Delexicalized Word Embeddings for Cross-lingual Dependency Parsing

Mathieu Dehouck, Pascal Denis

To cite this version:

Mathieu Dehouck, Pascal Denis. Delexicalized Word Embeddings for Cross-lingual Dependency Parsing. EACL, Apr 2017, Valencia, Spain. pp.241 - 250, 10.18653/v1/E17-1023 . hal-01590639

HAL Id: hal-01590639

https://hal.inria.fr/hal-01590639

Submitted on 20 Sep 2017

Delexicalized Word Embeddings for Cross-lingual Dependency Parsing

Mathieu Dehouck[1] **and Pascal Denis**[2]

[1]Univ. Lille, CNRS, UMR 9189 - CRIStAL
[1,2]Magnet Team, Inria Lille - Nord Europe
59650 Villeneuve d'Ascq, France
{mathieu.dehouck, pascal.denis}@inria.fr

Abstract

This paper presents a new approach to the problem of cross-lingual dependency parsing, aiming at leveraging training data from different source languages to learn a parser in a target language. Specifically, this approach first constructs word vector representations that exploit structural (i.e., dependency-based) contexts but only considering the morpho-syntactic information associated with each word and its contexts. These delexicalized word embeddings, which can be trained on any set of languages and capture features shared across languages, are then used in combination with standard language-specific features to train a lexicalized parser in the target language. We evaluate our approach through experiments on a set of eight different languages that are part the Universal Dependencies Project. Our main results show that using such delexicalized embeddings, either trained in a monolingual or multilingual fashion, achieves significant improvements over monolingual baselines.

1 Introduction

Over the recent years, distributional and distributed representations of words have become a critical component of many NLP systems (Turian et al., 2010; Collobert et al., 2011). The reason for this success is that these low-dimensional, dense word vectors address two major problems that appear in many NLP applications, namely data sparsity and the inherent lack of expressivity of one-hot representations, and they can also be trained on large unannotated corpora which are cheap to produce and easily available. While they have proven useful in a number of tasks, and especially in dependency parsing (Koo et al., 2008), these word vectors are often learned in a generic manner, only using linear bag-of-word contexts (F. Brown et al., 1992; Mikolov et al., 2013), without paying much attention to the specifics of the task to be solved. Only very recently, people have tried to learn dependency-based embeddings (Bansal et al., 2014; Levy and Goldberg, 2014; Madhyastha et al., 2014; Bansal, 2015), and these new structure-aware representations have been shown to improve parsing performance in a monolingual setting.

We would like to generalize this idea to a multilingual setting in a way that allows the transfer of structural information associated with words from one (or several) languages to others. While previous work have attempted to learn multilingual word clusters or embeddings (Guo et al., 2015) and use these for cross-lingual transfer, this paper explores a different research direction. Specifically, we investigate the use of vectorial representations in which lemma information have been abstracted away from both words and contexts, hence reduced to their morpho-syntactic attributes. The appeal of these de facto *delexicalized* word representations is that they further increase the coverage over the available training data, potentially allowing for better generalization. Furthermore, while words tend to be hard to align in a cross-lingual setting due to homonymy and polysemy, morpho-syntactic information tends to be much more robust to language barrier (depending on typological closeness), which make them particularly relevant for cross-lingual transfer. In contrast with delexicalized parsing approaches (McDonald et al., 2011), the proposed method uses delexicalization during word embedding learning, not during parsing. Once induced over different source language datasets, these language-shared representations are used as additional features, in combination with standard language-specific features, in a standard lexicalized monolingual graph-based dependency parser for the target language. As far as we know, this is the first attempt at constructing delexicalized word embeddings for cross-lingual dependency parsing.

Through the use of these delexicalized word embeddings, we wish to explore two main hypotheses. First and foremost, we want to assess to what extent a parser for a particular language can benefit from using morpho-syntactic regularities extracted from other languages. By comparing the performance of embeddings learned on different sets of source languages in parsing a specific target language, we also hope to assess whether and which typological similarities impact the learning of "good" embeddings. The second hypothesis looks at the choice of context types (sequential vs structural) used for learning these embeddings. That is, we wish to see if the use of syntactic structure delivers better parsing improvements, again in relation to typological similarities or differences. These hypotheses will be tested on treebanks from the recent Universal Dependencies Project (Nivre et al., 2016), which provides us with homogeneous syntactic dependency annotations in many languages.

In section 2, we provide some background and review previous work on graph-based dependency parsing for mono- and cross-lingual settings and on word embeddings. Section 3 details our approach for learning delexicalized word embeddings and using them in dependency parsing. In section 4, we present some experimental results that show the importance of grammatical embeddings for the task at hand. And finally in section 5, we draw some conclusions and present future perspectives.

