VarIDE at PARSEME Shared Task 2018: Are Variants Really *as Alike as Two Peas in a Pod*?

Caroline Pasquer
University of Tours
France

Carlos Ramisch
Aix Marseille Univ,
Université de Toulon,
CNRS, LIS, Marseille, France

Agata Savary
University of Tours
France

Jean-Yves Antoine
University of Tours
France

first.last@(univ-tours|lis-lab).fr

Abstract

We describe the *VarIDE* system (standing for *Variant IDEntification*) which participated in edition 1.1 of the PARSEME shared task on automatic identification of verbal multiword expressions (VMWEs). Our system focuses on the task of VMWE variant identification by using morphosyntactic information in the training data to predict if candidates extracted from the test corpus could be idiomatic, thanks to a naive Bayes classifier. We report results for 19 languages.

1 Introduction

Identifying multiword expressions (MWEs) such as *to* ***make ends meet*** and *to* ***give up*** in running text is a challenging problem (Baldwin and Kim, 2010; Constant et al., 2017). This is especially true for verbal MWEs (VMWEs), which, like verbs together with their subcategorization frames, are subject to complex morphological and syntactic transformations. As a consequence, VMWEs may occur under various forms, and it is especially important to identify expressions which are variants of each other.

Our system VarIDE, submitted to the PARSEME shared task 2018, focuses on the specific problem of variant identification. Shared task organizers provided training, development and test corpora (hereafter TRAIN, DEV, and TEST) manually annotated for VMWEs.[1] Given a VMWE (e.g. *to* ***have look*** 'to have appearance') that appears in TRAIN under a certain form as in ex. (1), VarIDE aims at identifying the different uses of this VMWE in the corresponding DEV and TEST corpora whatever their surface form, either identical – i.e. with the same sequence of words between the first and last lexicalized component[2] as in (4),(5) or (6) – or not – as in (2) or (3). Even though identifying the former may not seem challenging, especially for (4) that is completely identical to (1), it should be pointed out that (7), despite its apparent similarity, cannot be considered a valid variant because of the additional lexicalized determiner which characterizes a different VMWE (*to* ***have a look*** 'to examine'). Moreover, the other examples teach us that the VMWE ***have look*** tolerates the imperative in (5) or adverbial modifiers (advmod), adverbial clauses (advcl) and inflection for person in (6). With such a knowledge, we can establish the profile of the allowed morphosyntactic transformations for this VMWE, which should be useful when it appears with different surface form, as in (2). Therefore, VarIDE is based on the hypothesis that the variability phenomenon has to take into account the widest range of use of any VMWE, so that we consider all examples from (2) to (6) as *variants* (from now, this term will exclusively refer to this definition) of (1).

However, within the context of the shared task, a more restrictive definition is adopted: among all the occurrences in DEV/TEST corresponding to an annotated VMWE in TRAIN (called *Seen-in-train* VMWEs), only (2) and (3) are called *Variants-of-train*.[3] They exhibit differences within the lexicalized components (verbal inflection) and the insertions (e.g. a negation), contrary to the examples (4), (5) and (6) (called *Identical-to-train* VMWEs). In other words, what we call variants in this work corresponds to the Seen-in-train VMWEs in the shared task.

[1] http://multiword.sourceforge.net/sharedtask2018/

[2] The lexicalized components of the VMWE, i.e. those always realized by same lexemes, appear in bold.

[3] See details at http://multiword.sourceforge.net/sharedtaskresults2018.

Proceedings of the Joint Workshop on
Linguistic Annotation, Multiword Expressions and Constructions (LAW-MWE-CxG-2018), pages 283–289
Santa Fe, New Mexico, USA, August 25-26, 2018.

