UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

JULIO CESAR DE AZEREDO

Automated concatenation of embeddings for named-entity recognition in Portuguese

Work presented in partial fulfillment
of the requirements for the degree of
Bachelor in Computer Engineering

Advisor: Prof^{a}. Viviane P. Moreira

Porto Alegre
novembro 2021

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Reitor: Prof. Carlos André Bulhões
Vice-Reitora: Profª. Patricia Pranke
Pró-Reitora de Graduação: Profª. Cíntia Inês Boll
Diretora do Instituto de Informática: Profª. Carla Maria Dal Sasso Freitas
Coordenador do Curso de Engenharia de Computação: Prof. Walter Fetter Lages
Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

"If I have seen farther than others,

it is because I stood on the shoulders of giants."

— Sir Isaac Newton

AGRADECIMENTOS

First and foremost, I want to thank my mother, my father, and my entire family. For all the love, effort, and dedication you have given me, for the encouragement and support you have given me in both my academic and personal life. You are a great reference for me. I also thank all my friends from Brazil and all the wonderful people I met in Germany. I am infinitely grateful to have all of you in my life.

I would like to express my sincere gratitude to my advisors, Professor Viviane Pereira Moreira and Arghavan Hosseinzadeh. I was very fortunate to have been accepted by both of them. Thank you, Viviane, for your patience and for the teachings you have given me throughout this thesis. Thank you, Arghavan, for the great mentoring during the time I worked at Fraunhofer. Your hospitality and understanding were essential for me during my time in Germany.

I would also like to thank Professor Eduardo Zimmer and his entire research group. I was very lucky to have been part of this amazing group. Thank you so much, Professor Zimmer. Having such a positive and inspiring mentor who encourages us and makes us believe in our potential makes a huge difference in a student's life. I would also like to thank all my former lab mates who made my experience in this group wonderful.

I would like to thank the Federal University of Rio Grande do Sul (UFRGS), especially the Department of Computer Science, for the excellent training and the many opportunities I had during my studies here.

Finally, I would like to thank the Technical University of Kaiserslautern, which, in collaboration with the Department of Computer Science at UFRGS, gave me the unimaginable opportunity to complete an exchange program in Germany. Thank you for hosting me so well and for making these exchange experiences the best years of my life.

ABSTRACT

Nearly 80% of all potentially usable business information exists in unstructured form, primarily as text and images. Techniques such as named-entity recognition (NER) can provide a way to extract structured information from plain text. In general terms, NER aims to recognize information entities that refer to real-world objects, called named entities. Many applications use NER, but most studies have been done in English. In this work, we propose the use of automated concatenation of embeddings (ACE) approach for the Portuguese NER task. Given a set of candidate word embeddings, ACE is trained to find the best concatenation of embeddings to use for structured prediction. In addition, we propose the use of BERTimbau, a state-of-the-art Portuguese language model, as a candidate embedding. The results of the work show that our approach can outperform some previous works. However, it cannot achieve better results than the current state-of-the-art.

Keywords: Named-entity recognition. natural language processing. deep learning. HAREM. portuguese language.

Concatenação automática de embeddings para reconhecimento de entidades nomeadas em português

RESUMO

Quase 80% de todas as informações potencialmente utilizáveis existem na forma não-estruturada. Técnicas como o Reconhecimento de Entidade Nomeada (NER) podem nos fornecer uma maneira de extrair informações estruturadas desta categoria de dados. Em termos gerais, esse conjunto de técnicas visam reconhecer entidades de informação que se referem a objetos reais, chamados entidades nomeadas (NE). NER é usado em variadas aplicações, mas a maioria dos estudos desse campo estão relacionados à língua inglesa. Neste trabalho, propomos o uso da abordagem de concatenação automatizada de embeddings (ACE) para a tarefa de NER em português. Dado um conjunto de embeddings candidatos, ACE é treinado para encontrar a melhor concatenação de embeddings a ser usada para predição estruturada. Além disso, propomos o uso do BERTimbau, um modelo de linguagem em português de última geração, como um embedding candidato. Os resultados do trabalho mostram que nossa abordagem pode superar alguns trabalhos anteriores. Entretanto, não pode alcançar melhores resultados que o atual estado da arte.

Palavras-chave: Reconhecimento de entidades mencionadas, processamento de linguagem natural, aprendizagem profunda, HAREM, língua portuguesa.

LIST OF ABBREVIATIONS AND ACRONYMS

ACE	Automated Concatenation of Embeddings
BiLM	Bidirectional Language Model
BiLSTM	Bidirectional Long Short-Term Memory
CRF	Conditional Random Fields
DNN	Deep Neural Network
LM	Language Model
LSTM	Long Short-Term Memory
M-BERT	Multilingual BERT
M-Flair	Multilingual Flair
ML	Machine Learning
MLM	Masked Language Model
MUC	Message Understanding Conference
NAS	Neural Architecture Search
NE	Named entity
NER	Named-entity recogntion
NSP	Next Sentence Prediction
RNN	Recurrent Neural Network
TL	Transfer Learning

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

The creation of the internet has affected the way in which people carry out several activities. Through it, we can publish our opinions, communicate, read, watch videos, and do many other tasks. As a consequence of this practicality, the last years have witnessed a considerable volume of digital data being generated every day. An estimate published by the IDC (RYDNING, 2018) predicts that all the Global Datasphere will grow from 33 Zettabytes in 2018 to 175 Zettabytes by 2025. In the last years, companies and researchers have been working with part of the generated data, developing methods for understanding human behavior and taking advantage of it to provide new technologies. Recommendation systems, largely used in modern applications such as streaming services, are an example of this since they can use the data their users provide to the platform in order to recommend new products, videos, music, and others.

Through the internet, data can appear in different formats, as an image or a text, for example. Merrill Lynch in 1998 estimated that almost 80% of all potentially usable business information originates in unstructured form (BLUMBERG; ATRE, 2003). However, it is not simple to use them, due to their heterogeneity. To provide a way to take advantage of this data, the information extraction field has received much attention from researchers in the last decades. Its techniques contribute to the manipulation of unstructured information, making it possible to extract structured information from them.

Named-entity recognition (NER) is typical natural language processing (NLP) application widely used for information extraction. It aims to recognize information units that refer to real-world objects, called named entities (NE). In general terms, these objects belong to predefined semantic types, such as names of organizations (e.g. Google), people (e.g. Barack Obama), and geographic locations (e.g. Germany), as well as dates, time, and other numeric expressions (e.g. $54.3 billion) (NADEAU; SEKINE, 2007). NER systems have a fundamental role in many NLP applications, such as question answering, automatic text summarization, machine translation, text understanding, etc (YADAV; BETHARD, 2019; LI et al., 2020).

The research on NER has been widely discussed in the last decades after the term NE was coined for the sixth message understanding conference (MUC) (GRISHMAN; SUNDHEIM, 1996). However, recognizing a NE is not a straightforward task. Different categories of NEs can be written similarly or appear in related contexts. For example, people and place names start with a capital letter and temporal expressions contain numbers, as well numeric expressions. NEs can be recognized in different semantic types according to the context they appear (e.g. the NE Apple can refer to the fruit or the company).

As stated by Li et al. (2020), four main streams have been adopted within the techniques applied in NER: 1) Rule-based approaches, which rely on hand-crafted rules and do not require annotated data; 2) Unsupervised learning approaches, which eliminate the need for annotated data; 3) Feature-based supervised learning approaches, which require annotated data and a rigorous step of feature engineering; 4) Machine learning-based approaches, which applies algorithms that try to learn the abstract representation of the data.

In the last few decades, machine learning techniques have been widely used in various fields of research. In this way, NER research is also following this process. It can be said that the most recent work in the field of NER employs machine learning techniques to accomplish this task. In this sense, this paper focuses exclusively on solving NER using these techniques.

Although several studies have been conducted in English, only a few works have addressed the NER task for Portuguese. This scenario is changing since Santos et al. (2006) proposed HAREM, an initiative for NER systems in Portuguese that is widely used as a golden standard reference. However, we note that there is a gap between recent progress in NER systems for Portuguese and English. For this reason, and following the state-of-the-art approaches for English NER, we propose the use of automated concatenation of embeddings (ACE) (WANG et al., 2020) for NER in Portuguese. Given a set of candidate word embeddings, ACE is trained to find the best concatenation of embeddings to use for structured prediction. In addition, we explore the use of BERTimbau, a Portuguese language model (LM), as contextual embeddings within ACE.

The rest of this work is organized as follows: Chapter 2 presents the required background to comprehend our work, it details the HAREM dataset, neural network architectures, word embeddings, and the approach used in this work. Chapter 3 reviews existing work in the literature for Portuguese NER. Chapter 4 details our experiments. Chapter 5 presents and discusses the results achieved by our proposed methodology. Then, Chapter 6 presents the conclusions and perspectives for future research created by this study.

2 BACKGROUND

NER research is largely driven by shared evaluation contests that provide researchers with curated datasets and evaluation tools that help maintain a standard and robustness in evaluating and comparing different methods. Early NER systems were rule-based approaches based on hand-crafted rules. Nowadays, however, most recent studies use machine learning (ML) techniques to address this task. In this chapter, we describe HAREM, the shared evaluation contest that provides the Portuguese dataset we work with. We also explain the different ML techniques commonly used in NER systems, as well as the techniques we use in this work.

2.1 HAREM Evaluation Contest

A shared evaluation contest describes a common ground for approaches addressing a given task. Shared tasks make it possible to evaluate different methods fairly and impartially while promoting research in the field. Without them, each one of the published approaches would be evaluated exclusively by its authors, which would make it very difficult to compare different systems. Thus, these competitions were established as a way to standardize the evaluation method for different systems.

The sixth MUC (SUNDHEIM, 1995), was the first conference to propose measuring the NER task independently. It was followed by several other evaluation events focusing on NER, such as the MET (MERCHANT; OKUROWSKI; CHINCHOR, 1996), the CoNLL shared task (SANG; MEULDER, 2003), and the anaphora and coreference resolution (DODDINGTON et al., 2004) (SANTOS; CARDOSO, 2007).

For the Portuguese language, the most known shared evaluation contest was organized by Linguateca[1], a distributed resource center for the computerized processing of the Portuguese language. Linguateca was created in 1998 to support the community dedicated to the processing of this language. The initiative aims to facilitate access to existing resources and to organize shared evaluation contests, such as HAREM.

HAREM is the first shared evaluation contest for NER in Portuguese, organized by Linguateca in 2005 (SANTOS; CARDOSO, 2007). It was directly inspired by MUC. The contest was motivated by the fact that previous NER contests had not addressed the task in sufficient depth. HAREM started to be planned in June 2003, and the first edition was held

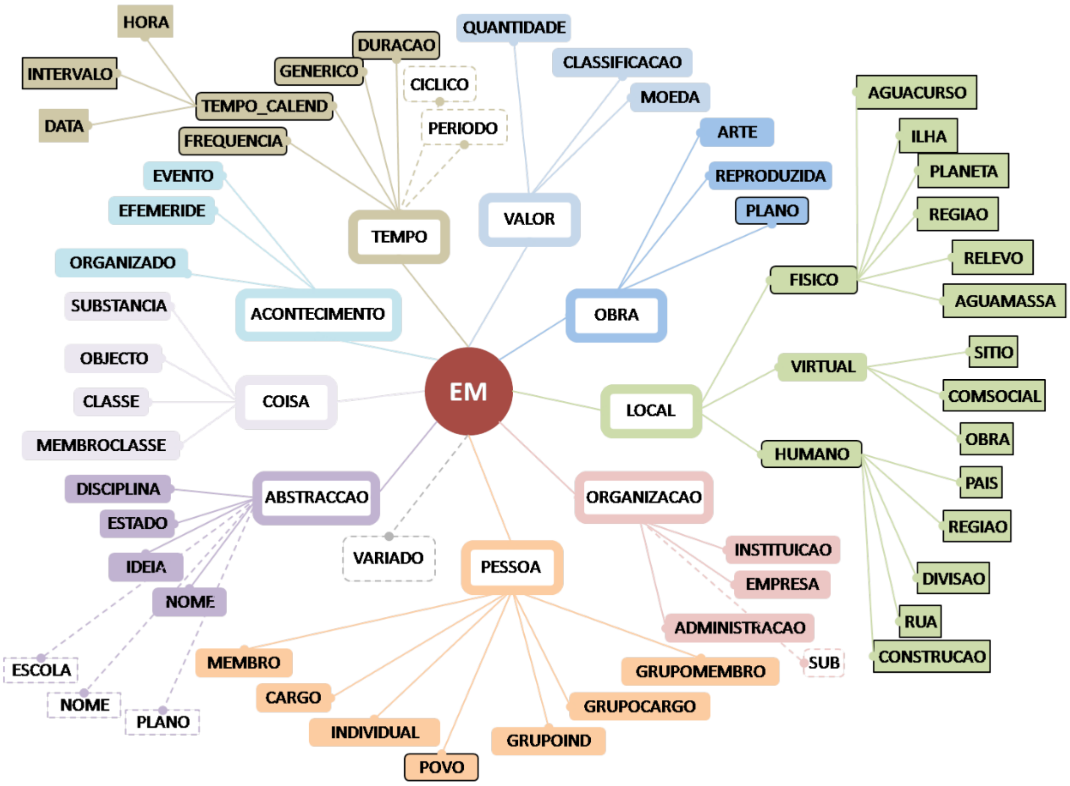
[1] https://www.linguateca.pt/

in 2004, followed by Mini-HAREM in 2006 and the second edition in 2007. Although its foundation was based on European Portuguese, HAREM serves as a collection for all dialects of Portuguese.

The guidelines for the HAREM evaluation were established together with the participants. These guidelines set the rules by which system results will be evaluated when compared to the Golden Collection, the comparative text developed in collaboration with the community (CARVALHO, 2012).

In the second edition of HAREM, changes and improvements led to a more precise and linguistically motivated characterization of certain NEs, as well as to a more unbiased evaluation of the systems. Figure 2.1 shows the category tree defined in the second edition of HAREM: the categories, types, and subtypes shown in the black-bordered boxes exist only in the second edition of HAREM; the categories, types, and subtypes shown in the dotted boxes exist only in the first edition of HAREM (CARVALHO et al., 2008).

Figure 2.1: The category tree defined in the second HAREM: the categories, types, and subtypes shown in the black-bordered boxes exist only in the second HAREM; the categories, types, and subtypes shown in the dashed-bordered boxes exist only in the first HAREM.



Source: Carvalho et al. (2008, p. 8)

In this sense, the HAREM categories, types, and subtypes are defined according to Santos and Cardoso (2007), Carvalho et al. (2008), which can be found in Appendix A. In general terms, according to the last edition of HAREM, the following categories are utilized in this dataset: ABSTRACCAO, ACONTECIMENTO, COISA, LOCAL, OBRA, ORGANIZACAO, OUTRO, PESSOA, TEMPO, and VALOR.