2 Preliminaries and Related Work

The approach proposed in this paper draws on three different lines of work in dependency parsing.

2.1 Graph-based Dependency Parsing

A dependency tree is a graphical representation of the syntactic structure of a sentence. The task of dependency parsing is to predict the dependency tree of a given sentence x, $\mathcal{T}_x$ being the set of all its possible trees. Assuming we have access to a scoring function $Score(\bullet, \bullet)$ that tells how well a dependency tree fits the syntactic structure of a sentence, the goal of dependency parsing is to find the tree $\hat{y}$ such that:

$$\hat{y} = \operatorname*{argmax}_{t \in \mathcal{T}_x} Score(x, t).$$

The size of $\mathcal{T}_x$ grows exponentially with the length of x, $|\mathcal{T}_x| = |x|^{|x|-2}$, making an exhaustive search for the best tree impractical in most cases. Thus in practice, some simplifying assumptions are made. Here we use the graph-based, edge-factored approach based on the assumption that the score of a tree can be computed as the sum of its edges scores (McDonald et al., 2005a). Let $score(\bullet, \bullet)$ be a scoring function for edges.

$$Score(x, t) = \sum_{e \in t} score(x, e).$$

In this case, finding the best parse tree for x boils down to finding the maximum spanning tree in the complete graph whose vertices are the words of x. The score of an edge e is here defined as the dot product between a model $\boldsymbol{w}$ (a weight vector) and the feature vector $\boldsymbol{\phi}(x, e)$ of this edge.

$$score(x, e) = \boldsymbol{w} \cdot \boldsymbol{\phi}(x, e).$$

In this paper, we learn the model $\boldsymbol{w}$ in an online manner with the Passive-Aggressive (PA) algorithm described in (Crammer et al., 2006). Specifically, we use the PA-II that uses a squared hinge loss in its predicted-loss cost-sensitive version, in which the cost is computed in terms of the symmetric difference on the edges with respect to the target tree.

2.2 Word Vectors for Dependency Parsing

Distributional and distributed word representations are dense vectorial representations of words that live in a multi-dimensional integer or real space whose size is much smaller than the size of the original language vocabulary. There are now a large variety of spectral and neural approaches for learning these representations, including several variants of Principal Component Analysis (Jolliffe, 2002) and several deep neural net approaches. Most of these approaches have in common that they solely exploit linear, bag-of-word co-occurrence between words to derive these low-dimensional representations (Mikolov et al., 2013; Lebret and Collobert, 2014).

Starting with the work of Koo (2008), the inclusion of this type of low-dimensional word representations as features has been shown to be a simple yet very effective way of improving dependency parsing performance. The main appeal of these low-dimensional dense representations is that they mitigate major shortcomings of standard one-hot encoding representations which are very sparse and live in very high dimensional spaces, thus lacking in expressivity and hindering generalization. While it is still unclear whether pretrained embeddings (Andreas and Klein, 2014) indeed capture interesting syntactic information, more recent work have concentrated on learning dependency-based word embeddings (Bansal et al., 2014; Levy and Goldberg, 2014; Madhyastha et al., 2014; Bansal, 2015). In these approaches, word co-occurrences are defined in terms of dependency contexts (x is the governor of word w), instead of linear contexts (x appears within a range of s around word w). Embedding techniques have also started to be applied to objects other than words, namely on dependency relations (Bansal, 2015; Kiperwasser and Goldberg, 2015).

In this paper, we depart from these approaches by learning a low-dimensional word vector representation that is based only on the morpho-syntactic information associated with that word, and learning is performed with simple PCA. Furthermore, we do not use any auto-parsed data in order to avoid errors from spreading into the embedding. Another point is that even if we use a first order parsing model, we use higher-order contexts for learning the embeddings. Bansal (2015) also uses higher-order contexts but combines them with a second-order dependency model.

2.3 Cross-lingual Dependency Parsing

Cross-lingual dependency parsing encompasses several problems ranging from learning a parser for a target language relying solely on annotated data from a source language (Lynn et al., 2014) to learning a unique parser that can handle various languages (Ammar et al., 2016). Delexicalized parsers (McDonald et al., 2011) have been used to avoid the problems the arise from lexical translation. More recently, cross-lingual parsers have been trained using cross-lingual word clusters as well as multilingual word embeddings (Guo et al., 2015) to alleviate the lack of lexical information.

