ELMoLex: Connecting ELMo and Lexicon features for Dependency Parsing

Ganesh Jawahar, Benjamin Muller, Amal Fethi, Louis Martin, Éric Villemonte de La Clergerie, Benoît Sagot, Djamé Seddah

ELMoLex: Connecting ELMo and Lexicon features for Dependency Parsing

Ganesh Jawahar **Benjamin Muller** **Amal Fethi** **Louis Martin**
Éric de La Clergerie **Benoît Sagot** **Djamé Seddah**
Inria
{firstname.lastname}@inria.fr

Abstract

In this paper, we present the details of the neural dependency parser and the neural tagger submitted by our team 'ParisNLP' to the CoNLL 2018 Shared Task on parsing from raw text to Universal Dependencies. We augment the deep Biaffine (BiAF) parser (Dozat and Manning, 2016) with novel features to perform competitively: we utilize an indomain version of ELMo features (Peters et al., 2018) which provide context-dependent word representations; we utilize disambiguated, embedded, morphosyntactic features from lexicons (Sagot, 2018), which complements the existing feature set. Henceforth, we call our system 'ELMoLex'. In addition to incorporating character embeddings, ELMoLex leverage pre-trained word vectors, ELMo and morphosyntactic features (whenever available) to correctly handle rare or unknown words which are prevalent in languages with complex morphology. ELMoLex[1] ranked 11^{th} by Labeled Attachment Score metric (70.64%), Morphology-aware LAS metric (55.74%) and ranked 9^{th} by Bilexical dependency metric (60.70%). In an extrinsic evaluation setup, ELMoLex ranked 7^{th} for Event Extraction, Negation Resolution tasks and 11^{th} for Opinion Analysis task by F1 score.

1 Introduction

The goal of this paper is to describe ELMoLex, the parsing system submitted by our team 'ParisNLP' to the CoNLL 2018 Shared Task on parsing from raw text to Universal Dependencies (Zeman et al., 2018). The backbone of ELMoLex is the BiAF parser (Dozat and Manning, 2016) consisting of a large, well-tuned network that generates word representations, which are then fed to an effective, biaffine classifier to predict the head of each modifier token and the class of the edge connecting these tokens. In their follow-up work (Dozat et al., 2017), the authors further enrich the parser by utilizing character embeddings for generating word representations which could help in generalizing to rare and unknown words (also called Out Of Vocabulary (OOV) words). They also train their own taggers using a similar architecture and use the resulting Part of Speech (PoS) tags for training the parser in an effort to leverage the potential benefits in PoS quality over off-the-shelf taggers.

We identify two potential shortcomings of the BiAF parser. The first problem is the *context independence* of the word embedding layer of the parser: the meaning of a word varies across linguistic contexts, which could be hard to infer automatically for smaller treebanks (especially) due to lack of data. To handle this bottleneck, we propose to use Embeddings from Language Model (ELMo) features (Peters et al., 2018) which are context dependent (function of the entire input sentence) and obtained from the linear combination of several layers of a pre-trained BiLSTM-LM[2]. The second problem is the *linguistic naivety*[3] of the character embeddings: they can generalize over relevant sub-parts of each word such as prefixes

[1] Code to reproduce our tagging and parsing experiments is publicly accessible at https://github.com/BenjaminMullerGit/NeuroTaggerLex and https://github.com/ganeshjawahar/ELMoLex respectively.

[2] Due to lack of time, we could train BiLSTM-LMs on the treebank data only (indomain version). We leave it for future work to train the model on large raw corpora from each language, which we believe could further strengthen our parser.

[3] The term *linguistic naivety* was introduced by Matthews et al. (2018) to refer to the fact that character-based embeddings for a sentence must discover that words exist and are delimited by spaces (basic linguistic facts that are built in to the structure of word-based models). In our context, we use a different meaning of this term as the term corresponds to the word-level character-based embeddings.

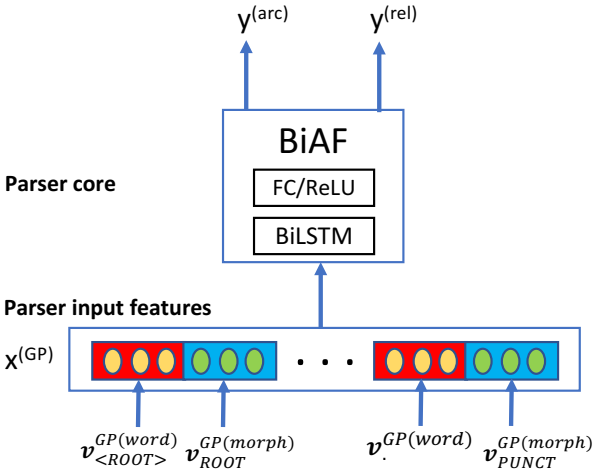

Figure 1: Architecture of ELMoLex which uses BiAF parser as its backbone. Arrows indicate structural dependence, but not necessarily trainable parameters.

or suffixes, which can be problematic for unknown words which do not always follow such generalizations (Sagot and Martínez Alonso, 2017). We attempt to lift this burden by resorting to external lexicons[4], which provides information for both word with an irregular morphology and word not present in the training data, without any quantitative distinction between relevant and less relevant information. To tap the information from the morphological features (such as gender, tense, mood, etc.) for each word present in the lexicon efficiently, we propose to embed the features and disambiguate them contextually with the help of attention (Bahdanau et al., 2014), before combining them for the focal word.

We showcase the potential of ELMoLex in parsing 82 treebanks provided by the shared task. ELMoLex ranked 11^{th} by Labeled Attachment Score (LAS) metric (70.64%), Morphology-aware LAS (MLAS) metric (55.74%) and ranked 9^{th} by BiLEXical dependency (BLEX) metric (60.70%). We perform ablation and training time studies to have a deeper understanding of ELMoLex. In an extrinsic evaluation setup (Fares et al., 2018), ELMoLex ranked 7^{th} for Event Extraction, Negation Resolution tasks and 11^{th} for Opinion Analysis task by F1 score. On an average, ELMoLex ranked 8^{th} with a F1 score of 55.48%.

2 ELMoLex

The model architecture of ELMoLex, which uses BiAF parser (Dozat and Manning, 2016) (which in turn is based on Kiperwasser and Goldberg (2016)) as its backbone, is displayed in Figure 1. For our shared task submission, we assume tokenization and segmentation is already done[5]; we henceforth train ELMoLex on gold tokens and PoS tags provided by UDPipe (Straka et al., 2016). We evaluate our model using the segmentation and PoS tags provided by UDPipe, except for certain languages where we use the tokens and PoS tag predicted by our own tokenizer and taggers (as respectively explained in Section 2.6 and 2.7)[6] respectively.

2.1 Backbone parser

ELMoLex uses the BiAF parser (Dozat and Manning, 2016), a state-of-the-art graph-based parser, as its backbone. BiAF parser consumes a sequence of tokens and their PoS tags, which is fed through a multilayer BiLSTM network. The output state of the final LSTM layer is then fed through four separate ReLU layers, producing four specialized vector representations: first for the word as a modifier seeking its head; second for the word as a head seeking all its modifiers; third for the word as a modifier deciding on its label; and lastly for the word as head deciding on the labels of its modifiers. These vectors become the input to two biaffine classifiers: one computes a score for each token pair, with the highest score for a given token indicating that token's most probable head; the other computes a score for each label for a given token/head pair, with the highest score representing the most probable label for the arc from the head to the

[4]We use lexicon information for treebanks from 43 languages provided by UDLexicons (Sagot, 2018).

[5]We explored a wide variety of tokenization and segmentation techniques in our last year submission (de La Clergerie et al., 2017). Our primary focus for this year is to explore novel neural network layers for both tagging and parsing.