2.2 Neural Network Architectures

ML is a powerful tool that allows computer algorithms to automatically learn and improve from experience and by using data without being explicitly programmed. In this way, many complex problems can be modeled using ML because it can capture relationships within a data set that a human would not readily perceive. ML algorithms are very commonly used in natural language processing applications, such as NER. However, there are many different algorithms and architectures used in ML, which makes choosing the right algorithm a non-trivial problem. Researchers have explored and reported on the behavior of these options over the past few decades.

In this section, we present several common architectures and algorithms used for NER tasks. We focus here only on the most commonly used methods for NER in English and Portuguese.

2.2.1 Long Short-Term Memory

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in deep learning. Proposed in 1997 by Hochreiter and Schmidhuber (1997), LSTMs have been widely used not just in NLP problems, but also in many applications as time series and market prediction (SCHMIDHUBER; WIERSTRA; GOMEZ, 2005; ISLAM; HOSSAIN, 2020), protein homology detection (HOCHREITER; HEUSEL; OBERMAYER, 2007), human action recognition (BACCOUCHE et al., 2011), and others. The main idea of LSTM is to improve the classic RNN architecture (RUMELHART; HINTON; WILLIAMS, 1985), to avoid the vanishing gradient problem. This problem is caused by the computations involved in the backpropagation algorithm that use finite-precision numbers, which can lead gradients to be null or make them tend to be infinite.

The architecture of an LSTM unit is shown in Figure 2.2 and is modeled by the following set of equations:

$$\begin{aligned} i_t &= \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \\ f_t &= \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \\ o_t &= \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_0\right) \end{aligned} \tag{2.1}$$

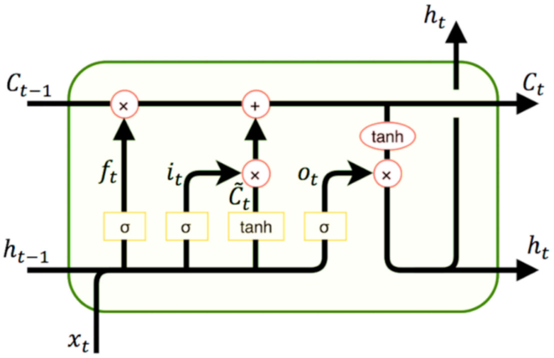
$$\tilde{C}_t = \tanh\left(W_C \cdot [h_{t-1}, x_t] + b_C\right) \tag{2.2}$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \tilde{C}_t \tag{2.3}$$

$$h_t = o_t \otimes \tanh\left(C_t\right) \tag{2.4}$$

Where i_t denotes the input gate and o_t denotes the output gate. The forget gate, memory cell, and hidden state are denoted by f_t, C_t, and h_t, respectively. The set of equations 2.1 are sigmoid functions where W's and b's are the parameters (weights and biases) for input, forget and output gates. In Eq. 2.3, the $tanh$ layer computes the vector of the new candidate value $\tilde{C}*t$ which is added to the cell state (GRAVES; SCHMIDHUBER, 2005; KULL; KUHAUPT,).

Figure 2.2: Architecture of an LSTM unit.

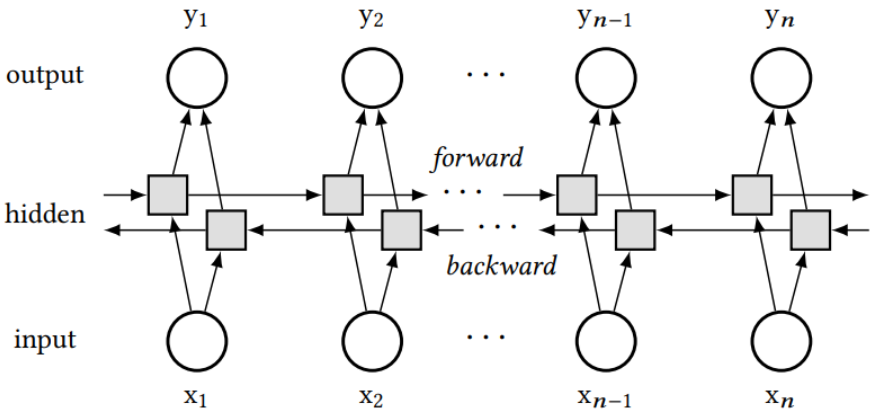


Source: Bouktif et al. (2019, p. 5)

2.2.1.1 Bidirectional Long Short-Term Memory

In a forward LSTM network, the information stored in the hidden state h_t is only available from the past. However, when using a bidirectional LSTM network we can capture the information flow both ways, providing additional context to the network and resulting in faster and even fuller learning on the problem. The network consists of a forward hidden layer and a backward hidden layer. The architecture of a bidirectional long short-term memory (BiLSTM) network is shown in figure 2.3. Here, the hidden layer is used to maintain the long-distance information in the matrix weights (ALZAIDY; CARAGEA; GILES, 2019).

Figure 2.3: BiLSTM network.



Source: Alzaidy, Caragea and Giles (2019, p. 2553)

2.2.2 LSTM and Conditional Random Fields

2.2.2.1 Conditional Random Fields

In this work, ML is used to detect potential NEs given an input sequence. The model reads the sequence of words and computes the probabilities of each word to be classified as a category. As the focus of the study involves many possible categories for each prediction, it can be listed as a multiclass classification problem.

In most multiclass classification problems, activation functions like the Softmax are used as in the output layer that computes the probability distribution for the input sequence. However, given the shape of the NER problem, this is not the optimal method. Since it computes the model prediction based on its local input, i.e, only in the current word being analyzed. In other words, it means the model could skip important features when not using the word's context. In this case, it is necessary an alternative approach, which considers the influence of neighboring words.

Conditional Random Fields (CRF) is a statistical modeling method introduced in Lafferty, McCallum and Pereira (2001) used to model the structure of conditional dependence between random variables that can be represented as an undirected graph. It is been widely used for pattern recognition and ML problems, such as NER. The main idea of CRF is to learn the mapping function $\boldsymbol{x} \rightarrow \boldsymbol{y}$ using conditional probability, considering that each output $\boldsymbol{y}$ is not independent. In a sequence labeling problem, it is assumed that a prediction for output $\boldsymbol{y_i}$ is not only dependent on its feature x_i, but also on other outputs and features in the sequence. For example, when using the IOB2 tagging, a word cannot be classified as I-PERSON if the previous one is not classified as B-PERSON, since the prefix I- designates that the word is an intern token of an entity. Therefore, the CRF algorithm can learn the correlation between neighboring words, using their features and labels for achieving a better prediction for each one of the words.

The most common structure of dependencies between variables is presented in a linear chain CRF that represents these dependencies in a time sequence. In other words, it predicts the output variables as a sequence. As shown in Alzaidy, Caragea and Giles (2019), a linear-chain CRF is a conditional distribution over the label sequence $\boldsymbol{y}$ given $\boldsymbol{x}$, as presented in Equation 2.5

$$p(\mathbf{y} \mid \mathbf{x}; \mathbf{W}, \mathbf{b}) \propto \exp\left(\sum_{i=1}^{n} \mathbf{W}_{y_{i-1},y_i}^{T} \mathbf{x}_i + \mathbf{b}_{y_{i-1},y_i}\right) \tag{2.5}$$

where the model parameters $\mathbf{W}_{y_{i-1},y_i}$ (weight vector) and $\mathbf{b}_{y_{i-1},y_i}$ (bias) represents neighboring labels information.

During the training process, the model parameters are estimated by the log-likelihood function 2.6, where $\mathcal{D} = \left\{\left(\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\right)\right\}_{j=1}^{N}$ represents the dataset being used. The estimation of the outputs is represented by the Viterbi (VITERBI, 1967) decoding Equation 2.7, which computes the sequence label that maximizes the likelihood.

$$L(\mathbf{W}, \mathbf{b}) = \sum_{j=1}^{N} \log p\left(\mathbf{y}^{(j)} \mid \mathbf{x}^{(j)}; \mathbf{W}, \mathbf{b}\right) \tag{2.6}$$

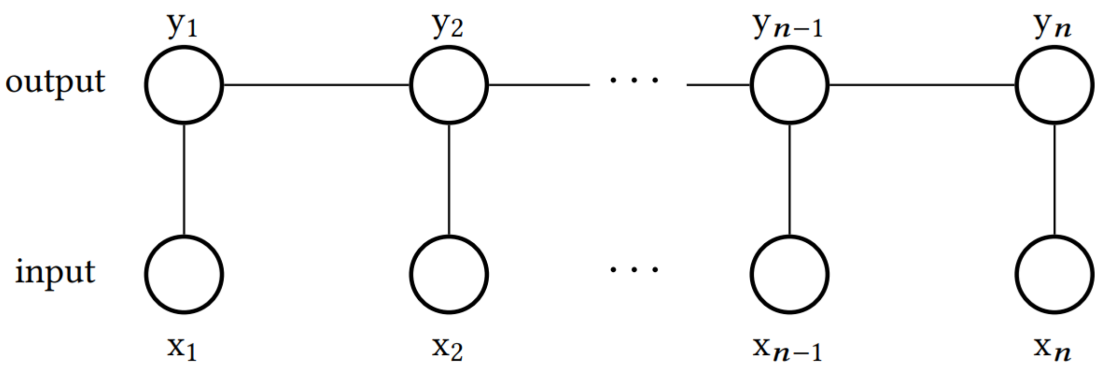
$$\mathbf{y}^{*} = \operatorname{argmax}_{y \in \mathcal{Y}(\mathbf{x})} p(\mathbf{y} \mid \mathbf{x}; \mathbf{W}, \mathbf{b}) \tag{2.7}$$

Figure 2.4 illustrates the architecture of a simple CRF model. Since the output nodes are connected, the model can learn dependencies between elements.

2.2.2.2 BiLSTM-CRF

The advantage of BiLSTM neural networks lies in the automatic selection and maintaining of sequential input information. On the other hand, the CRF algorithm is very good when computing global optimal predictions of a sequence. Therefore, we can combine a bidirectional LSTM and a CRF network to build a BiLSTM-CRF model

Figure 2.4: Simple linear-chain CRF architecture.



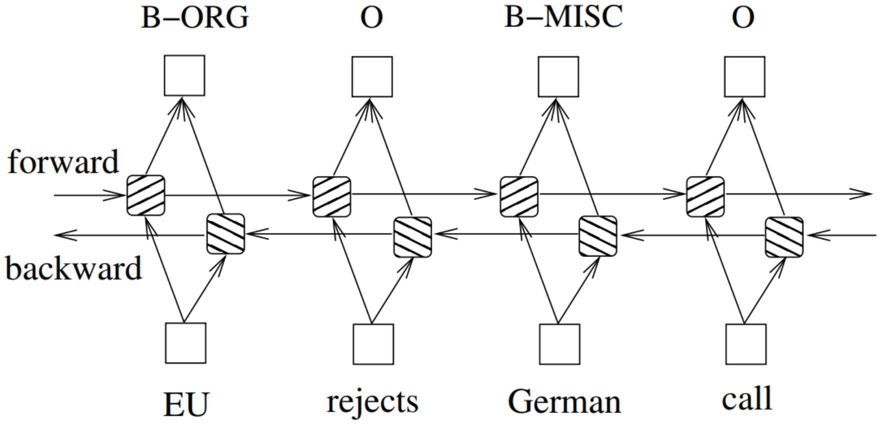
Source: Alzaidy, Caragea and Giles (2019, p. 2553)

(HUANG; XU; YU, 2015), which can use future input features while boosting tagging accuracy, allowing the model to capture as much context as possible while preventing information loss.

Figure 2.5 shows the architecture of a BiLSTM-CRF model. The first layer, represented by the BiLSTM network, captures the semantics of the given input sequence. Then, its outputs are connected as an input to a CRF layer, responsible for producing a probability distribution over the label sequence using the dependencies between the tags of the sequence.

Basically, CRF is added as a decoder layer taking the output of BiLSTM as input while the neural networks act as an encoder. In order to find the best sequence of labels for an input sequence, the Viterbi algorithm (VITERBI, 1967) is used. The CRF used takes advantage of the best label chains in a given input sentence instead of individual position (ALZAIDY; CARAGEA; GILES, 2019).

Figure 2.5: BiLSTM-CRF network.



Source: Huang, Xu and Yu (2015, p. 2553)

The LSTM has proven to be very successful, but the Transformers model (VASWANI et al., 2017) has recently changed the perception of the NLP area. The recent emergence of transformers has led to a huge amount of research on neural architectures, especially in the field of NLP, which has not received much attention since the release of Transformer. Among the most efficient natural language understanding frameworks, many models have the influence of this neural architecture in some way.

2.2.3 Transformers

Unlike RNNs, Transformer (VASWANI et al., 2017) does not use a temporal relationship between time steps through recurrence. Instead, its architecture relies entirely on the attention mechanism, making it computationally efficient and highly parallelizable. For this reason, it eliminates the memory constraints which sequential computation suffers. The main idea is to use self-attention to find relevant units (e.g. words) to each one of the units in a given input sequence. Beyond its use in this architecture, self-attention was successfully implemented in many NLP problems before the emergence of Transformers.

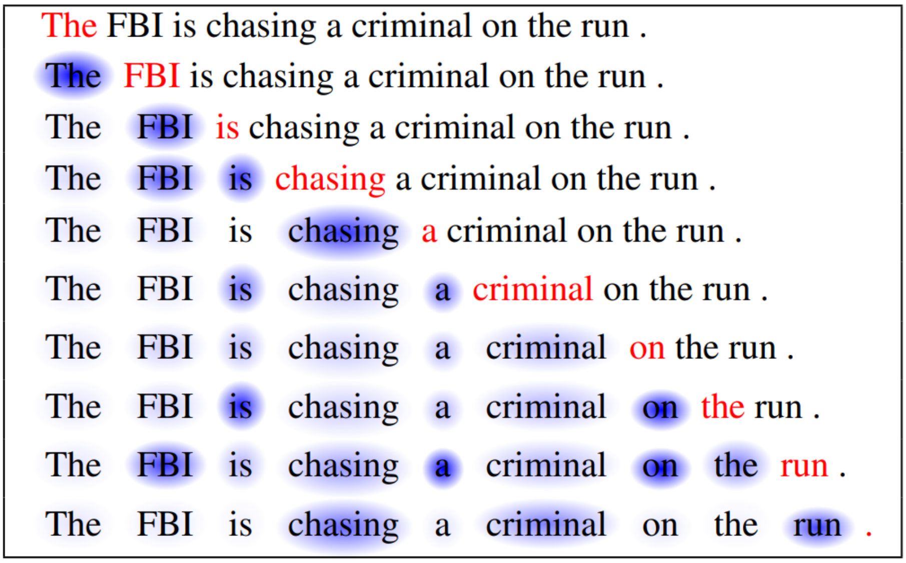
The transformer architecture uses an encoder-decoder scheme, where the decoder relies on additive attention, as explained in Bahdanau, Cho and Bengio (2014). Furthermore, it fully depends on multi-head attention layers, referred by the authors when applying attention both in the encoder and decoder, in combination with a regular feed-forward neural network. The main benefit of using attention mechanisms is that dependencies between units can be computed without taking their position in the sequence or its distance to a token in the output into consideration, contrary to Machine Translation where is essential to learn dependencies among distant units to correctly map words between two languages (BAHDANAU; CHO; BENGIO, 2014; LUONG; PHAM; MANNING, 2015; VASWANI et al., 2017).

