Romanian Part of Speech Tagging using LSTM Networks

Beáta Lőrincz[a,b], Maria Nuţu[a,b], Adriana Stan[a]
[a]Communications Department, Technical University of Cluj-Napoca, Romania
[b]Department of Computer Science, "Babeş-Bolyai" University, Cluj-Napoca, Romania
{beata.lorincz, maria.nutu, adriana.stan}@com.utcluj.ro

***Abstract*—In this paper we present LSTM based neural network architectures for determining the part of speech (POS) tags for Romanian words. LSTM networks combined with fully-connected output layers are used for predicting the root POS, and sequence-to-sequence models composed of LSTM encoders and decoders are evaluated for predicting the extended MSD and CTAG tags. The highest accuracy achieved for the root POS is 99.18% and for the extended tags is 98.25%. This method proves to be efficient for the proposed task and has the advantage of being language independent, as no expert linguistic knowledge is used in the input features.**

***Index Terms*—POS tagging, recurrent neural networks, LSTM, sequence-to-sequence, Romanian, MSD, CTAG**

I. Introduction

Part of speech (POS) tagging is one of the key tasks of natural language processing (NLP). It refers to the identification of the part of speech of a given word and optionally, additional grammatical properties inherent to a particular POS. Along with other components of NLP, such as lemmatization, stemming, syllabification, word or sentence boundary detection and many others it aims to help computers understand and process natural language.

The difficulty of the task lays in the fact that the same orthographic form of a word can have a different meaning depending on the context (i.e. homographs). Another problem is that the declination and inflections of the words are not regular, especially in morphological rich languages, such as the case of Romanian. As a result, to define the correct POS of a word aside from its spelling, we also need to take into account the semantic links between words.

The task of POS tagging can have several annotation levels. The most basic one refers to determining the root POS (noun, verb, adjective, pronoun, determiner, article, adverb, adposition, conjunction, numeral, interjections, residual, abbreviation, part particle) and can usually be obtained from the dictionary entry of the word's lemma. The most complex tagset is the Morpho-Syntactic Descriptions (MSD) set, which adds several grammatical properties depending on the root POS [1]. Compared to MSD, the C-tagset introduced by [2] is a reduced size tagset that adds a maximum of 3 additional properties to the root POS. For ease of expression we will refer to the process of determining the root or an extended tag of a word as POS-tagging in general and denote the specific tagset where it is necessary.

Several single or multilingual text processing tools have been developed to perform the task of POS tagging with the scope of being incorporated into applications such as speech recognition, speech synthesis, machine translation or textual information extraction. The most common approaches among the previous studies rely on probabilistic and rule-based methods such as Hidden Markov Models, Maximum Entropy Classifiers, Bayesian Networks and Conditional Random Fields [3]. According to [3] these methods do not perform well on languages that use a lot of inflection, such as Romanian. In addition, rule based methods are language dependent and might require hand-crafted rules. Tools for building these rules have also been developed [4].

With the increasing popularity of machine learning, deep neural networks have also been successfully applied to NLP tasks. Neural network based algorithms have been used and compared to probabilistic POS-taggers since the 1990's [5] for the English language. To the best of our knowledge the first accuracy results for POS-tagging with neural networks applied to the Romanian language were reported in [3].

Studies concerning the Romanian language are also numerous with a relatively high reported accuracy. Tufiş et al. [2] achieved an accuracy of 98.39% using a tiered tagging with C-tagset for the language model, extended with a post-processor using probabilistic methods to reconstruct the MSD tag. [4] and [6] present hybrid methods combining statistical models with a rule based system that classifies tagging errors and reduces the need for manual rule construction. The best accuracy obtained by Simionescu is 97.03%. [3] reports an accuracy of 98.17% for MSD-tagging by implementing feed-forward neural networks with genetic algorithms used for designing the network topology. The BAILE multilanguage system [8] obtains an accuracy of 95.03% for POS-tagging for the Romanian language. Similar accuracy, 96.12% is achieved by [7] using a Naive Bayes model with a word database. All

TABLE I
POS-tagging accuracy results for Romanian reported in the literature

Authors	Method	Accuracy	Tagset
Tufis & Mason [2]	Probabilistic	98.39%	MSD
Boros & Dumitrescu [3]	Deep Neural Networks	98.19%	MSD
Simionescu [6]	Probabilistic & Rule-based	97.03%	MSD
Teodorescu et al. [7]	Probabilistic	96.12%	Root POS
Frunza et al. [8]	Machine Learning	95.30%	Root POS