[6]Due to lack of time, we could not train the taggers effectively for all the languages and use the predicted PoS from UDPipe for training our parser.

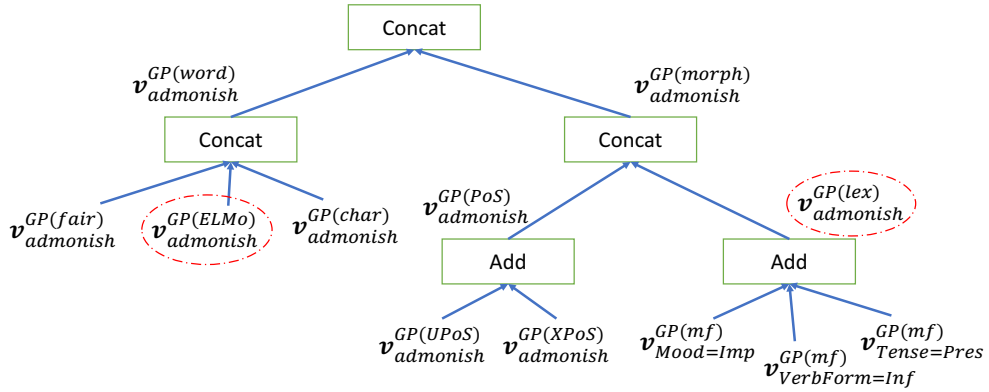

Figure 2: Architecture of embedding model used by ELMoLex for the word 'admonish'. $\mathbf{v}^{GP(ELMo)}$ and $\mathbf{v}^{GP(lex)}$ (red ellipses) are our major contributions. Arrows indicate structural dependence, but not necessarily trainable parameters.

modifier. We refer the readers to Dozat and Manning (2016) for further details.

Formally, the BiAF parser consumes a sequence of n word embeddings ($\mathbf{v}_1^{GP(word)}, \ldots, \mathbf{v}_n^{GP(word)}$) and n tag embeddings ($\mathbf{v}_1^{GP(tag)}, \ldots, \mathbf{v}_n^{GP(tag)}$) as input, which can be written as:

$$\begin{aligned}
\mathbf{x}_i^{GP} &= \mathbf{v}_i^{GP(word)} \oplus \mathbf{v}_i^{GP(tag)}, \\
\mathbf{v}_i^{GP(word)} &= \mathbf{v}_i^{GP(token)} + \mathbf{v}_i^{GP(w2v)} + \mathbf{v}_i^{GP(char)}, \\
\mathbf{v}_i^{GP(tag)} &= \mathbf{v}_i^{GP(UPoS)} + \mathbf{v}_i^{GP(XPoS)}.
\end{aligned} \tag{1}$$

In the equation 1, $\mathbf{v}_i^{GP(token)}$, $\mathbf{v}_i^{GP(w2v)}$, $\mathbf{v}_i^{GP(char)}$ represent the holistic frequent word embedding, pre-trained word embedding (fixed) and LSTM based character-level embeddings respectively, whereas the Universal PoS (UPoS) and language-specific (XPoS) tag embeddings are represented by $\mathbf{v}_i^{GP(UPoS)}$ and $\mathbf{v}_i^{GP(XPoS)}$ respectively. Note that '$+$' denotes element-wise addition operator, while '$\oplus$' denotes concatenation operator.

ELMoLex reformulates the input embedding layer of BiAF parser in a few ways (as illustrated in Figure 2): we utilize an indomain version of ELMo features ($\mathbf{v}_i^{GP(ELMo)}$) which provide context-dependent word representation (as discussed in Section 2.2); we utilize disambiguated, embedded, morphosyntactic features from lexicons ($\mathbf{v}_i^{GP(lex)}$), which provide information that is especially relevant for word with an irregular morphology (Sagot and Martínez Alonso, 2017), thereby complementing the existing feature set (as discussed in Section 2.3). Incorporating them, equation 1 now becomes:

$$\begin{aligned}
\mathbf{x}_i^{GP} &= \mathbf{v}_i^{GP(word)} \oplus \mathbf{v}_i^{GP(morph)}, \\
\mathbf{v}_i^{GP(word)} &= \mathbf{v}_i^{GP(fair)} \oplus \mathbf{v}_i^{GP(char)} \oplus \mathbf{v}_i^{GP(ELMo)}, \\
\mathbf{v}_i^{GP(morph)} &= \mathbf{v}_i^{GP(PoS)} \oplus \mathbf{v}_i^{GP(lex)}, \\
\mathbf{v}_i^{GP(PoS)} &= \mathbf{v}_i^{GP(UPoS)} + \mathbf{v}_i^{GP(XPoS)}.
\end{aligned} \tag{2}$$

In the equation 2, $\mathbf{v}_i^{GP(fair)}$ and $\mathbf{v}_i^{GP(char)}$ represent the learnable embeddings that are associated with frequent words in the vocabulary (pre-initialized from FAIR word vectors (Bojanowski et al., 2017)) and convolution-based character-level embeddings (Ma et al., 2018)[7] respectively. Apart from these changes in the embedding layer, we replace the decoding strategy (tree construction from the predicted graph) of our parser from greedy decoding (used by BiAF parser) to Chu-Liu-Edmonds algorithm (Chu and Liu, 1967), which further improves performance during evaluation.

2.2 ELMo features

In natural language, the meaning of a word changes when the underlying linguistic context changes. This fact is not captured by static word embeddings due to their *context independence*. Employing a deep, contextualized word representation, ELMo (Peters et al., 2018), which is a function of the entire sentence, yields promising result for several downstream tasks such as Question Answering, Textual Entailment and Sentiment Analysis. We attempt to test whether this hypothesis holds for dependency parsing. This is an interesting experiment as the authors of ELMo obtain larger improvements for tasks with small train set

[7]For obtaining character emebddings, we prefer convolution operation (introduced in dos Santos and Zadrozny (2014)) over LSTM (used by BiAF and introduced in Ballesteros et al. (2015)) as the former is parallelizable and efficient especially for large treebanks.

(sample efficient), indicating that smaller treebanks deprived of useful information could potentially enjoy good improvements.[8]

The backbone of ELMo is a BiLSTM-based neural Language Model (BiLSTM-LM), which is trained on a large raw corpus. We attempt to explore in this work whether we can train an indomain version of a BiLSTM-LM effectively using the available training data. The main challenge to accomplish this task is to learn transferable features in the absence of abundant raw data. Inspired by the authors of BiAF who use a large, well-tuned network to create a high performing graph parser, we implement a large BiLSTM-LM network (independent of the ELMoLex parser) which is highly regularized to prevent data overfitting and able to learn useful features. Our BiLSTM-LM consumes both the word and tag embedding as input, which can be formally written as:

$$\begin{aligned}\mathbf{x}_i^{LM} &= \mathbf{v}_i^{LM(word)} \oplus \mathbf{v}_i^{LM(UPoS)}, \\ \mathbf{v}_i^{LM(word)} &= \mathbf{v}_i^{LM(fair)} \oplus \mathbf{v}_i^{LM(char)}.\end{aligned} \tag{3}$$

In equation 3, the notations $\mathbf{v}_i^{LM(fair)}$, $\mathbf{v}_i^{LM(char)}$ and $\mathbf{v}_i^{LM(UPoS)}$ are the counterparts of $\mathbf{v}_i^{GP(fair)}$, $\mathbf{v}_i^{GP(char)}$ and $\mathbf{v}_i^{GP(UPoS)}$ respectively.

