarc a is represented as a binary feature vector f^a such that $f^a_k = 1$ iff the k-th feature of the model holds within the context of a, and (ii) $f^h = \sum_{a \in h} f^a$.

Let θ be a vector of real-valued model parameters of length n. The conditional probability of a particular hyperpath $h \in \mathcal{H}(d)$ given dependency tree d is defined as:

$$p_\theta(h|d) = \exp(\theta \cdot f^h) \; / \; \sum_{h' \in \mathcal{H}(d)} \exp(\theta \cdot f^{h'}), \tag{1}$$

where $\cdot$ is a dot product. Determining the highest-probability hyperpath, as well as determining the marginal probabilities of the individual hyperarcs (important for parameter estimation), can be performed

efficiently using the inside-outside algorithm. For parameter learning, we rely on the maximum likelihood estimates (w.r.t. training data T) with normal priors over θ. The log-likelihood takes the form:

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

where the regularization parameter $1/2\sigma^2$ determines the strength of the penalty on high parameter values. The maximum likelihood estimates are then approximated using stochastic gradient descent with momentum (Qian, 1999; Ruder, 2016).

4 Experimental setup

In this section we describe how the system detailed in Sec. 3 was adapted to the shared task. In Sec. 4.1, we explain the process used to determine the set of feature templates. In Sec. 4.2 we describe the pre-processing steps used to facilitate MWE prediction for the individual languages. In Sec. 4.3 we return to the problem of MWE segmentation and outline the heuristics used to solve it. Finally, in Sec. 4.4, we give some details on how training of our system was performed.

4.1 Features templates

We used the Polish and French DEV datasets to determine a reasonable set of feature templates for our system. We started with a small set and incrementally augmented it with new templates as long as this led to better results on one or both languages. After a couple of iterations we ended up with the following set of templates, where 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, `Sib`-prefixed templates apply to sibling relations, `Par`-prefixed templates apply to parent-child relations, and `Bin`-prefixed templates apply to both types of binary relations:

- `Lem`: (l_v, m_v)
- `SibMwe` and `ParMwe`: (m_v, m_w)
- `SibLem` and `ParLem`: (l_v, l_w, m_v, m_w)
- `BinLem`: unordered pair $\{(l_v, m_v), (l_w, m_w)\}$
- `ParMweDep`: (m_v, m_w, d_v)
- `SibMweDep`: (m_v, m_w, d_v, d_w)
- `ParLemPosDep`: $(l_v, p_w, m_v, m_w, d_v)$
- `SibLemPos`: (l_v, p_w, m_v, m_w)
- `ParPosLemDep`: $(p_v, l_w, m_v, m_w, d_v)$
- `SibPosLem`: (p_v, l_w, m_v, m_w)

We used the above set of templates consistently for all languages and for all VMWE categories, except for Lithuanian – dependency trees were not available for this language. We therefore converted each sentence in the `LT` dataset to a pseudo-dependency tree in which (i) the first token is the root, (ii) every other token is the child of the preceding token, thus obtaining a model equivalent to a 2-order sequential CRF. We also adapted the default set of templates to Lithuanian by replacing the sibling templates with selected grandparent-related templates (possible due to the structure of pseudo-dependency trees).

4.2 Pre-processing

We applied various pre-processing procedures to 9 datasets in order to facilitate prediction of MWE labels. The pre-processing method most often applied, case lifting, consisted in reattaching case dependents to their grandparents so as to make MWEs of certain categories – notably, inherently adpositional verbs – connected.[4] We applied it to `BG`, `DE`, `EN`, `ES`, `HI`, `HR`, `PL`, and `SL` datasets. For Turkish, we removed tokens marked as `DERIV`s (with unspecified word form, and never marked as MWEs in training data) and copied word forms to lemmas where the latter were not present. In case of Slovak, we relied on language-specific POS tags rather than universal tags, since the latter were not provided, and assumed

[4]Consider the expression *based on data*, with the underlined words annotated as an inherently adpositional verb in the `EN` dataset. Case lifting changes the set of its internal dependency arcs from {based → data, on ← data} to {based → on, based → data}, thus making *based* and *on* adjacent in the resulting dependency structure.