Our work differs from previous studies in that it assumes the availability of annotated data from the target language for training the parser, but uses multilingual embeddings to benefit from annotated data in other languages. Another important difference is that while multilingual word embeddings are usually used to replace lexical items, we use morpho-syntactic embeddings that are less language dependent.

3 Dependency Parsing with Delexicalized Word Embeddings

Standard word embeddings have two major drawbacks for our purposes: they represent word forms which are not easy to transfer from one language to another, and they rely on sequential contexts which are not grammatically motivated for languages with free word order.

To circumvent these problems, we propose to create embeddings for morpho-syntactic attribute sets using structural information from dependency trees. As we abstract away the lexical form of words we call our embeddings *delexicalized word embeddings*. The advantage of embedding sets of morpho-syntactic attributes over word forms is that morpho-syntactic attributes are shared across languages much more frequently than lexical features, and they also tend to be more stable through translation. This allows a more reliable transfer of linguistic knowledge from one language to another. This also increases the vocabulary coverage as the number of morpho-syntactic attributes is far smaller than the number of word forms. Here, we choose to learn representations for *full* attribute sets (i.e., the set containing all the morpho-syntactic attributes associated with a word form) instead of learning representations for single attributes and then composing those for each word. This is in line with standard work embedding approaches which implicitly do the same in learning a different representation for each distinct word form of a lemma (e.g., *be*, *am*, *is*, *were*) without any further analysis. We discuss this issue in more detail in the experiment section.

3.1 Delexicalized Words

Let us briefly illustrate the kinds of morpho-syntactic attributes we want to embed with some examples. For these examples, we are using the notation of the Universal Dependencies project.

The English word *a* is a *determiner* (its part-of-speech or POS is DET), its number is *singular* and its definiteness is *indefinite*. As a determiner, it does not have tense or mood, and as most English words, it does not have gender. We would thus embed the feature set *DET:Definite=Ind|Number=Sing*.

Looking at an example from a morphologically richer language, we have Finnish verb form *oli* (meaning *was*) which we encode as *VERB:Mood=Ind|Number=Sing|Person=3|Tense-=Past|VerbForm=Fin|Voice=Act*. This means that *oli* is a finite verb form (not a participle) in the indicative past, its voice is active and its agreement is third person, singular.

This gives vocabulary sizes ranging from a hundred or less for analytic languages (English has about 120 such combinations) to several thousands for synthetic ones (the Finnish part of the UD Project has about 4000 such combinations).

3.2 Structural Contexts

Having decided to embed morpho-syntactic attribute sets rather than words, we need to map these new objects to dense vectors in $\mathbb{R}^d$. To this end, we apply principal component analysis (PCA) to a co-occurrence matrix (possibly reweighted) whose lines represent our new attribute sets (i.e., our delexicalized words) and whose columns are their contexts, which are also expressed in terms of morpho-syntactic attributes. We describe those contexts later on, but it is worth noting that as we allow them to diverge from the morpho-syntactic features sets, not including every features for example, the number of context can potentially reach a few tens of thousands.

As we want to learn embeddings specifically tailored to dependency parsing, we use not only sequential (i.e. linear) contexts but also structural contexts based on dependency trees. Sequential contexts are of the kind: "word *x* appears in a window of size l centered on word *y*". Structural contexts are instead defined on the dependency tree: "word *x* is the governor of word *y*", or "*x* is a dependent of *y*"or again "*x* is a sibling of *y*".

The new structural contexts can be seen as following a certain path in the dependency tree linking two words. Let $up(x,t)$ be the function that maps a word x to its governor (following the upgoing edge) in tree t. As the root of a tree has no governor we add a dummy token called nil for that purpose. Then $down(x,t)$ is the function that maps x to the set of its dependents in t.

$$down(x,t) = \{y \in t \mid up(y,t) = x\}$$

Then we can define our new contexts as combinations of up and $down$, for example the governor of x is $up(x,t)$, its dependents are $down(x,t)$ and its siblings are $down(up(x,t),t) \setminus \{x\}$.

We can also define similar functions over sequences. Let $right(x,s)$ (respectively $left(x,s)$) be the word in sequence s standing just at the right (respectively at the left) of x, then we can also express the sequential contexts in the same framework. We also add two new dummy tokens $begin$ and end to avoid ill definition at the borders of s. Using the notation $f^n(\bullet)$ for $f \circ f \circ \cdots f(\bullet)$ where f is applied n times, we have that the window of size l centered on x is $\{right^i(x,s) \mid i \in [i..l]\} \cup \{left^i(x,s) \mid i \in [i..l]\}$. We can also define new contexts such as left or right siblings.

