Boosting Named Entity Recognition with Neural Character Embeddings

Cícero dos Santos
IBM Research
138/146 Av. Pasteur
Rio de Janeiro, RJ, Brazil
cicerons@br.ibm.com

Victor Guimarães
Instituto de Computação
Universidade Federal Fluminense (UFF)
Niterói, RJ, Rio de Janeiro
victorguimaraes@id.uff.br

Abstract

Most state-of-the-art named entity recognition (NER) systems rely on handcrafted features and on the output of other NLP tasks such as part-of-speech (POS) tagging and text chunking. In this work we propose a language-independent NER system that uses automatically learned features only. Our approach is based on the CharWNN deep neural network, which uses word-level and character-level representations (embeddings) to perform sequential classification. We perform an extensive number of experiments using two annotated corpora in two different languages: HAREM I corpus, which contains texts in Portuguese; and the SPA CoNLL-2002 corpus, which contains texts in Spanish. Our experimental results give evidence of the contribution of neural character embeddings for NER. Moreover, we demonstrate that the same neural network which has been successfully applied to POS tagging can also achieve state-of-the-art results for language-independet NER, using the same hyperparameters, and without any handcrafted features. For the HAREM I corpus, CharWNN outperforms the state-of-the-art system by 7.9 points in the F1-score for the total scenario (ten NE classes). For the SPA CoNLL-2002 corpus, CharWNN outperforms the state-of-the-art system by 0.8 point in the F1.

1 Introduction

Named entity recognition is a natural language processing (NLP) task that consists of finding names in a text and classifying them among several predefined categories of interest such as person, organization, location and time. Although machine learning based systems have been the predominant approach to achieve state-of-the-art results for NER, most of these NER systems rely on the use of costly handcrafted features and on the output of other NLP tasks (Tjong Kim Sang, 2002; Tjong Kim Sang and De Meulder, 2003; Doddington et al., 2004; Finkel et al., 2005; Milidiú et al., 2007). On the other hand, some recent work on NER have used deep learning strategies which minimize the need of these costly features (Chen et al., 2010; Collobert et al., 2011; Passos et al., 2014; Tang et al., 2014). However, as far as we know, there are still no work on deep learning approaches for NER that use character-level embeddings.

In this paper we approach language-independent NER using CharWNN, a recently proposed deep neural network (DNN) architecture that jointly uses word-level and character-level embeddings to perform sequential classification (dos Santos and Zadrozny, 2014a). CharWNN employs a convolutional layer that allows effective character-level feature extraction from words of any size. This approach has proven to be very effective for language-independent POS tagging (dos Santos and Zadrozny, 2014a; dos Santos and Zadrozny, 2014b).

We perform an extensive number of experiments using two annotated corpora: HAREM I corpus, which contains texts in Portuguese; and the SPA CoNLL-2002, which contains texts in Spanish. In our experiments, we compare the performance of the joint and individual use of character-level and word-level embeddings. We provide information on the impact of unsupervised pre-training of word embeddings in the performance of our proposed NER approach. Our experimental results evidence that CharWNN is effective and robust for Portuguese and Spanish NER. Using the same CharWNN configuration used by dos Santos and Zadrozny (2014) for POS Tagging,

Proceedings of the Fifth Named Entity Workshop, joint with 53rd ACL and the 7th IJCNLP, pages 25–33,
Beijing, China, July 26-31, 2015.

we achieve state-of-the-art results for both corpora. For the HAREM I corpus, CharWNN outperforms the state-of-the-art system by 7.9 points in the F1-score for the *total scenario* (ten NE classes), and by 7.2 points in the F1 for the *selective scenario* (five NE classes). For the SPA CoNLL-2002 corpus, CharWNN outperforms the state-of-the-art system by 0.8 point in the F1.

This work is organized as follows. In Section 2, we briefly describe the CharWNN architecture. Section 3 details our experimental setup and Section 4 discuss our experimental results. Section 6 presents our final remarks.