(1) *Rooms **have**.Ind.Pres.3rd a personalized **look** with curtains* 'Rooms have a personalized appearance' (TRAIN)

(2) *This room **has**.*Ind.Pres.3rd *not a personalized **look*** (TEST)

(3) *He **has** a **look** of innocence* (TEST)

(4) *Rooms **have**.*Ind.Pres.3rd *a personalized **look** with curtains* (TEST)

(5) *Please **have**.*Imp *a personalized **look** today* (TEST)

(6) *You always.*advmod ***have** a personalized **look** by using.*advcl *this color* (TEST)

(7) *Applicants **have a** personalized **look** at their resume* 'Applicants examine their resume in a personalized way' (TEST)

On average, Seen-in-train VMWEs represent 59.8% of all VMWEs in TEST, and, therefore, deserve dedicated analysis and processing. To this aim, our system first extracts a large set of candidates to cover a large proportion of annotated VMWEs (but also a considerable amount of noise). This extraction is based on the most frequent POS patterns of annotated VMWEs in TRAIN. Then, we extract features for VMWE classification based on the morphological and syntactic characteristics of candidates. Finally, we train a naive Bayes classifier which tries to predict, given these features, whether extracted candidates are true VMWEs or ordinary word combinations. VarIDE is a generic multilingual system for VMWE identification. It was evaluated on 19 of the 20 shared task languages. This paper describes the submitted system (Sec. 2) with the variant identification understood as above and analyzes its results (Sec. 3).

2 System description: variant identification

VarIDE is designed to identify variants of VMWEs observed in the training data. It relies on the hypothesis that the more a candidate expression c is similar to at least one annotated VMWE occurrence e, the higher the probability that c and e are variants of the same VMWE. We estimate this similarity by comparing features exhibited by c and e. These features are then used by a classifier to determine whether c is a true VMWE (that is, a variant of an observed one) or an ordinary word combination. We describe the classifier training (Sec. 2.1), and then the variant prediction and categorization on TEST (Sec. 2.2).

2.1 Training data

To train the binary classifier, we need both positive (IDIOMATIC) and negative (LITERAL) examples of VMWEs. Only the former are provided in TRAIN. Therefore, we extract VMWEs and candidates by searching for co-occurrences of the same lemmas as in annotated VMWEs, according to certain patterns. We specify the patterns to be respected since (i) this reduces noise with respect to free lemma co-occurrences, and (ii) features strongly depend on c and e's syntactic patterns. Among the steps described below for TRAIN, candidate and feature extraction will also be applied on TEST for the prediction phase.

2.1.1 Normalization and pattern generation

We aim at obtaining the most frequent patterns of annotated VMWEs in each language. Therefore, two normalization steps are required to accommodate for POS tag variability and morphological inflection.

POS sequence normalization A given VMWE annotated in TRAIN, e.g. *they **build** a **bridge*** 'they create a relationship', can be represented as a sequence of POS tags of its lexicalized components, here: ⟨VERB,NOUN⟩. The same VMWE may exhibit other POS sequences because of syntactic transformations (e.g. ⟨NOUN,VERB⟩ for *the **bridges** that were **built***). We define the *normalized POS sequence* (hereafter *POSnorm*) as the lexicographically sorted sequence of POS tags of the lexicalized components of a VMWE, e.g. the two occurrences of *to **build** a **bridge*** above have the same *POSnorm* ⟨NOUN,VERB⟩.

Lemma sequence normalization Inflection and word order should also be neutralized so as to consider e.g. both ***builds** a **bridge*** and ***bridges built*** as variants of the same VMWE. We, thus, define the *normalized lemmatized form* (hereafter *LemmNorm*) as the sequence of lexicographically ordered lemmas (e.g. ⟨bridge,build⟩). Although this form could potentially conflate distinct VMWEs sharing the same *LemmNorm*, such spurious conflation was rarely observed in practice upon inspection of a sample.