Figure 2.6 shows the distribution of the attention between each word in the sentence, which is responsible for the efficient capture of long-distance dependencies. This ensures that long-distance dependencies can be captured efficiently (VASWANI et al., 2017). Furthermore, the different properties in the sequence are captured by each attention head.

In terms of sequential computation, the input sequence is encoded as a hidden representation, where each unit at time step t is dependent on its current input at t and on the hidden representation of time step $t-1$. As a result, early positions can be weighted less or overwritten by succeeding positions. On the other hand, the self-attention mechanism is capable of learning dependencies between positions independent of their distances, providing a very rich and informative representation of an input sequence (BAHDANAU; CHO; BENGIO, 2014; LUONG; PHAM; MANNING, 2015; VASWANI et al., 2017; DEVLIN et al., 2018). Therefore, this mechanism relieves the limitation that is observed on recurrence-based models.

The Transformer is entirely built by self-attention and point-wise, a stack of fully connected layers. As such, this is the first neural architecture that uses a self-attention mechanism without employing any recurrence in its computation algorithm. In addition, recent researches have been proving that the self-attention mechanisms leveraged by Transformers can capture NEs (RAGANATO; TIEDEMANN et al., 2018), as well as dependency relations (VIG; BELINKOV, 2019) and part-of-speech tags (RAGANATO; TIEDEMANN et al., 2018; VIG; BELINKOV, 2019).

Figure 2.6: Illustration of the self-attention mechanism working with the sentence “The FBI is chasing a criminal on the run.”.


The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .
The FBI is chasing a criminal on the run .

Source: Cheng, Dong and Lapata (2016, p. 2)

2.3 Word Embeddings

In NLP, word embedding is a term for the representation of words that allows machines to work with this abstract concept. There are many techniques to achieve this representation. Typically, words are mapped to a vector space where words that are close to each other are assumed to be similar in meaning. In this sense, it is common to use LM and feature learning techniques, as well as statistics within a corpus, such as word co-occurrences.

In this section, we will introduce several common methods for computing word embeddings. Here, we divide them into two sections: (1) fixed embeddings, where the context of the words is not considered during the algorithm, and (2) contextualized embeddings, where the final embedding takes into account the context in which the words are inserted.

2.3.1 Fixed embeddings

2.3.1.1 GloVe

Semantic vector space models of language represent each word with a real-valued vector. Traditional methods typically use the distance or angle between pairs of words to achieve such a representation. However, these approaches only focus on the information obtained from the local context without using global statistical information. Motivated by these drawbacks of traditional methods, Pennington, Socher and Manning (2014) proposed GloVe (Global Vectors), a model for distributed word representation (WANG; ZHOU; JIANG, 2020).

The proposed model uses semantic similarity to represent words in a vector space. Moreover, the author claims that global log-bilinear regression models are suitable to achieve linear directions of meaning on word representation. In this sense, the model training uses aggregated global word co-occurrence statistics from a given corpus.

The resulting representations of the model reveal interesting linear substructures of the word vector space (RAO et al., 2019). As a result, GloVe is a global log-bilinear regression model for unsupervised word representation learning that outperforms other models in word analogy, word similarity, and NER tasks (PENNINGTON; SOCHER; MANNING, 2014).

2.3.1.2 fastText

Many word representation approaches treat words as atomic units and ignore the characters that compose them. This means that these approaches ignore sub-word-level information such as prefixes and suffixes, roots, and compound words. This is a limitation as word morphology could be used to increase model accuracy, especially for highly morphological languages. To address these drawbacks, Bojanowski et al. (2017) has presented a method that takes morphology information into account when computing word embeddings.

fastText represents words using n-grams of characters. For example, with n-grams of size 3, the word "where" would be represented as <wh, whe, her, ere, and re>, plus <where>. In addition, each n-gram has its vector of characters "<" and ">" to define the beginning and end of a word. In practice, the work tries to extract multiple n-grams simultaneously ($3 \leq n \leq 6$). During the training process, the n-grams are summed by the vector of all original words. All word vectors and character-level n-gram vectors are summed simultaneously and averaged as input to the training model (WANG; ZHOU; JIANG, 2020). Moreover, it allows the capture of the order relationship between characters and the internal semantics of words.

The results presented in the paper show that fastText outperforms word embeddings that do not use subword information, as well as approaches based on morphological analysis. Due to the simplicity of its architecture, fastText has a fast training time and requires no processing or monitoring during this task.

2.3.2 Contextualized embeddings

2.3.2.1 ELMo

Peters et al. (2018) has introduced a new type of deep contextualized word representation that allows modeling complex features of word usage (e.g., syntax and semantics) and their behavior when used in different linguistic contexts. In contrast to traditional word embeddings, where each token is assigned a representation that is a function of the whole input sentence, this work exploits the use of a linear combination of vectors derived from a biLSTM that is trained on a large text corpus using a coupled LM objective.

The word vectors are learned functions of the internal states of a bidirectional Language Model (biLM), as shown in Figure 2.3. In this sense, ELMo representations are deep since they are a function of all internal layers of the biLM, which allows for a very rich word representation. Due to this architecture, the approach is called ELMo (Embeddings from Language Models) representations.

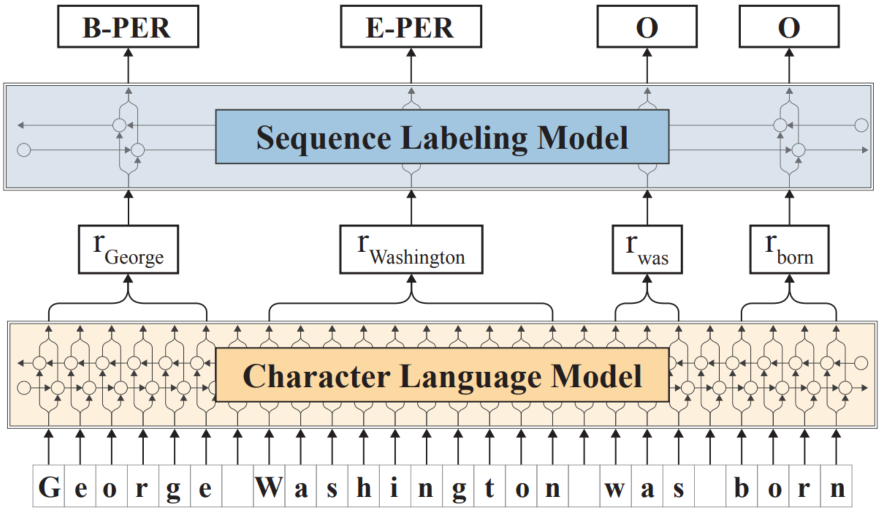
In this work, ELMo was evaluated in six different tasks, and in all of them, the state-of-the-art was improved by the simple addition of ELMo. In the NER task, the addition of ELMo increased the F1 score by 2.06 points. The authors claim that the good performance is because the contextual representations of biLMs can encode information that is not captured by using word vectors. Moreover, biLM disambiguates the meaning of words based on their context. Moreover, ELMo-enhanced models use smaller training sets more efficiently than models without ELMo.

2.3.2.2 Contextual String Embeddings (Flair)

Akbik, Blythe and Vollgraf (2018) proposed a novel approach to contextualized character-level word embedding. The work aimed to combine several good features of previous word embedding methods using distributions over character sequences generated by LMs. In general, the author presented contextual string embeddings and their use in state-of-the-art sequence labeling tasks.

The author claims that highly effective word-level embeddings can be generated if a good selection of the hidden states of an LM is made. As mentioned in the paper, each sentence serves as input to a bidirectional character-level neural LM, from which the internal character states for each word are retrieved to create a contextual string embedding. For a downstream NLP task, the generated embedding is used as input to a BiLSTM-CRF architecture. This process is illustrated in Figure 2.7.

Figure 2.7: A high-level overview of the proposed approach. A sentence is an input as a character sequence into a pre-trained bidirectional character LM.

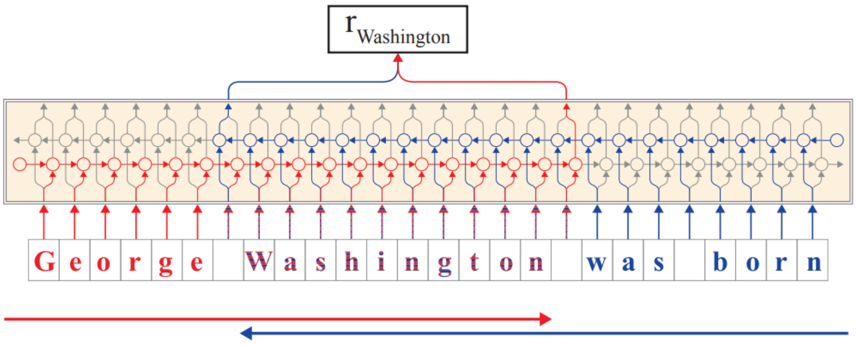


Source: Akbik, Blythe and Vollgraf (2018, p. 2)

Moreover, an LSTM architecture is used for language modeling because it can encode long-term dependencies with their hidden states. The atomic units of the model are characters, which means that the input of the LSTM used is a sequence of characters, and each point in the sequence is trained to predict the next character. In this sense, for each character in the input sequence, the model has a hidden state that is used to generate the embedding for the characters of a word and also for the characters of the surrounding context.

The procedure for extracting these word representations is shown in Figure 2.8. More precisely, the hidden state of the output is extracted after the last character of the word. Thus, this hidden state contains information that propagates from the beginning of the sentence to this point. From the backward LM (shown in blue), the output hidden state is extracted before the first character of the word. Thus, it contains information that propagates from the end of the sentence to this point. Both hidden initial states are linked together to form the final embedding (AKBIK; BLYTHE; VOLLGRAF, 2018).

Figure 2.8: Extraction of a contextual string embedding for a word ("Washington") in a sentential context.



Source: Akbik, Blythe and Vollgraf (2018, p. 4)

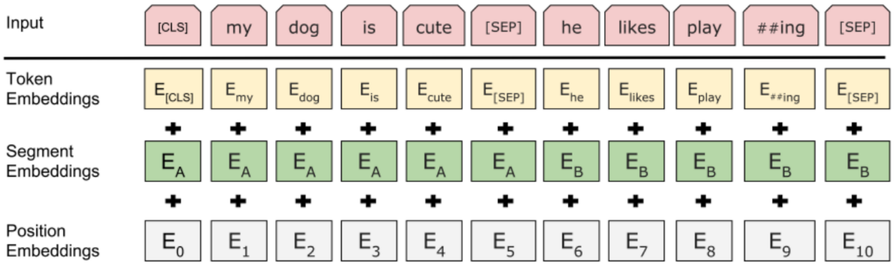
The work performed an evaluation of contextual string embeddings for sequence tagging. For the NER task, stacked combinations of embeddings were used as input to a BiLSTM-CRF architecture. The paper shows that the proposed approach performs well on this task when a stacked combination of three embeddings is used: (1) contextual string embeddings, (2) classical word embeddings, and (3) character-level features. The result, evaluated using the CONLL 2003 common task dataset, outperforms the previous state-of-the-art approach by 0.87 percentage points on the F1-score.

2.3.2.3 BERT

BERT was introduced by Google AI researchers and stands for Bidirectional Encoder Representations from Transformers (DEVLIN et al., 2018). It is a deep contextual LM based on the Transformer architecture. Prior to BERT, all LMs that relied on LSTMs or Transformers were unidirectional or slightly bidirectional, which made it impossible for the model to contextualize its inputs given all the context in which they appear. Therefore, BERT uses context from both directions of a word and makes use of fully connected linear layers and self-awareness mechanisms that can compute token relationships regardless of their position. BERT can be easily fine-tuned with an additional output layer, allowing us to implement excellent models for many NLP problems (XU et al., 2020; GLASS et al., 2019; ZHANG et al., 2020).

Regarding its architecture, BERT consists of a stack of Transformer layers, each of them containing two kinds of sublayers. The first, called the multi-head self-attention mechanism, helps the model compute the importance of other words in encoding a given word. The second is the position-wise fully connected feed-forward network, which is responsible for applying linear transformations to each unit. It is important to note that in the input of the model, in BERT, the words are not used in their full structure, but are represented by WordPiece embeddings (WU et al., 2016). For example, the word "embeddings" would be split into four WordPieces: "em", "bed", "ding" and "s". WordPieces allow us to represent words that are outside the model vocabulary without the complexity that exists with character-based models. Later, BERT performs a transformation on these WordPieces to obtain their numerical representations and pass them as input to the model. Besides these token embeddings, BERT also uses segment and position embeddings. Consequently, the final input of the model is the sum of the token, segment, and position embeddings. Figure 2.9 shows how the input embeddings are calculated.

Figure 2.9: BERT input representations.



Source: Devlin et al. (2018, p. 5)

The proposed work uses two training strategies, Masked Language Model (MLM) and Next Sentence Prediction (NSP). In MLM, part of the words in the input sequence is replaced by a $[MASK]$ token. The model then tries to predict the original value of the corrupted token based on the words that have not been changed. In NSP, on the other hand, pairs of sentences are input to a model that attempts to predict whether the second sentence of the pair is the next sentence of the first. In this case, $[CLS]$ and $[SEP]$ tokens are used to indicate the beginning and end of each sentence, respectively.

BERT is considered to be a huge improvement in the use of ML for NLP. Due to the good results it can achieve, its ease of use, and the fact that it is open-source, many other BERT-based LM have emerged after its release. Also, several pre-trained checkpoints are available online, allowing the use of these models in downstream NLP tasks, such as NER.

2.3.2.4 BERTimbau

Transfer learning (TL) is an ML technique in which a model trained for a specific task can be fine-tuned and applied to another related task of interest. This technique had a huge impact on many research areas in recent years, as it is possible to save resources and train models when only unlabeled datasets are available. In the field of NLP, TL has been used extensively and has helped the community to reach the state-of-art in many NLP tasks.

Many works use a TL strategy that consists of fine-tuning a large pre-trained LM, which has been shown to work well in many applications (RADFORD et al., 2018; YANG et al., 2019). However, this strategy requires a large amount of data and computational resources. Therefore, these drawbacks have been a limiting factor for the availability of these models in different languages.

BERT (DEVLIN et al., 2018) is one of the most adopted pre-trained LM. Although BERT has a multilingual model (mBERT), many researchers have made efforts to pre-train monolingual BERT and derived models for individual languages, which has been shown to perform better than mBERT. Moreover, monolingual pre-trained models can be very useful for languages that have few annotated records but abundant unlabeled data.

Motivated by this idea, Souza et al. (2020) presented BERTimbau, a BERT model trained on unlabeled Brazilian Portuguese data. The approach used replicates the BERT architecture and pre-training procedures with few modifications. The transformer architecture used in BERT allows the pre-training of deep bidirectional word representations by training the model using MLM. More specifically, this means that the model can see all input tokens at once, which allows for high parallelization of the task while improving dependency modeling in long input sequences.