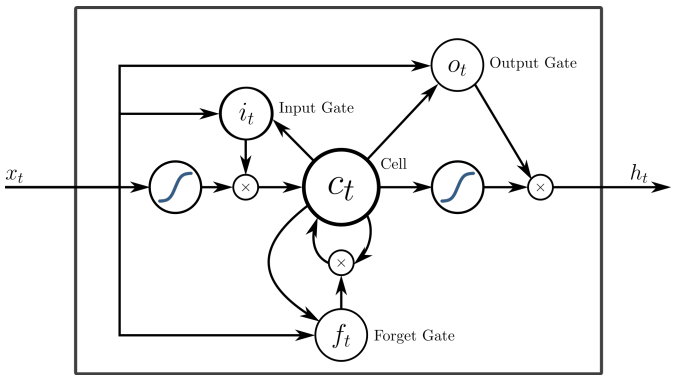

Fig. 1. LSTM memory cell [13]

the above results are summarized in Table I.

Starting from this overview, in this work we investigate the use of a recurrent neural network model with Long Short Term Memory (LSTM) layers for POS-tagging taking into consideration previous studies evaluated on several other languages (not including Romanian) [9], [10]. They conclude that LSTM networks work well for determining the POS-tag, not only with limited, but also with extended tagsets. We compare the results obtained with LSTM networks to a sequence-to-sequence model composed of LSTM encoders and decoders used for determining extended tags.

The paper is structured as follows: the proposed methods for POS-tagging are described in Section II, and the details of training data and implementation are elaborated in Section III. We discuss the results in Section IV. The conclusions and possible future work are summarized in Section V.

II. Method overview

As neural network based learning methods are now widely used in many areas of NLP and speech processing applications, in this article we focus on experimenting with different neural network architectures to determine the root POS or the extended tag.

Recurrent neural networks (RNN) are highly efficient in sequential data modelling. Their main advantage is that their output combines the current input with the output of the previous time step. Therefore it can extend its understanding of the data to the temporal connections between sequential inputs. However, vanilla RNNs cannot expand across long temporal sequences [11]. To overcome this disadvantage, Long Short Term Memory (LSTM) structures can be used [12]. These structures are based on recurrent nodes, but are extended with a memory cell which can model long-term dependencies in the input data. The memory cell contains a set of gates which decides what previous information needs to be saved in the cell state (memory), what needs to be discarded and how much the current input and stored memory will influence the output of the cell. The memory cell components are depicted in Figure 1.

The behaviour of the cell gates are described by Equations 1, 2, and 3:

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad (1)$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \quad (2)$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \quad (3)$$

where σ is the sigmoid function, and i, f, o are the input, forget, output gates, respectively, at time step t. w_x are the weights of the appropriate *gate(x)*, x_t is the input at time step t and b_x is the bias for the respective *gate(x)*. The states of the cell at the current time step t are stored in cell vector c_t and hidden vector h_t. These state vectors have the same dimension as the cell gate vectors. Equations 4 and 5 describe the state vectors:

$$c_t = f_t C_{t-1} + i_t \tanh\left(w_c[h_{t-1}, x_t] + b_c\right) \quad (4)$$

$$h_t = o_t \tanh c_t \quad (5)$$

where c_t – the cell state is computed based on the previous state with a sigmoid activation function computed in training time and with the candidate for the current state.

The hidden state is calculated from the cell state passed through an activation function and element-wise multiplied with the output vector at time step t.

In our experiments to predict the root POS-tag (containing a single character) we added to the LSTM layers a fully-connected (dense) layer. The unit size of this layer is set to the number of possible POS-tags. The behaviour of this layer is described in Equation 6:

$$o_x = (Wx) + b \quad (6)$$

where the output is calculated based on the inputs (x), weight (W) and bias (b).

To determine the MSD tag of the word, a sequence-to-sequence (seq2seq) [14] learning method was applied. RNN networks can be used when the input and output vectors can be encoded with fixed-dimension vectors and are not appropriate for defining the variable length MSD tag of a word. The seq2seq model is composed of an *encoder* and a *decoder*. The encoder and decoder are both neural networks that are frequently implemented with LSTM cells.