Note that ELMo, as proposed in Peters et al. (2018), builds only on character embeddings, automatically inferring the PoS information in the lower layers of the LSTM network. Since we have less training data to work with, we feed the PoS information explicitly which helps in easening the optimization process of our BiLSTM-LM network. Given a sequence of n words, $x_1^{LM}, \ldots, x_n^{LM}$, BiLSTM-LM learns by maximizing the log likelihood of forward LSTM and backward LSTM directions, which can be defined as:

$$\begin{aligned}&\sum_{i=1}^{n} (log\ \Pr(x_i^{LM} | x_1^{LM}, \ldots, x_{i-1}^{LM}; \Theta_x; \overrightarrow{\Theta}_{LSTM}, \overrightarrow{\Theta}_s)) \\ &+ (log\ \Pr(x_i^{LM} | x_{i+1}^{LM}, \ldots, x_n^{LM}; \Theta_x; \overleftarrow{\Theta}_{LSTM}, \overleftarrow{\Theta}_s)).\end{aligned} \tag{4}$$

We share the word embedding layer (Θ_x) of both LSTMs and learn the rest of the parameters independently. Unlike Peters et al. (2018), we do not tie the Softmax layer (Θ_s) in both the LSTM directions. Essentially, ELMo features are computed by a task-specific linear combination of the BiLSTM-LM's intermediate layer representations. If L represents the number of layers in BiLSTM-LM, ELMo computes a set of $2L + 1$ representations:

$$\begin{aligned}R_k &= \{\mathbf{x}_k^{LM}, \overleftarrow{\mathbf{h}}_{k,j}^{LM}, \overrightarrow{\mathbf{h}}_{k,j}^{LM} | j = 1, \ldots, L\} \\ &= \{\mathbf{h}_{k,j}^{LM} | j = 1, \ldots, L\},\end{aligned} \tag{5}$$

where $\mathbf{h}_{k,0}^{LM}$ is the word embedding layer (Equation 3) and $\mathbf{h}_{k,j}^{LM} = \overleftarrow{\mathbf{h}}_{k,j}^{LM} \oplus \overrightarrow{\mathbf{h}}_{k,j}^{LM}$, for each BiLSTM layer.

The authors of ELMo (Peters et al., 2018) show that different layers of BiLSTM-LM carry different types of information: lower-level LSTM states capture syntactic aspects (e.g., they can be used for PoS tagging); higher-level LSTM states model context-dependent aspects of word meaning (e.g., they can be used for word sense disambiguation); This observation is exploited by ELMoLex which can smartly select among all of these signals the useful information for dependency parsing. Thus, ELMo features for a word are computed by attending (softly) to the informative layers in R, as follows:

$$\mathbf{v}_i^{GP(ELMo)} = \mathbb{E}(R_k; \Theta^{elmo}) = \gamma^{elmo} \sum_{j=0}^{L} s_j^{elmo} \mathbf{h}_{k,j}^{LM}. \tag{6}$$

In equation 6, s^{elmo} corresponds to the softmax-normalized weights, while γ^{elmo} lets ELMoLex to scale the entire ELMo vector.

2.3 Lexicon features

Character-level models depend on the internal character-level make-up of a word. They exploit the relevant sub-parts of a word such as suffixes or prefixes to generate word representations. They can generalize to unknown words if these unknown words follow such generalizations. Otherwise, they fail to add any improvement (Sagot and Martínez Alonso, 2017) and we may need to look for other sources to complement the information provided by character-level embeddings. We term this problem as *linguistic naivety*.

ELMoLex taps into the large inventory of morphological features (gender, number, case, tense, mood, person, etc.) provided by external resources, namely the UDLexicons (Sagot, 2018) lexicon collection, which cover words with an irregular morphology as well as words not present in the training data. Essentially, these lexicons consist of ⟨word, UPoS, morphological features⟩ triplets, which we query using ⟨word, UPoS⟩ pair resulting in one or more hits. When we attempt to integrate the information from these hits, we face the challenge of disambiguation as not all the morphological features returned by the query are relevant to the focal ⟨word, UPoS⟩ pair. ELMoLex relies

[8] ELMoLex rank 8th overall for small treebanks by LAS metric.

Target	Source(s)
hy_armtdp	grc_perseus, grc_proiel
kk_ktb	tr_imst, ug_udt
hsb_ufal	*mixed*
kmr_mg	fa_seraji
bxr_bdt	tr_imst, ug_udt, ko_gsd, ko_kaist, ja_gsd
pcm_nsc	*mixed*
en_pud	en_ewt
th_pud	*mixed*
ja_modern	ja_gsd
br_keb	*mixed*
fo_oft	da_ddt, sv_talbanken, sv_lines, no_nynorsklia, no_bokmaal, no_nynorsk
fi_pud	fi_tdt
sv_pud	sv_talbanken
cs_pud	cs_pdt

Table 1: Treebanks to source the training data for Delexicalized Parsing of a given target treebank.

on attention mechanism (Bahdanau et al., 2014) to select the relevant morphological features, thereby having the capability to handle noisy or irrelevant features by paying no attention.

Put formally, given a sequence of m morphological feature embeddings ($\mathbf{v}^{GP(mf)}_{mf_1}$, ..., $\mathbf{v}^{GP(mf)}_{mf_m}$) for a word i, the lexicon-based embedding for the word ($\mathbf{v}^{GP(lex)}_{i}$) can be computed as follows:[9]

$$\mathbf{v}^{GP(lex)}_{i} = \sum_{j=1}^{m} s^{lex}_{mf_j} \mathbf{v}^{GP(mf)}_{mf_j}. \quad (7)$$

In equation 7, $s^{lex}_{mf_j}$ corresponds to the softmax-normalized weight which is a learnable parameter for each available morphological feature (in this case, it is mf_j). The general idea to perform a weighted sum to extract relevant features has been previously studied in the context of sequence labeling (Rei et al., 2016) for integrating word and character level features. Combining the distributional knowledge of words along with the semantic lexicons has been extensively studied for estimating high quality word vectors, also referred to as 'retrofitting' in literature (Faruqui et al., 2015).

2.4 Delexicalized Parsing

We perform delexicalized "language family" parsing for treebanks with less than 50 or no train sentences (as shown in Table 1). The delexicalized version of ELMoLex throws away word-level information such as $\mathbf{v}^{GP(word)}$ and $\mathbf{v}^{GP(char)}$ and works with the rest. The source treebanks are concatenated to form one large treebank, which is then used to train the delexicalized parser for the corresponding target treebank. In case of "mixed model", we concatenate at most 300 sentences from each treebank to create the training data.

2.5 Handling OOV words

Out Of Vocabulary (OOV) word problem is prevalent in languages with rich morphology and an accurate parser should come up with smart techniques to perform better than substituting a learned unknown vocabulary token ('UNK') during evaluation. To circumvent this problem, ELMoLex relies on four signals from the proposed embedding layer:

- $\mathbf{v}^{GP(fair)}$: If an OOV word is present in the FAIR word vectors, ELMoLex directly substitute the word embedding without any transformation.[10] If OOV word is absent, we resort to using 'UNK' token.
- $\mathbf{v}^{GP(ELMo)}$: For an OOV word, the ELMo layer of ELMoLex computes the context-dependent word representation based on the other vocabulary words present in the focal sentence.
- $\mathbf{v}^{GP(char)}$: Character-level embedding layer of ELMoLex computes the representation based on the known characters extracted from the OOV word naturally.
- $\mathbf{v}^{GP(lex)}$: If an OOV word is present in the external lexicon, ELMoLex queries with the ⟨word, PoS⟩ pair and computes the representation based on the known set of morphological features.

2.6 Neural Tagger with embedded lexicon

As described in Dozat et al. (2017), the BiAF model benefits from Part-of-Speech (PoS) inputs only if their accuracy is high enough. Our idea was therefore to design an accurate tagger in order to improve the performance of the parser.