BERTimbau was trained on a large corpus of web pages named brWaC (FILHO et al., 2018). The dataset consists of documents from many different domains, which is desirable for BERT pre-training. The final processed corpus contains 17.5 GB of raw text.

As in the BERT model, MLM and NSP tasks were used in the training process for BERTimbau. In this sense, the input examples are generated by concatenating two sequences of tokens $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_m)$ that are separated by special tokens [CLS] and [SEP] as follows:

$$[CLS]\, x_1 \cdots x_n \,[SEP]\, y_1 \cdots y_m \,[SEP] \tag{2.8}$$

Given an input sentence $\boldsymbol{x}$, 50% of the time $\boldsymbol{y}$ is chosen to form a contiguous piece of text and 50% of the time $\boldsymbol{y}$ is assigned as a random sentence from another document. Later, each example is corrupted by replacing 15% of the tokens of $\boldsymbol{x}$ and $\boldsymbol{y}$ with 1 of 3 randomly assigned options: a special [MASK] token, a random token from the vocabulary, or otherwise the original token.

Each of the corrupted sentences is then used as input to BERT, while the encoded representations of the tokens are used as input to other pre-training tasks. In this sense, the NSP task predicts whether $\boldsymbol{y}$ is the original continuation of $\boldsymbol{x}$ or not, while the MLM task tries to predict the original form of the corrupted token.

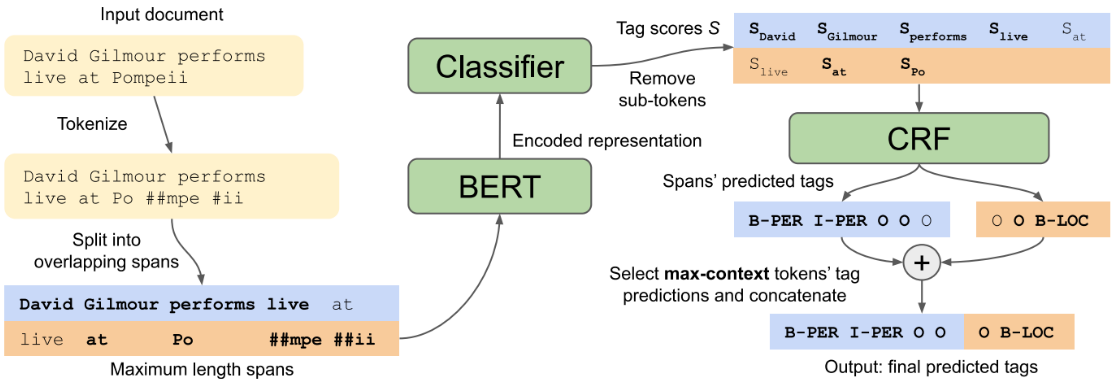
Two models were pre-trained in the work: the BERTimbau Base model, in which weights were initialized with the checkpoint of Multilingual BERT Base, and BERTimbau Large, in which weights were initialized with the checkpoint of English BERT Large.

BERTimbau models were evaluated on 3 downstream tasks: Sentence Textual Similarity, Recognizing Textual Entailment, and NER. For NER, the Golden Collections of the HAREM dataset were used. In this case, First HAREM was used for training, while MiniHAREM served as a test set. The author uses two different scenarios for the dataset: a Total scenario that considers all 10 classes of HAREM, and the Selective scenario that considers only 5 classes (Person, Organization, Location, Value, and Date). In addition, 7% of the First HAREM documents were put aside as a holdout validation set. Performance was evaluated using an evaluation script from CoNLL 2003 (SANG; MEULDER, 2003) that measures entity-level precision, recall, and micro-F1 score for exact matches.

The approach used for NER considered document context as input instead of sentence context, following the approach of (DEVLIN et al., 2018). In this method, illustrated by Figure 2.10, examples longer than $\boldsymbol{S}$ tokens are divided into spans of up to $\boldsymbol{S}$ length with a step of $\boldsymbol{D}$ tokens. Each span is used as a separate example during training. The final prediction for each token is taken from the span where the token is closer to the central position, i.e., the span where it has the most contextual information (SOUZA et al., 2020).

To use BERTimbau for downstream tasks, the MLM and NSP classification heads used in the pre-training phase must be removed and the headers required for each task

Figure 2.10: Illustration of the proposed method for the NER task.


Input document
David Gilmour performs live at Pompeii
Tokenize
David Gilmour performs live at Po ##mpe #ii
Split into overlapping spans
David Gilmour performs live at
live at Po ##mpe ##ii
Maximum length spans
BERT
Encoded representation
Classifier
Tag scores S
Remove sub-tokens
S_{David} $S_{Gilmour}$ $S_{performs}$ S_{live} S_{at}
S_{live} S_{at} S_{Po}
CRF
Spans' predicted tags
B-PER I-PER O O O
O O B-LOC
Select max-context tokens' tag predictions and concatenate
+
B-PER I-PER O O O B-LOC
Output: final predicted tags

Source: Souza et al. (2020, p. 408)

added in their place. For the NER, a linear classification layer was placed on top of BERTimbau to independently predict the tag of each token. Motivated by the large use of CRF in sequence labeling tasks, the work also experimented with employing a CRF layer after the linear layer.

BERTimbau outperforms the other approaches in both the Total and Selective scenarios. Moreover, the large gap between the results of mBERT and BERTimbau confirms that monolingual models pre-trained on multiple domains can perform better compared to mBERT. Furthermore, it has been shown that using the CRF layer on BERTimbau improves the results.

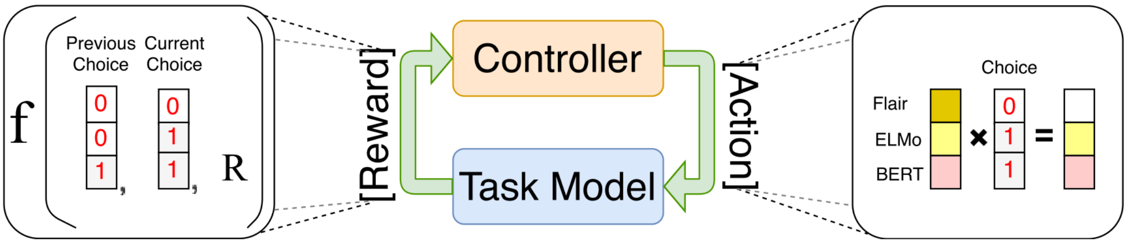
2.4 Automated Concatenation of Embeddings

Recent research in NLP has shown that the use of pre-trained contextualized embeddings can improve performance on structured prediction tasks. In this sense, ELMo (PETERS et al., 2018), Flair (AKBIK; BLYTHE; VOLLGRAF, 2018), BERT (DEVLIN et al., 2018), and XLM-R (CONNEAU et al., 2020) have consistently advanced the state-of-the-art for many structured prediction tasks. Moreover, concatenating different types of embeddings can lead to better word representations and further improve performance (PETERS et al., 2018; AKBIK; BLYTHE; VOLLGRAF, 2018).

Wang et al. (2020) presented an automated approach to find concatenations of embeddings for structured prediction tasks. The work, called Automated Concatenation of Embeddings (ACE), was inspired by recent advances in neural architecture search (NAS), an area of Deep Learning that aims to find better model architectures. However, unlike most previous work in NAS, ACE focuses on finding better word representations rather than better model architectures.

ACE implements an iterative search process driven by a controller and a task model that interact repeatedly. The controller is tasked with performing a concatenation of embeddings and passes this as input to the task model, which predicts the output of the task. The task model is thus trained on the dataset and returns its performance results as a reward signal to the controller, which uses this information to search for a better embedding concatenation. Figure 2.11 shows the architecture proposed by ACE.

Figure 2.11: The main paradigm of ACE is shown in the middle, with an example of a reward function on the left and an example of a concatenation action on the right.



Source: Wang et al. (2020, p. 5)

For the task model, depending on the target task, a probability distribution such as the Equation 2.9 is used. For the NER, the BiLSTM-CRF architecture has defined the appropriate probability distribution model. In this sense, given an input sentence $\boldsymbol{x}$, the structured output $\boldsymbol{y}$ is computed for this structured prediction task.

$$P(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{\exp(\operatorname{Score}(\boldsymbol{x}, \boldsymbol{y}))}{\sum_{\boldsymbol{y}' \in \mathbb{Y}(\boldsymbol{x})} \exp\left(\operatorname{Score}\left(\boldsymbol{x}, \boldsymbol{y}'\right)\right)} \tag{2.9}$$

On the other hand, the controller defines a search space in which each embedding candidate is defined as a node in a directed acyclic graph. Thus, the input to these nodes is a sentence x, while the outputs are the embeddings. To find the best concatenation of embeddings, the controller iteratively defines a masking vector representing the selected concatenation and uses it to train the task model until it converges. Then, the accuracy of the task model is computed for the development dataset and then used as a reward signal for the controller. In this case, the reward function accumulates all rewards based on the transformation between the current concatenation and all previously sampled concatenations, which improves the efficiency of the search process.

3 RELATED WORK

In this chapter, we briefly review the existing work in the NER literature. Although there are numerous works on NER, very few of them deal with the Portuguese language.

Most of the related work presented here deals with the NER task for Portuguese using the HAREM dataset. It is important to note that HAREM has two editions, for each of which different guidelines and labels have been used. However, most works in the literature evaluate their approaches using the golden collection of the first edition, while using the guidelines of the second edition. Furthermore, it is common practice to evaluate papers within the miniHAREM dataset for two scenarios: (1) a total scenario where all ten classes are considered, and (2) a selective scenario where only five classes are considered (PESSOA, ORGANIZACAO, LOCAL, VALOR and TEMPO).

Santos and Guimaraes (2015) reported on an approach for language-independent NER using a deep neural network (DNN) architecture. The work was based on CharWNN (SANTOS; ZADROZNY, 2014), a DNN that uses word-level and character-level embeddings to perform sequence labeling. CharWNN uses a convolutional layer that allows character-level feature extraction from words and has shown good results in POS tagging. Given an input sequence, each of the tokens is assigned a score for each class. More specifically, the score is calculated based on complex features extracted from the sequential layers of the network. The Viterbi algorithm (VITERBI, 1967) is then used to compute the outputs for this structured prediction task. The proposed approach presented experiments for HAREM I and SPA CoNLL-2002, a dataset for the Spanish language. The experimental results show that CharWNN is effective and robust for Portuguese and Spanish NER. In this context, CharWNN significantly outperformed the previous state-of-the-art in both the overall and selective scenarios, achieving an F1 score of 65.4% and 71.2% for the total and selective scenarios, respectively. Moreover, the presented results show that CharWNN also achieves state-of-the-art results for the Spanish dataset. The authors claim that the main difference with previous work is the use of neural character embeddings, which is responsible for the good performance on NER.

Castro, Silva and Soares (2018) investigated the LSTM-CRF model in Portuguese using character-based word representations and word embeddings. The architecture consists of a BiLSTM network connected to a CRF layer responsible for sequential classification. In the work, four different pre-trained word embeddings were tried: fastText (BOJANOWSKI et al., 2017), GloVe (PENNINGTON; SOCHER; MANNING, 2014), Wang2Vec (LING et al., 2015) and Word2Vec (MIKOLOV et al., 2013). The MiniHAREM dataset was used for evaluation, while the HAREM I was used for training. The best-reported result was an F1-score of 70.33% for the total scenario and 76.27% for the selective scenario when Wang2Vec was used as the word embedding model. In addition, it was reported that the results were improved by a technique in which words were normalized to their lower case before creating the dictionaries used for word-embedding lookup.

The use of Flair Embeddings was explored by Santos et al. (2019), which pre-trained it on a corpus of 4.9 billion words corpus of raw Portuguese text and named it FlairBBP. Three large corpora were used for training: BlogSet-BR, brWaC, and ptwiki-20190301. In this context, the combination of Flair Embeddings with traditional word embeddings was studied. The result was the concatenation of the FlairBBP with the Word2Vec embeddings. Finally, the final architecture combines FlairBBP as input with a BiLSTM-CRF. The BiLSTM-CRF+FlairBBP model evaluated on MiniHAREM achieved an F1-score of 74.64% and 82.26% for the total and selective scenario, respectively.

Souza et al. (2020) pre-trained an LM for the Portuguese language. The model, named BERTimbau, was trained over the BERT model and with data from the dataset brWaC. The work was evaluated on three NLP tasks: sentence textual similarity, recognizing textual entailment, and NER. BERTimbau improved the state-of-the-art on the three mentioned tasks. For the NER task, six different models were developed over the pre-trained LM. The authors claim that of the six models, BERTimbau Large + CRF was the one that performed better on the NER, achieving an F1-score of 78.5% for the total scenario and 83.7% for the selective scenario.

Motivated by the same idea, Carmo et al. (2020) also pre-trained an LM for the Portuguese language, called PTT5. In this work, a T5 model (RAFFEL et al., 2019) was pre-trained on the BrWac corpus and later validated on sentence entailment prediction and NER. The authors were motivated by the work presented by (SOUZA et al., 2020). However, the idea of using T5 was driven by its ability to generate text and perform tasks that BERT cannot, such as summarization, abstractive question answering, and translation. The LM follows a common technique of masking part of the tokens of the input sequence. This masked token sequence is fed into the model, which is then trained to produce the original sequence. The model was evaluated on MiniHAREM for the selective scenario only, where it achieved an F1-score of 82.00%.

Tables 3.1 and 3.2 show a comparison of the results of all the approaches mentioned in this chapter. The results were obtained from the original papers mentioned in the tables. Table 3.1 shows that for the total scenario, the BERTimbau Large + CRF architecture outperforms all other works by a large margin. For the selective scenario, Table 3.2 shows that the BERTimbau Large + CRF architecture is still ahead of the other works. However, in this scenario, the difference with other approaches is not so large. Moreover, PTT5 Base + CRF reported the best precision metric among the works.

Table 3.1: Results of the NER task (Precision, Recall, and micro F1-score) on the test set (MiniHAREM) for different approaches for the total scenario. All listed methods were trained on HAREM I golden collection. The best results are shown in bold.

Architecture	**Total scenario**		
	Prec.	**Rec.**	**F1**
CharWNN (SANTOS; GUIMARAES, 2015)	67.2	63.7	65.4
LSTM-CRF (CASTRO; SILVA; SOARES, 2018)	72.8	68.0	70.3
BiLSTM-CRF + FlairBBP (SANTOS et al., 2019)	74.9	74.4	74.6
BERTimbau Large + CRF (SOUZA et al., 2020)	**79.6**	**77.4**	**78.5**

Source: the authors

Table 3.2: Results of the NER task (Precision, Recall, and micro F1-score) on the test set (MiniHAREM) for different approaches for the selective scenario. All listed methods were trained on HAREM I golden collection. The best results are shown in bold.