Pattern generation Candidate extraction is based on *LemmNorm*, hence we keep the correspondence between a *LemmNorm* and its observed/allowed lemma sequences. However, VMWEs may not exhibit their whole range of possible lemma sequences in TRAIN. Therefore, we apply an extrapolation procedure to avoid missing VMWEs with low frequency or unobserved variants in TRAIN, as exemplified in Table 1. The *LemmNorm* of each VMWE in TRAIN is associated with all its observed POS sequences and their frequencies, and with its *POSnorm*. When the same *LemmNorm* is associated with more than one *POSnorm* (e.g. due to annotation errors), only the most frequent one is kept like for $\langle out, turn \rangle$ in Table 1. When the table is read from left to right, this leads to a list of allowed permutations for each VMWE sharing the same *POSnorm*. For instance, VMWEs associated to the *POSnorm* ⟨NOUN,VERB⟩ (e.g. ***make decisions***) may occur in both orders VERB-NOUN and NOUN-VERB, as opposed to those associated with the *POSnorm* ⟨PRON,VERB⟩ (e.g. ***take it***), which only appears in the VERB-PRON order. As a consequence, the *LemmNorm* ⟨adjustment,make⟩ will be associated with the POS sequence NOUN-VERB, even if this order (e.g. ***adjustments*** *which were* ***made***) was never observed. This enlarges the possible word-order combinations to be searched during candidate extraction, but does not mean that the full set of POS permutations is allowed by all VMWEs sharing the same *POSnorm*.

2.1.2 Extraction of positive and negative VMWE examples

To be extracted, a candidate must respect one of the POS sequences allowed by its *POSnorm*. For instance, this condition is not fulfilled by *Tower Bridge built*[4] (PROPN instead of NOUN) or by *it takes* (PRON-VERB order instead of VERB-PRON). We select the 10 most frequent *POSnorms* for each language and their associated POS sequences. For each *LemmNorm* in TRAIN whose *POSnorm* belongs to this top-10 list, we generate all allowed permutations of lemmas and search them in the corpus allowing for discontinuities. Given that candidate extraction relies on the *LemmNorm*, we can never find candidates whose lemma sequence never occurs in TRAIN, i.e. we focus on Seen-in-train VMWEs, as explained in Sec. 1. To further limit the quantity of spurious candidates in some languages (e.g. because of sentence segmentation errors), we limit the number of words that can occur between the the first and last components of an extracted candidate to 20. This constraint is referred to as *Filter20*. Moreover, a post-processing script checks whether all annotated VMWEs within the top-10 *POSnorms* were actually extracted as candidates, and adds them automatically if missing (which could occur because of lemmatization errors). To develop the training set for the classifier, the extracted candidates in TRAIN are labeled IDIOMATIC if they were manually annotated as VMWEs, and LITERAL otherwise.

LemmNorm	Occurrence (freq.)	Observed POS sequence	*POSnorm*	Most frequent *POSnorm*	Allowed POS sequences
⟨decision,make⟩	***decisions made*** (1) ***make decisions*** (1)	NOUN-VERB VERB-NOUN	⟨NOUN,VERB⟩	⟨NOUN,VERB⟩	NOUN-VERB VERB-NOUN
⟨adjustment,make⟩	***make*** *an* ***adjustment*** (2) ***make adjustments*** (2) ***make*** *the* ***Adjustments*** (1)	VERB-NOUN VERB-PROPN	⟨NOUN,VERB⟩ ⟨PROPN,VERB⟩	⟨NOUN,VERB⟩	
⟨take,vote⟩	*the* ***vote*** *will be* ***taken*** (1)	NOUN-VERB	⟨NOUN,VERB⟩	⟨NOUN,VERB⟩	
⟨it,take⟩	*we can* ***take it*** (1) ***take it*** *from me* (1)	VERB-PRON	⟨PRON,VERB⟩	⟨PRON,VERB⟩	VERB-PRON
⟨it,make⟩	*He* ***made it*** (1)	VERB-PRON	⟨PRON,VERB⟩	⟨PRON,VERB⟩	
⟨out,turn⟩	*The pics* ***turned out*** *ok* (1) *It* ***turns out*** [...] *is fine* (1)	VERB-ADP	⟨ADP,VERB⟩	⟨ADP,VERB⟩	VERB-ADP
	It ***turns out*** *that* (1)	VERB-ADV	⟨ADV,VERB⟩		

Table 1: Example of VMWEs, their *LemmNorm*, list of POS sequences, and *POSnorm*.