The encoder is responsible for interpreting the input data, one time step at a time, and converting it into a fixed dimension vector representation. The decoder uses the hidden or output state of the encoder to condition its own output. In general, the decoder is trained with one time step-delayed sequences. This means that it learns to predict the next character or word in the output sequence. Figure 2 presents the seq2seq model for the word "acasă" as input and "Rg" as output sequence, where <SS> marks the start and <SE> the end of sequence.

III. Evaluation

A. Tagsets

Different tagsets can be used for POS-tagging to indicate the POS and/or additional grammatical categories. The tagset is language dependent and contains all possible parts of speech in the respective language. For Romanian, we used the following tagsets: root tagset (ID:**RPOS**), MSD-tagset (ID:**MSD**) and

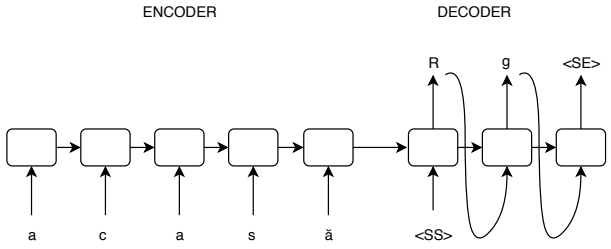

Fig. 2. Sequence-to-sequence model input and output sequence example

TABLE II
NUMBER OF TAGS PER TAGSET

Tagset	No. of tags used
Basic	13
MSD	334
CTAG	89

TABLE III
MSD TAG EXAMPLES FOR A VERB AND A NOUN

Vmp3pf		Ncmsrn	
V	verb	N	noun
m	main	c	common
p	participle	m	masculin
3	third person	s	singular
p	plural	r	nominative
f	feminine	n	not definite

C-tagset (ID:**CTAG**). The number of different tags used from each tagset is shown in Table II.

The most basic tagset contains only the part of speech of the word's lemma as extracted from a linguistic dictionary (e.g 'V' for verb, 'N' for noun, 'A' for adjective etc.). We refer to this as the *root POS* tag.

The MSD (Morpho-Syntactic Descriptions) is an extended tagset containing the codes defined by the MULTEXT-EAST project [1]. The number of MSD tags varies per dataset and according to [15] a number of 615 tags were described for the Romanian word-form lexicon. The MSD is a hierarchical POS representation, where the first upper case letter represents the root POS followed by the lexical attributes of the POS. Examples for a verb and noun MSD tag are shown in Table III.

A different tagset, called C-tagset (CTAG) was introduced in [2]. CTAG is a reduced version of the MSD and can be mapped to it directly. For example 'NSRN' stands for a common noun, that is singular, direct and indefinite. We used this tagset when training on large amounts of text where the tag is defined based on the context of the word.

B. Datasets

Three different datasets were used for the training part. Most of the experiments were run on the morphological dictionary created by Simionescu [4] with the help of DexOnline database and Wikipedia[1] proper nouns collection (ID:**WPT**).[2] This dataset consists of 1,715,881 words associated with one or more MSD tags. For the POS-tagging training we used the first character of the MSD tag. If multiple tags were available for the same orthographic form, only the first entry was used. For evaluation purposes the test data was validated against all possible tags assigned to the word sample.

TABLE IV
NUMBER OF TRAINING AND TEST SAMPLES PER DATASET

Dataset	Total samples	No. of training samples	No. of test samples
WPT	1,715,881	897,328	224,331
DEX	1,994,412	936,611	234,152
CoRoLa	3,075,165	2,460,132	615,033

The second dataset we used is the Romanian Explicative Dictionary database (ID:**DEX**)[3] that contains 1,994,412 words, each associated with one or more POS-tags and a word frequency. The words with 0 frequency were removed from the training, resulting in a set of 1,158,197 samples.

The third dataset consists of 125,316 sentences selected from the CoRoLa [16] corpus (ID:**CoRoLa**).[4] The sentences contain 3,075,165 words with a number of 175,946 unique words. The CoRoLa corpus provides linguistic attributes for all words including the MSD and CTAG annotations and contains texts of different styles such as juridical or scientific. The word samples were collected from these texts randomly.

The first two datasets contain individual words with single or multiple POS or MSD information attached. The third dataset provides context-related information as the tags are assigned to words of the sentences. From each dataset 20% of the samples was randomly selected for testing and the remaining 80% used for training. The number of train and test samples is summarized in Table IV.