Moreover, the shared task allows the use of external resources such as lexicons. A lexicon is simply a collection of possibilities in terms of PoS and morphological features usually provided for a large amount of words. In the context of neural taggers, an external lexicon can be seen as an external memory that can be useful in two ways:

[9] We experienced inferior results with other lexicon-based representations such as n-hot vector (having active value corresponding to the each of the morphological feature), unweighted average of morphological feature embeddings and morphological feature group based embedded attention.

[10] Inspired by Mikolov et al. (2013), we experimented with a linear transformation of the FAIR word vectors to the trained word embedding space, which resulted in poor performance. This confirms the intuition that the learned word embedding space is a non-linear transformation of the pre-trained word vectors.

- *For making training faster.* At initialization, for a given token, all possible PoS tags are equiprobable of being predicted by the network. The model only learns from the example it sees. By providing the model with a constrained set of possible tags as input features we can expect the training process to be faster.

- *For helping the model with OOV tokens.* Indeed, the lexicon provides information - potentially complementary to the character based representation - on OOV tokens that could be useful at inference.

Generally speaking, this experience is interesting because it challenges the idea that neural models, if deep enough and trained on enough data, don't require external resources and can learn everything in an end-to-end manner. As we will see for tagging, external pre-computed resources such as lexicons are of a great help.

The tagger we design is based on the neural tagging model with lexicon described in Sagot and Martínez Alonso (2017) and adapted using architectural insights from Dozat and Manning (2016). In short, words are represented in three ways. The first part is a trainable word vector (initialized with the FAIR vectors described in Bojanowski et al. (2017)). The second part is a character-based representation either computed using 1-dimensional convolution or a recurrent LSTM cell. The third component is an n-hot encoded vector of the tags that appear in an external lexicon, possibly embedded in a continuous space. These three components are summed, providing a *morphologically and lexically-enriched word representation*. This vector is then fed to a two layer BiLSTM that encodes the sentence level context, followed by two heads, one for predicting UPoS and the other for predicting morphological features. Each head is composed of a dense layer followed by a softmax layer.

2.7 Specific Tokenization post-processing for Arabic

To improve the Arabic tokenizer, we noticed that tokenization is very error-prone wherein most of the errors come from wrong analysis of the letter 'و'. Indeed, in Arabic, this letter (which is a coordinate conjunction, "and") is usually concatenated to the next word (e.g 'وقطر', "and Qata") but is sometimes just a part of the word (e.g 'وافق', "he agrees"). This ambiguity confuses the UDPipe tokenizer. Our fix consists in splitting that letter from its word whenever UDPipe was unable to provide a proper UPoS tag. This simple fix led to a 0.7% improvement in word segmentation compared to the UDPipe baseline and led us to rank 4^{th} on Arabic in the final LAS metric.

Treebank	Neural Tagger	UDPipe
af_afribooms	95.33	**95.53**
da_dadt	**95.54**	95.18
el_gdt	**95.48**	94.80
fr_sequoia	**96.13**	95.78
fr_spoken	**95.71**	93.70
hu_szeged	**93.07**	92.56
sv_lines	**95.27**	94.37
tr_imst	**91.08**	91.02
vi_vtb	77.10	**77.80**
zh_gsd	**84.26**	83.24

Table 2: UPoS F1 on Dev. datasets (used as test)

3 Results

The implementation of ELMoLex as well as the neural tagger are based on the publicly available BiAF parser code provided by CMU (Ma et al., 2018). Similar to Dozat et al. (2017), we use mostly the same set of hyper-parameters (as displayed in Appendix A), which makes ELMoLex robust across a wide variety of treebanks present in the shared task (Zeman et al., 2018). For treebanks with no development data, we perform a 5-fold cross validation to identify the average number of epochs taken to train each fold. By setting the maximum number of epochs to this average number, we then train ELMoLex on 90% of the training data and use the rest of the training data for selecting our best model. When we do not find external lexicon in UDLexicons (Sagot, 2018) for a given language, we skip the lexicon based features ($\mathbf{v}^{GP(lex)}$) and work with the rest. ELMoLex ran for ∼26 hours on the TIRA virtual machine (Potthast et al., 2014) sequentially, which can be trivially parallelized to run within two hours. Our shared task results are displayed in Appendix B.

3.1 Performance Analysis of the Tagger

Given the general architecture we presented in Section 2.6, we are able to test a few key questions: Is recurrent cell better suited at encoding word morphology compared to 1-D convolution layer ? Is embedding the lexical information into a continuous space useful for improving the performance ? And finally, is using an external lexicon always useful for better UPoS tagging ? We summarize our results as follows:

- Convolution layer works better than recurrent

Treebank	Vanilla		NLM Init.		ELMo		Lex		ELMoLex	
nl_lassysmall	73.57	(230)	74.08	(751)	74.10	(779)	**74.23**	(979)	74.05	(500)
fr_spoken	61.83	(94)	61.97	(141)	62.07	(175)	62.28	(275)	**62.49**	(383)
el_gdt	81.43	(148)	82.06	(687)	82.09	(764)	**82.48**	(1358)	**82.48**	(2028)
it_postwita	64.95	(369)	66.35	(225)	65.27	(266)	**65.67**	(600)	65.63	(518)
ro_rrt	81.21	(430)	81.08	(1139)	81.39	(1325)	**81.60**	(1694)	81.47	(890)
tr_imst	53.72	(249)	53.67	(624)	54.00	(735)	**54.16**	(670)	53.97	(1559)
uk_iu	78.62	(316)	78.73	(271)	**79.21**	(330)	79.14	(731)	78.87	(1891)

Table 3: Ablation study of ELMoLex. LAS dev. score along with training time (with 90% of the training data with the rest used for selecting the best model) in minutes is reported for selected treebanks. For NLM Init. and ELMoLex models, we report the time taken to train the parser (excluding the time taken to train the underlying BiLSTM-LM). All the reported models uses Chu-Liu-Edmonds algorithm (Chu and Liu, 1967) for constructing the final parse tree.

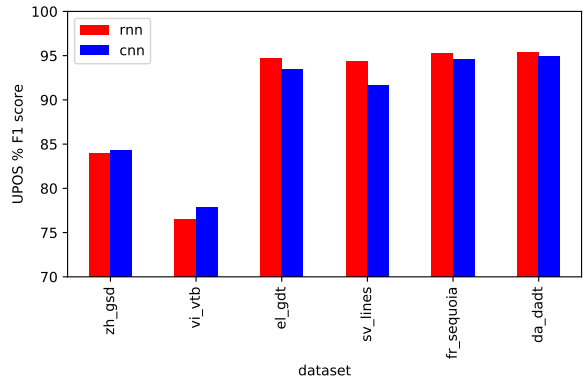

Figure 3: Comparing morphological embedding technique: RNN vs CNN

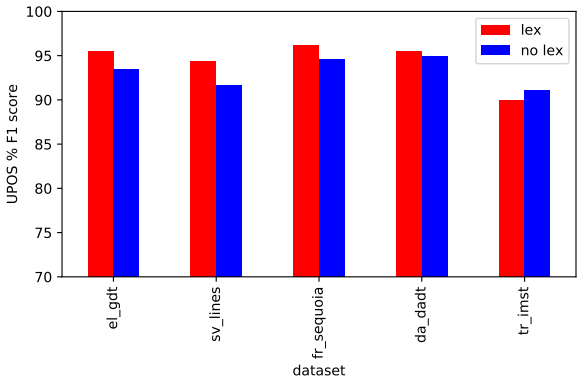

Figure 4: Impact of external lexicons

cell for languages such as Vietnamese and Chinese.