2.1.3 Features

Language-adaptable features We describe each candidate VMWE using a set of feature-value pairs. For that purpose, we adapt the methodology presented for French in (Pasquer et al., 2018) to a multilingual scale. Its main principle is that a *feature* is defined as a named property (e.g. the UD verbal form VERBFORM) which is associated with a value taken from a set of possible values (e.g. *Inf*, *Ger*, *Conv*).

[4]Wavy underlining means a non-VMWE.

However, we cannot define a fixed set of features and values due to language specificities (e.g. VERB-FORM=*Conv(erb)* exists in Croatian but not in English). Such specificities occur in the POS tagsets, dependency relations, and morphological features. Therefore, we first scan the corpora to list all features and their possible values for each language. As a result, all existing features for each language are considered for all candidates, even if some of them are irrelevant, like the gender of an invariable token. When a feature is irrelevant for a candidate, its value is set to *-1*.

Features represent morphological[5] and syntactic properties, thanks to information available in the shared task corpora in the *.cupt* format. Syntactic features involve both insertions and outgoing dependency relations when available. For the elements inserted between the VMWE components, we disregard adjacency and only consider their POS (e.g. ADV-PRON in *They believed that genuine democracy* ***was*** *now.ADV* ***on*** *its.PRON* ***way***). Features can be classified into two classes: absolute and relative.

Absolute (ABS) vs. relative (REL) features For a given candidate, which can be either positive or negative VMWE, ABS features are obtained directly, based on its local properties and on the properties of its component words. For instance, in ex. (8a), the noun is singular, hence the feature ABS_morph_NOUN_Number = *singular*. On the other hand, REL features are obtained by comparing a candidate with all annotated VMWEs in TRAIN that share the same *LemmNorm* (except itself). These features aim at capturing the similarity of a candidate with annotated VMWEs. REL features can take three values: *false*, if no equivalence with any annotated VMWE was found, *true* if at least one equivalence was found[6] , or *-1* if comparison is impossible (e.g. for hapaxes). In other words, this similarity relies on the most similar annotated VMWE (i.e. the REL values are assigned after all the VMWEs in TRAIN have been scanned) even though the considered properties are only observed once. For instance, to obtain the REL feature-values for the VMWE $\langle photo, take \rangle$ in ex. (8a), we compare it with the annotated occurrences (8b) and (8c). First, as synthesized in Table 2 cell (5,4), one determiner (*a, some*) is inserted in both (8a) and (8b), so that the REL_insert_DET value is *true* whatever the insertions in (8c). Second, (8a) and (8b) differ regarding the mood/tense of the verb (imperative vs. preterite) but the imperative is also used in (8c) so that REL_morph_*VERB*_Mood is *true*. Third, the number inflection of the noun *photo* differs from (8b) or (8c), hence REL_morph_*NOUN*_Number = *false* – cf. cell (13,5).

Features can refer to the whole VMWE candidate (e.g. *LemmNorm*) or to its individual tokens. In the latter case, each token is identified by its POS, hence the three cases in Table 2: no duplicated POS so each component can be identified by its POS (Case 1, illustrated by the examples 8a-8b-8c); duplicated POS that can be distinguished by the tokens' incoming dependencies (Case 2, ex. 9a-9b); duplicated POS that cannot be distinguished by the tokens' incoming dependencies (Case 3, ex. 10a-10b).

(8) CASE 1
- a. ***Take***.VERB *a*.DET ***photo***.NOUN *of a very light plain subject* [...]
- b. *I* ***took***.VERB *some*.DET ***photos***.NOUN *of my model girlfriend* [...]
- c. *Please* ***take***.VERB *four*.NUM *new*.ADJ ***photos***.NOUN

(9) CASE 2
- a. *we'll* ***let***.VERB.root *you* ***know***.VERB.xcomp
- b. ***Let***.VERB.root *me* ***know***.VERB.xcomp[...]