C. Data input format

The input data was coded either with one-hot-encoding (ID:**OHE**) or letter embedding (ID:**LE**). In the former encoding the words are represented by a two-dimensional binary matrix. The size of the matrix depends on the number of input characters and the maximum length of the input words. The letter embedding uses a dense representation of each character that contains information about the context of the selected character which is important for POS-tagging [17]. The Gensim[5] library was used to create a letter embedding of order 30 based on the Romanian Wikipedia pages' database dump. The order was selected to be close to the number of characters in the Romanian alphabet.

Word embeddings were not considered in our study, as the focus was to predict the POS tags using only the orthographic form of the words, independent of their linguistic context.

[1]https://ro.wikipedia.org/
[2]http://nlptools.infoiasi.ro/WebPosTagger
[3]https://dexonline.ro/
[4]http://corola.racai.ro/
[5]https://radimrehurek.com/gensim/

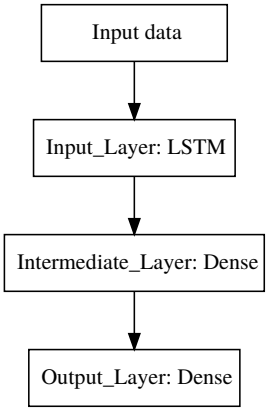

Fig. 3. LSTM with dense layers model

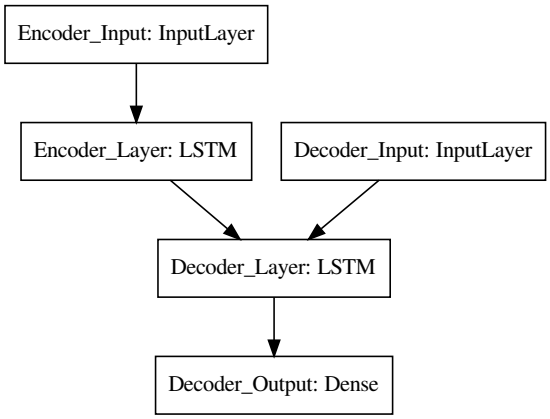

Fig. 4. LSTM sequence-to-sequence model

D. System architectures

For the implementation we used the Keras[6] deep learning library with TensorFlow[7] backend. We grouped our network architectures into the following categories:

1) LSTM with fully-connected (dense) layers: The input data was one-hot encoded or used letter embeddings and provided to a first LSTM layer. The latent dimension of the LSTM layer varied from 64 to 1024. On top of the LSTM layer two dense layers were stacked. The second dense layer serves as the output layer, having the size equal to the number of possible POS-tags. The architecture of this network is shown in Figure 3.

2) Sequence-to-Sequence Model: A character-level sequence-to-sequence (seq2seq) model is used for the MSD tagging. The input sequences are the words and the MSD tags of the target sequences. The encoder LSTM layer converts the input sequence into state vectors. The decoder LSTM layer uses the initial state vectors of the encoder and turns the target sequences into the same sequence offset by one time step. Teacher enforcing is used to generate the next character, as the decoder generates the $target[t+1, ...]$ sequence given the sequence $target[..., t]$. We add a starting and ending character to each target sequence. An example of an input and target sequence composed of 4 words is shown below:

Input sequence: 'Absolvent al Facultăţii de'
Target sequence: '#NSN TS NSOY S@'

When an unknown input sequence is decoded, we begin with the starting character and use the decoder to predict the next character until the end of sequence character or a maximum sequence length is reached.

The sequence-to-sequence model was used to determine the CTAGs within the context of sentences. The encoder and decoder models were trained with a sliding window over the words. The networks we trained used 3 to 5 words and were shifted to the right by one word for each sample. The best accuracy for the tags was achieved when the training samples were composed of 3 words. The architecture of this network is shown in Figure 4.

[6]https://keras.io/

[7]https://www.tensorflow.org/

IV. Results and discussions

Each network was evaluated on 20% of the initial dataset held out of the training data. The efficiency of the training method was measured by accuracy: dividing of the number of correct predictions by the total number of predictions. When calculating the accuracy for the MSD tags we only considered as correct outputs the predicted sequences that fully matched the target MSD-tag. The parameters and accuracy results are shown in Table V. The asterisk (*) marks the systems where we considered all possible tags associated with the words as correct answers, and not only the first entry tag.