Let us now turn to an example to make our approach more concrete. Figure 1 shows a dependency tree for a Gothic sentence[1] from the Universal Dependencies Project. Each word is accompanied with its part-of-speech and the corresponding morpho-syntactic attributes. Colored links represent examples of contexts, orange links standing for contexts of length 1 and blue links standing for contexts of length 2.

Embedding Delexicalized Words With Structural Contexts

Given the above definition of structural contexts, there are still several design parameters to set in order to construct embeddings. First, we distinguish different *types of contexts*, depending on whether the contexts are sequential (with a distinction between left and right), governor, dependents and siblings (with a distinction between first left sibling, first right sibling and others). Second, there are different *context spans*, where the span is the maximum length (in term of function applications) of a context. It is equivalent to the window size for sequential context. For example, the sibling context has a fixed span of 2, but the governor of span 2 means the governor of the governor (if it exists, *nil* otherwise). Third, we distinguish different *granularity levels* of contexts, corresponding to the maximum number of morpho-syntactic attributes used to model contexts. Like words, contexts are also defined in terms of morpho-syntactic attributes, but we do not require complete sets for them, hence allowing for different granularity

[1]The reader may notice that there is no punctuation in that sentence, it is because there is originally no punctuation in Ulfila's Bible from which the sentence comes, but if there were some (as in modern texts) we would use them just as usual, treating any token as a word.

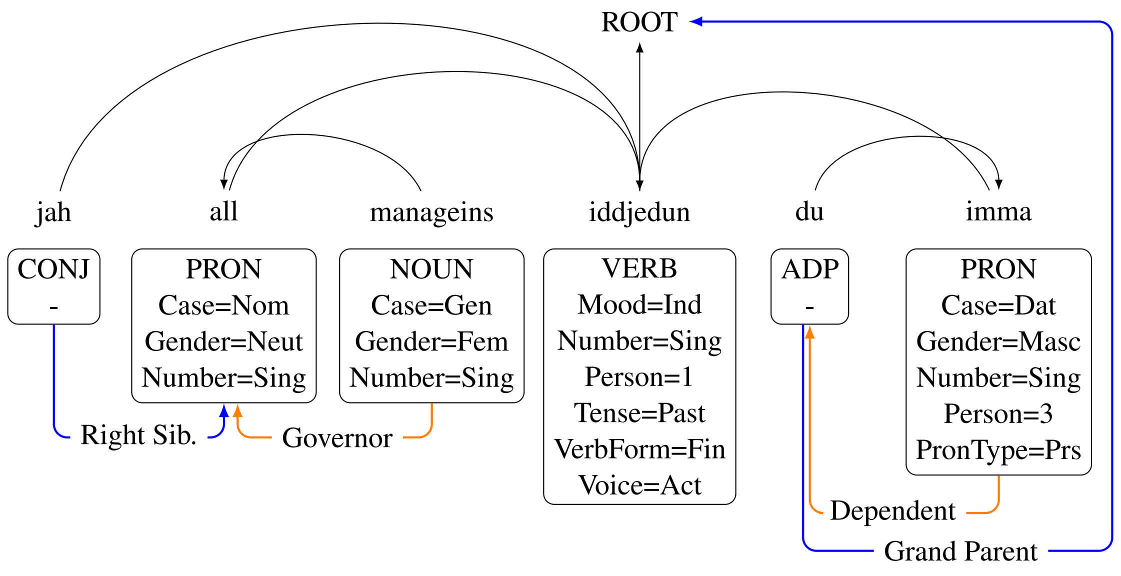

Figure 1: Dependency tree of sentence *jah all manageins iddjedun du imma* (*and all of the crowd went to him*) from Ulfila's Bible, from the Gothic part of the Universal Dependencies Project. Under the words are their morpho-syntactic analysis and the colored links represent some possible structural contexts.

levels. Taking an example from the sentence of Figure 1, with a granularity of 0, the word *manageins* triggers the context *NOUN* only (i.e., the POS tag alone). With a granularity of 1, it triggers *NOUN:Case=Gen*, *NOUN:Gender=Fem* and *NOUN:Number=Sing* (i.e., the word POS tag is crossed with each of the other attributes). And with *full* granularity, the union over all subsets of attributes (of size 0 to 3) is combined with the POS tag. This gives 2^a possible contexts for a morpho-syntactic attributes set with a attributes.

Finally, other parameters are the embedding algorithm (here we use PCA), the matrix normalization method (he we use a simple L2-norm row normalization), the size of the embedding space and the number of contexts used. We can keep all the contexts as their numbers range from 36 (each part-of-speech counted twice for before and after a word and two extra context representing the beginning and the end of the sentence) to a few tens of thousands.