2 CharWNN

CharWNN extends Collobert et al.'s (2011) neural network architecture for sequential classification by adding a convolutional layer to extract character-level representations (dos Santos and Zadrozny, 2014a). Given a sentence, the network gives for each word a score for each class (tag) $\tau \in T$. As depicted in Figure 1, in order to score a word, the network takes as input a fixed-sized window of words centred around the target word. The input is passed through a sequence of layers where features with increasing levels of complexity are extracted. The output for the whole sentence is then processed using the Viterbi algorithm (Viterbi, 1967) to perform structured prediction. For a detailed description of the CharWNN neural network we refer the reader to (dos Santos and Zadrozny, 2014a).

2.1 Word- and Character-level Embeddings

As illustrated in Figure 1, the first layer of the network transforms words into real-valued feature vectors (embeddings). These embeddings are meant to capture morphological, syntactic and semantic information about the words. We use a fixed-sized word vocabulary V^{wrd}, and we consider that words are composed of characters from a fixed-sized character vocabulary V^{chr}. Given a sentence consisting of N words $\{w_1, w_2, ..., w_N\}$, every word w_n is converted into a vector $u_n = [r^{wrd}; r^{wch}]$, which is composed of two sub-vectors: the *word-level embedding* $r^{wrd} \in \mathbb{R}^{d^{wrd}}$ and the *character-level embedding* $r^{wch} \in \mathbb{R}^{cl_u}$ of w_n. While word-level embeddings capture syntactic and semantic information, character-level embeddings capture morphological and shape information.

Word-level embeddings are encoded by column vectors in an embedding matrix $W^{wrd} \in \mathbb{R}^{d^{wrd} \times |V^{wrd}|}$, and retrieving the embedding of a particular word consists in a simple matrix-vector multiplication. The matrix W^{wrd} is a parameter to be learned, and the size of the word-level embedding d^{wrd} is a hyperparameter to be set by the user.

The *character-level embedding* of each word is computed using a convolutional layer (Waibel et al., 1989; Lecun et al., 1998). In Figure 1, we illustrate the construction of the character-level embedding for the word *Bennett*, but the same process is used to construct the character-level embedding of each word in the input. The convolutional layer first produces local features around each character of the word, and then combines them using a max operation to create a fixed-sized character-level embedding of the word.

Given a word w composed of M characters $\{c_1, c_2, ..., c_M\}$, we first transform each character c_m into a character embedding r^{chr}_m. Character embeddings are encoded by column vectors in the embedding matrix $W^{chr} \in \mathbb{R}^{d^{chr} \times |V^{chr}|}$. Given a character c, its embedding r^{chr} is obtained by the matrix-vector product: $r^{chr} = W^{chr} v^c$, where v^c is a vector of size $\left|V^{chr}\right|$ which has value 1 at index c and zero in all other positions. The input for the convolutional layer is the sequence of character embeddings $\{r^{chr}_1, r^{chr}_2, ..., r^{chr}_M\}$.

The convolutional layer applies a matrix-vector operation to each window of size k^{chr} of successive windows in the sequence $\{r^{chr}_1, r^{chr}_2, ..., r^{chr}_M\}$. Let us define the vector $z_m \in \mathbb{R}^{d^{chr} k^{chr}}$ as the concatenation of the character embedding m, its $(k^{chr} - 1)/2$ left neighbors, and its $(k^{chr} - 1)/2$ right neighbors:

$$z_m = \left(r^{chr}_{m-(k^{chr}-1)/2}, ..., r^{chr}_{m+(k^{chr}-1)/2}\right)^T$$

The convolutional layer computes the j-th element of the vector r^{wch}, which is the character-level embedding of w, as follows:

$$[r^{wch}]_j = \max_{1<m<M} \left[W^0 z_m + b^0\right]_j \qquad (1)$$

where $W^0 \in \mathbb{R}^{cl_u \times d^{chr} k^{chr}}$ is the weight matrix of the convolutional layer. The same matrix is used to extract local features around each character window of the given word. Using the max over all character windows of the word, we extract a fixed-sized feature vector for the word.

