# Applying Deep Neural Networks to Named Entity Recognition in Portuguese Texts

Ivo Fernandes[1], Henrique Lopes Cardoso[2] and Eugenio Oliveira[2]
[1] Faculdade de Engenharia, Universidade do Porto
Rua Dr. Roberto Frias 4200-465, Porto, Portugal
Email: up201303199@fe.up.pt
[2] LIACC, DEI, Faculdade de Engenharia, Universidade do Porto
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
Email: {hlc,eco}@fe.up.pt

*Abstract*—**There is currently few research in using deep learning (DL) applied to Named Entities Recognition (NER) in Portuguese texts. This work exposes some challenges and limitations but also the benefits of applying DL architectures to NER in Portuguese. Four different DL architectures are applied to Portuguese datasets. All architectures are heavily influenced by previous published work in NER applied to English. Annotated data is used to train and test NER models, while non-annotated data is used to train word embeddings, as well as being a key part of a bootstrapping approach, where raw textual data is used to create NER models.**

## 1. INTRODUCTION

Natural Language Processing (NLP) is a field of computer science that intersects artificial intelligence and computational linguistics. NLP is focused on creating ways for computers to process large natural language corpora. To fully process, extract knowledge from, and understand the content of corpora, scientists and researchers have subdivided this macro problem into smaller subtasks with specific goals.

Named Entity Recognition (NER) is one of those subtasks and aims at identifying and classifying entity mentions in free text. Research in NER applied to the Portuguese language started with the HAREM contest in 2004 (Santos et al., 2006). This contest received submissions from multiple countries and is recognized as being the first evaluation contest for named entity recognition in Portuguese.

Current focus of research in NER is the application of deep learning (DL) methods, but this is only true for languages with big research communities, such as English. More research interest means there is more annotated corpora available, which is a requirement for all supervised machine learning algorithms; DL approaches, in particular, benefit the most from corpora with a larger size (Goodfellow et al., 2016). Looking back at recent research on NER in Portuguese texts, it is still hard to find DL approaches, with just one particular instance standing out – the work of dos Santos and Guimarães, 2015.

DL techniques are very appealing since they tend to avoid hand-crafted features, something needed for previous machine learning techniques. When provided with enough data, DL methods are capable of automatically identifying relevant features, leading to good performance without external resources or time-intensive feature engineering.

This work focuses on applying multiple DL architectures to NER in Portuguese. A total of four different architectures are tested and compared with previous work. Section 2 reviews several developments on NER. Section 3 describes DL architectures used both for NER and for training word embeddings. In Section 4 experimental results are presented and discussed. Section 5 introduces the bootstrapping approach and its experimental evaluation. Section 6 concludes.

## 2. RELATED WORK

The term *named entity* was first introduced at the $6^{th}$ Message Understanding Conference (MUC-6) back in 1996 (Grishman and Sundheim, 1996), and the task of *named entity recognition* is defined as identifying and classifying named entity mentions in free texts, taking into consideration a predefined set of categories. For the English language, a common dataset used to test models and compare results is the CoNLL-2003 dataset, created in the scope of the CoNLL-2003 shared task, which was focused on NER. As for the Portuguese language, the standard datasets used for testing and comparing systems are the HAREM GC datasets (Freitas et al., 2010; Santos et al., 2006).

The great appeal of approaches based on DL is the possibility of discarding the need of feature engineering (Chiu and Nichols, 2015). A DL model is capable of learning what features to detect and using these features to identify and classify named entities into predefined categories.

The versatility and power of DL architectures, combined with advancements in hardware, pushed DL to many computer science areas, including NLP and more specifically NER. The last few years of research in the English language has mainly focused on DL approaches (Lample et al., 2016; Sutskever et al., 2014; Yang et al., 2017; dos Santos and Guimarães, 2015; Chiu and Nichols, 2015; Namazifar, 2017; Collobert et al., 2011; Collobert and Weston, 2008; Huang et al., 2015; Socher and Manning, 2013).