- Leveraging an external lexicon helps the tagger for most of the languages, specifically for languages such as French (tested on $fr\_sequoia$ and fr_spoken), Greek, Danish, Hungarian and Swedish. The only language for which the lexicon did not help is Turkish.

- Using a continuous embedding layer for lexicon features always leads to better performance compare to straight n-hot encoding.

We now present in more details the performance of our model regarding these three dimensions. The results are reported on development datasets treated as a strict test set.

As we notice in Figure 3, using a convolution layer as a morphological embedding technique provides poorer results compared to a recurrent cell except for two cases: Chinese and Vietnamese. This suggests that the different morphology and tokenization complexity that we find for Europeans languages compared to Chinese and Vietnamese might well require different kind of embedding architectures. Intuitively, we could say that the character-wise sequential structure of the Europeans languages is better modeled by a recurrent cell, while a language like Chinese with overlaying phenomenons is better modeled by a convolution layer.

We now describe the impact of an external lexicon for UPoS tagging (Figure. 4). We present the results only for the datasets for which the RNN cell was providing the best results. We compare two architectures: the neural tagger using a recurrent cell for morphology with an external lexicon (embedded in a continuous space) and the same architecture without external lexicons. For all the treebanks (except Turkish), the lexicon helps the UPoS tagging performance.

The last component of the neural tagger we analyze is the input technique of the lexical information. We compare two techniques. The first one is the architecture described in Sagot and Martínez Alonso (2017) for which the lexical information is feeded using a n-hot encoded vector which is then concatenated with the other word representation vectors. The second one embeds in a continuous space the lexicon tags before averaging them providing a single vector that summarizes the lexical information of a given word. As we see in Figure 5, in all languages we experi-

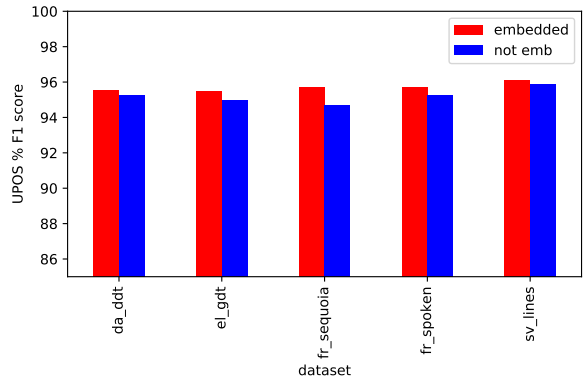

Figure 5: Impact of lexicon embedding technique

mented with, the embedding layers provides better results than a simple n-hot encoded representation.

For most of the treebanks, we performed significantly above the UDPipe baseline for UPoS tagging. Our results are summarized in Table 2. Unfortunately, we reached these results too late before the deadline, and we did not get the time to retrain our parser on our own, predicted PoS tags. Therefore, at test time, we only used the predicted tags for treebanks for which we were confident that our predicted tags would help the parser. It resulted in using our system for four treebanks: *el_gdt*, *hu_szeged*, *sv_lines* and *tr_imst*. In Appendix C,you will find the performance of our Neural tagger and our ELMoLex parser trained on our predicted tags for which we were able to retrain the models after the system submission deadline.

3.2 Ablation Study of the Parser

To unpack the benefits obtained from each of our contributions, we perform ablation studies with the following variants of ELMoLex: ELMoLex without ELMo and lexicon features, which is effectively the *vanilla* BiAF model (Ma et al., 2018); initializing the BiLSTM layer of the vanilla model with that of a pre-trained BiLSTM-LM (*NLM Init.*)[11]; ELMoLex with the ELMo features only (ELMo); ELMoLex with the lexicon features only (Lex); ELMoLex with both ELMo and Lexicon features (full version); The models used in the ablation study are trained on 90% of the training data, tuned on the remaining 10% and tested on the development set provided by the organizers.

The results are displayed in Table 3[12]. We make the following important observations: (1) Utilizing either ELMo or Lexicon or both always outperform the BiAF model; (2) External lexicons brings in valuable information about OOV words and words with irregular morphology, thereby outperforming BiAF (which relies for those cases on character embeddings only) by a large margin; (3) ELMo entails high train time due to the additional LSTM operation over the entire sentence, but exhibits strong performance over BiAF model which naturally leads us to combine it along with the lexicon information to create ELMoLex (our final submission system).

[11] Our final submission to the shared task did not have the NLM pre-initialization feature.

[12] The training time reported in the table is an over-estimate, as it is captured when several parsers (at most four of them) are running together in a single GTX 1070 GPU.

Treebank	ParisNLP-2017	ParisNLP-2018
el_edt	82.25	86.83 (+4.58)
hu_szeged	66.82	74.36 (+7.54)
sv_lines	76.61	79.92 (+3.31)
tr_imst	56.03	61.27 (+5.24)

Table 4: Comparing LAS scores for 2017 and 2018 participation of ParisNLP. The increase in the absolute LAS points are enclosed in the ellipses.

In summary, in contrast with our participation to the shared task last year (de La Clergerie et al. (2017)[13], we decided to focus on neural models wherein we explored many architectures and ideas both for tagging and parsing. As a result we reached superior performance in the final LAS score compared to our last year submission. To illustrate this, we compare the results of our current submission with that of the last year for four treebanks (Table 4) and observe significant improvements in the LAS score.

3.3 Extrinsic Evaluation of the Parser

A "good" parser should not only perform well in the intrinsic metrics such as LAS, BLAS and BLEX, but should strengthen a real world NLP system by providing relevant syntactic features. To understand the impact of ELMoLex in a downstream NLP application, we participiated in the shared task on Extrinsic Parser Evaluation (Fares et al., 2018). The goal of this task is to evaluate the parse trees predicted by ELMoLex on three downstream applications: biological event extraction, fine-grained opinion analysis, and negation resolution, for its usefulness. Since all the tasks are based on English language, we train ELMoLex on *en_ewt* treebank (which is the largest English treebank provided by the organizers (Zeman et al., 2018)) without changing the hyper-parameters (as disclosed in Appendix A). We refer the readers to Fares et al. (2018) for details about each of the downstream task and the accompanying system (which takes the fea-

[13] Our team 'ParisNLP' ranked 6^{th} in the unofficial ranking.

Downstream Task	Precision	Recall	F1 Score
Event Extraction	55.66 (-3.6)	43.56 (-9.87)	48.87 (-4.72)
Negation Resolution	100 (0)	40.68 (-2.07)	57.83 (-1.91)
Opinion Analysis	63.01 (-3.66)	56.78 (-6.1)	59.73 (-4.99)

Table 5: Results on the downstream tasks for our ELMoLex system trained on the en_ewt treebank with the corresponding difference from the best system enclosed in ellipses.

tures dervied from ELMoLex) used to solve the task. Our extrinsic evaluation results[14] are displayed in Table 5. ELMoLex ranked 7^{th} for Event Extraction, Negation Resolution tasks and 11^{th} for Opinion Analysis task by F1 score. On an average, ELMoLex ranked 8^{th} with a F1 score of 55.48%.

4 Conclusion

We presented our parsing system, ELMoLex, which successfully integrates context-dependent ELMo and lexicon-based representations to overcome the context independency and linguistic naivety problem in the embedding layer of the BiAF model respectively. We showed the analysis of our neural tagger, whose competitive performance in PoS estimation is capitalized by ELMoLex to achieve strong gains in parsing quality for four treebanks. We also performed an ablation study to understand the source of gains brought by ELMoLex. We evaluated ELMoLex on three downstream applications to understand its usefulness.