(10) CASE 3
- a. ***It's raining cats.***NOUN.obj ***and dogs.***NOUN.obj
- b. ***It was*** *sometimes*.ADV ***raining cats.***NOUN.obj ***and dogs.***NOUN.obj

2.2 VMWE prediction and category assignment

Once the training is complete, in the prediction phase we extract candidates from TEST following the procedure described in Sec. 2.1.2, except that we do not know whether they are negative or positive. Absolute feature-values are obtained as described for the candidate extraction in the TRAIN corpus. As for the relative feature-values, they are obtained by comparison with all VMWEs in TRAIN with the same *LemmNorm*: for any given feature, if the same absolute value is found at least once in TEST as in TRAIN, then the Boolean relative feature is set to *true*, and *false* otherwise.

[5]Inflection and typology e.g. NUMTYPE $\in$ {*Ord(inal)*, *Card(inal)*}

[6]For instance, as shown in Table 2, similarities are found between the variants (8a) and (8b) whether the presence of an inserted determiner or the absence of an inserted verb.

	Feature description	Feature name (without the ABS/REL prefix)	ABSolute and RELative feature-values					
			CASE 1		CASE 2		CASE 3	
			ABS (8a)	REL (8a) vs. (8b/c)	ABS (9a)	REL (9a) vs. (9b)	ABS (10a)	REL (10a) vs. (10b)
not component-sensitive features	Normalized lemma sequence	*LemmNorm*	⟨photo,take⟩	n/a	⟨know,let⟩	n/a	⟨and,be,cat, dog,it,rain⟩	n/a
	VMWE category	typeMWE	*LVC.full*	n/a	*VID*	n/a	*VID*	n/a
	POS insertions	insertALL	*DET*	*true*	*PRON*	*true*		*false*
	All existing POS per language (value=*true* if present, *false* otherwise)	insert_*DET*	*true*	*true*	*false*	*true*	*false*	*true*
		insert_*VERB*	*false*	*true*	*false*	*true*	*false*	*true*
	Number of inserted elements between VMWE components in tree	distSyn_VerbNoun	*0*	*true*	*-1*	*true*	*-1*	*true*
	Syntactic distance iff two components in the VMWE	distSyn_2elts	*0*	*true*	*0*	*true*	*-1*	*true*
	Type of syntactic relation: *direct* (parent-child),*serial* (indirect ancestor) or *parallel* (shared ancestor)	typeSyn_VerbNoun	*direct*	*true*	*-1*	*true*	*-1*	*true*
	Type of syntactic distance	typedistSyn_2elts	*direct*	*true*	*direct*	*true*	*-1*	*true*
component-sensitive	Lemma of each component	lemma_*NOUN*	*photo*	n/a	*-1*	n/a	*-1*	n/a
		lemma_*VERB*_xcomp	*-1*	*-1*	*know*	*true*	*-1*	*-1*
	Morphological features per component (*-1* if irrelevant)	morph_*VERB*_VerbForm	*Fin*	*true*	*-1*	*-1*	*-1*	*-1*
		morph_*VERB*_Mood	*Imp*	*true*	*-1*	*-1*	*-1*	*-1*
		morph_*VERB*_Gender	*-1*	*true*	*-1*	*-1*	*-1*	*-1*
		morph_*NOUN*_Number	*singular*	*false*	*-1*	*-1*	*-1*	*-1*
		morph_*VERB*_xcomp_VerbForm	*-1*	*-1*	*Inf*	*true*	*-1*	*-1*
	Outgoing dependencies per component (*1* if satisfied at least once)	depSyn_*NOUN*_obj	*1*	*true*	*-1*	*-1*	*-1*	*-1*
		depSyn_*NOUN*_punct	*0*	*true*	*-1*	*-1*	*-1*	*-1*
		depSyn_*VERB*_xcomp_advcl	*-1*	*-1*	*0*	*true*	*-1*	*-1*

Table 2: Absolute and relative features for examples 8a (RELative to 8b/c), 9a (RELative to 9b), and 10a (RELative to 10b). The table should be read by compositing the ABS or REL prefix with the feature names from column 3, e.g. the cells in line 4 and columns 4 and 5 represent the feature-value pairs ABS_insertALL=*DET* and REL_insertALL=*true*.