978-1-5386-9588-3/18/$31.00 ©2018 IEEE

This work explores the approaches of Collobert et al., 2011, Chiu and Nichols, 2015, Lample et al., 2016 and Ma and Hovy, 2016. All architectures were, when proposed, regarded as the state of the art for NER in English. In this work these architectures are applied to Portuguese texts.

# 3. METHODOLOGY

We here provide details about the data, DL NER architectures and word embedding models explored in this work. Since all models are sourced from previous research on NER in English texts, we limit ourselves to provide brief descriptions and point the reader to the appropriate references, where low level implementation details can be found.

## 3.1. Data

Two different dataset categories are explored: annotated and non-annotated datasets. Annotated datasets are used to train and test the different DL models, while non-annotated data is used to train word embeddings.

The datasets used in this work are originally distributed in different formats, and were transformed into the CoNLL-2003 format (Sang and De Meulder, 2003) before being used to train the models. To do so, it is necessary to first tokenize the text into sentences and then each sentence into words. The NLTK (Loper and Bird, 2002) Portuguese word and sentence tokenizers were used.

**3.1.1. Annotated Data.** Following the same approach as Santos *et.al.* (dos Santos and Guimarães, 2015), the first HAREM I GC ($HAREM_{first}$) dataset is split into train set and development (or validation) set and the miniHAREM GC ($HAREM_{mini}$) is used as the test set for the performed experiments. The development set contains the last 5% of the $HAREM_{first}$ dataset. When referring to the train or development subsections of the datasets, we make use of subscripts *dev* or *train*. Both $HAREM_{first}$ and $HAREM_{mini}$ datasets have a total of 10 different named entity categories.

In addition to these original datasets, some others were derived from the originals: $HAREM_{first\_selective}$ and *HAREMmini_selective*. These derived datasets only includes 4 named entity categories: *organization* (*ORG*), *abstraction* (*MISC*), *location* (*LOC*) and *person* (*PER*). The derived datasets created so that we can assess the impact of the number of different named entity categories on the performance of the models. Two scenarios are created: a *Complete* scenario where models are trained and tested using the original datasets ($HAREM_{first}$) and $HAREM_{mini}$) with 10 named entity categories and the *Selective* scenario where models are trained and tested using the derived datasets ($HAREM_{first\_selective}$ and $HAREM_{mini\_selective}$) with only 4 named entity categories.

Detailed statistics about the annotated datasets used in this work are available in Table 1.

TABLE 1. ANNOTATED DATASETS.

| Dataset | Tokens | Entities |
|---|---|---|
| $HAREM_{first}$ | 92 228 | 4 972 |
| $HAREM_{first\_train}$ | 87 594 | 4 805 |
| $HAREM_{first\_dev}$ | 4 634 | 167 |
| $HAREM_{first\_selective}$ | 92 228 | 3 578 |
| $HAREM_{first\_selective\_train}$ | 87 594 | 3 458 |
| $HAREM_{first\_selective\_dev}$ | 4 634 | 120 |
| $HAREM_{mini}$ | 62 440 | 3 624 |
| $HAREM_{mini\_selective}$ | 62 440 | 2 507 |

**3.1.2. Raw Data.** To obtain pre-trained word embeddings, large amounts of raw textual data are required. The raw data used in this work is sourced from Portuguese Wikipedia articles, obtained from a dump of April 1, 2018.

An adapted version of the Perl script written by [1] was used to process the Wikipedia dump, obtaining a text file to train embeddings. This parser excludes all meta-data and structure but also transforms all words to lower case and all digits into their word representation. Despite excluding images and links, all captions are preserved. A total of 892 834 Wikipedia articles were parsed to obtain a corpus with a total of 422 024 462 tokens.

## 3.2. Models

In this paper, the architectures explored are named based on their characteristics. These are not the original names given to the architectures by their authors. Original names were replaced by more expressive names that highlight the details of the architecture. For all networks, the learning algorithm used was mini-batch stochastic gradient descent. For all experiments the mini-batch size used was 16.