Architecture	**Selective scenario**		
	Prec.	**Rec.**	**F1**
CharWNN (SANTOS; GUIMARAES, 2015)	74.0	68.7	71.2
LSTM-CRF (CASTRO; SILVA; SOARES, 2018)	78.3	74.4	76.3
BiLSTM-CRF + FlairBBP (SANTOS et al., 2019)	83.4	81.2	82.3
BERTimbau Large + CRF (SOUZA et al., 2020)	84.9	**82.5**	**83.7**
PTT5 Base + CRF (CARMO et al., 2020)	**85.5**	78.8	82.0

Source: the authors

4 MATERIALS AND METHODS

In this work, we propose a new approach to NER in Portuguese based on recent advances in NER in English. The current state-of-the-art in this task for English was achieved by ACE (WANG et al., 2020) on the CoNNL03 dataset. With this in mind, we propose to use the same architecture to automate the process of finding better concatenations of embeddings for the NER task in Portuguese. In this chapter, we describe all the experiments we performed and how they were evaluated.

4.1 Dataset

We evaluated our experiments with the golden collections of the first HAREM evaluation contests (SANTOS et al., 2006), first HAREM and MiniHAREM, as training and test sets, respectively. Additionally, we use 7% of First HAREM documents as a holdout validation set. The decision to use the first HAREM rather than the second is based on the fact that most related works use the first HAREM golden collections, which allows a better comparison between our approach and others. Both the first HAREM and MiniHAREM contain documents annotated with ten NE classes: ABSTRACCAO, ACONTECIMENTO, COISA, LOCAL, OBRA, ORGANIZACAO, OUTRO, PESSOA, TEMPO, and VALOR. However, we also use the dataset for two different scenarios, *total* and *selective*, following previous work (SANTOS; GUIMARAES, 2015; CASTRO; SILVA; SOARES, 2018; SANTOS et al., 2019; SOUZA et al., 2020; CARMO et al., 2020). The total scenario contains all ten classes present in HAREM, while the selective scenario contains only five of them: PESSOA, ORGANIZACAO, LOCAL, VALOR, and TEMPO.

It is a common practice in the NER task to use a tagging scheme when labeling NEs. In this work, we use two of them, as described by Kudo and Matsumoto (2001):

- IOB2: Introduced in Sang and Veenstra (1999), this method uses the following set of three tags for representing NEs:

  I Current token is inside of a NE.

  O Current token is outside of any NE.

  B Current token is the beginning of a NE.

- IOBES: Introduced in Uchimoto et al. (2000), this method uses the following set of five tags for representing NEs:

I Current token is a middle of a NE consisting of more than two tokens.

O Current token is outside of any NE.

B Current token is the start of a NE consisting of more than one token.

E Current token is the end of a NE consisting of more than one token.

S Current token is a NE consisting of only one token.

In this case, the available ACE algorithm requires as input a dataset according to the IOB2 scheme. However, this scheme is changed to IOBES during the training process. Table 4.1 contains the difference between the tagging schemes IOB2 and IOBES tagging schemes when applied to the sentence "Alex está indo com Bruno H. Silva para São Paulo".

Table 4.1: Comparative of IOB2 and IOBES tagging schemes applied to the sentence "*Alex está indo com Bruno H. Silva para São Paulo*".

	IOB2	**IOBES**
Alex	B-PESSOA	S-PESSOA
está	O	O
indo	O	O
com	O	O
Bruno	B-PESSOA	B-PESSOA
H.	I-PESSOA	I-PESSOA
Silva	I-PESSOA	E-PESSOA
para	O	O
São	B-LOCAL	B-LOCAL
Paulo	I-LOCAL	E-LOCAL

Source: the authors

ACE allows us to use both sentence and document-level for the training process. However, the document-level requires much more RAM and GPU memory. For this reason, we chose to use a sentence-level approach. Nevertheless, HAREM golden collections consist of documents that are not partitioned into sentences. For this reason, we applied a sentence tokenizer to the entire dataset. For this purpose, we used a Python library published by Bird, Klein and Loper (2009). The statistics for the HAREM I corpora are listed in Table 4.2. Notice that the selective scenario, which has only five classes, corresponds to 82% of the entities.

Table 4.2: Dataset statistics for the HAREM I corpora. The Tokens column refers to whitespace and punctuation tokenization.

Dataset	Documents	Tokens	Entities in scenario	
			Selective	Total
First HAREM	129	95,585	4,151	5,017
MiniHAREM	128	64,853	3,018	3,642

Source: Souza et al. (2020, p. 407)

4.2 Experimental Setup

We use the ACE code published by its authors to automate the process of finding better concatenations of embeddings for the Portuguese NER. First, we define the candidate embeddings used in the algorithm: BERTimbau base, multilingual BERT (M-BERT) base cased, ELMo, fastText word embeddings, GloVe word embeddings, Flair and, multilingual Flair (M-Flair). For ELMo and Flair, we used the Portuguese model provided by them. The choice of these embeddings was based on the experiments reported by the authors of ACE (WANG et al., 2020), which present results for English, German, Spanish, and Dutch. However, in this work we use BERTimbau as a candidate embedding, since it is considered the current state-of-the-art for Portuguese NER. The sources of the embeddings we use are listed in Table 4.3.

Table 4.3: The embeddings we used in our experiments. The URL is where we downloaded the embeddings.

Embedding	Resource	URL
GloVe	Pennington, Socher and Manning (2014)	<https://nlp.stanford.edu/projects/glove>
fastText	Bojanowski et al. (2017)	<https://github.com/facebookresearch/fastText>
ELMo	Schuster et al. (2019)	<https://github.com/TalSchuster/CrossLingualContextualEmb>
M-BERT	Devlin et al. (2018)	<https://huggingface.co/bert-base-multilingual-cased>
BERTimbau	Souza et al. (2020)	<https://huggingface.co/neuralmind/bert-base-portuguese-cased>
Flair	Akbik, Blythe and Vollgraf (2018)	<https://github.com/flairNLP/flair-lms>
M-Flair	Akbik, Blythe and Vollgraf (2018)	<https://github.com/flairNLP/flair-lms>

Source: the authors

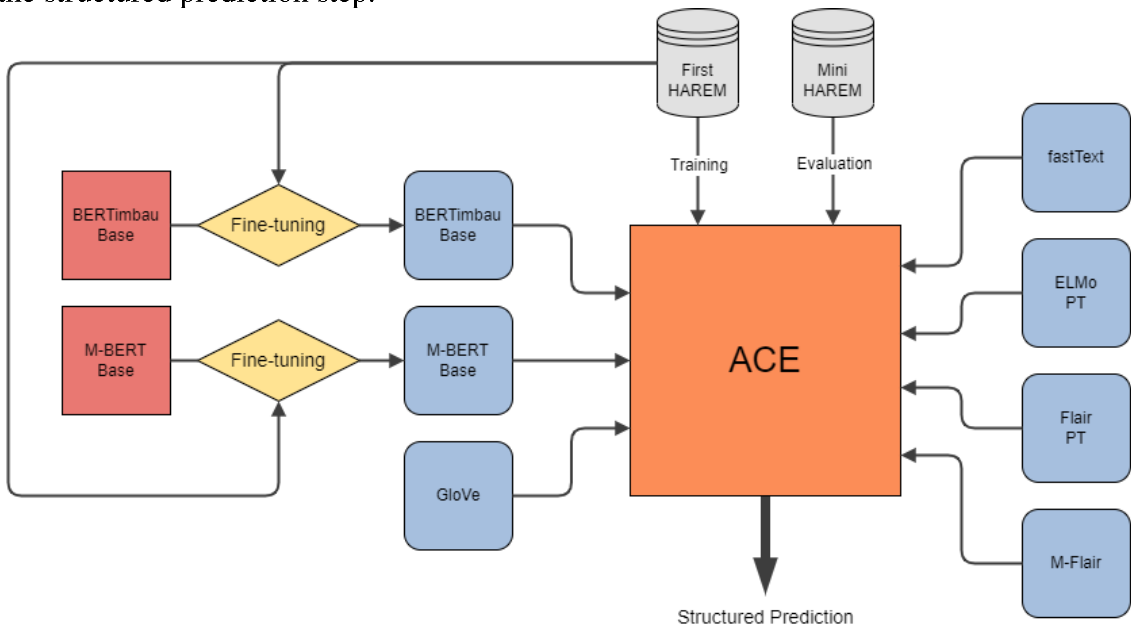
Wang et al. (2020) suggests that the transformer-based embeddings should be fine-tuned before they are set as a candidate embedding. This is a common approach in the literature that can lead to better accuracy. In the case of NER, previous work uses the fine-tuning pipeline of BERT, which combines the BERT model with a linear layer for word-level classification. This step must be done before applying ACE to the embeddings, as this would make the algorithm very slow.

In this case, we fine-tuned BERTimbau and M- BERT for both total and selective scenarios. Here we use the optimizer AdamW (LOSHCHILOV; HUTTER, 2017) with a learning rate of 5×10^{-6}. Both models were trained for 50 epochs for the NER task, with a batch size of 12 for BERTimbau and 8 for M-BERT.

The controller is trained in 30 steps and the task model with the highest accuracy on the development set is stored for later evaluation. For this process, we use a stochastic gradient descent optimizer with a learning rate of 0.1 and a batch size of 16 sentences. We increase the learning rate by 0.5 if the accuracy on the development set has not improved for 5 epochs. We set the maximum training epoch to 150. Each of the controller's parameters is initially set to 0, so that each candidate is selected equally in the first two time steps. The choice of these hyper-parameters values was based on the experiments reported by the authors of ACE (WANG et al., 2020), which performed a grid search for the model.

Figure 4.1 illustrates a general overview of the proposed approach. Blue boxes represent candidate embeddings. Red boxes represent transformer-based embeddings that go through a fine-tuning process before becoming candidate embedding. ACE is represented by an orange box, which in this case includes the search for concatenations of embeddings and the structured prediction step. In this case, First HAREM is used as the training dataset, while MiniHAREM is used as the test dataset. It is important to note that the whole process shown in the figure takes place once for each defined scenario.

Figure 4.1: The proposed architecture for the Portuguese NER. Blue boxes represent candidate embeddings. Red boxes represent transformer-based embeddings that go through a fine-tuning process before becoming a candidate embedding. ACE is represented by an orange box, which in this case includes the search for concatenations of embeddings and the structured prediction step.



Source: the authors

4.3 Evaluation

NER performance is commonly evaluated using the following metrics, as described in Derczynski (2016):

- Precision: represents the proportion of elements - in this case entities - returned by the system that are exactly right. It rewards careful selection and penalises overzealous systems that return too many results: to achieve high precision, discard anything that might not be correct. False positives - false entities - reduce precision. Precision is defined as:

$$P = \frac{|\text{ true positives }|}{|\text{ true positives } | + | \text{ false positives }|}$$

- Recall: indicates how much of all items that should be found were found. This metric rewards comprehensiveness: to get a high recall, it is better to include entities you are not sure about. False negatives - entities - result in a low recall. It balances out precision. Recall is defined as:

$$R = \frac{|\text{ true positives }|}{|\text{ true positives } | + | \text{ false negatives }|}$$

- $F_\beta - Score$: precision and recall can be balanced together. These extreme situations in which they are exploited contrast with each other: finding everything results in only a base precision, and finding only one thing usually results in a very low recall score. Therefore, it is common to combine precision and recall with a weighted harmonic mean (RIJSBERGEN, 1974). The F1-score is defined as:

$$F_\beta = \left(1 + \beta^2\right) \frac{PR}{\beta^2 P + R}$$

  In this equation, the coefficient β determines the balance between precision and recall, with high values favouring recall. This is a harmonic weighted average of precision and recall. In our evaluation, as well in the related works, we use $\beta = 1.0$, and due to this reason, we refer to this metric as F1-Score.

These metrics are computed for each class of NEs. However, we can combine them in different ways to get an overview of the results for the whole set of classes. In the case of the F1-Score, we refer to these combinations as averaged F1-Scores. Following previous work in NER (SOUZA et al., 2020; SANTOS et al., 2019; CARMO et al., 2020), we evaluate our results based on the micro F1-Score, which is computed based on micro-averaged precision and micro-averaged recall. Takahashi et al. (2021) describe them as:

$$\text{miP} = \frac{\sum_{i=1}^{r} TP_i}{\sum_{i=1}^{r} (TP_i + FP_i)} = \frac{\sum p_{ii}}{\sum p_{i\cdot}} = \sum_{i=1}^{r} p_{ii}$$

$$\text{miR} = \frac{\sum_{i=1}^{r} TP_i}{\sum_{i=1}^{r} (TP_i + FN_i)} = \frac{\sum p_{ii}}{\sum p_{\cdot i}} = \sum_{i=1}^{r} p_{ii}$$

Where miP is the micro-averaged precision and miR is the micro-averaged recall. Finally, the micro-averaged F1-Score is defined as the harmonic mean of these quantities:

$$miF_1 = 2\frac{miP \times miR}{miP + miR} = \sum_{i=1}^{r} p_{ii}$$

Typically, NER performance is evaluated using the evaluation script published by CoNLL 2003 (SANG; MEULDER, 2003), which computes precision, recall, and micro F1-Score for entity-level. However, this script requires the use of the IOB1 tagging scheme originally used for the CoNLL shared task. Since the prediction results computed by ACE use the IOBES tagging scheme, we evaluated the performance of our experiments using Nakayama (2018), a very well-known Python framework for sequence labeling evaluation that has been well tested using the evaluation script published by CoNLL.

5 RESULTS

We propose the use of ACE (WANG et al., 2020) to find better concatenations of embeddings for the Portuguese NER. ACE is the current state-of-the-art approach for English NER, which has been evaluated on the CoNLL2003 shared task and can find good concatenation of embeddings using NAS techniques. Moreover, we believe that recent progress on Portuguese LMs (SOUZA et al., 2020) could be useful for our approach.

In our experiments, we trained ACE with the HAREM I dataset for the following set of candidate embeddings: BERTimbau base, multilingual BERT (M-BERT) base cased, ELMo, fastText word embeddings, GloVe word embeddings, Flair and, multilingual Flair (M-Flair). For ELMo and Flair, we use the Portuguese model provided by them. In addition, we fine-tuned BERTimbau and M- BERT on the dataset before using them with ACE.

We evaluated the performance of our experiments using seqeval (NAKAYAMA, 2018), a Python library. In this case, the embeddings were concatenated as input to a BiLSTM-CRF model that predicted the results for the MiniHAREM, our test set. We then compute the precision, recall, and micro F1-Score for the entity-level using the predicted results. Since we consider both the total and selective scenarios, the fine-tuning of BERTimbau and M-BERT as well as the training of ACE with the candidate embeddings were performed twice. All experiments were conducted in an environment with a single NVIDIA GTX 980 Ti 6GB GPU, 128GB RAM, and an Intel Xeon CPU E5-1650 v3 @ 3.50GHz processor. It takes 22 GPU hours to train the controller for each of the scenarios.

We compare the precision, recall, and F1-Score between our approach and related works that have published results on the same datasets. We include the BERTimbau Base + CRF model in our comparisons because we used BERTimbau Base and not the BERTimbau Large in our approach. The results of the other papers in this chapter were obtained from the respective original papers.

5.1 Total Scenario

Table 5.1 lists the comparative performance on NEs for the total scenario between ACE and BiLSTM-CRF + FlairBBP (SANTOS et al., 2019). A general overview of the performance of the different approaches for the total scenario is listed in Table 5.2 and shown in Figure 5.1.