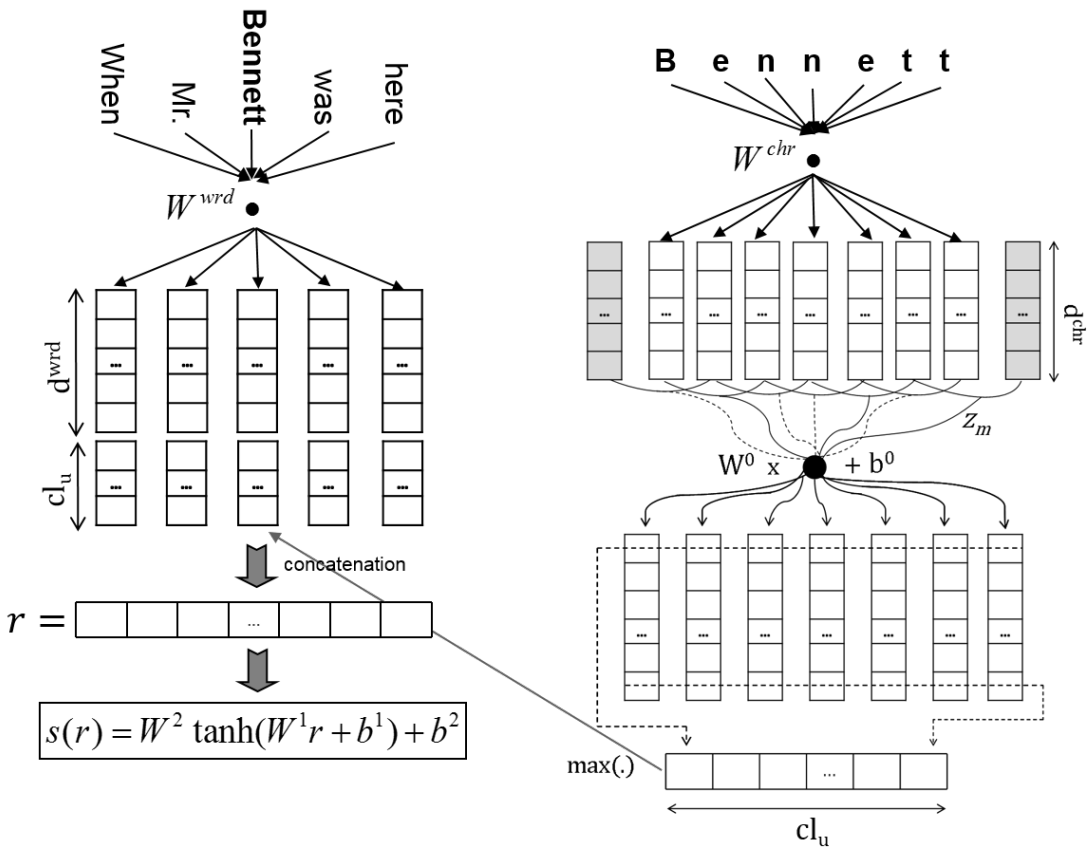

Figure 1: CharWNN Architecture

Matrices W^{chr} and W^0, and vector b^0 are parameters to be learned. The size of the character vector d^{chr}, the number of convolutional units cl_u (which corresponds to the size of the character-level embedding of a word), and the size of the character context window k^{chr} are hyperparameters.

2.2 Scoring and Structured Inference

We follow Collobert et al.'s (Collobert et al., 2011) window approach to score all tags T for each word in a sentence. This approach follows the assumption that in sequential classification the tag of a word depends mainly on its neighboring words. Given a sentence with N words $\{w_1, w_2, ..., w_N\}$, which have been converted to joint word-level and character-level embedding $\{u_1, u_2, ..., u_N\}$, to compute tag scores for the n-th word w_n in the sentence, we first create a vector r resulting from the concatenation of a sequence of k^{wrd} embeddings, centralized in the n-th word:

$$r = \left(u_{n-(k^{wrd}-1)/2}, ..., u_{n+(k^{wrd}-1)/2}\right)^T$$

We use a special *padding token* for the words with indices outside of the sentence boundaries.