3.3 From Word Embeddings to Edge Embeddings to Parsing Features

As the dependency model factors on edges, we need to turn the word embeddings into edge embeddings. We want an aggregating function that preserves edge orientation and the dependency between the edge endpoints. So we chose to use the outer product of the two original embeddings because it is not commutative and each output dimension depends on the two inputs. Let $\otimes$ denote the outer product and let $\boldsymbol{u}$ and $\boldsymbol{v}$ be two vectors of $\mathbb{R}^d$, then:

$$\boldsymbol{u} \otimes \boldsymbol{v} = \boldsymbol{u}\boldsymbol{v}^\top$$

This operation yields a matrix in $\mathbb{R}^{d\times d}$ but we need a vector, so we take the vector $vec(\boldsymbol{u}\boldsymbol{v}^\top)$. In the following, whenever we use a matrix where a vector is expected, we implicitly assume the presence of a $vec(\bullet)$[2]. There is a slight scalability problem as the output size grows quadratically with the size of the inputs. But in the case of dependency parsing, where feature vectors are commonly a few millions dimensions long, for typical embeddings size (between 100 and 500) an increase of a few tens of thousands dimensions is acceptable.

We would like to use more context than just the two nodes of the edge to represent it. In order to define higher-order contexts, we use triplets of delexicalized words centered on each side of the edge. Specifically, we concatenate the representations associated with the triplets to keep a tractable model of size $9d^2$. Note that applying an outer product across each word of the triplet would be

[2] In this case, it does not matter if a matrix is vectorized by the columns or by the rows, as soon as the same convention is used consistently throughout the algorithm.

prohibitive, leading to vectors of the order of 10^{12} dimensions which would not even fit in memory.

More formally, let $\oplus$ denote the concatenation operator, $Dep(e)$ (respectively $Gov(e)$) be the dependent (respectively the governor) of an edge e and let $\boldsymbol{emb}(\bullet)$ be the embedding function. $\boldsymbol{\phi}_{emb}(\bullet,\bullet)$ being the embedding part of the feature vector. Using the notation of section 2.1, we have:

$$\boldsymbol{\phi}_{emb}(x,e) = \bigoplus_{i\in\{-1,0,+1\}} \boldsymbol{emb}(Gov(e)_i) \otimes \bigoplus_{i\in\{-1,0,+1\}} \boldsymbol{emb}(Dep(e)_i).$$

$$score(x,e) = \boldsymbol{w}\cdot(\boldsymbol{\phi}(x,e) \oplus \alpha\boldsymbol{\phi}_{emb}(x,e)).$$

Where α is a scalar allowing to tune the relative importance of each part of the feature vector, and $\phi(\bullet,\bullet)$ is the traditional dependency feature vector. This follows the same approach as Kiperwasser and Goldberg (2015) and Chen et al.(2014). As mentioned previously, we have special tokens such as *begin* and *end* to represent the word before the beginning and the end of a sentence. We also have a *root* token that stands for the extra root node added by the graphical dependency model. And we also have a back-off embedding of raw part-of-speech to handle unseen delexicalized words in the test set.

4 Experiments

4.1 Data set

We have tested our parsing models based on delexicalized word embeddings on eight languages using the data of the Universal Dependencies (UD) v1.3 (Nivre et al., 2016). We have chosen to work on English (en), Basque (eu), Finnish (fi), French (fr), Gothic (got), Hebrew (he), Hungarian (hu) and Romanian (ro). These languages belong to four different families, which are Indo-European (en, fr, got, ro), Finno-Ugric (fi, hu), Semitic (he), and Basque which forms a separate group. They also display various levels of morphological complexity not correlated with the families (en, fr and he do not have case marking while the other five do to various degrees) as well as different grammatical typologies (eu is an ergative language, while the other seven are accusative ones). When several corpora are available for a language, we picked the standard one. Table 1 provides some basic statistics on the language datasets. Also note that our experiments follow the train/dev/test split as provided by the UD Project.

4.2 Features

Dependency Features

For parsing, we use standard graphical dependency parsing features that include word forms and POS-tags of edge nodes and surrounding words, edge length and direction and conjunction of those basic features. The main difference with the original MSTparser features of McDonald et al.(2005b) is that instead of using truncated words of length 5 as back-off features, we use the lemmas that are provided in the UD Project.