Considering Table 5.1, ACE outperformed BiLSTM-CRF + FlairBBP (SANTOS et al., 2019) by a wide margin in terms of precision, recall, and F1-Score for almost all entities in the total scenario. In this case, the results for two entities stand out: ACONTECIMENTO and OBRA, which achieved a positive delta of 20.29% and 19.87%, respectively.

In terms of overall performance for the total scenario, Table 5.2 and Figure 5.1 show that BERTimbau + CRF architecture, both base and large models, outperform all other works, including our approach, in precision, recall and F1-Score. ACE lags behind BERTimbau Base + CRF with the third-highest results.

Table 5.1: Comparison of performance on entities between the proposed method and BiLSTM-CRF + FlairBBP (SANTOS et al., 2019) for the total scenario.

Entity	**BiLSTM-CRF + FlairBBP (2019)**			**ACE**			
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	Δ
ABSTRACCAO	**53.90%**	42.13%	47.29%	50.00%	**55.24%**	**52.49%**	+5.20%
ACONTECIMENTO	22.37%	34.00%	26.98%	**50.00%**	**44.83%**	**47.27%**	+20.29%
COISA	54.72%	35.80%	43.28%	**59.43%**	**36.63%**	**45.32%**	+2.04%
LOCAL	81.40%	**85.75%**	83.52%	**85.32%**	83.69%	**84.50%**	+0.98%
OBRA	46.82%	43.09%	44.88%	**65.61%**	**63.92%**	**64.75%**	+19.87%
ORGANIZACAO	67.23%	**77.07%**	**71.82%**	**69.45%**	71.94%	70.67%	−1.15%
OUTRO	10.00%	7.14%	8.33%	**15.38%**	**12.50%**	**13.79%**	+5.46%
PESSOA	82.44%	77.08%	79.67%	**83.99%**	**82.94%**	**83.46%**	+3.79%
TEMPO	91.43%	**90.40%**	90.91%	**91.60%**	90.33%	**90.96%**	+0.05%
VALOR	**82.92%**	**81.90%**	**82.41%**	81.23%	79.14%	80.17%	−2.24%
Overall	74.91%	74.37%	74.64%	**77.74%**	**76.12%**	**76.92%**	+2.28%

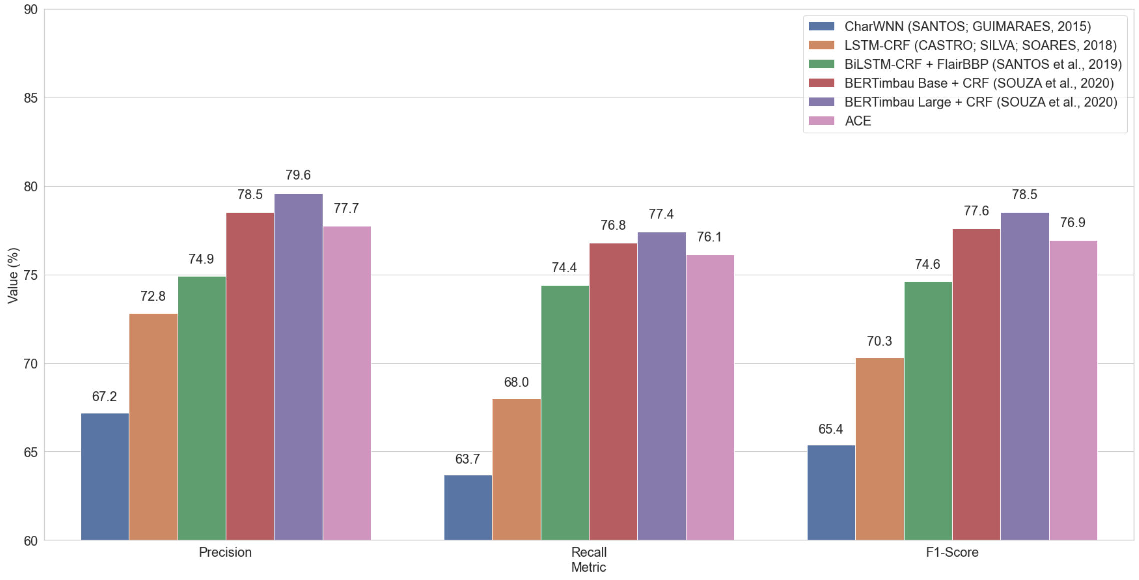
Source: the authors

Table 5.2: Results of the NER task (Precision, Recall and micro F1-score) on the test set (MiniHAREM) for different approaches on the total scenario. All presented methods were trained on HAREM I golden collection. The best results are shown in bold.

Architecture	**Total scenario**		
	Prec.	**Rec.**	**F1**
CharWNN (SANTOS; GUIMARAES, 2015)	67.2%	63.7%	65.4%
LSTM-CRF (CASTRO; SILVA; SOARES, 2018)	72.8%	68.0%	70.3%
BiLSTM-CRF + FlairBBP (SANTOS et al., 2019)	74.9%	74.4%	74.6%
BERTimbau Base + CRF (SOUZA et al., 2020)	78.5%	76.8%	77.6%
BERTimbau Large + CRF (SOUZA et al., 2020)	**79.6%**	**77.4%**	**78.5%**
ACE	77.7%	76.1%	76.9%

Source: the authors

Figure 5.1: Comparison of overall performance for the total scenario.



Source: the authors

5.2 Selective Scenario

Considering the selective scenario, we compare the performance on NEs between ACE and CharWNN (SANTOS; GUIMARAES, 2015), BiLSTM-CRF + FlairBBP (SANTOS et al., 2019), and PTT5 Base + CRF (CARMO et al., 2020) in Tables 5.3, 5.4, and 5.5, respectively. This comparison is also shown in Figure 5.2. A general overview of the performance of the different approaches for the selective scenario is given in Table 5.6 and shown in Figure 5.3.

Considering the performance on entities for the selective scenario, ACE outperformed CharWNN (SANTOS; GUIMARAES, 2015) by a wide margin almost all entities according to the results listed in Table 5.3 and Figure 5.2. The only entity in which CharWNN performed better than ACE was ORGANIZACAO, however, the F1-Score of both approaches are very similar.

Table 5.4 and Figure 5.2 show that the BiLSTM-CRF + FlairBBP architecture (SANTOS et al., 2019) performs better than ACE for almost all entities in the selective scenario on recall and F1-Score. However, our proposed approach can achieve better precision results. We note that the performance for the entity PESSOA, which was the only entity with a positive delta, resulted in an F1-Score very similar to that of the BiLSTM-CRF + FlairBBP architecture due to the choice of micro F1-Score.

Table 5.5 and Figure 5.2 show that ACE outperforms PTT5 Base + CRF (CARMO et al., 2020) for three entities in the selective scenario. The VALOR and TEMPO entities had a large lead in F1-Score, 9.44% and 5.77%, respectively. However, the performance of the PTT5 Base + CRF architecture was 8.37% higher than the F1-Score score obtained by ACE for the entity ORGANIZACAO. In terms of general performance, we can say that both approaches achieved very similar results on all metrics.

In terms of overall performance, Table 5.6 and Figure 5.3 show that BERTimbau + CRF architecture, both base and large models, outperform all other works, including our approach, in recall and F1-Score for the selective scenario. In this case, the best precision is reported by PTT5 Base + CRF. Considering the F1-Score, ACE only lags behind the two BERTimbau + CRF architectures.

For comparisons related to performance on NEs, not all papers published the results needed for a better comparison. Some published results only for the selective scenario. In the case of PTT5 Base + CRF (CARMO et al., 2020), the model was only evaluated in the selective scenario.

Table 5.3: Comparison of performance on entities between the proposed method and CharWNN (SANTOS; GUIMARAES, 2015) for the selective scenario.

Entity	**CharWNN (2015)**			**ACE**			
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	Δ
LOCAL	76.91%	78.55%	77.72%	**84.98%**	**83.65%**	**84.31%**	+6.59%
ORGANIZACAO	70.65%	**71.56%**	**71.10%**	**75.09%**	67.21%	70.93%	−0.17%
PESSOA	81.35%	77.07%	79.15%	**87.98%**	**81.11%**	**84.40%**	+5.25%
TEMPO	90.27%	81.32%	85.56%	**97.44%**	**88.49%**	**91.37%**	+5.81%
VALOR	78.08%	74.99%	76.51%	**82.94%**	**80.57%**	**81.74%**	+5.23%
Overall	78.38%	77.49%	77.93%	**84.81%**	**79.88%**	**82.27%**	+4.79%

Source: the authors

[1]

Table 5.4: Comparison of performance on entities between the proposed method and BiLSTM-CRF + FlairBBP (SANTOS et al., 2019) for the selective scenario.

Entity	**BiLSTM-CRF + FlairBBP (2019)**			**ACE**			
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	Δ
LOCAL	84.78%	**84.68%**	**84.73%**	**84.98%**	83.65%	84.31%	−0.42%
ORGANIZACAO	72.61%	**76.19%**	**74.35%**	**75.09%**	67.21%	70.93%	−3.42%
PESSOA	85.07%	76.84%	80.75%	**87.98%**	**81.11%**	**84.40%**	+3.65%
TEMPO	93.81%	**89.83%**	**91.77%**	**97.44%**	88.49%	91.37%	−0.40%
VALOR	**84.81%**	**82.21%**	**83.49%**	82.94%	80.57%	81.74%	−1.75%
Overall	83.38%	**81.17%**	82.26%	**84.81%**	79.88%	**82.27%**	+0.01%

Source: the authors

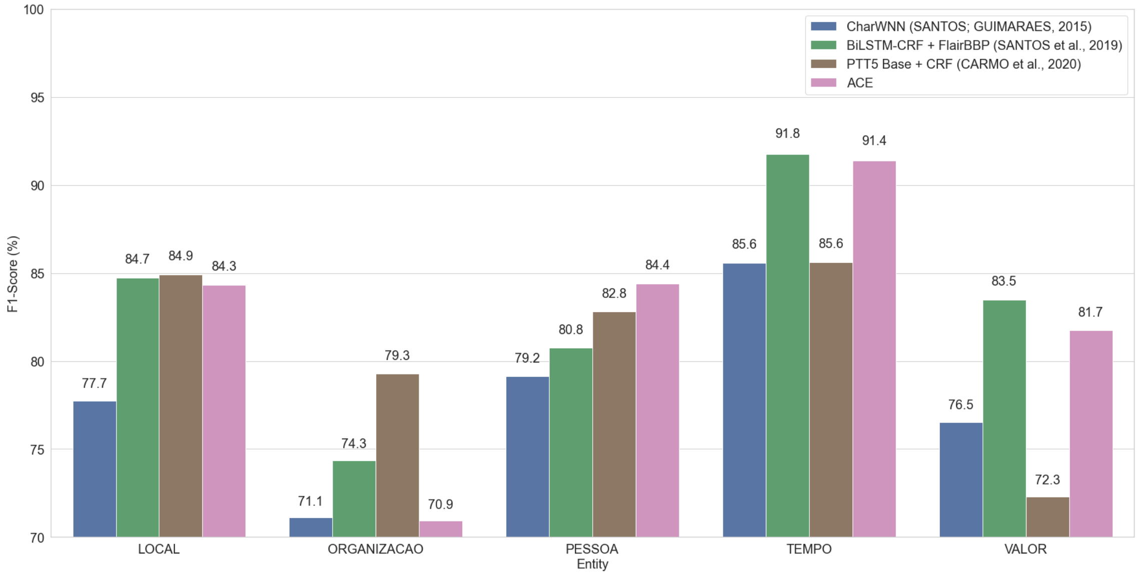
Table 5.5: Comparison of performance on entities between the proposed method and PTT5 Base + CRF (CARMO et al., 2020) for the selective scenario.

Entity	**PTT5 Base + CRF (2020)**			**ACE**			
	Prec.	**Rec.**	**F1**	**Prec.**	**Rec.**	**F1**	Δ
LOCAL	**87.6%**	82.3%	**84.9%**	84.98%	**83.65%**	84.31%	−0.59%
ORGANIZACAO	**80.2%**	**78.4%**	**79.3%**	75.09%	67.21%	70.93%	−8.37%
PESSOA	86.8%	79.2%	82.8%	**87.98%**	**81.11%**	**84.40%**	+1.60%
TEMPO	87.2%	84.1%	85.6%	**97.44%**	**88.49%**	**91.37%**	+5.77%
VALOR	84.4%	63.2%	72.3%	**82.94%**	**80.57%**	**81.74%**	+9.44%
Overall	**85.5%**	78.8%	82.0%	84.81%	**79.88%**	**82.27%**	+0.27%

Source: the authors

[1] Although the results presented in Table 5.3 do not match those presented in Tables 5.6 and 3.2 and Figure 5.3, they were taken from their respective papers, which did not provide more detailed information on how these metrics were calculated.

Figure 5.2: Comparison of performance on entities for the selective scenario.



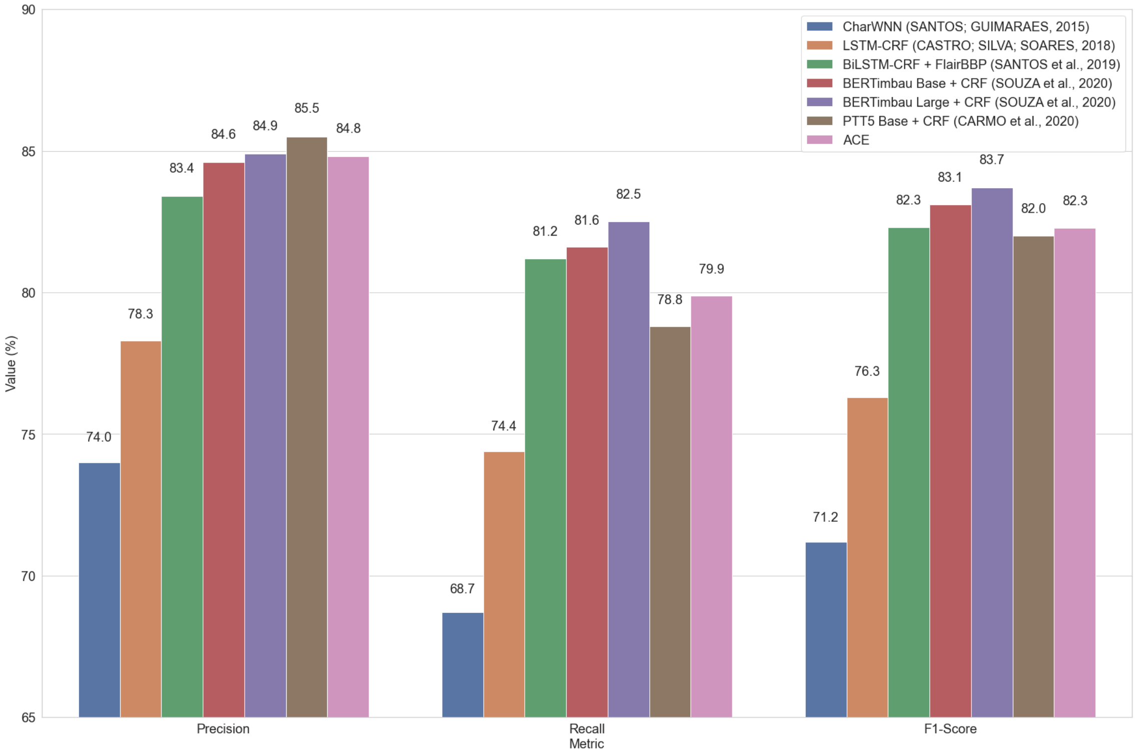
Source: the authors

Table 5.6: Results of the NER task (Precision, Recall and micro F1-score) on the test set (MiniHAREM) for different approaches on the selective scenario. All presented methods were trained on HAREM I golden collection. The best results are shown in bold.