Next, the vector r is processed by two usual neural network layers, which extract one more level of representation and compute the scores:

$$s(w_n) = W^2 h(W^1 r + b^1) + b^2 \qquad (2)$$

where matrices $W^1 \in \mathbb{R}^{hl_u \times k^{wrd}(d^{wrd}+cl_u)}$ and $W^2 \in \mathbb{R}^{|T| \times hl_u}$, and vectors $b^1 \in \mathbb{R}^{hl_u}$ and $b^2 \in \mathbb{R}^{|T|}$ are parameters to be learned. The transfer function $h(.)$ is the hyperbolic tangent. The size of the context window k^{wrd} and the number of hidden units hl_u are hyperparameters to be chosen by the user.

Like in (Collobert et al., 2011), CharWNN uses a prediction scheme that takes into account the sentence structure. The method uses a transition score A_{tu} for jumping from tag $t \in T$ to $u \in T$ in successive words, and a score A_{0t} for starting from the t-th tag. Given the sentence $[w]_1^N = \{w_1, w_2, ..., w_N\}$, the score for tag path

$[t]_1^N = \{t_1, t_2, ..., t_N\}$ is computed as follows:

$$S\left([w]_1^N, [t]_1^N, \theta\right) = \sum_{n=1}^{N} \left(A_{t_{n-1}t_n} + s(w_n)_{t_n}\right) \quad (3)$$

where $s(w_n)_{t_n}$ is the score given for tag t_n at word w_n and θ is the set of all trainable network parameters $\left(W^{wrd}, W^{chr}, W^0, b^0, W^1, b^1, W^2, b^2, A\right)$.

After scoring each word in the sentence, the Viterbi algorithm (Viterbi, 1967) is used to find the most likely tag sequence $[t^*]_1^N$, which consists in the tag path that leads to the maximal score:

$$[t^*]_1^N = _{[t]_1^N \in T^N} S\left([w]_1^N, [t]_1^N, \theta\right) \quad (4)$$

2.3 Network Training

We train CharWNN by minimizing a negative likelihood over the training set D. In the same way as in (Collobert et al., 2011), we interpret the sentence score (3) as a conditional probability over a path. For this purpose, we exponentiate the score (3) and normalize it with respect to all possible paths. Taking the log, we arrive at the following conditional log-probability:

$$\log p\left([t]_1^N | [w]_1^N, \theta\right) = S\left([w]_1^N, [t]_1^N, \theta\right) - \log\left(\sum_{\forall [u]_1^N \in T^N} e^{S\left([w]_1^N, [u]_1^N, \theta\right)}\right) \quad (5)$$

The log-likelihood in Equation 5 can be computed efficiently using dynamic programming (Collobert, 2011). We use stochastic gradient descent (SGD) to minimize the negative log-likelihood with respect to θ. We use the backpropagation algorithm to compute the gradients of the neural network. We implemented CharWNN using the *Theano* library (Bergstra et al., 2010).

3 Experimental Setup

3.1 Unsupervised Learning of Word Embeddings

The word embeddings used in our experiments are initialized by means of unsupervised pre-training. We perform pre-training of word-level embeddings using the skip-gram NN architecture (Mikolov et al., 2013) available in the word2vec [1] tool.

In our experiments on Portuguese NER, we use the word-level embeddings previously trained by dos Santos and Zadrozny (2014a). They have used a corpus composed of the Portuguese Wikipedia, the CETENFolha[2] corpus and the CETEMPublico[3] corpus.

In our experiments on Spanish NER, we use the Spanish Wikipedia. We process the Spanish Wikipedia corpus using the same steps used by dos Santos and Zadrozny (2014a): (1) remove paragraphs that are not in Spanish; (2) substitute non-roman characters by a special character; (3) tokenize the text using a tokenizer that we have implemented; (4) remove sentences that are less than 20 characters long (including white spaces) or have less than 5 tokens; (5) lowercase all words and substitute each numerical digit by a 0. The resulting corpus contains around 450 million tokens.