In the next step of our work, we plan to: (1) compare the performance in utilizing recurrent layer over the convolution layer for character embeddings (similar to our neural tagger experiment) which underlies our parser; (2) pursue the NLM initialization feature further to inspect if using it can enrich ELMoLex; (3) observe the performance when we augment the in-domain train data for our BiLSTM-LM with massive raw data (such as Wikipedia); (4) train our parser and tagger jointly with the gold PoS tags; and (5) exploit lattice information (More et al., 2018; Buckman and Neubig, 2018) which captures rich linguistic knowledge.

Acknowledgments

We thank the organizers and the data providers who made this shared task possible within the core Universal Dependencies framework (Nivre et al., 2016), namely the authors of the UD version 2.0 datasets (Nivre et al., 2018), the baseline UDPipe models (Straka et al., 2016), and the team behind the TIRA evaluation platform (Potthast et al., 2014) to whom we owe a lot. We also thank José Carlos Rosales Núñez for useful discussions. This work was funded by the ANR projects ParSiTi (ANR-16-CE33-0021) and SoSweet (ANR15-CE38-0011-01).

[14] In our last year joint submission with Stanford (Schuster et al., 2017), we evaluated different semantic dependencies representations and also compared different parsing strategies.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473. http://arxiv.org/abs/1409.0473.

Miguel Ballesteros, Chris Dyer, and Noah A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. pages 349–359. http://aclweb.org/anthology/D/D15/D15-1041.pdf.

Piotr Bojanowski, Édouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL* 5:135–146. https://transacl.org/ojs/index.php/tacl/article/view/999.

Jacob Buckman and Graham Neubig. 2018. Neural lattice language models. *CoRR* abs/1803.05071. http://arxiv.org/abs/1803.05071.

Yoeng-Jin Chu and Tseng-Hong Liu. 1967. On the shortest arborescence of a directed graph. In *Science Sinica*. pages 1396–1400.

Éric Vilemonte de La Clergerie, Benoît Sagot, and Djamé Seddah. 2017. The ParisNLP entry at the CoNLL UD Shared Task 2017: A Tale of a # ParsingTragedy. In *Conference on Computational Natural Language Learning*. pages 243–252.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing,*

China, 21-26 June 2014. pages 1818–1826. http://jmlr.org/proceedings/papers/v32/santos14.html.

Timothy Dozat and Christopher D. Manning. 2016. Deep Biaffine Attention for Neural Dependency Parsing. *CoRR* abs/1611.01734. http://arxiv.org/abs/1611.01734.

Timothy Dozat, Peng Qi, and Christopher D Manning. 2017. Stanford's Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task. *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies* pages 20–30.

Murhaf Fares, Stephan Oepen, Lilja Øvrelid, Jari Björne, and Richard Johansson. 2018. The 2018 Shared Task on Extrinsic Parser Evaluation. On the downstream utility of English Universal Dependency parsers. In *Proceedings of the 22nd Conference on Natural Language Learning*. Brussels, Belgia.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard H. Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*. pages 1606–1615. http://aclweb.org/anthology/N/N15/N15-1184.pdf.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics (TACL)* 4:313–327. https://transacl.org/ojs/index.php/tacl/article/view/885.

Xuezhe Ma, Zecong Hu, Jingzhou Liu, Nanyun Peng, Graham Neubig, and Eduard H. Hovy. 2018. Stack-pointer networks for dependency parsing. *CoRR* abs/1805.01087. http://arxiv.org/abs/1805.01087.

Austin Matthews, Graham Neubig, and Chris Dyer. 2018. Using Morphological Knowledge in Open-Vocabulary Neural Language Models. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018*. New Orleans, Louisiana, USA, pages 1435–1445. https://aclanthology.info/papers/N18-1130/n18-1130.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *CoRR* abs/1309.4168. http://arxiv.org/abs/1309.4168.

Amir More, Özlem Çetinoglu, Çagri Çöltekin, Nizar Habash, Benoît Sagot, Djamé Seddah, Dima Taji, and Reut Tsarfaty. 2018. Conll-ul: Universal morphological lattices for universal dependency parsing. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018*. Miyazaki, Japan.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, Aitziber Atutxa, Liesbeth Augustinus, Elena Badmaeva, Miguel Ballesteros, Esha Banerjee, Sebastian Bank, Verginica Barbu Mititelu, John Bauer, Sandra Bellato, Kepa Bengoetxea, Riyaz Ahmad Bhat, Erica Biagetti, Eckhard Bick, Rogier Blokland, Victoria Bobicev, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Adriane Boyd, Aljoscha Burchardt, Marie Candito, Bernard Caron, Gauthier Caron, Gülşen Cebiroğlu Eryiğit, Giuseppe G. A. Celano, Savas Cetin, Fabricio Chalub, Jinho Choi, Yongseok Cho, Jayeol Chun, Silvie Cinková, Aurélie Collomb, Çağrı Çöltekin, Miriam Connor, Marine Courtin, Elizabeth Davidson, Marie-Catherine de Marneffe, Valeria de Paiva, Arantza Diaz de Ilarraza, Carly Dickerson, Peter Dirix, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Ali Elkahky, Binyam Ephrem, Tomaž Erjavec, Aline Etienne, Richárd Farkas, Hector Fernandez Alcalde, Jennifer Foster, Cláudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Kim Gerdes, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökırmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Céline Guillot-Barbance, Nizar Habash, Jan Hajič, Jan Hajič jr., Linh Hà Mỹ, Na-Rae Han, Kim Harris, Dag Haug, Barbora Hladká, Jaroslava Hlaváčová, Florinel Hociung, Petter Hohle, Jena Hwang, Radu Ion, Elena Irimia, Tomáš Jelínek, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşıkara, Sylvain Kahane, Hiroshi Kanayama, Jenna Kanerva, Tolga Kayadelen, Václava Kettnerová, Jesse Kirchner, Natalia Kotsyba, Simon

Krek, Sookyoung Kwak, Veronika Laippala, Lorenzo Lambertino, Tatiana Lando, Septina Dian Larasati, Alexei Lavrentiev, John Lee, Phương Lê Hồng, Alessandro Lenci, Saran Lertpradit, Herman Leung, Cheuk Ying Li, Josie Li, Keying Li, KyungTae Lim, Nikola Ljubešić, Olga Loginova, Olga Lyashevskaya, Teresa Lynn, Vivien Macketanz, Aibek Makazhanov, Michael Mandl, Christopher Manning, Ruli Manurung, Cătălina Mărănduc, David Mareček, Katrin Marheinecke, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Gustavo Mendonça, Niko Miekka, Anna Missilä, Cătălin Mititelu, Yusuke Miyao, Simonetta Montemagni, Amir More, Laura Moreno Romero, Shinsuke Mori, Bjartur Mortensen, Bohdan Moskalevskyi, Kadri Muischnek, Yugo Murawaki, Kaili Müürisep, Pinkey Nainwani, Juan Ignacio Navarro Horñiacek, Anna Nedoluzhko, Gunta Nešpore-Bērzkalne, Lương Nguyễn Thị, Huyền Nguyễn Thị Minh, Vitaly Nikolaev, Rattima Nitisaroj, Hanna Nurmi, Stina Ojala, Adédayọ̀stling Robert Olúòkun, Lilja Øvrelid, Niko Partanen, Elena Pascual, Marco Passarotti, Agnieszka Patejuk, Siyao Peng, Cenel-Augusto Perez, Guy Perrier, Slav Petrov, Jussi Piitulainen, Emily Pitler, Barbara Plank, Thierry Poibeau, Martin Popel, Lauma Pretkalniņa, Sophie Prévost, Prokopis Prokopidis, Adam Przepiórkowski, Tiina Puolakainen, Sampo Pyysalo, Andriela Rääbis, Alexandre Rademaker, Loganathan Ramasamy, Taraka Rama, Carlos Ramisch, Vinit Ravishankar, Livy Real, Siva Reddy, Georg Rehm, Michael Rießler, Larissa Rinaldi, Laura Rituma, Luisa Rocha, Mykhailo Romanenko, Rudolf Rosa, Davide Rovati, Valentin Ro□ca, Olga Rudina, Shoval Sadde, Shadi Saleh, Tanja Samardžić, Stephanie Samson, Manuela Sanguinetti, Baiba Saulīte, Yanin Sawanakunanon, Nathan Schneider, Sebastian Schuster, Djamé Seddah, Wolfgang Seeker, Mojgan Seraji, Mo Shen, Atsuko Shimada, Muh Shohibussirri, Dmitry Sichinava, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Isabela Soares-Bastos, Antonio Stella, Milan Straka, Jana Strnadová, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Dima Taji, Yuta Takahashi, Takaaki Tanaka, Isabelle Tellier, Trond Trosterud, Anna Trukhina, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Zdeňka Urešová, Larraitz Uria, Hans Uszkoreit, Sowmya Vajjala, Daniel van Niekerk, Gertjan van Noord, Viktor Varga, Veronika Vincze, Lars Wallin, Jonathan North Washington, Seyi Williams, Mats Wirén, Tsegay Woldemariam, Taksum Wong, Chunxiao Yan, Marat M. Yavrumyan, Zhuoran Yu, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, Manying Zhang, and Hanzhi Zhu. 2018. Universal dependencies 2.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics (ÚFAL), Faculty of Mathematics and Physics, Charles University. http://hdl.handle.net/11234/1-2837.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portorož, Slovenia, pages 1659–1666.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. pages 2227–2237. https://aclanthology.info/papers/N18-1202/n18-1202.