Architecture	**Selective scenario**		
	Prec.	**Rec.**	**F1**
CharWNN (SANTOS; GUIMARAES, 2015)	74.0%	68.7%	71.2%
LSTM-CRF (CASTRO; SILVA; SOARES, 2018)	78.3%	74.4%	76.3%
BiLSTM-CRF + FlairBBP (SANTOS et al., 2019)	83.4%	81.2%	82.3%
BERTimbau Base + CRF (SOUZA et al., 2020)	84.6%	81.6%	83.1%
BERTimbau Large + CRF (SOUZA et al., 2020)	84.9%	**82.5**%	**83.7**%
PTT5 Base + CRF (CARMO et al., 2020)	**85.5**%	78.8%	82.0%
ACE	84.8%	79.8%	82.2%

Source: the authors

Figure 5.3: Comparison of overall performance for the selective scenario.

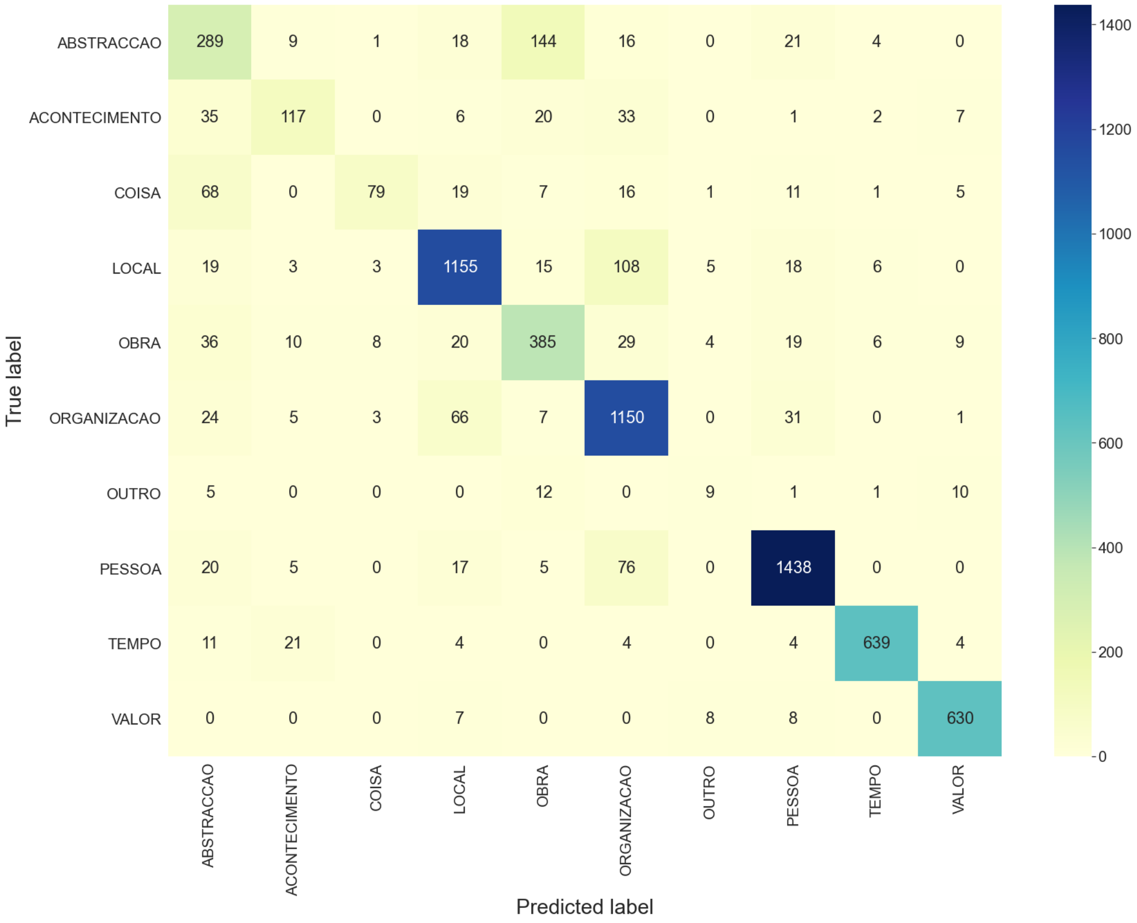


Source: the authors

5.3 Error Analysis

We also evaluate common errors of our system by creating a simplified confusion matrix for the total scenario in Figure 5.4 and the selective scenario in Figure 5.5. These confusion matrices are simplified versions of the original confusion matrices. For example, a complete confusion matrix would be a 40 × 40 matrix for the total scenario, since there are 10 classes and 4 possible prefixes for each class according to the IOBES scheme: B-, I-, S-, E-. In the case of a simplified confusion matrix, we group the results for all possible prefixes for each class. It allows us to see which pairs of NEs are frequently predicted incorrectly.

Figure 5.4: Confusion matrix for the total scenario. This is a simplified token-level confusion matrix. A complete confusion matrix would be a 40 × 40 matrix for the total scenario since there are 10 classes and 4 possible prefixes for each class according to the IOBES scheme: B-, I-, S-, E-. In the case of a simplified confusion matrix, we group the results for all possible prefixes for each class.

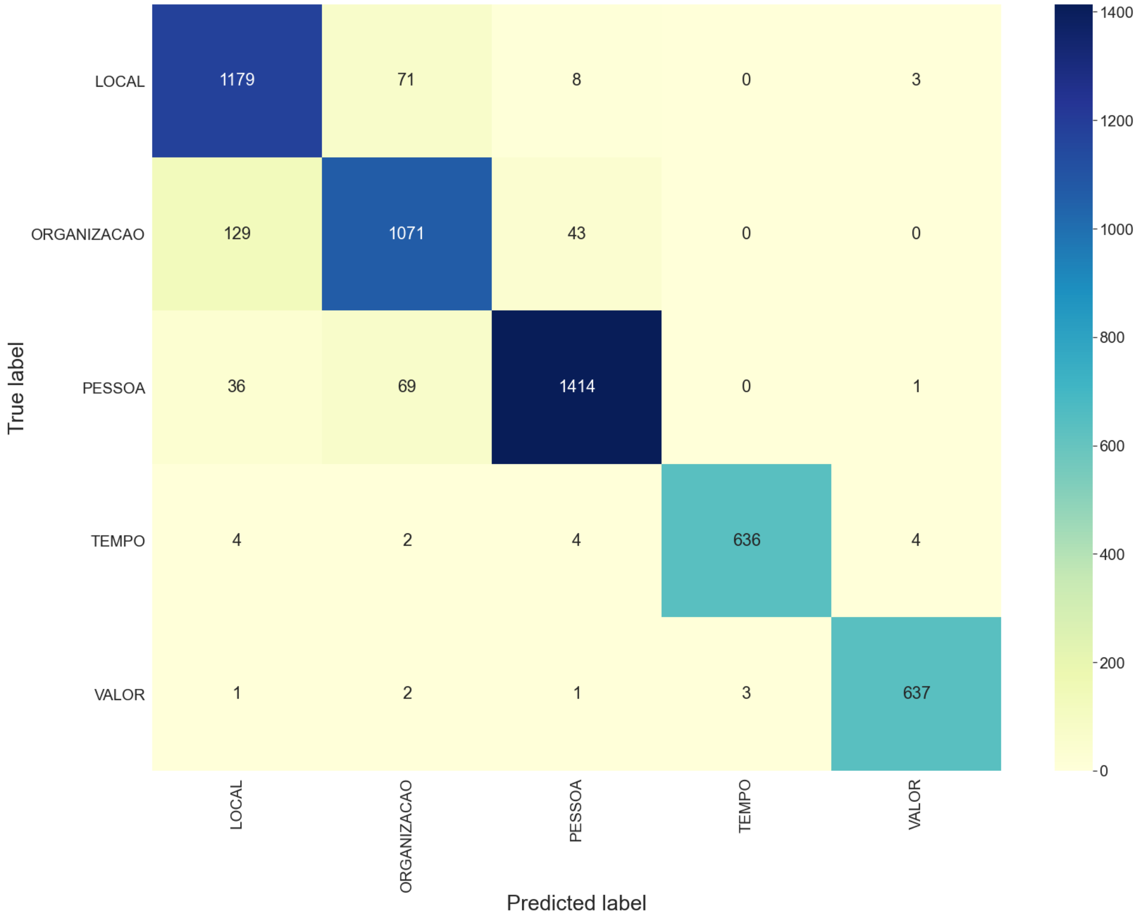


	ABSTRACCAO	ACONTECIMENTO	COISA	LOCAL	OBRA	ORGANIZACAO	OUTRO	PESSOA	TEMPO	VALOR
ABSTRACCAO	289	9	1	18	144	16	0	21	4	0
ACONTECIMENTO	35	117	0	6	20	33	0	1	2	7
COISA	68	0	79	19	7	16	1	11	1	5
LOCAL	19	3	3	1155	15	108	5	18	6	0
OBRA	36	10	8	20	385	29	4	19	6	9
ORGANIZACAO	24	5	3	66	7	1150	0	31	0	1
OUTRO	5	0	0	0	12	0	9	1	1	10
PESSOA	20	5	0	17	5	76	0	1438	0	0
TEMPO	11	21	0	4	0	4	0	4	639	4
VALOR	0	0	0	7	0	0	8	8	0	630

Source: the authors

Looking at the total scenario, Figure 5.4 shows that the NE COISA is frequently misclassified as ABSTRACCAO, with a total of 68 occurrences, while there are only

Figure 5.5: Confusion matrix for the total scenario. This is a simplified token-level confusion matrix. A complete confusion matrix would be a 20 × 20 matrix for the total scenario since there are 5 classes and 4 possible prefixes for each class according to the IOBES scheme: B-, I-, S-, E-. In the case of a simplified confusion matrix, we group the results for all possible prefixes for each class.



Source: the authors

79 correct predictions for this class. The predictions for the NE ABSTRACCAO also stand out, as it is misclassified as OBRA 144 times, which is almost half of the correct predictions. The ACONTECIMENTO class was misclassified as ABSTRACCAO, OBRA, and ORGANIZACAO very frequently, with 35, 20, and 33 occurrences respectively, while there are 117 correct predictions for this class. For the NE OUTRO, most of the predictions are wrong and are mainly distributed among the classes OBRA and VALOR.

In the selective scenario, on the other hand, Figure 5.4 shows that the NE ORGANIZACAO is frequently misclassified as LOCAL with 129 occurrences and PESSOA with 43 occurrences, while there are 1071 correct predictions for this class. The class PESSOA was very often misclassified as ORGANIZACAO and LOCAL with 69 and 36 occurrences respectively, in comparison it was correctly predicted 1414 times. The NE LOCAL is frequently misclassified as ORGANIZACAO, 71 times compared to 1179 instances where the prediction was correct.

6 CONCLUSION

In this work, we tested whether applying the ACE approach to the problem of NER in Portuguese would lead to an improvement in the state-of-the-art. The method chosen was to adapt the work by Wang et al. (2020), which represents the state-of-the-art for structured prediction tasks over six tasks, including English NER, to the HAREM evaluation contest and compare its performance with previous works. To do this, we also used the state-of-the-art for Portuguese LM (SOUZA et al., 2020) as contextual embeddings. We evaluated our system using the metrics Precision, Recall, and F1-Score and found that our results are behind the state-of-the-art achieved by BERTimbau + CRF architecture (SOUZA et al., 2020), which is the only work our system cannot outperform, considering the total scenario. In the selective scenario, the proposed approach lags behind BERTimbau Large + CRF (SOUZA et al., 2020) and BiLSTM-CRF + FlairBBP (SANTOS et al., 2019), but performs better than other related works, including PTT5 Base + CRF (PETERS et al., 2018).

Our approach has produced results that are not as good as we expected. Since we use BERTimbau, we expect that using it together with other embeddings would give better results than using it alone, or at least similar performance. However, our experiments are not sufficient to explain why we did not achieve this.

In future work, we plan to continue using ACE, but add other embeddings and change the parameters. In Wang et al. (2020), the author suggests that the use of document-level word representations and other transformer-based embeddings, such as XLNet (YANG et al., 2019) and XLM-R (CONNEAU et al., 2019), could improve the performance of this NER approach. In this work, we only use sentence-level word representation and not XLNet or XLM-R. The decision not to use them was because they would require a more powerful computing environment. However, we believe that using these new features in future work would lead to better results than those shown in this work.

REFERENCES

AKBIK, A.; BLYTHE, D.; VOLLGRAF, R. Contextual string embeddings for sequence labeling. In: **Proceedings of the 27th International Conference on Computational Linguistics**. Santa Fe, New Mexico, USA: Association for Computational Linguistics, 2018. p. 1638–1649. Available from Internet: <https://aclanthology.org/C18-1139>.

ALZAIDY, R.; CARAGEA, C.; GILES, C. L. Bi-lstm-crf sequence labeling for keyphrase extraction from scholarly documents. In: **The world wide web conference**. [S.l.: s.n.], 2019. p. 2551–2557.

BACCOUCHE, M. et al. Sequential deep learning for human action recognition. In: SPRINGER. **International workshop on human behavior understanding**. [S.l.], 2011. p. 29–39.

BAHDANAU, D.; CHO, K.; BENGIO, Y. Neural machine translation by jointly learning to align and translate. **arXiv preprint arXiv:1409.0473**, 2014.

BIRD, S.; KLEIN, E.; LOPER, E. **Natural language processing with Python: analyzing text with the natural language toolkit**. [S.l.]: " O'Reilly Media, Inc.", 2009.

BLUMBERG, R.; ATRE, S. The problem with unstructured data. **Dm Review**, POWELL PUBLISHING INC, v. 13, n. 42-49, p. 62, 2003.

BOJANOWSKI, P. et al. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 5, p. 135–146, 2017.

BOUKTIF, S. et al. Single and multi-sequence deep learning models for short and medium term electric load forecasting. **Energies**, Multidisciplinary Digital Publishing Institute, v. 12, n. 1, p. 149, 2019.

CARMO, D. et al. Ptt5: Pretraining and validating the t5 model on brazilian portuguese data. **arXiv preprint arXiv:2008.09144**, 2020.

CARVALHO, P. et al. Segundo harem: Modelo geral, novidades e avaliaçao. **quot; In Cristina Mota; Diana Santos (ed) Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM Linguateca 2008**, Linguateca, 2008.