It is important to note that although we perform unsupervised pre-training of word embeddings, we also leave the word embeddings be updated during the supervised step, i.e., during the training with the NER labeled data.

Following dos Santos and Zadrozny (2014a), we do not perform unsupervised learning of character-level embeddings. The character-level embeddings are initialized by randomly sampling each value from an uniform distribution: $\mathcal{U}(-r, r)$, where $r = \sqrt{\frac{6}{|V^{chr}| + d^{chr}}}$. The weight matrices W^0, W^1 and W^2 are initialized in a similar way.

3.2 Corpora

We use the corpus from the first HAREM evaluation (Santos and Cardoso, 2007) in our experiments on Portuguese NER. This corpus is annotated with ten named entity categories: Person (PESSOA), Organization (ORGANIZACAO), Location (LOCAL), Value (VALOR), Date (TEMPO), Abstraction (ABSTRACCAO), Title (OBRA), Event (ACONTECIMENTO), Thing (COISA) and Other (OUTRO). The HAREM corpus is already divided into two subsets: First HAREM and MiniHAREM. Each subset corresponds to a different Portuguese NER contest. In our experiments, we call HAREM I the setup where we use the First HAREM corpus as the training set and the MiniHAREM corpus as the test set. This is the same setup used by dos Santos and Milidiú (2012). Additionally, we tokenize the

[1] http://code.google.com/p/word2vec/

[2] http://www.linguateca.pt/cetenfolha/

[3] http://www.linguateca.pt/cetempublico/

Table 1: Named Entity Recognition Corpora.

Corpus	Language	Training Data		Test Data	
		Sentenc.	Tokens	Sentenc.	Tokens
HAREM I	Portuguese	4,749	93,125	3,393	62,914
SPA CoNLL-2002	Spanish	8,323	264,715	1,517	51,533

HAREM corpus and create a development set that comprises 5% of the training set. Table 1 present some details of this dataset.

In our experiments on Spanish NER we use the SPA CoNLL-2002 Corpus, which was developed for the CoNLL-2002 shared task (Tjong Kim Sang, 2002). It is annotated with four named entity categories: Person, Organization, Location and Miscellaneous. The SPA CoNLL-2002 corpus is already divided into training, development and test sets. The development set has characteristics similar to the test corpora.

We treat NER as a sequential classification problem. Hence, in both corpora we use the *IOB2* tagging style where: *O*, means that the word is not a NE; *B-X* is used for the leftmost word of a NE type *X*; and *I-X* means that the word is inside of a NE type *X*. The *IOB2* tagging style is illustrated in the following example.

```
Wolff/B-PER ,/O currently/O a/O
journalist/O in/O Argentina/B-LOC ,/O
played/O with/O Del/B-PER Bosque/I-PER
in/O the/O final/O years/O of/O the/O
seventies/O in/O Real/B-ORG
Madrid/I-ORG
```

3.3 Model Setup

In most of our experiments, we use the same hyperparameters used by dos Santos and Zadrozny (2014) for part-of-speech tagging. The only exception is the learning rate for SPA CoNLL-2002, which we set to 0.005 in order to avoid divergence. The hyperparameter values are presented in Table 2. We use the development sets to determine the number of training epochs, which is six for HAREM and sixteen for SPA CoNLL-2002.

We compare CharWNN with two similar neural network architectures: CharNN and WNN. CharNN is equivalent to CharWNN without word embeddings, i.e., it uses character-level embeddings only. WNN is equivalent to CharWNN without character-level embeddings, i.e., it uses word embeddings only. Additionally, in the same way as in (Collobert et al., 2011), we check the impact of adding to WNN two handcrafted features that contain character-level information, namely capitalization and suffix. The capitalization feature has five possible values: all lowercased, first uppercased, all uppercased, contains an uppercased letter, and all other cases. We use suffix of size three. In our experiments, both capitalization and suffix embeddings have dimension five. The hyperparameters values for these two NNs are shown in Table 2.