The sentence-level log-likelihood score function is used for all networks. The sentence-level log-likelihood score function requires additional network parameters: a transition score $[A]_{i,j}$ for jumping from $i$ to $j$ tags in successive words; initial score $[A]_{i,0}$ for starting from the $i^{th}$ tag. The number of operations needed to calculate this sentence-level cost function grows exponentially with sentence length, as all the combinations of tag sequences need to be tested. However, it is possible to compute it in linear time (Collobert et al., 2011). During training, the objective is to maximize the log-likelihood. When testing, given a sentence $[x]_1^T$ to process, it is necessary to find the path that minimizes the sentence score. The Viterbi algorithm (Forney, 1973) is used.

**3.2.1. $Window_{Network}$.** The $Window_{Network}$ was introduced by Collobert et al., 2011, this architecture has shifted the focus of NLP tasks to neural networks and DL. From all tested architectures, this is the only feed-forward neural network (all others are recurrent neural networks). Extra features used in combination with the word embeddings include capitalization and suffix.

[1] http://mattmahoney.net/dc/textdata.html

**3.2.2.** $BiLSTM_{Network}$**.** The $BiLSTM_{Network}$ used in the experiments follows the same structure as the one presented by Chiu and Nichols, 2015, but discards CNN-extracted character features. Character-level word representations are not included to provide a fair comparison with the feed-forward neural network ($Window_{Network}$), which has no character-level word representation.

**3.2.3.** $BiLSTMChar_{Network}$**.** The $BiLSTMChar_{Network}$ was introduced in the work of Lample et al., 2016 and just as the $BiLSTM_{Network}$ it uses a one layer bidirectional LSTM network. In order to improve word representation, character-based representations are created and used side by side with word embeddings. The character representation of a word in the $BiLSTMChar_{Network}$ is created by feeding a word char-by-char to a Bidirectional LSTM and at the end concatenating the outputs of both the forward and backward LSTM. The code used to run the experiments was made available by the authors (Lample, 2016).

**3.2.4.** $BiLSTM\_CNN_{Network}$**.** The $BiLSTM\_CNN_{Network}$ was introduced in the work of Ma and Hovy, 2016, and its architecture includes a bidirectional LSTM to create a sequence-to-sequence model, a CNN to extract character-level features from words and a CRF layer to jointly decode the best chain of NER tags for a given sentence. To create character representations, Ma *et.al.* make use of a simple CNN. The authors made available their implementation (Ma, 2017), written with the Pytorch DL framework. All experiments involving this network were ran using this implementation. Gradient clipping and variable learning rate are used for this network.

## 3.3. Pre-trained Embeddings

For this work, word embeddings were trained from scratch using Portuguese Wikipedia data. Two different embedding training architectures were used: *word*2*vec* (Mikolov et al., 2013) and *wang*2*vec* (Ling et al., 2015), a slight modification of *word*2*vec*. The work of Hartmann et al., 2017, which focuses on evaluating different embedding models for Portuguese, highlighted the *wang*2*vec* structured *skipngram* embedding model as being the best performing in extrinsic evaluation for the tasks of part of speech tagging and semantic similarity.

Embeddings were trained using the original implementations of *word*2*vec* [2] and *wang*2*vec* [3]. A total of 3 different word embeddings were trained all with a vocabulary of around 620 000 words and using a window size of 8. For the *wang*2*vec* architecture, two embeddings were trained: one with dimension 64 ($wang2vec_{64D}$) and another with dimension 100 ($wang2vec_{100D}$). *word*2*vec* embeddings were trained with dimension 100 ($word2vec_{100D}$). Pre-trained word embeddings are used to initialize the different networks during the experiments, see Table 2.