Martin Potthast, Tim Gollub, Francisco Rangel, Paolo Rosso, Efstathios Stamatatos, and Benno Stein. 2014. Improving the reproducibility of PAN's shared tasks: Plagiarism detection, author identification, and author profiling. In Evangelos Kanoulas, Mihai Lupu, Paul Clough, Mark Sanderson, Mark Hall, Allan Hanbury, and Elaine Toms, editors, *Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 14)*. Springer, Berlin Heidelberg New York, pages 268–299. https://doi.org/10.1007/978-3-319-11382-1_22.

Marek Rei, Gamal K. O. Crichton, and Sampo Pyysalo. 2016. Attending to characters in neural sequence labeling models. In *COLING 2016, 26th International Conference on Computational Linguis-*

tics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan. pages 309–318. http://aclweb.org/anthology/C/C16/C16-1030.pdf.

Benoît Sagot. 2018. A multilingual collection of conll-u-compatible morphological lexicons. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018*. Miyazaki, Japan.

Benoît Sagot and Héctor Martínez Alonso. 2017. Improving neural tagging with lexical information. In *Proceedings of the 15th International Conference on Parsing Technologies, IWPT 2017*. Pisa, Italy, pages 25–31. https://aclanthology.info/papers/W17-6304/w17-6304.

Sebastian Schuster, Éric De La Clergerie, Marie Candito, Benoît Sagot, Christopher D. Manning, and Djamé Seddah. 2017. Paris and Stanford at EPE 2017: Downstream evaluation of graph-based dependency representations. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies*. Pisa, Italy, page 47 – 59.

Milan Straka, Jan Hajič, and Jana Straková. 2016. UDPipe: trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing. In *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*. European Language Resources Association, Portorož, Slovenia.

Daniel Zeman, Jan Hajič, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*. Association for Computational Linguistics, Brussels, Belgium, pages 1–20.

Appendix A: Hyper-parameters Used

	Hyper-parameter	ELMoLex Value	BiLSTM-LM Value	Neural Tagger Value
Embed	CNN window	3	3	3
	char-LSTM	-	-	300
	CNN filters	100	100	300
	character embed. size	100	100	100
	word embed. size	300	300	300
	PoS embed. size	100	100	-
	lexicon based embed. size	100	-	300
LSTM	layers	3	3	2
	hidden state	512	150	400
	arc MLP size	512	-	-
	label MLP size	128	-	400
	BPTT	10	-	-
Dropout	embeddings	0.33	0.33	0.33
	LSTM hidden states	0.33	0.33	0.5
	LSTM layers	0.33	0.33	0.33
Optimization	optimizer	Adam	Adam	Adam
	init learning rate	0.001	0.001	0.001
	(β_1, β_2)	(0.9, 0.9)	(0.9, 0.9)	(0.9, 0.9)
	decay rate	0.75	0.75	0.75
	gradient clipping	5.0	5.0	5.0
	schedule	10	10	10
	maximum epochs	1000	variable [15]	1000
	batch size	variable[16]	variable[16]	12

Table 6: Hyper-parameters for all the models used in the experiments.