4 Experimental Results

4.1 Results for Spanish NER

In Table 3, we report the performance of different NNs for the SPA CoNLL-2002 corpus. All results for this corpus were computed using the CoNLL-2002 evaluation script[4]. CharWNN achieves the best precision, recall and F1 in both development and test sets. For the test set, the F1 of CharWNN is 3 points larger than the F1 of the WNN that uses two additional handcrafted features: suffixes and capitalization. This result suggests that, for the NER task, the character-level embeddings are as or more effective as the two character-level features used in WNN. Similar results were obtained by dos Santos and Zadrozny (2014) in the POS tagging task.

In the two last lines of Table 3 we can see the results of using word embeddings and character-level embeddings separately. Both, WNN that uses word embeddings only and CharNN, do not achieve results competitive with the results of the networks that jointly use word-level and character-level information. This is not surprising, since it is already known in the NLP community that jointly using word-level and character-level features is important to perform named entity recognition.

In Table 4, we compare CharWNN results with the ones of a state-of-the-art system for the SPA CoNLL-2002 Corpus. This system was trained using AdaBoost and is described in (Carreras et al., 2002). It employs decision trees as a base learner

[4] http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt

Table 2: Neural Network Hyperparameters.

Parameter	Parameter Name	CharWNN	WNN	CharNN
d^{wrd}	Word embedding dimensions	100	100	-
k^{wrd}	Word context window size	5	5	5
d^{chr}	Char. embedding dimensions	10	-	50
k^{chr}	Char. context window size	5	-	5
cl_u	Convolutional units	50	-	200
hl_u	Hidden units	300	300	300
λ	Learning rate	0.0075	0.0075	0.0075

Table 3: Comparison of different NNs for the SPA CoNLL-2002 corpus.

NN	Features	Dev. Set			Test Set		
		Prec.	Rec.	F1	Prec.	Rec.	F1
CharWNN	word emb., char emb.	**80.13**	**78.68**	**79.40**	**82.21**	**82.21**	**82.21**
WNN	word emb., suffix, capit.	78.33	76.31	77.30	79.64	78.67	79.15
WNN	word embeddings	73.87	68.45	71.06	73.77	68.19	70.87
CharNN	char embeddings	53.86	51.40	52.60	61.13	59.03	60.06

and uses handcrafted features as input. Among others, these features include gazetteers with people names and geographical location names. The AdaBoost based system divide the NER task into two intermediate sub-tasks: NE identification and NE classification. In the first sub-task, the system identifies NE candidates. In the second sub-task, the system classifies the identified candidates. In Table 4, we can see that even using only automatically learned features, CharWNN achieves state-of-the-art results for the SPA CoNLL-2002.

4.2 Results for Portuguese NER

In Table 5, we report the performance of different NNs for the HAREM I corpus. The results in this table were computed using the CoNLL-2002 evaluation script. We report results in two scenarios: total and selective. In the *total* scenario, all ten categories are taken into account when scoring the systems. In the *selective* scenario, only five chosen categories (Person, Organization, Location, Date and Value) are taken into account. We can see in Table 5, that CharWNN and WNN that uses two additional handcrafted features have similar results. We think that by increasing the training data, CharWNN has the potential to learn better character embeddings and outperform WNN, like happens in the SPA CoNLL-2002 corpus, which is larger than the HAREM I corpus. Again, CharNN and WNN that uses word embeddings only, do not achieve results competitive with the results of the networks that jointly use word-level and character-level information.

In order to compare CharWNN results with the one of the state-of-the-art system, we report in tables 6 and 7 the precision, recall, and F1 scores computed with the evaluation scripts from the HAREM I competition[5] (Santos and Cardoso, 2007), which uses a scoring strategy different from the CoNLL-2002 evaluation script.