	Corpus						TRAVERSAL results									
	#VMWE			%VMWE$_{(train+dev)}$			MWE-based					Token-based				
	Train	Dev	Test	Con	Con$_p$	Iso$_p$	P	R	**F1**	Rank	Delta	P	R	**F1**	Rank	Delta
BG	5364	670	670	95.0	97.34	93.42	75.59	47.61	**58.42**	4/11	-4.1	82.36	49.79	**62.06**	3/11	-1.8
DE*	2820	503	500	90.76	93.17	87.69	62.93	32.73	**43.06**	3/11	-2.21	76.17	38.36	**51.02**	2/11	-0.64
EL	1404	500	501		92.86	84.66	65.7	36.33	**46.79**	2/11	-2.97	82.16	42.01	**55.59**	1/11	1.89
EN**	331	0	501	91.84	98.19	95.17	55.5	21.16	**30.64**	2/10	-2.24	58.31	20.33	**30.15**	3/10	-4.22
ES	1739	500	500	76.55	90.13	84.59	28.84	33.4	**30.95**	4/11	-3.03	39.91	40.26	**40.09**	1/11	0.34
EU*	2823	500	500		95.28	84.38	78.28	58.4	**66.9**	4/10	-8.9	83.42	65.01	**73.07**	4/10	-3.76
FA*	2451	501	501		94.48	57.83	73.8	58.48	**65.26**	7/10	-12.57	90.19	65.23	**75.7**	6/10	-5.58
FR**	4550	629	498		98.17	92.62	77.19	44.18	**56.19**	1/13	5.65	84.72	48.76	**61.9**	1/13	6.18
HE	1236	501	502		76.45	73.69	50.33	15.14	**23.28**	1/10	0.59	74.64	18.1	**29.13**	2/10	-0.52
HI	534	0	500	98.5	98.88	85.58	66.3	60.6	**63.32**	5/10	-9.66	73.15	67.12	**70**	4/10	-3.35
HR	1450	500	501	63.38	96.56	87.03	68.04	46.59	**55.3**	1/10	11.03	78.14	50.73	**61.52**	1/10	11.55
HU**	6205	779	776		99.53	90.35	88.01	74.74	**80.84**	4/10	-9.47	89.91	79.61	**84.45**	4/10	-3.55
IT	3254	500	503		94.62	88.12	63.09	40.32	**49.2**	1/12	10.68	74.42	42.11	**53.78**	1/12	7.27
LT†	312	0	500		58.33	58.33	29.61	13.8	**18.83**	3/10	-13.34	55.56	16.92	**25.94**	3/10	-8.49
PL*	4122	515	515	80.87	86.09	82.42	77.02	59.22	**66.96**	1/11	6.42	81.85	59.03	**68.59**	1/11	3.67
PT*	4430	553	553		95.54	90.01	76.8	52.08	**62.07**	1/13	1.23	85.14	54.69	**66.6**	1/13	4.22
RO	4713	589	589		98.64	95.38	86.06	79.63	**82.72**	3/10	-2.56	88.84	82.26	**85.42**	2/10	-0.27
SL*	2378	500	500	71.61	88.12	86.14	79.41	54	**64.29**	1/10	21.95	83.61	54.54	**66.01**	1/10	14.04
TR	6125	510	506	55.54	98.21	91.08	81.48	26.09	**39.52**	5/10	-5.72	88.38	27.66	**42.13**	5/10	-10.89
AVG							*67.58*	*44.97*	***54***	*1/13*	*4.26*	*77.41*	*48.55*	***59.67***	*1/13*	*5.04*

Table 1: Detailed results of TRAVERSAL for 19 languages (identified by their ISO 639-1 codes)

that tokens with an unspecified dependency head are attached to the artificial root node (with ID=0). The same pre-processing steps were applied to TRAIN, DEV, and (blind) TEST data.

4.3 Segmentation

Once the labeling of a given dependency tree is determined, we need to determine the boundaries of the detected MWEs. To this end, we considered two heuristics: (i) all adjacent nodes marked as `MWE`s of the same category are considered as a single MWE occurrence, and (ii) if a group of adjacent nodes is marked as `MWE`s but it contains two (or more) verbs, the group is divided into two (or more) distinct MWEs. We applied the first heuristic for all languages except Farsi, where the second heuristic yielded better results, notably due to a relatively high frequency of neighboring MWEs in the `FA` dataset.

4.4 Training

We trained the models over the combined TRAIN+DEV datasets with $\sigma^2 = 10$ (see Sec. 3.3). For a given language, we trained one model per MWE category so as to handle the phenomenon of overlapping MWEs of different categories, often occurring in the provided annotated datasets.