# 4. EXPERIMENTAL RESULTS

In order to obtain results in a timely manner hyper-parameters of the architectures were set to the hyper-parameters reported in the paper for that specific architecture. The reported hyper-parameters are tuned for the dataset where they have been tested, typically datasets in the English language, and there is no guarantee of optimality when used with different datasets or different languages.

For all experimental results, the models were trained and tested using two combinations of datasets. The *Complete* combination uses the $HAREM_{first\_train}$, $HAREM_{first\_dev}$ and $HAREM_{mini}$ datasets. The *Selective* combination uses the $HAREM_{first\_selective\_train}$, $HAREM_{first\_selective\_dev}$ and $HAREM_{mini\_selective}$ datasets. All metrics are obtained using the official CoNLL-2003 (Sang and De Meulder, 2003) evaluation script. All models are trained on a machine with an Intel(R) Core(TM) i7-3770K CPU @ 3.50GHz processor, 32 GB of RAM and a GeForce GTX 1080 (8GB).

Analysing the obtained F1 measures (Table 2) for the various experiments, some interesting observations can be extracted. A more substantial difference between the top performing models on the *Complete* dataset combination and the *Selective* dataset combination was expected. The *Complete* dataset combination has a total of 10 different named entity categories, while the *Selective* dataset combination has only 4 different named entity categories. The real difference between the best performing models for the two dataset combinations is under 1% of F1.

Just like previous published work (Collobert et al., 2011; Chiu and Nichols, 2015; Lample et al., 2016; Huang et al., 2015; Liu et al., 2011; dos Santos and Guimarães, 2015; Socher and Manning, 2013; Yang et al., 2017; Derczynski et al., 2015; Nothman et al., 2013; Ma and Hovy, 2016), major improvements were observed when using pre-trained embeddings to initialize the models. This was verified for both scenarios: improvements of up to 18.07% in F1 were observed for $BiLSTMChar_{Network}$.

The $CompleteWindow_{Network}$ was the only feed-forward neural network architecture. All others include a sequence to sequence model based on RNNs. Looking at Table 2 we can observe that the RNN architectures perform significantly better than the feed-forward neural networks. This observation matches the latests developments in NLP where state of the art systems for many tasks were improved after using recurrent neural network architectures to train new models.

All the architectures described and tested performed much better in the English language, in some cases showing differences in F1 score of up to 23%. The $BiLSTMChar_{Network}$ and $BiLSTM\_CNN_{Network}$ both have F1 scores above 90.0% (Lample et al., 2016; Ma and Hovy, 2016) for the NER task in the CoNLL-2003 English dataset while in the *Complete* scenario neither one gets scores above 70% F1. These abrupt differences in performance between the two languages can be due to a multitude of factors: the inherent differences between the two languages; the quality of the pre-trained word embeddings used to initialize the

[2] https://github.com/dav/word2vec

[3] https://github.com/wlin12/wang2vec

TABLE 2. EXPERIMENT RESULTS FOR THE *Complete* SCENARIO AND THE *Selective* SCENARIO.

| **Model** | **Embeddings** | *Complete* | | | *Selective* | | |
|---|---|---|---|---|---|---|---|
| | | F1 | Precision | Recall | F1 | Precision | Recall |
| $CompleteWindow_{Network}$ | $Wang2Vec_{64D}$ | 43.26 | 42.40 | 44.15 | 47.13 | 52.81 | 42.55 |
| $BiLSTM_{Network}$ | $Wang2Vec_{64D}$ | 59.61 | 64.86 | 55.16 | 61.64 | 65.00 | 43.70 |
| | $word2vec_{100D}$ | 55.05 | 62.16 | 49.40 | 46.54 | 57.22 | 39.22 |
| $BiLSTMChar_{Network}$ | — | 54.86 | 57.70 | 52.29 | 52.08 | 54.31 | 50.02 |
| | $word2vec_{100D}$ | 67.02 | 68.31 | 65.78 | **70.15** | **71.90** | 68.49 |
| | $Wang2Vec_{64D}$ | 67.35 | 68.94 | 65.84 | 69.50 | 70.12 | **68.89** |
| $BiLSTM\_CNN_{Network}$ | $word2vec_{100D}$ | 68.52 | 71.57 | 65.71 | 49.04 | 54.18 | 44.79 |
| | $Wang2Vec_{64D}$ | **69.97** | **72.64** | **67.50** | 68.44 | 70.67 | 66.35 |
| | $Wang2Vec_{100D}$ | 53.72 | 64.88 | 45.83 | 11.70 | 19.36 | 8.38 |