In Table 6, we compare CharWNN results with the ones of ETL$_{CMT}$, a state-of-the-art system for the HAREM I Corpus (dos Santos and Milidiú, 2012). ETL$_{CMT}$ is an ensemble method that uses Entropy Guided Transformation Learning (ETL) as the base learner. The ETL$_{CMT}$ system uses handcrafted features like gazetteers and dictionaries as well as the output of other NLP tasks such as POS tagging and noun phrase (NP) chunking. As we can see in Table 6, CharWNN outperforms the state-of-the-art system by a large margin in both total and selective scenarios.

In Table 7, we compare CharWNN results by entity type with the ones of ETL$_{CMT}$. These results were computed in the selective scenario. CharWNN produces a much better recall than ETL$_{CMT}$ for the classes LOC, PER and ORG. For the ORG entity, the improvement is of 21 points in the recall. We believe that a large part of this boost in the recall is due to the unsupervised pre-

[5]http://www.linguateca.pt/primeiroHAREM/harem_Arquitectura.html

Table 4: Comparison with the state-of-the-art for the SPA CoNLL-2002 corpus.

System	Features	Prec.	Rec.	F1
CharWNN	word embeddings, char embeddings	**82.21**	**82.21**	**82.21**
AdaBoost	words, ortographic, POS tags, trigger words, bag-of-words, gazetteers, word suffixes, word type patterns, entity length	81.38	81.40	81.39

Table 5: Comparison of different NNs for the HAREM I corpus.

NN	Features	Total Scenario			Selective Scenario		
		Prec.	Rec.	F1	Prec.	Rec.	F1
CharWNN	word emb., char emb.	67.16	**63.74**	65.41	73.98	**68.68**	71.23
WNN	word emb., suffix, capit.	**68.52**	63.16	**65.73**	**75.05**	68.35	**71.54**
WNN	word embeddings	63.32	53.23	57.84	68.91	58.77	63.44
CharNN	char embeddings	57.10	50.65	53.68	66.30	54.54	59.85

Table 6: Comparison with the State-of-the-art for the HAREM I corpus.

System	Features	Total Scenario			Selective Scenario		
		Prec.	Rec.	F1	Prec.	Rec.	F1
CharWNN	word emb., char emb.	74.54	**68.53**	**71.41**	**78.38**	**77.49**	**77.93**
ETL_{CMT}	words, POS tags, NP tags, capitalization, word length, dictionaries, gazetteers	**77.52**	53.86	63.56	77.27	65.20	70.72

training of word embeddings, which can leverage large amounts of unlabeled data to produce reliable word representations.

4.3 Impact of unsupervised pre-training of word embeddings

In Table 8 we assess the impact of unsupervised pre-training of word embeddings in CharWNN performance for both SPA CoNLL-2002 and HAREM I (selective). The results were computed using the CoNLL-2002 evaluation script. When unsupervised pre-training is not used, the word embeddings are initialized by randomly sampling each value from an uniform distribution: $\mathcal{U}(-r, r)$, where $r = \sqrt{\frac{6}{|V^{wrd}| + d^{wrd}}}$.

We can see in Table 8 that, for both corpora, CharWNN results are improved when using unsupervised pre-training. The impact of unsupervised pre-training is larger for the HAREM I corpus (13.2 points in the F1) than for the SPA CoNLL-2002 (4.3 points in the F1). We believe one of the main reasons of this difference in the impact is the training set size, which is much smaller in the HAREM I corpus.

5 Related Work

Some recent work on deep learning for named entity recognition include Chen et al. (2010), Collobert et al. (2011) and Passos et al. (2014).

Chen et al. (2010) employ deep belief networks (DBN) to perform named entity categorization. In their system, they assume that the boundaries of all the entity mentions were previously identified, which makes their task easier than the one we tackle in this paper. The input for their model is the character-level information of the entity to be classified. They apply their system for a Chinese corpus and achieve state-of-the-art results for the NE categorization task.

Collobert et al. (2011) propose a deep neural network which is equivalent to the WNN architecture described in Section 3.3. They achieve state-of-the-art results for English NER by adding a feature based on gazetteer information.