Embedding Contexts

For the embedding contexts, we consider four parameters, namely the type and span of contexts, the granularity of the morpho-syntactic attributes used in those contexts and the dimension of the embedding space. Regarding the type of contexts we have experiments with three settings: (i) strictly sequential contexts (*Seq*), (ii) strictly structural contexts that use governor, dependents and siblings information (*Struct*) and (iii) mixed contexts using both dependency-based and sequential contexts (*Mix*). Regarding the span, we have tried 1 and 2. Siblings are only used in structural and mixed contexts of span 2 because that is the length of the path between a vertex and its siblings in a tree. We have tried granularity of 0 and full granularity. For the embedding space dimension we have tried 50, 150 and 500 dimensions, or the maximum possible size[3] if smaller than 500. For better readability, we will use shortcuts to refer to the different parameter settings: 1 = (span 1,granularity 0), 2 = (span 2, granularity 0) and 3 = (span 2, full granularity for contexts span 1, granularity 0 for context span 2).

4.3 Experimental Settings

We have carried out two sets of experiments: monolingual and cross-lingual. In the first set, embeddings are learned on a single language training set and then used to parse that same language. In the second set, we have defined several clusters of languages based on their phylogenetic relationship and typological similarities. For a given cluster, embeddings are learned on the training sets of

[3]Spectral-based dimension reduction such as PCA are limited by the number of eigenvectors of the matrix to be reduced. For example a matrix of size 200×500 can at most be reduced into a 200×200 matrix via PCA. When the number of eigenvectors is smaller than 500, we use that value instead.

	Train		Test			
	sentences	words	sentences	words	morpho-syntactic tokens	POS
English	12 543	204 586	2 077	25 096	118	17
Basque	5 396	72 974	1 799	24 374	845	16
Finnish	12 217	162 721	648	9 140	1 592	15
French	14 557	356 216	298	7 018	195	17
Gothic	4 360	44 722	485	5 158	662	13
Hebrew	5 142	15 558	491	12 125	480	16
Hungarian	1 433	23 020	188	4 235	651	16
Romanian	4 759	108 618	794	18 375	412	17

Table 1: Number of sentences and words in the training and test sets, number of delexicalized word and of POS-tags for each language. The total number or embedded tokens is $|\text{morpho-syntactic feature set}| + |POS| + 3$ because of the POS back-offs and the special *begin*, *end* and *root* tokens.

each language in that cluster, and in turn used to parse each language in that cluster. It is possible not to use any data from the target language when learning the embeddings, but in this study we stick to using target language data.

Besides embeddings, there are three additional hyper-parameters that need to be tuned: the C aggressiveness parameter of the PA-II algorithm, the scaling factor α that controls the relative weight of the embedding features in the edge scores, and the number i of training iterations of the PA-II algorithm. We have tuned these hyper-parameters through a grid search on the development sets and picked the values that were behaving best on average, giving $C = 0.001$, $\alpha = 0.001$, $i = 5$.

All the scores reported below are Unlabeled Attachment Scores (UAS) measured on the test sets ignoring the punctuation marks. As a baseline comparison we use our implementation of the MSTparser without morpho-syntactic attributes representation of any kind. We computed the significance of the scores using the McNemar's test.

4.4 Monolingual Experiments

Table 2 displays UAS scores for the monolingual setting. Except for French and Romanian that do not show real improvement, the six other languages show substantial performance increases with the embeddings. These improvements are statistically significant for all languages, except for Basque and Hebrew. One of our hypotheses was that structure is important when learning an embedding for dependency parsing and indeed our results support it. The largest improvements appear with structural or mixed embeddings which rely on syntactic structures.

The results for English are significant and close to each other for all types of embeddings, this tends to show that in English, sentence structure and word order are very correlated and both contribute information. Indeed that is what one expects for English which has a rigid syntax and a poor morphology.

On the other side of the picture, Basque and Gothic display the largest improvements with structural morpho-syntactic embeddings. This is also expected as those are both morphologically rich languages with more flexible word order. Even though the argument is less clear for Hungarian and Finnish, they both show that structure is important for learning informative dependency embeddings.

4.5 Cross-lingual Experiments

Table 3 summarizes the UAS scores achieved using delexicalized embeddings learned on several languages. Parsing accuracy improve for four languages (en, eu, hu, ro) in the cross-lingual setting compared to the best monolingual setting. While the multilingual embeddings do not outperform the monolingual ones for the other four languages, they still deliver parsing performance that are better than with the baseline MST parser for all languages (but Gothic). That shows that indeed using data from other languages is beneficial for learning good embeddings for dependency parsing, which was the second hypothesis we wanted to evaluate. We also notice that the largest gains are achieved with structural (or mixed) embeddings, giving more evidence to the importance of structure for learning embeddings for dependency parsing.