5 Results

TRAVERSAL ranked 1st in the general ranking among the systems submitted to the closed track, according to both official evaluation measures: MWE-based F_1 and token-based F_1. Table 1 summarizes the performance of our system across 19 languages of the shared task (all except Arabic). For each language, the MWE-based and token-based precision (`P`), recall (`R`), and F_1 (`F1`) scores are reported, as well as the rank (`Rank`) of our system, and the difference (`Delta`) between the TRAVERSAL's F_1 score and the score of the other best closed-track system. 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. `Con` is the % of connected (via parental or sibling relation) VMWEs in the TRAIN+DEV dataset (no value $\Longrightarrow$ `Con`=`Con`$_p$), and `Con`$_p$ is the same measure after pre-processing. Finally, `Iso`$_p$ is the % of connected and isolated (with no adjacent VMWEs of the same category) VMWEs after pre-processing, for which the baseline segmentation heuristic is sufficient.

Language-wise, our system performed particularly well for Slavic and Romance languages, which is likely related to our choice of Polish and French for feature template engineering. `FA` was the most

	Continuous	Discont.	Multi-token	Single-token	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 2: Macro-average MWE-based F_1-scores for different 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 3: Macro-average MWE-based F_1-scores for different MWE categories[5]

challenging dataset for our system, which is clearly due to the low % of isolated VMWEs in this dataset and, consequently, low effectiveness of the implemented segmentation heuristics. TRAVERSAL performed well on datasets with both manually annotated (`FR`) and automatically obtained (`IT`, `HR`, `EL`) dependencies, thus showing robustness w.r.t. the quality of dependency annotations.

Concerning specialized phenomena (see Table 2), TRAVERSAL performed particularly well on discontinuous MWEs. This might be related to the view on MWEs adopted in our system, where continuous and discontinuous MWEs are not really distinguished – both are "continuous" in dependency trees. TRAVERSAL proved better in handling multi-token VMWEs than single-token (e.g. *to pretty-print*) VMWEs (outperformed by TRAPACC and TRAPACC_S in case of the latter) and exhibited certain preference for VMWEs already seen during training (outperformed by GBD-NER-standard and GBD-NER-resplit in case of unseen VMWEs). Besides, TRAVESAL turned out less efficient in identifying identical VMWEs and variants of VMWEs seen during trainig than TRAPACC and GBD-NER-resplit, respectively, even though it outperformed the two systems for the class of seen VMWEs in general.

Category-wise (see Table 3), TRAVERSAL was quite successful in identifying inherently adpositional verbs (IAV), which suggests that the pre-processing strategy of re-attaching case markers was effective. It also performed very well for inherently reflexive verbs (IRV), verbal idioms (VID), light-verb constructions (LVC.full), and inherently clitic verbs (LS.ICV, occurring only in the IT dataset). In case of "causative" LVCs (LVC.cause), TRAVERSAL was outperformed by varIDE, and in case of multi-verb constructions (MVC) – by TRAPACC and CRF-DepTree-categs.

6 Conclusions and future work

This paper presents a system dedicated to identification of verbal MWEs based on the explicit assumption that MWEs form connected components in dependency trees. It divides the task of MWE identification into two subsequent sub-tasks: (i) tree labeling (with two possible labels: `MWE` and `not-MWE`), and (ii) MWE segmentation (determining the boundaries of MWEs). For the former task, it relies on the multiclass logistic regression model in order to find the globally optimal labeling for a given tree. The system ranked 1st in the closed track of the PARSEME shared task, thus showing the viability of applying CRF-like models to identification of verbal – in particular, discontinuous – MWEs.

For future work, we consider improving the MWE segmentation component of our system. In particular, we would like to explore the idea of enriching labels with information about MWE boundaries so as to perform MWE segmentation jointly with MWE labeling. Another direction for future work would be to see to what extent external resources (MWE dictionaries, word embeddings) could be incorporated in our system. We could also explore usefulness of ternary features, so far ignored in our experiments. Finally, we plan to perform a more fine-grained error analysis, so as to get better insight into the advantages and limitations of the implemented method.

Acknowledgements

The author thanks Behrang QasemiZadeh and the anonymous reviewers for their valuable comments.

[5] In Tab. 1 and Tab. 2, macro-average F_1 is calculated based on macro-average precision and recall. In Tab. 3, macro-average F_1 is calculated directly as the mean of the relevant F_1-scores obtained for the individual languages.