TABLE 3. DATASET STATISTICS.

| Variable | CoNLL-2003 | HAREM I |
|---|---|---|
| Number tokens | 204 567 | 92 228 |
| Number sentences | 14 987 | 3 682 |
| Avg tokens per sent. | 13.65 | 25.05 |
| Categories | 4 | 10 |

models; the hyper-parameters used; or other characteristics of available training datasets.

To better understand the difference in performance between a model trained using English datasets and the same model trained using Portuguese datasets, a comparison between the datasets was performed. In Table 3 it is possible to observe some clear differences between the datasets, namely, total size in terms of tokens and the average sentence length.

Looking strictly at the number of tokens in each dataset, the HAREM I GC seems to be almost half the size of the CoNLL-2003 English dataset. However, the models explored are trained on a sentence level, which means that the true number of training examples is the number of sentences in the dataset. In terms of number of sentences the HAREM I GC has only approximately 25% of the number of sentences present in the CoNLL-2003 English dataset. This change in relative size is due to the fact that the HAREM I GC has an average number of tokens per sentence much higher than the CoNLL-2003 English dataset. Longer sentences mean that more tokens are processed before making a tag prediction. This increase in the number of tokens consumed by the model before prediction might be one of the reasons behind the drop in performance.

Another important difference that most certainly has impact in model performance is the quality of the pre-trained word embeddings used. Large amounts of raw textual data are essential to obtain good quality embeddings. All the pre-trained word embeddings used in this work were trained with a total of around 422 million tokens, while pre-trained word embeddings for the English language are trained in billions of tokens. The publicly available GloVe (Pennington et al., 2014) word embeddings for the English language were trained in a total of 6 billion tokens.

# 5. BOOTSTRAPPING

Annotated datasets are scarce and hard to obtain for most languages. Teixeira et al., 2011 concluded that HAREM datasets are not adequate to be used on an up-to-date NER system due to the age of the articles that compose the datasets. On the other hand, non annotated data or raw text is for the most part freely available with virtually no cost and a constant stream of fresh data. The bootstrapping approach can exploit raw text to train NER models. As explained in Teixeira et al., 2011, bootstrapping consists on:

1) Identifying named entities in an unannotated corpus using a dictionary-based approach;
2) Training a model with the newly annotated corpus;
3) Testing the trained model on an external annotated corpus;
4) Re-annotating the corpus using the model;
5) Repeat until performance measures drop.

The stopping condition is the performance measures drop, any further iterations do not improve the model and the new names detected become false positives, leading to a model that evolves to be worse and worse over time.

Usually, NER focuses on multiple entity classes. However, to obtain the needed initial list of annotations for bootstrapping it is necessary to identify a pattern that is followed by entities of that specific type. Identifying patterns that match enough entities with 100% precision is not trivial. Just like Teixeira *et al.*, for these bootstrapping experiments only entities of the type *person* (*PER*) are considered.

## 5.1. Data

Two datasets were gathered to be used in bootstrapping experiments: one based on Portuguese news articles, just like the work by Teixeira *et.al.*; the other based on the Portuguese Wikipedia dump of April 1, 2018.

In order to test the NER models produced using bootstrapping, annotated datasets are required. Two annotated datasets are used: *WikiNER* (Nothman et al., 2013) and $HAREM_{second}$ (Freitas et al., 2010). Both datasets were processed to remove all named entity annotation except for named entities of category *person*.