Language	Baseline	Seq 1	Seq 2	Seq 3	Struct 1	Struct 2	Struct 3	Mix 1	Mix 2	Mix 3
English	85.62	$86.07^{\star}_{40}$	85.92^{*}_{80}	$86.06^{\diamond}_{121}$	$86.01^{\diamond}_{37}$	85.95^{*}_{121}	85.94^{*}_{50}	$86.10^{\star}_{77}$	$\mathbf{86.19}^{\star}_{50}$	85.84_{121}
Basque	76.65	76.60_{38}	76.70_{76}	76.73_{500}	76.67_{35}	76.85_{119}	$\mathbf{76.90}_{150}$	76.66_{50}	76.72_{150}	76.80_{500}
Finnish	79.97	80.44_{36}	80.44_{72}	$80.71^{\diamond}_{150}$	80.58^{*}_{33}	80.35_{114}	80.58^{*}_{50}	$\mathbf{80.92}^{\star}_{69}$	80.40_{50}	80.60^{*}_{50}
French	**83.99**	83.48_{40}	83.55_{50}	83.47_{150}	83.42_{37}	83.68_{128}	83.71_{150}	83.61_{77}	83.89_{150}	83.84_{198}
Gothic	79.16	78.95_{32}	79.10_{50}	79.57_{500}	79.24_{28}	79.31_{50}	$\mathbf{80.09}^{\diamond}_{500}$	79.59_{50}	79.62_{150}	79.62_{500}
Hebrew	84.05	83.87_{38}	83.86_{50}	84.24_{50}	84.18_{35}	84.32_{50}	84.14_{150}	84.32_{50}	84.34_{50}	$\mathbf{84.36}_{150}$
Hungarian	79.15	79.23_{38}	$\mathbf{80.13}^{*}_{76}$	79.83^{*}_{500}	79.67_{34}	$\mathbf{80.13}^{\diamond}_{118}$	79.69_{500}	79.94^{*}_{50}	79.64_{50}	79.80_{50}
Romanian	81.35	81.34_{40}	81.29_{80}	81.21_{415}	81.00_{37}	$\mathbf{81.38}_{50}$	81.29_{150}	81.30_{50}	81.26_{150}	81.21_{415}

Table 2: Best UAS scores for each embedding type in monolingual setting. The best score for each language are in bold and in gray are the results above the baseline. The statistical significance (using McNemar's test) of an improvement over the baseline is indicated with a superscript mark: * stands for a significance with a p-value inferior than 0.05, $^{\diamond}$ stands for $p \leq 0.01$ and * for $p \leq 0.001$. The length of the embeddings is reported as a subscript.

Language	Baseline	Best Mono	Best All	Best Multilingual
English	85.62	$86.19^{\star}$	$86.18^{\star}_{seq3,50}$	$\mathbf{86.32}^{\star}_{en,got,mix3,50}$
Basque	76.65	76.90	$\mathbf{76.97}^{*}_{struct3,50}$	$76.68_{decl,seq2,50}$
Finnish	79.97	$\mathbf{80.92}^{\star}$	$80.89^{\star}_{struct2,50}$	$80.81^{\diamond}_{decl,seq2,50}$
French	**83.99**	83.89	$83.87_{struct1,37}$	$83.89_{en,fr,ro,mix1,77}$
Gothic	79.16	$\mathbf{80.09}^{\diamond}$	$79.80_{struct2,50}$	$79.99^{*}_{got,ro,mix3,500}$
Hebrew	84.05	**84.36**	$84.32_{seq3,150}$	$84.13_{en,fr,he,mix1,77}$
Hungarian	79.15	80.13^{*}	$80.05^{*}_{mix2,150}$	$\mathbf{80.30}^{\diamond}_{decl,struct1,37}$
Romanian	81.35	81.38	$81.31_{seq3,150}$	$\mathbf{81.52}_{hu,ro,mix3,50}$

Table 3: Best UAS scores in cross-lingual setting. Under *Best All* are the results using the embeddings learned on the set of all languages, while under *Best Multilingual* are given the best results for each language using only a subset of the languages for learning the embedding. The subscript represents the context types and the number of dimensions of the embedding. The baselines and best monolingual scores are also reported. Significance of scores uses the same conventions as in Table 2.

Let us now look more closely at which groups of source languages are most helpful for specific target languages. First, note in general the best performing embeddings are never those obtained by using the full set of languages (this is only the case for Basque). This is expected since we have picked languages with very different grammars thus the full embeddings can be very noisy with regard to a single language. In fact, the Basque results are rather surprising since this language differs the most from the others in terms of morphology, but also one for which we had rather small training data.