References

Anne Abeillé and Yves Schabes. 1989. Parsing Idioms in Lexicalized TAGs. In *Proceedings of the Fourth Conference on European Chapter of the Association for Computational Linguistics*, EACL '89, pages 1–9, Stroudsburg, PA, USA. Association for Computational Linguistics.

Eduard Bejček, Jarmila Panevová, Jan Popelka, Pavel Straňák, Magda Ševčíková, Jan Štěpánek, and Zdeněk Žabokrtský. 2012. Prague dependency treebank 2.5 – a revisited version of pdt 2.0. In *In Proceedings of the 24th International Conference on Computational Linguistics (Coling 2012)*, pages 231–246.

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. 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. Association for Computational Linguistics.

Matthieu Constant, Anthony Sigogne, and Patrick Watrin. 2012. Discriminative Strategies to Integrate Multiword Expression Recognition and Parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 204–212. Association for Computational Linguistics.

Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. 1993. Directed Hypergraphs and Applications. *Discrete Appl. Math.*, 42(2-3):177–201, April.

Spence Green, Marie-Catherine de Marneffe, and Christopher D. Manning. 2013. Parsing Models for Identifying Multiword Expressions. *Computational Linguistics*, 39(1).

Aravind K Joshi and Yves Schabes. 1997. Tree-Adjoining Grammars. In Grzegorz Rozenberg and Arto Salomaa, editors, *Handbook of Formal Languages*, pages 69–123. Springer Berlin Heidelberg.

Dan Klein and Christopher D. Manning. 2001. Parsing and Hypergraphs. In *Seventh International Workshop on Parsing Technologies (IWPT-2001)*, October.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. Dependency parsing. *Synthesis Lectures on Human Language Technologies*, 1(1):1–127.

Joseph Le Roux, Antoine Rozenknop, and Matthieu Constant. 2014. Syntactic Parsing and Compound Recognition via Dual Decomposition: Application to French. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1875–1885. Dublin City University and Association for Computational Linguistics.

Alfredo Maldonado, Lifeng Han, Erwan Moreau, Ashjan Alsulaimani, Koel Dutta Chowdhury, Carl Vogel, and Qun Liu. 2017. Detection of Verbal Multi-Word Expressions via Conditional Random Fields with Syntactic Dependency Features and Semantic Re-Ranking. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 114–120. Association for Computational Linguistics.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics*.

Ryan McDonald and Giorgio Satta. 2007. On the complexity of non-projective data-driven dependency parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 121–132. Association for Computational Linguistics.

István Nagy T. and Veronika Vincze. 2014. VPCTagger: Detecting Verb-Particle Constructions With Syntax-Based Methods. In *Proceedings of the 10th Workshop on Multiword Expressions (MWE)*, pages 17–25, Gothenburg, Sweden, April. 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. Association for Computational Linguistics.

Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151.

Carlos Ramisch, Silvio Ricardo Cordeiro, Agata Savary, Veronika Vincze, Verginica Barbu Mititelu, Archna Bhatia, Maja Buljan, Marie Candito, Polona Gantar, Voula Giouli, Tunga Güngör, Abdelati Hawwari, Uxoa Iñurrieta, Jolanta Kovalevskaitė, Simon Krek, Timm Lichte, Chaya Liebeskind, Johanna Monti, Carla Parra Escartín, Behrang QasemiZadeh, Renata Ramisch, Nathan Schneider, Ivelina Stoyanova, Ashwini Vaidya, and Abigail Walsh. 2018. Edition 1.1 of the PARSEME Shared Task on Automatic Identification of Verbal Multiword Expressions. In *Proceedings of the Joint Workshop on Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG 2018)*, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747.

Manon Scholivet and Carlos Ramisch. 2017. Identification of Ambiguous Multiword Expressions Using Sequence Models and Lexical Resources. In *Proceedings of the 13th Workshop on Multiword Expressions (MWE 2017)*, pages 167–175. Association for Computational Linguistics.

Charles Sutton, Andrew McCallum, et al. 2012. An introduction to conditional random fields. *Foundations and Trends® in Machine Learning*, 4(4):267–373.

Veronika Vincze, István Nagy T., and János Zsibrita. 2013. Learning to Detect English and Hungarian Light Verb Constructions. *ACM Trans. Speech Lang. Process.*, 10(2):6:1–6:25, June.

Jakub Waszczuk, Agata Savary, and Yannick Parmentier. 2016. Promoting multiword expressions in A* TAG parsing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 429–439. The COLING 2016 Organizing Committee.