A very small annotated dataset, $News_{test}$, was produced to evaluate bootstrapping experiments, composed of 15 news articles from March of 2018 and containing only named entities of category *person*. It was created with the intention of assessing the factor of dataset age from the performance measures of models trained with the bootstrapping method.

To obtain the raw textual data from the Wikipedia dump file, Wikiextractor [4] was used. Only the first 7 million tokens are part of the training data used for bootstrapping; this constraint was necessary due to GPU memory restrictions.

The news dataset is composed of articles from Portuguese news websites, published between September 2017 and April 2018 and fetched using an API provided by SAPO Labs [5]. The dataset contains 57 000 news articles but for the bootstrapping experiments only a subset of the first 15 100 news articles is used as the training set; just like for the Wikipedia data this is done so that the training set fits into GPU memory.

## 5.2. Experimental Setup

The strategy to obtain the initial name list is the same for both datasets: identify a common pattern in which names of people appear with 100% precision, model a regular expression to capture the name of the person and finally extract all names from the text.

The pattern identified is vaguely the same for both news and Wikipedia data, and follows a structure of $\langle[CapitalizedSequence],[ergonym]\rangle$ just like in Teixeira et al., 2011. Some examples of words present in the ergonym list are: *presidente*, *jogador*, *vocalista*, *pai*, *marido*, *mulher*.

This pattern proved to work fine for news articles. However, to obtain the initial name list from Wikipedia text it was necessary to further restrict the pattern. Preceding words are introduced into the pattern, modifying the initial name pattern for Wikipedia to: $\langle[PrecedingWord]$ $[CapitalizedSequence],[ergonym]\rangle$. Preceding words include mostly words that refer to the nationality of the person in question, such as *brasileiro*, *chileno*, *argentino*, *inglesa*. In order to fully guarantee that only correct names are present in the initial name list, some further conditions were added:

1) The capitalized sequence length must be at least 2, that is, single word names are excluded.
2) For a name to be included in the initial name list it must occur at least $N$ times in the name extraction dataset. We used $N = 3$ for the *NewsExperiment* and $N = 2$ for the *WikipediaExperiment*.

After obtaining the initial name list it is necessary to annotate the training dataset, which consists of raw textual data with no annotations. The initial name list is sorted by length so that names with the most number of words are processed first.

After some initial experimentation with different networks, $BiLSTM\_CNN_{Network}$ was chosen for the bootstrapping experiments. Further details on the $BiLSTM\_CNN_{Network}$ architecture are available in Section 3.2.4. The PyTorch implementation of Ma[6] was modified to perform bootstrapping. The word embeddings used to initialize the network were $wang2vec_{64D}$.

TABLE 4. BOOTSTRAPPING RESULTS FOR THE BEST MODELS FOR ALL BOOTSTRAPPING ITERATIONS. *NewsExperiment* AND THE WORK OF TEIXEIRA *et.al.* IS TESTED WITH THE $HAREM_{second\_per}$ DATASET, WHILE *WikipediaExperiment* IS TESTED WITH THE $WikiNER_{per}$ DATASET.

| Experiment | Iter | F1 | Prec | Rec | Avg. new names |
|---|---|---|---|---|---|
| *NewsExperiment* | 29 | 25.3 | 71.23 | 15.38 | 153 |
| *WikipediaExperiment* | 20 | 0.49 | 5.59 | 0.26 | 1.3 |
| Teixeira *et.al.* | 12 | 69 | 90 | 56 | 233 |

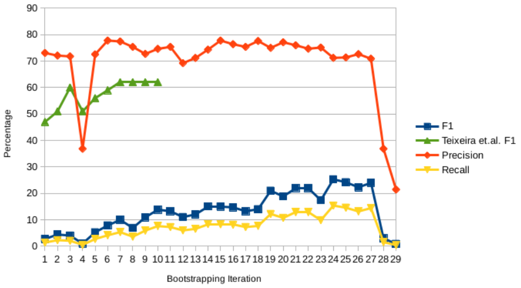

Figure 1. $HAREM_{second}$ test scores for the *NewsExperiment* setup compared to the work of Teixeira et al., 2011.