The test data was randomly selected, and both training and test samples were shuffled before training. Based on initial tests we choose to set the batch sizes to 256, 512 or 1024. The best results were obtained with a batch size of 512. The latent dimension of the LSTM cells was evaluated with 64, 128, 256, 512 respectively 1024, the best accuracy numbers were achieved with size 256.

The best accuracy for the root POS-tagging (when predicting only a single character) was 99.18% (System ID 1). This outperforms all the systems presented in Table I which predict the root POS. When the same network was validated against only the most frequent POS and not all the possible POS tags available in the dataset, the accuracy was 94.85% (System ID 2). The accuracy breakdown per root POS is shown in Table VI. The POS types that did not end up in the test set due to their low count and the random selection, are marked with a not applicable (N/A) accuracy. The loss was calculated

TABLE V
NETWORK PARAMETERS AND ACCURACY RESULTS

System ID	Dataset	Tag	Network type	Character encoding	Latent dimension	Batch size	Epochs	Accuracy
1	WPT	RPOS	LSTM + Dense (*)	OHE	256	512	50	**99.18%**
2	WPT	RPOS	LSTM + Dense	OHE	256	512	50	94.85%
3	WPT	RPOS	LSTM + Dense	LE	256	256	25	54.80%
4	WPT	RPOS	seq2seq LSTM	LE	256	256	25	94.99%
5	WPT	RPOS	seq2seq + Embedding layer	OHE	256	256	20	93.88%
6	WPT	MSD	seq2seq LSTM (*)	OHE	512	1024	50	98.25%
7	WPT	MSD	seq2seq LSTM	OHE	512	1024	50	75.28%
8	WPT	MSD	seq2seq + Embedding layer	OHE	256	512	50	76.62%
9	DEX	RPOS	LSTM + Dense	OHE	256	512	50	94%
10	CoRoLa	CTAG	seq2seq LSTM	OHE	256	512	100	97.15%

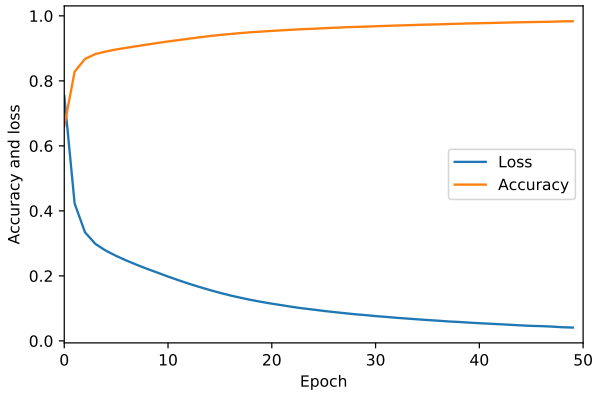

Fig. 5. The model accuracy and loss during training for System ID 1

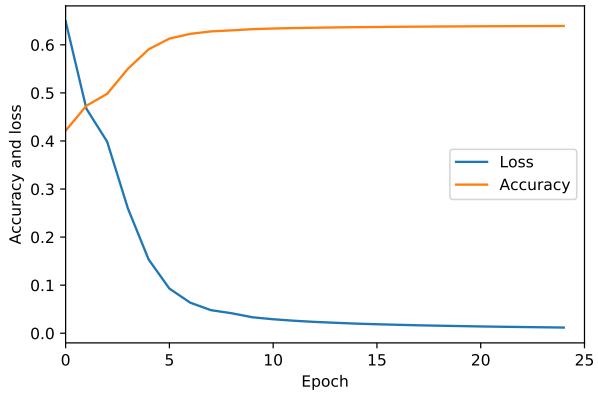

Fig. 6. The model accuracy and loss during training for System ID 6

TABLE VI
ACCURACY SUMMARY PER POS FOR SYSTEM ID 1

POS	No. of train samples	No. of test samples	Accuracy
Noun	471470	118133	99.86%
Adjective	232731	58457	99.86%
Verb	191489	47320	99.76%
Adverb	783	212	98.29%
Numeral	214	65	98.21%
Pronoun	220	48	97.76%
Determiner	161	37	98.99%
Interjections	156	37	96.37%
Abbreviation	58	0	N/A
Adposition	39	9	97.92%
Conjunction	12	0	N/A
Article	6	0	N/A
Part Particle	2	0	N/A

with categorical cross entropy method. The training accuracy and loss values of this model are shown in Figure 5.