The best parsing performance for English are achieved when using additional data from Gothic. As both are Germanic languages, this tends to show that data from genetically related languages can help in learning a better representation. Even though they do not achieve the best results, similar patterns occur for French (French and Romanian are Romance languages and English has been heavily influenced by Romance languages) and for Gothic (Gothic and Romanian are both Indo-European languages). Similarly, Hungarian and Romanian reach their best scores when parsed with typologically close languages that have case marking. And again, Basque, Finnish and Gothic display similar patterns. Hebrew performs reasonably well with French and English which are two languages with fairly restricted word orders.

As to why some languages have better monolingual parsing results than multilingual results, we think this is at least partly due to the lack of flexibility of our model. That is, morpho-syntactic attributes sets are treated independently from one another making some of them hard to use in the cross-lingual setting. For example, Hebrew verbs display 'binyanim' (internal flection classes) that do not appear in any other language, similarly Finnish has a lot of cases that are not found in other languages. Those are indeed two languages that do not perform well with other languages. We thus believe that introducing compositionality in our embedding model should help in solving those problems and enhance the results further.

5 Conclusion

In this paper, we have described a new way to induce multilingual embeddings, namely delexicalized word embeddings, that solely rely on the morpho-syntactic attributes of words which can easily be transferred across languages. This new approach to multilingual embeddings allows one to use annotated data from other languages to further improve the resulting embeddings and parsers, avoiding the problems that arise from lexicon alignment or cross-lingual word clustering.

In line with previous recent work, we have shown that the syntactic structure is crucial when it comes to learning embeddings for dependency parsing. In addition, we have seen that the impact of the structure on the quality of an embedding depends on language typology.

In future work, we should see how those morpho-syntactic embeddings can help in labeled dependency parsing, as edge types and word morpho-syntactic attributes are related. We would like to investigate the impact of the embedding algorithms (here we use PCA) on the final embeddings. We would also like to try other ways to turn word embeddings into edge embeddings in order to benefit more from the local neighborhoods. Finally, we would like to work on the embedding of clusters of morpho-syntactic attributes to induce higher-order embeddings for noun-phrases or verb-phrases and to deal with agreement and morpho-syntactic attributes hierarchy.

6 Acknowledgment

We thank the three anonymous reviewers for their comments. This work was supported by a grant from CPER Nord-Pas de Calais/FEDER DATA Advanced data science and technologies 2015-2020.

References

Waleed Ammar, George Mulcaire, Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2016. One parser, many languages. *CoRR*, abs/1602.01595.

Jacob Andreas and Dan Klein. 2014. How much do word embeddings encode about syntax? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 822–827. Association for Computational Linguistics.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 809–815. Association for Computational Linguistics.

Mohit Bansal. 2015. Dependency link embeddings: Continuous representations of syntactic substructures. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 102–108, Denver, Colorado, June. Association for Computational Linguistics.

Wenliang Chen, Yue Zhang, and Min Zhang. 2014. Feature embedding for dependency parsing. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 816–826. Dublin City University and Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *J. Mach. Learn. Res.*, 7:551–585, December.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, T. J. Watson, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics, Volume 18, Number 4, December 1992.*

Jiang Guo, Wanxiang Che, David Yarowsky, Haifeng Wang, and Ting Liu. 2015. Cross-lingual dependency parsing based on distributed representations. 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 1234–1244. Association for Computational Linguistics.

Ian T. Jolliffe. 2002. *Principal component analysis*. Springer series in statistics. Springer, New York, Berlin, Heidelberg. Autre(s) tirage(s) : 2004.

Eliyahu Kiperwasser and Yoav Goldberg. 2015. Semi-supervised dependency parsing using bilexical contextual features from auto-parsed data. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1348–1353. Association for Computational Linguistics.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603. Association for Computational Linguistics.

Rémi Lebret and Ronan Collobert. 2014. Word embeddings through hellinger pca. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 482–490. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 302–308. Association for Computational Linguistics.

Teresa Lynn, Jennifer Foster, Mark Dras, and Lamia Tounsi. 2014. Cross-lingual transfer parsing for low-resourced languages: An irish case study. In *Proceedings of the First Celtic Language Technology Workshop*, pages 41–49, Dublin, Ireland, August. Association for Computational Linguistics and Dublin City University.

Swaroop Pranava Madhyastha, Xavier Carreras, and Ariadna Quattoni. 2014. Learning task-specific bilexical embeddings. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 161–171. Dublin City University and Association for Computational Linguistics.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005a. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 91–98. Association for Computational Linguistics.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005b. Spanning Tree Methods for Discriminative Training of Dependency Parsers.

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. Multi-source transfer of delexicalized dependency parsers. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 62–72. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may. European Language Resources Association (ELRA).

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.