Second, we use the NLTK's naive Bayes classifier[7] to classify candidates as negative/positive on the basis of their features. After binary classification, the VMWE category of the predicted candidate is obtained thanks to the most frequent category associated to its *LemmNorm* in TRAIN.

3 Results

Recall that VarIDE aims at identifying VMWEs occurrences which correspond to the Seen-in-train category of the shared task. Therefore, Unseen-in-train VMWEs were not expected to be identified. However, VarIDE achieves a non-zero recall for Unseen-in-train (R = 3.31), which can be due, in French, to language-specific lemma homogenization for reflexive clitics (e.g. *nous* '*us*' can be lemmatized either as *nous* '*us*' or as *se* '*oneself*').

Table 3 shows the number of true and false VMWEs, called IDIOMATIC (ID) and LITERAL (LIT), respectively, extracted from TRAIN to train the classifier, with the ratio of IDIOMATIC (% ID) examples. Recall (R) for Variant-of-train before classification (i.e. after candidate extraction) and after classification is also presented. The comparison between the global and the Variant-of-train F1-score in Table 3 shows to what extent our variant-centered identification system specifically performs on identifying Variant-of-train occurrences, which is a narrower and more challenging task than the Seen-in-train identification.

Candidate extraction We notice that we obtain satisfactory coverage of the top-10 *POSnorm*, with R $>$ 0.8 for 17 languages (0.62 and 0.75 for IT and DE). Moreover, extraction recall on Variants-of-train depends on their proportion in corpora which varies from 12% (RO) to 83% (LT). Despite few Variants-of-train in RO, global F1 is satisfactory due to well identified Identical-to-train occurrences.

Candidate classification Variant-of-train classification performance (F1 and R) is sensitive to the reliability of the annotated corpora, being affected by both false positives (e.g. *UV lights.NOUN up.VERB the temperature* was falsely annotated, probably by analogy to *to* ***light****.VERB* ***up****.ADP*) and false nega-

[7] http://www.nltk.org/

Lang.	Candidates from TRAIN for classifier training			Var-of-t % TEST	Var-of-t extraction Recall	Var-of-t classif. Recall	Global F1 MWE -based	Seen-in-train F1 MWE-based	Var-of-t F1 MWE -based	UD tags	dep syn	*Filter 20*
	# ID	# LIT	% ID									
ES	1580	3414	32	52%	0.8966	0.8345	0.253	0.2854	0.1883	x	x	x
FR	4303	5089	46	50%	0.9286	0.8968	0.5054	0.7003	0.5722	x	x	
IT	2755	5721	32	62%	0.625	0.5707	0.325	0.4024	0.3226		x	x
PT	4171	4014	51	59%	0.9442	0.7082	0.6084	0.728	0.6574	x	x	x
RO	4636	5501	46	12%	0.9692	0.8923	0.7115	0.7243	0.2613	x	x	x
DE	2437	1114	69	59%	0.7568	0.1554	0.153	0.2809	0.2614	x	x	
EN	316	336	48	53%	0.9474	0.5526	0.2417	0.5609	0.525	x	x	x
BG	5031	6637	43	36%	0.9625	0.8562	0.6252	0.7495	0.5842	x	x	x
HR	1381	843	62	73%	0.9698	0.1457	0.1257	0.2152	0.2447	x	x	
LT	301	**96**	**76**	83%	0.9946	0.0269	0.0196	0.0427	0.0515	x		x
PL	3954	2119	65	60%	0.9507	0.1256	0.1125	0.1523	0.2205	x	x	
SL	2281	13330	15	73%	0.9812	0.9624	0.4234	0.4612	0.3908		x	x
EL	1270	1341	49	68%	0.9239	0.3299	0.3477	0.523	0.4676	x	x	
EU	2499	5147	33	39%	0.9451	0.9268	0.5231	0.5527	0.3482		x	x
FA	2437	1707	59	53%	1	0.4311	0.4495	0.6274	0.5806	x	x	
HE	932	820	53	41%	0.8472	0.1528	0.1862	0.4082	0.2157	x	x	
HI	526	463	53	49%	0.95	0.6786	0.568	0.7948	0.7224	x	x	x
HU	6187	516	**92**	21%	1	0.0336	0.1869	0.2041	0.0649		x	
TR	5802	**156652**	**4**	60%	0.9733	0.9733	0.0787	0.3595	0.2598		x	x