Analyzing the performance results shown in Table 4, it is clear that the bootstrapping process is not fit to handle Wikipedia articles, at least using the same parameters as the ones used for the *NewsExperiment*. The reasons behind the poor results in the *WikipediaExperiment* could be related to the textual genre, as Portuguese Wikipedia articles contain more foreign names of people than Portuguese media articles and the context in which names appear within the article is different in the two textual genres. Furthermore, the test set for the *WikipediaExperiment*, $WikiNER_{per}$, is a much larger dataset than the test set used for the *NewsExperiment*.

Since the results from the *WikipediaExperiment* only proved that the described bootstrapping process is not compatible with that textual genre, the discussion will focus on the results of the *NewsExperiment*.

It is possible to view the evolution of the performance scores in the $HAREM_{second}$ dataset for each of the bootstrapping iterations in Figure 1, performance on the $News_{test}$ dataset is roughly 10% higher in terms of F1 score in all iterations. Despite starting with a much lower number of initial names and the training set being smaller than the one used in the work of Teixeira *et.al.*, the number of total names in the *NewsExperiment* after finishing the bootstrapping process is similar (around 5000 names).

Looking at Figure 1 it is clear that the difference in the F1 score is due to differences in recall and not in precision. Precision values are very similar for both the $HAREM_{second}$ and $News_{test}$ datasets. The observed difference in recall may be due to the age difference between the train and test datasets. $HAREM_{second}$ is a collection of texts from the late 90s, while the $News_{test}$ dataset is made up of a small number of articles from the same time frame as the train data. The performance measures for both test datasets are correlated and vary in the same way throughout the bootstrapping iterations. A drop in performance also means less new names

[4] https://github.com/attardi/wikiextractor
[5] http://labs.sapo.pt/
[6] https://github.com/XuezheMax/NeuroNLP2

are discovered at that iteration.

Comparing the results reported by Teixeira *et.al.* with the results obtained in the *NewsExperiment*, it seems that using DL in combination with bootstrapping does not provide good results. However, the work of Teixeira *et.al.* can not be directly compared with the *NewsExperiment*. Teixeira *et.al.* used larger datasets from a different time frame, the preprocessing of the datasets may have been done differently and the ergonym list used may differ greatly. These differences are a source of inconsistency in the experiment setup and prohibit a fair comparison of results.

## 6. CONCLUSIONS

State of the art DL architectures for NER proved to be adequate for Portuguese datasets. The drop in performance when compared to English can be attributed to multiple factors, as discussed in Section 4, but not to the architectures themselves or to the Portuguese language specifically.

There are multiple ways to improve and develop this work. Most improvements have to do with exploring more architectures for both the models and pre-trained embeddings but also including more data into the training process. Future work for NER in Portuguese should include the creation of a large annotated dataset so that state of the art models, such as DL architectures, can be fully explored. This would help Portuguese NER research to stay up to date with all NER developments to come. Creating an easy and free way to obtain large amounts of Portuguese textual data would benefit research on not only NER but Portuguese NLP in general. Large amounts of textual data are essential to train better word embeddings.

## REFERENCES

Chiu, J. P. C. and Nichols, E. (2015). Named Entity Recognition with Bidirectional LSTM-CNNs. *CoRR*, abs/1511.0.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing. In *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 160–167, New York, New York, USA. ACM Press.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural Language Processing (almost) from Scratch. *CoRR*, abs/1103.0.

Derczynski, L., Maynard, D., Rizzo, G., Van Erp, M., Gorrell, G., Troncy, R., Petrak, J., and Bontcheva, K. (2015). Analysis of named entity recognition and linking for tweets. *Information Processing and Management*, 51(2):32–49.

dos Santos, C. N. and Guimarães, V. (2015). Boosting Named Entity Recognition with Neural Character Embeddings. *CoRR*, abs/1505.0.