The sequence-to-sequence model achieved an accuracy of 98.25% for the MSD tags (System ID 6). Compared to the POS prediction where the model only needed to predict one of the 13 root POS tags, the MSD model needs to predict 334 possible tags. Our results are comparable to the ones reported in [2]. However, the evaluation datasets are different, and [2] also uses context information.

The sequence-to-sequence model on CTAGs obtained an accuracy of 97.15%. The loss and accuracy values during model training are shown on Figure 6. This model uses context related information which improves the performance and could be extended to hierarchically predict MSD tags.

The models were tested with letter embeddings as input and also with the input layer replaced by an embedding layer. Changing the input did not result in higher accuracy. This could be explained by the fact that in the case of LE the positional information of a letter does not change the POS. In the case of the embedding layer the available data might not be sufficient for a good representation learning of the input.

V. CONCLUSIONS AND FUTURE WORK

In this paper we evaluated two different types of neural network architectures for POS-tagging, applied on the Romanian language. LSTM networks with fully-connected layers performed well on predicting the root POS, while sequence-to-sequence models achieved high accuracy for defining extended tags, such as the MSD and CTAGs. The addition of the letter embeddings and using embedding layers instead of the one-hot-encoding for representing the input data did not result in higher accuracy.

Despite the very good results obtained by our network architectures, there is still a need to explore other network architectures, such as convolutional neural networks, especially

for the extended tagsets. Adding other linguistic information, such as lemma or lexical stress could improve the accuracy of the POS taggers, especially in the case of homographs.

ACKNOWLEDGMENT

This work was supported by a grant of the Romanian Ministry of Research and Innovation, PCCDI – UEFISCDI, project number PN-III-P1-1.2-PCCDI-2017-0818/73, within PNCDI III.

REFERENCES

[1] T. Erjavec, "Multext-east morphosyntactic specifications: Version 3.0," *Supported By EU Projects Multext-East, Concede And TELRI*, 2004.

[2] D. Tufis and O. Mason, "Tagging romanian texts: a case study for qtag, a language independent probabilistic tagger," in *Proceedings of the First International Conference on Language Resources and Evaluation (LREC)*, vol. 1, no. 589-596, 1998, p. 143.

[3] T. Boroş and S. D. Dumitrescu, "Improving the racai neural network msd tagger," in *International Conference on Engineering Applications of Neural Networks*. Springer, 2013, pp. 42–51.

[4] R. Simionescu, "Graphical grammar studio as a constraint grammar solution for part of speech tagging," in *The Conference on Linguistic Resources and Instruments for Romanian Language Processing*, vol. 152, 2011.

[5] H. Schmid, "Part-of-speech tagging with neural networks," in *Proceedings of the 15th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 1994, pp. 172–176.

[6] R. Simionescu, "Hybrid pos tagger," in *Proceedings of Language Resources and Tools with Industrial Applications Workshop (Eurolan 2011 Summer School), Cluj-Napoca, Romania*. Citeseer, 2011, pp. 21–28.

[7] L. R. Teodorescu, R. Boldizsar, M. Ordean, M. Duma, L. Detesan, and M. Ordean, "Part of speech tagging for romanian text-to-speech system," in *2011 13th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*. IEEE, 2011, pp. 153–159.

[8] O. Frunza, D. Inkpen, and D. Nadeau, "A text processing tool for the romanian language," in *Proc. of the EuroLAN 2005 Workshop on Cross-Language Knowledge Induction*. Citeseer, 2005.

[9] T. Horsmann and T. Zesch, "Do lstms really work so well for pos tagging?–a replication study," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 727–736.

[10] B. Plank, A. Søgaard, and Y. Goldberg, "Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss," *arXiv preprint arXiv:1604.05529*, 2016.

[11] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, pp. 1735–80, 12 1997.

[13] H. Graves, Abdel-Rahman. Peephole long short-term memory, wikimedia commons. [Online]. Available: https://commons.wikimedia.org/wiki/

[14] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[15] D. Tufiş, "Tiered tagging and combined language models classifiers," in *International Workshop on Text, Speech and Dialogue*. Springer, 1999, pp. 28–33.

[16] V. B. Mititelu, E. Irimia, and D. Tufis, "Corola—the reference corpus of contemporary romanian language." in *LREC*, 2014, pp. 1235–1239.

[17] C. D. Santos and B. Zadrozny, "Learning character-level representations for part-of-speech tagging," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1818–1826.