Table 3: VarIDE results, with a focus on Variant-of-train (Var-of-t) identification.

tives. Imbalance between IDIOMATIC and LITERAL, i.e. either an over-representation of LITERAL, as in Turkish (96%) or the contrary, as in Hungarian (8%), may also have a detrimental impact. Not only percentages should be considered: in Lithuanian, only 96 candidates are classified as LITERAL. In this case, the classifier may not have enough counter-examples to learn from the features. Finally, after classification, 8 languages remain at R>0.7, but only 3 with F1>0.5: FR, PT, and BG. Features should therefore be improved to optimize both classification recall and precision.

Other problems are non-UD tagsets (which required adjustments for a few languages), sentence segmentation errors (handled with *Filter20*) and missing lemmatization (e.g. in Turkish). Other parameters may also influence the results: despite similarities between FR and ES (both are Romance languages, exhibit similar % of Variants-of-train in Table 3, and similar recall of Variants-of-train after classification) F1 is significantly lower for ES (0.57 vs. 0.19) due to lower precision. A thorough analysis might explain those results and determine whether language families share properties about the alikeness of variants of VMWEs. We believe that the similarity between variants cannot be only evaluated by the visual similarity between strings but also by taking their morphosyntactic properties into account .

4 Conclusions and future work

Our VarIDE system classifies VMWE candidates as VMWEs[8] on the basis of their morphosyntactic features by comparison with annotated VMWEs. After candidate classification, F1 for the Variants-of-train is higher than 0.5 for 6 languages (FR, PT, EN, BG, FA, HI) whereas it does not exceed 0.2 for 3 languages (ES, LT, HU). For Lithuanian or Hungarian, this low performance can be explained by the imbalance in the TRAIN data, but such explanation is not valid for Spanish. A more detailed analysis should be therefore conducted to explain the discrepancies between the observed performances for the 19 languages. For that purpose, we should look more precisely at the most informative features found in TRAIN. For instance, for Hindi, for which the system presents the best Seen-in-train and Variant-of-train performances (respectively, F1 = 0.79 and F1 = 0.72), the insertion of an auxiliary or a verb appears as a determining factor for LITERAL labels. In Basque, Farsi or Italian, the insertion of punctuation also appears among the first features that favor the LITERAL label.

Furthermore, we could also evaluate other classifiers such as a linear SVM or a multilayer perceptron. Finally, we aim at correlating the absolute and relative features used during the classification task with linguistic justifications in order to define a more task-independent variability profile of any VMWE.

[8]Input data, scripts and metrics are available at: `https://gitlab.com/cpasquer/SharedTask2018_varIDE`

Acknowledgments

This work was supported by the IC1207 PARSEME COST action[9] and by the PARSEME-FR project (ANR-14-CERA-0001).[10]

References

[Baldwin and Kim2010] Timothy Baldwin and Su Nam Kim. 2010. Multiword expressions. In Nitin Indurkhya and Fred J. Damerau, editors, *Handbook of Natural Language Processing*, pages 267–292. CRC Press, Taylor and Francis Group, Boca Raton, FL, USA, 2 edition.

[Constant et al.2017] Mathieu Constant, Gülşen Eryiğit, Johanna Monti, Lonneke van der Plas, Carlos Ramisch, Michael Rosner, and Amalia Todirascu. 2017. Multiword expression processing: A survey. *Computational Linguistics*, 43(4):837–892.

[Pasquer et al.2018] Caroline Pasquer, Agata Savary, Carlos Ramisch, and Jean-Yves Antoine. 2018. If you've seen some, you've seen them all: Identifying variants of multiword expressions. In *Proceedings of COLING 2018, the 27th International Conference on Computational Linguistics*. The COLING 2018 Organizing Committee.

[9] http://www.parseme.eu

[10] http://parsemefr.lif.univ-mrs.fr/