Forney, G. D. (1973). The Viterbi Algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Freitas, C., Mota, C., Santos, D., Oliveira, H. G., and Carvalho, P. (2010). Second HAREM : Advancing the State of the Art of Named Entity Recognition in Portuguese. In *Procs. 7th Int. Conf. Language Resources and Evaluation (LREC'10)*, number 3, pages 3630–3637.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*, volume 22. MIT Press.

Grishman, R. and Sundheim, B. (1996). Message Understanding Conference-6: A Brief History. In *Procs. 16th Conf. on Computational Linguistics*, volume 1, pages 466–471. ACL.

Hartmann, N., Fonseca, E., Shulby, C., Treviso, M., Rodrigues, J., and Aluisio, S. (2017). Portuguese Word Embeddings: Evaluating on Word Analogies and Natural Language Tasks. *CoRR*, abs/1708.0.

Huang, Z., Xu, W., and Yu, K. (2015). Bidirectional LSTM-CRF Models for Sequence Tagging. *CoRR*, abs/1508.0.

Lample, G. (2016). Named entity recognition tool source code. https://github.com/glample/tagger. Accessed: 2018-06-28.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural Architectures for Named Entity Recognition. *CoRR*, abs/1603.0.

Ling, W., Dyer, C., Black, A. W., and Trancoso, I. (2015). Two/Too Simple Adaptations of Word2Vec for Syntax Problems. In *Procs. 2015 Conf. of the North American Chapter of the ACL: Human Language Technologies*, pages 1299–1304.

Liu, X., Zhang, S., Wei, F., and Zhou, M. (2011). Recognizing Named Entities in Tweets. *In Procs. 48th Annual Meeting of the ACL*, 1(2008):359–367.

Loper, E. and Bird, S. (2002). Nltk: The natural language toolkit. In *Procs. ACL-02 Workshop on Effective Tools and Methodologies for Teaching NLP and Computational Linguistics - Volume 1*, pages 63–70, Stroudsburg, PA, USA. ACL.

Ma, X. (2017). Deep neural models for core nlp tasks source code. https://github.com/XuezheMax/NeuroNLP2. Accessed: 2018-06-28.

Ma, X. and Hovy, E. (2016). End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. *CoRR*, abs/1603.0.

Mikolov, T., Corrado, G., Chen, K., and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *CoRR*, abs/1301.3.

Namazifar, M. (2017). Named Entity Sequence Classification. *ArXiv e-prints*.

Nothman, J., Ringland, N., Radford, W., Murphy, T., and Curran, J. R. (2013). Learning multilingual named entity recognition from Wikipedia. *Artificial Intelligence*, 194:151–175.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global Vectors for Word Representation. In *Procs. EMNLP 2014*, pages 1532–1543.

Sang, E. F. T. K. and De Meulder, F. (2003). Introduction to the CoNLL-2003 Shared Task: Language-Independent NER. In *Procs. 7th Conf. on Natural language learning at HLT-NAACL 2003*, volume 4, pages 142–147. ACL.

Santos, D., Seco, N., Cardoso, N., and Vilela, R. (2006). HAREM: An Advanced NER Evaluation Contest for Portuguese. In *Procs. 5th Int. Conf. on Language Resources and Evaluation, LREC'2006*, pages 1986–1991.

Socher, R. and Manning, C. (2013). Deep Learning for NLP. *HLT-NAACL Tutorials*, pages 1–204.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *CoRR*, abs/1409.3.

Teixeira, J., Sarmento, L., and Oliveira, E. (2011). A bootstrapping approach for training a ner with conditional random fields. In *Progress in Artificial Intelligence*, LNCS 7026.

Yang, J., Zhang, Y., and Dong, F. (2017). Neural Reranking for Named Entity Recognition. *CoRR*, abs/1707.0.