Passos et al. (2014) extend the Skip-Gram language model (Mikolov et al., 2013) to produce *phrase embeddings* that are more suitable to be used in a linear-chain CRF to perform NER. Their linear-chain CRF, which also uses additional handcrafted features such as gazetteer

Table 7: Results by entity type for the HAREM I corpus.

Entity	CharWNN			ETL$_{CMT}$		
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**
DATE	90.27	81.32	85.56	88.29	82.21	85.14
LOC	76.91	78.55	77.72	76.18	68.16	71.95
ORG	70.65	71.56	71.10	65.34	50.29	56.84
PER	81.35	77.07	79.15	81.49	61.14	69.87
VALUE	78.08	74.99	76.51	77.72	70.13	73.73
Overall	78.38	77.49	77.93	77.27	65.20	70.72

Table 8: Impact of unsup. pre-training of word emb. in CharWNN performance.

Corpus	Pre-trained word emb.	**Precision**	**Recall**	**F1**
SPA CoNLL-2002	Yes	**82.21**	**82.21**	**82.21**
	No	78.21	77.63	77.92
HAREM I	Yes	**73.98**	**68.68**	**71.23**
	No	65.21	52.27	58.03

based, achieves state-of-the-art results on two English corpora: CoNLL 2003 and Ontonotes NER.

The main difference between our approach and the ones proposed in previous work is the use of neural character embeddings. This type of embedding allows us to achieve state-of-the-art results for the full task of identifying and classifying named entities using only features automatically learned. Additionally, we perform experiments with two different languages, while previous work focused in one language.

6 Conclusions

In this work we approach language-independent NER using a DNN that employs word- and character-level embeddings to perform sequential classification. We demonstrate that the same DNN which was successfully applied for POS tagging can also achieve state-of-the-art results for NER, using the same hyperparameters, and without any handcrafted features. Moreover, we shed some light on the contribution of neural character embeddings for NER; and define new state-of-the-art results for two NER corpora in two different languages: Portuguese and Spanish.

Acknowledgments

Victor Guimarães' contributions were made during an internship at IBM Research.

References

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*.

Xavier Carreras, Lluís Màrques, and Lluís Padró. 2002. Named entity extraction using adaboost. In *Proceedings of CoNLL-2002*, pages 167–170. Taipei, Taiwan.

Yu Chen, You Ouyang, Wenjie Li, Dequan Zheng, and Tiejun Zhao. 2010. Using deep belief nets for chinese named entity categorization. In *Proceedings of the Named Entities Workshop*, pages 102–109.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

R. Collobert. 2011. Deep learning for efficient discriminative parsing. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 224–232.

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ace) program tasks, data, and evaluation. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC-2004)*, Lisbon, Portugal, May.

Cícero Nogueira dos Santos and Ruy Luiz Milidiú, 2012. *Entropy Guided Transformation Learning: Algorithms and Applications*, chapter Named entity

recognition, pages 51–58. Springer Briefs in Computer Science. Springer.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014a. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning, JMLR: W&CP volume 32*, Beijing, China.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014b. Training state-of-the-art portuguese POS taggers without handcrafted features. In *Proceedings of the 11th International Conference Computational Processing of the Portuguese Language*, pages 82–93, São Carlos, Brazil.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370.

Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of Workshop at International Conference on Learning Representations*.

Ruy Luiz Milidiú, Julio Cesar Duarte, and Roberto Cavalcante. 2007. Machine learning algorithms for portuguese named entity recognition. *Revista Iberoamericana de Inteligencia Artificial*, pages 67–75.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 78–86, Ann Arbor, Michigan.

Diana Santos and Nuno Cardoso. 2007. *Reconhecimento de entidades mencionadas em português*. Linguateca, Portugal.

Buzhou Tang, Hongxin Cao, Xiaolong Wang, Qingcai Chen, and Hua Xu. 2014. Evaluating word representation features in biomedical named entity recognition tasks. *BioMed Research International*, 2014.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Erik F. Tjong Kim Sang. 2002. Introduction to the conll-2002 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2002*, pages 155–158. Taipei, Taiwan.

A. J. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, April.

A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang. 1989. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(3):328–339.