Appendix B: CoNLL Final Results

Treebank	LAS	MLAS	BLEX	Treebank	LAS	MLAS	BLEX
af_afribooms	82.97 (-2.50)	69.59 (-6.08)	72.57 (-3.87)	hy_armtdp	12.49 (-24.52)	1.98 (-11.38)	6.24 (-12.80)
ar_padt	72.77 (-4.29)	58.95 (-9.59)	63.62 (-6.44)	id_gsd	77.44 (-2.61)	65.06 (-3.30)	65.58 (-10.98)
bg_btb	88.85 (-2.37)	79.07 (-4.05)	77.92 (-6.39)	it_isdt	89.51 (-2.49)	80.43 (-3.46)	81.33 (-3.43)
br_keb	10.53 (-28.11)	0.34 (-13.57)	2.23 (-18.47)	it_postwita	69.73 (-9.66)	55.95 (-12.55)	57.63 (-11.71)
bxr_bdt	15.06 (-4.47)	1.78 (-1.20)	3.61 (-3.04)	ja_gsd	74.44 (-8.67)	60.38 (-12.24)	63.05 (-10.74)
ca_ancora	89.96 (-1.65)	81.63 (-2.44)	83.03 (-2.44)	ja_modern	13.63 (-14.70)	3.06 (-8.76)	4.49 (-9.30)
cs_cac	90.31 (-1.30)	75.72 (-7.70)	84.87 (-1.92)	kk_ktb	23.50 (-8.43)	5.88 (-3.05)	8.62 (-2.71)
cs_fictree	89.39 (-2.63)	74.18 (-10.05)	82.86 (-4.95)	kmr_mg	17.93 (-12.48)	4.09 (-3.89)	7.99 (-5.67)
cs_pdt	90.27 (-1.41)	79.13 (-5.97)	86.31 (-1.60)	ko_gsd	82.22 (-2.92)	75.17 (-5.68)	69.25 (-7.06)
cs_pud	82.68 (-3.45)	67.87 (-7.94)	76.23 (-4.30)	ko_kaist	85.49 (-1.42)	77.49 (-3.80)	72.29 (-7.26)
cu_proiel	70.54 (-5.19)	57.01 (-6.30)	63.56 (-7.75)	la_ittb	84.64 (-2.44)	74.66 (-5.18)	81.33 (-3.04)
da_ddt	81.22 (-5.06)	70.47 (-6.84)	71.74 (-6.33)	la_perseus	55.02 (-17.61)	32.08 (-17.69)	37.15 (-15.60)
de_gsd	77.64 (-2.72)	36.95 (-21.09)	67.94 (-3.46)	la_proiel	67.90 (-5.71)	53.02 (-6.34)	61.85 (-5.75)
el_gdt	86.83 (-2.82)	67.92 (-10.74)	74.12 (-5.97)	lv_lvtb	78.16 (-5.81)	60.19 (-7.70)	66.52 (-5.88)
en_ewt	81.42 (-3.15)	71.53 (-4.80)	74.97 (-3.47)	nl_alpino	85.17 (-4.39)	69.52 (-7.00)	74.06 (-5.09)
en_gum	78.43 (-6.62)	65.12 (-8.12)	66.25 (-7.32)	nl_lassysmall	80.79 (-6.05)	67.44 (-6.67)	69.79 (-6.75)
en_lines	76.64 (-5.33)	66.73 (-5.52)	68.95 (-6.34)	no_bokmaal	88.90 (-2.33)	79.58 (-4.10)	82.26 (-3.56)
en_pud	80.75 (-7.14)	67.89 (-6.97)	72.19 (-8.34)	no_nynorsk	88.67 (-2.32)	78.08 (-3.78)	81.57 (-2.87)
es_ancora	89.13 (-1.80)	81.18 (-2.75)	82.59 (-2.33)	no_nynorsklia	54.39 (-15.95)	39.68 (-17.83)	45.47 (-15.51)
et_edt	81.78 (-3.57)	71.59 (-5.38)	69.82 (-9.55)	pcm_nsc	11.32 (-18.75)	3.49 (-1.81)	9.80 (-16.24)
eu_bdt	80.45 (-3.77)	63.40 (-8.33)	72.50 (-5.65)	pl_lfg	92.96 (-1.90)	76.96 (-9.97)	84.14 (-6.28)
fa_seraji	85.14 (-2.97)	77.73 (-3.10)	75.48 (-4.96)	pl_sz	89.03 (-3.20)	67.31 (-13.46)	78.85 (-7.44)
fi_ftb	83.73 (-4.80)	71.43 (-8.22)	69.33 (-13.11)	pt_bosque	86.53 (-1.28)	71.73 (-4.21)	78.02 (-2.60)
fi_pud	71.64 (-18.59)	62.46 (-21.32)	57.40 (-25.04)	ro_rrt	84.75 (-2.12)	76.02 (-2.66)	77.28 (-3.69)
fi_tdt	83.06 (-5.67)	73.16 (-7.68)	67.47 (-13.77)	ru_syntagrus	91.41 (-1.07)	81.74 (-5.02)	84.65 (-4.00)
fo_oft	29.48 (-19.95)	0.34 (-0.73)	7.16 (-7.24)	ru_taiga	62.15 (-12.09)	38.46 (-23.13)	45.77 (-18.59)
fr_gsd	85.00 (-1.89)	75.38 (-3.06)	78.51 (-2.67)	sk_snk	84.22 (-4.63)	58.52 (-16.49)	67.08 (-13.66)
fr_sequoia	85.32 (-4.57)	74.70 (-7.85)	79.13 (-5.54)	sl_ssj	84.48 (-6.99)	67.85 (-14.53)	76.57 (-6.66)
fr_spoken	66.41 (-9.37)	53.30 (-11.37)	55.96 (-9.67)	sl_sst	50.62 (-10.77)	35.21 (-10.72)	41.82 (-9.12)
fro_srcmf	85.53 (-1.59)	76.96 (-3.32)	82.14 (-1.97)	sme_giella	65.95 (-3.92)	48.73 (-8.74)	49.28 (-10.82)
ga_idt	67.24 (-3.64)	39.68 (-6.11)	46.29 (-8.89)	sr_set	87.02 (-1.64)	72.20 (-5.53)	78.10 (-5.18)
gl_ctg	81.25 (-1.51)	68.48 (-2.44)	71.77 (-3.37)	sv_lines	79.92 (-4.16)	61.59 (-4.99)	72.27 (-4.74)
gl_treegal	70.48 (-3.77)	51.10 (-9.53)	56.75 (-7.54)	sv_pud	71.77 (-8.58)	42.69 (-9.05)	55.48 (-10.64)
got_proiel	66.37 (-3.18)	51.20 (-5.25)	59.18 (-4.80)	sv_talbanken	84.27 (-4.36)	73.98 (-5.34)	76.49 (-4.95)
grc_perseus	71.76 (-7.63)	39.22 (-15.76)	49.39 (-9.29)	th_pud	0.23 (-13.47)	0.00 (-6.29)	0.01 (-10.76)
grc_proiel	74.51 (-4.74)	54.87 (-5.40)	62.83 (-6.20)	tr_imst	61.71 (-4.73)	48.32 (-7.41)	52.34 (-7.79)
he_htb	62.17 (-13.92)	47.82 (-15.56)	51.52 (-13.52)	ug_udt	61.27 (-5.78)	40.73 (-5.05)	49.48 (-5.94)
hi_hdtb	90.98 (-1.43)	72.25 (-6.05)	84.80 (-1.94)	uk_iu	81.33 (-7.10)	59.99 (-12.28)	70.06 (-8.32)
hr_set	84.53 (-2.83)	61.82 (-11.62)	75.94 (-4.56)	ur_udtb	81.10 (-2.29)	52.34 (-5.64)	68.57 (-5.22)
hsb_ufal	28.15 (-18.27)	4.88 (-4.21)	15.06 (-6.03)	vi_vtb	43.04 (-12.18)	35.45 (-12.16)	38.89 (-5.13)
hu_szeged	74.36 (-8.30)	55.98 (-11.15)	63.76 (-9.41)	zh_gsd	62.80 (-13.97)	52.47 (-14.15)	58.01 (-14.96)

	LAS	MLAS	BLEX
All treebanks	70.64 (-5.20)	55.74 (-5.51)	60.70 (-5.39)
Big treebanks only	80.29 (-4.08)	65.88 (-6.79)	70.95 (-4.88)
PUD treebanks only	64.09 (-10.11)	48.79 (-9.96)	53.16 (-10.09)
Small treebanks only	60.84 (-8.69)	40.71 (-8.53)	46.08 (-8.81)
Low-resource languages only	16.52 (-11.37)	2.53 (-3.60)	6.75 (-7.23)

Table 7: Results on each treebank in the shared task along with the macro average over all of them (with the corresponding difference from the best system enclosed in ellipses).

Appendix C: Performance of ELMoLex trained and tested using the NeuroTagger tags

Treebank	UPOS	LAS	MLAS	BLEX
da_ddt	95.53 (+0.09)	81.69(+0.47)	69.19 (-1.28)	72.57(+0.83)
de_gsd	91.97 (+0.39)	77.09(-0.55)	37.24(+0.29)	67.71(-0.23)
eu_bdt	92.74 (+0.4)	79.94(-0.51)	61.84(-1.56)	72.02(-0.48)
el_gdt**	96.35 (-)	87.31(+0.48)	68.68(+0.76)	75.07(+0.95)
fr_sequoia	96.66 (+0.82)	87.31 (+1.99)	76.11(+1.41)	81.43(+2.3)
fr_spoken	94.63 (+1.69)	66.98(+0.57)	54.1(+0.8)	56.6(+0.64)
hr_set	96.96 (+0.63)	84.56 (+0.03)	61.88 (+0.06)	75.98(+0.06)
hu_szeged**	91.82 (-)	73.86(-0.5)	55.59 (-0.39)	63.38 (-0.38)
sv_lines**	95.28 (-)	79.57 (-0.35)	61.35(-0.24)	72.2(-0.07)
zh_gsd**	83.59 (+0.12)	64.56 (+1.76)	52.85(+0.38)	60.24(+2.23)

	UPOS	LAS	MLAS	BLEX
Average gain	+0.59	+0.34	0.02	+0.58

Table 8: Performance of ELMOLEX trained and tested using tags from the neural tagger (with the corresponding absolute difference from our submission final results) and average absolute gain of using the neural tagger compared to tags used at the submissions
**indicates datasets for which tags from the neural tagger where used at test time for the submission