CARVALHO, W. S. **Reconhecimento de entidades mencionadas em português utilizando aprendizado de máquina**. Thesis (PhD) — Universidade de São Paulo, 2012.

CASTRO, P. V. Q. de; SILVA, N. F. F. da; SOARES, A. da S. Portuguese named entity recognition using lstm-crf. In: SPRINGER. **International Conference on Computational Processing of the Portuguese Language**. [S.l.], 2018. p. 83–92.

CHENG, J.; DONG, L.; LAPATA, M. Long short-term memory-networks for machine reading. **arXiv preprint arXiv:1601.06733**, 2016.

CONNEAU, A. et al. Unsupervised cross-lingual representation learning at scale. **arXiv preprint arXiv:1911.02116**, 2019.

CONNEAU, A. et al. Unsupervised cross-lingual representation learning at scale. In: **Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics**. Online: Association for Computational Linguistics, 2020. p. 8440–8451. Available from Internet: <https://aclanthology.org/2020.acl-main.747>.

DERCZYNSKI, L. Complementarity, f-score, and nlp evaluation. In: **Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)**. [S.l.: s.n.], 2016. p. 261–266.

DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018.

DODDINGTON, G. R. et al. The automatic content extraction (ace) program-tasks, data, and evaluation. In: LISBON. **Lrec**. [S.l.], 2004. v. 2, n. 1, p. 837–840.

FILHO, J. A. W. et al. The brwac corpus: A new open resource for brazilian portuguese. In: **Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)**. [S.l.: s.n.], 2018.

GLASS, M. et al. Span selection pre-training for question answering. **arXiv preprint arXiv:1909.04120**, 2019.

GRAVES, A.; SCHMIDHUBER, J. Framewise phoneme classification with bidirectional lstm and other neural network architectures. **Neural networks**, Elsevier, v. 18, n. 5-6, p. 602–610, 2005.

GRISHMAN, R.; SUNDHEIM, B. M. Message understanding conference-6: A brief history. In: **COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics**. [S.l.: s.n.], 1996.

HOCHREITER, S.; HEUSEL, M.; OBERMAYER, K. Fast model-based protein homology detection without alignment. **Bioinformatics**, Oxford University Press, v. 23, n. 14, p. 1728–1736, 2007.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HUANG, Z.; XU, W.; YU, K. Bidirectional lstm-crf models for sequence tagging. **arXiv preprint arXiv:1508.01991**, 2015.

ISLAM, M. S.; HOSSAIN, E. Foreign exchange currency rate prediction using a gru-lstm hybrid network. **Soft Computing Letters**, Elsevier, p. 100009, 2020.

KUDO, T.; MATSUMOTO, Y. Chunking with support vector machines. In: **Second Meeting of the North American Chapter of the Association for Computational Linguistics**. [S.l.: s.n.], 2001.

KULL, M.; KUHAUPT, N. Application and evaluation of lstm architectures for energy time-series forecasting.

LAFFERTY, J.; MCCALLUM, A.; PEREIRA, F. C. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

LI, J. et al. A survey on deep learning for named entity recognition. **IEEE Transactions on Knowledge and Data Engineering**, IEEE, 2020.

LING, W. et al. Two/too simple adaptations of word2vec for syntax problems. In: **Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies**. [S.l.: s.n.], 2015. p. 1299–1304.

LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. **arXiv preprint arXiv:1711.05101**, 2017.

LUONG, M.-T.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. **arXiv preprint arXiv:1508.04025**, 2015.

MERCHANT, R.; OKUROWSKI, M. E.; CHINCHOR, N. **The multilingual entity task (MET) overview**. [S.l.], 1996.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

NADEAU, D.; SEKINE, S. A survey of named entity recognition and classification. **Lingvisticae Investigationes**, John Benjamins, v. 30, n. 1, p. 3–26, 2007.

NAKAYAMA, H. **seqeval: A Python framework for sequence labeling evaluation**. 2018. Software available from https://github.com/chakki-works/seqeval. Available from Internet: <https://github.com/chakki-works/seqeval>.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: **Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)**. [S.l.: s.n.], 2014. p. 1532–1543.

PETERS, M. E. et al. Deep contextualized word representations. In: **Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)**. New Orleans, Louisiana: Association for Computational Linguistics, 2018. p. 2227–2237. Available from Internet: <https://aclanthology.org/N18-1202>.

RADFORD, A. et al. Improving language understanding with unsupervised learning. Technical report, OpenAI, 2018.

RAFFEL, C. et al. Exploring the limits of transfer learning with a unified text-to-text transformer. **arXiv preprint arXiv:1910.10683**, 2019.

RAGANATO, A.; TIEDEMANN, J. et al. An analysis of encoder representations in transformer-based machine translation. In: THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP**. [S.l.], 2018.

RAO, Y. et al. Hybrid feature-based sentiment strength detection for big data applications. In: **Multimodal analytics for next-generation big data technologies and applications**. [S.l.]: Springer, 2019. p. 73–91.

RIJSBERGEN, C. J. V. Foundation of evaluation. **Journal of documentation**, MCB UP Ltd, 1974.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. **Learning internal representations by error propagation**. [S.l.], 1985.

RYDNING, D. R.-J. G.-J. The digitization of the world from edge to core. **Framingham: International Data Corporation**, 2018.

SANG, E. F.; MEULDER, F. D. Introduction to the conll-2003 shared task: Language-independent named entity recognition. **arXiv preprint cs/0306050**, 2003.

SANG, E. F.; VEENSTRA, J. Representing text chunks. **arXiv preprint cs/9907006**, 1999.

SANTOS, C. D.; ZADROZNY, B. Learning character-level representations for part-of-speech tagging. In: PMLR. **International Conference on Machine Learning**. [S.l.], 2014. p. 1818–1826.

SANTOS, C. N. d.; GUIMARAES, V. Boosting named entity recognition with neural character embeddings. **arXiv preprint arXiv:1505.05008**, 2015.

SANTOS, D.; CARDOSO, N. **Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área**. 2007.

SANTOS, D. et al. Harem: An advanced ner evaluation contest for portuguese. In: **quot; In Nicoletta Calzolari; Khalid Choukri; Aldo Gangemi; Bente Maegaard; Joseph Mariani; Jan Odjik; Daniel Tapias (ed) Proceedings of the 5 th International Conference on Language Resources and Evaluation (LREC'2006)(Genoa Italy 22-28 May 2006)**. [S.l.: s.n.], 2006.

SANTOS, J. et al. Assessing the impact of contextual embeddings for portuguese named entity recognition. In: IEEE. **2019 8th Brazilian Conference on Intelligent Systems (BRACIS)**. [S.l.], 2019. p. 437–442.

SCHMIDHUBER, J.; WIERSTRA, D.; GOMEZ, F. J. Evolino: Hybrid neuroevolution/optimal linear search for sequence prediction. In: **Proceedings of the 19th International Joint Conferenceon Artificial Intelligence (IJCAI)**. [S.l.: s.n.], 2005.

SCHUSTER, T. et al. Cross-lingual alignment of contextual word embeddings, with applications to zero-shot dependency parsing. **arXiv preprint arXiv:1902.09492**, 2019.

SOUZA, F. C. d. et al. Bertimbau: pretrained bert models for brazilian portuguese= bertimbau: modelos bert pré-treinados para português brasileiro. [sn], 2020.

SUNDHEIM, B. M. Overview of results of the muc-6 evaluation. NAVAL COMMAND CONTROL AND OCEAN SURVEILLANCE CENTER SAN DIEGO CA, 1995.

TAKAHASHI, K. et al. Confidence interval for micro-averaged f1 and macro-averaged f1 scores. **Applied Intelligence**, Springer, p. 1–12, 2021.

UCHIMOTO, K. et al. Named entity extraction based on a maximum entropy model and transformation rules. In: **proceedings of the 38th Annual Meeting of the Association for Computational Linguistics**. [S.l.: s.n.], 2000. p. 326–335.

VASWANI, A. et al. Attention is all you need. **arXiv preprint arXiv:1706.03762**, 2017.

VIG, J.; BELINKOV, Y. Analyzing the structure of attention in a transformer language model. **arXiv preprint arXiv:1906.04284**, 2019.

VITERBI, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. **IEEE transactions on Information Theory**, IEEE, v. 13, n. 2, p. 260–269, 1967.

WANG, S.; ZHOU, W.; JIANG, C. A survey of word embeddings based on deep learning. **Computing**, Springer, v. 102, n. 3, p. 717–740, 2020.

WANG, X. et al. Automated concatenation of embeddings for structured prediction. **arXiv preprint arXiv:2010.05006**, 2020.

WU, Y. et al. Google's neural machine translation system: Bridging the gap between human and machine translation. **arXiv preprint arXiv:1609.08144**, 2016.

XU, W. et al. Symmetric regularization based bert for pair-wise semantic reasoning. In: **Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval**. [S.l.: s.n.], 2020. p. 1901–1904.

YADAV, V.; BETHARD, S. A survey on recent advances in named entity recognition from deep learning models. **arXiv preprint arXiv:1910.11470**, 2019.

YANG, Z. et al. Xlnet: Generalized autoregressive pretraining for language understanding. **Advances in neural information processing systems**, v. 32, 2019.

ZHANG, Z. et al. Semantics-aware bert for language understanding. In: **Proceedings of the AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2020. v. 34, n. 05, p. 9628–9635.

APPENDIX A — HAREM CATEGORIES

HAREM categories, types, and subtypes defined according to Santos and Cardoso (2007) and Carvalho et al. (2008).

A.1 Category ABSTRACCAO

A.1.1 Type DISCIPLINA

It encompasses scientific disciplines, theories, technologies, and practices (e.g. Inteligência Artificial, Neurofisiologia, Teoria da Relatividade, GSM, Tai-Chi, Futebol de 5, Java).

A.1.2 Type ESTADO

Physical states, conditions, or functions (e.g. Doença de Alzheimer, Sistema Nervoso Central).

A.1.3 Type IDEIA

Ideas or ideals are often NEs that represent abstract concepts but are usually referenced by other more concrete concepts. In the example "A honra da França estava em jogo", the abstract concept is "honra", taken from the reference "França".

A.1.4 Type NOME

Represents only a name. For example, in the sentence "Achei um cão. Vou dar-lhe o nome de Bobi", "Bobi" is a NE of the NOME type.

A.2 Category ACONTECIMENTO

A.2.1 Type EFEMERIDE

An event that occurred in the past and cannot be repeated (e.g. Revolução Francesa, Segunda Guerra Mundial).

A.2.2 Type EVENTO

An event that can occur for a period in time and can contain other sub-events (e.g. Copa do Mundo, Jogos Olímpicos).

A.2.3 Type ORGANIZADO

One-time event, organized or not (e.g. Rolling Stones em Copacabana).

A.3 Category COISA

A.3.1 Type CLASSE

A collection of objects is called by a single name, such as brands, models, and pedigrees. For example, in the sentence "a lâmpada de Edison", "de Edison" is a NE of the CLASSE type.

A.3.2 Type MEMBROCLASSE

This type includes NEs, which refers to an instantiation of classes, that is, to particular objects referred to by the class to which they belong. This includes products that are marketed and referred to by a brand or company. For example, in the sentence "Os pastéis de Belém têm muita fama", "de Belém" is a NE of the MEMBROCLASSE type.

A.3.3 Type OBJECTO

Refers to a specific object or structure designated by a proper name. Includes planets, stars, comets, and suns. It may also include specific objects (e.g. Marte, Titanic).

A.3.4 Type SUBSTANCIA

Refers to elementary substances that cannot be considered objects because they cannot be counted (e.g. paracetamol, água).

A.4 Category LOCAL

A.4.1 Type FISICO

1. Subtype AGUACURSO: Rivers, streams, creeks, waterfalls, etc;
2. Subtype AGUAMASSA: Lakes, seas, oceans, gulfs, straits, channels, basins, dams, etc;
3. Subtype ILHA: Islands and archipelagos;
4. Subtype PLANETA: All celestial bodies;
5. Subtype REGIAO: Designates a geographical/natural region or the continents viewed as a region of physical geography (e.g. Balcãs, região do Amazonas, Deserto do Sahara);
6. Subtype RELEVO: Mountains, ranges, hills, sierras, plains, plateaus, valleys, etc;
7. Subtype OUTRO: Every other NEs that may belong to this type.

A.4.2 Type HUMANO

1. Subtype CONSTRUCAO: All kinds of constructions, from buildings, clusters of buildings, or specific areas of a building (e.g. a room, gallery, garden, or swimming pool);
2. Subtype DIVISAO: Population aggregations such as metropolises, cities, villages, or towns, as well as other administrative divisions such as states, districts, provinces,

continents, or fiscal districts;

3. Subtype PAIS: Countries, principalities, and unions of countries, as is, for example, the case of the European Union;
4. Subtype REGIAO: Cultural or traditional location, with no administrative value (e.g. o Grande Porto, o Médio-Oriente, o Terceiro Mundo ou o Nordeste);
5. Subtype RUA: All kinds of streets and alleys, such as streets, avenues, roads, lanes, squares, squares, alleys, squares, etc;
6. Subtype OUTRO: Every other NEs that may belong to this type.

A.4.3 Type VIRTUAL

1. Subtype COMSOCIAL: All media, such as newspapers, television, radio;
2. Subtype SITIO: All virtual sites in the electronic sense: Web, WAP, FTP, etc;
3. Subtype OBRA: Reference to a printed work;
4. Subtype OUTRO: Every other NEs that may belong to this type;

A.5 Category OBRA

A.5.1 Type ARTE

Works or objects of which there is a single copy (e.g. Torre Eiffel, Capela Sistina).

A.5.2 Type PLANO

Political, administrative and financial measures, projects, and treaties (e.g. Constituição, Tratado de Tordesilhas).

A.5.3 Type REPRODUZIDA

Works of which there are many copies, the name represents the original from which reproductions are made (e.g. Titanic, Tropa de Elite).

A.6 Category ORGANIZACAO

A.6.1 Type ADMINISTRACAO

It identifies organizations that are involved in the administration or governance of a region (e.g. Parlamento, Brasil, Administração Bush). In addition, it includes organizations that are involved in international or supranational governance (e.g ONU, UE).

A.6.2 Type EMPRESA

In this category are for-profit organizations, including companies, societies, clubs, etc (e.g. Ferrari, Zara).

A.6.3 Type INSTITUICAO

All organizations that are neither profit-making nor play a direct role in the government administration. This type includes institutions in the strict sense, associations, and other cooperative organizations, universities, collectives, schools, or political parties (e.g. Igreja Católica, Sindicato dos Enfermeiros).

A.7 Category PESSOA

A.7.1 Type CARGO

This type refers to occupations that are being held by individual persons at a certain moment, meaning that they can be held by others in the future (e.g.Presidente da ONU, Papa, Ministro da Economia).

A.7.2 Type GRUPOCARGO

Analogous to the GRUPOIND, designating NEs that refers to a set of people through a position (e.g Ministros da Economia, Professores da UFRGS).

A.7.3 Type GRUPOIND

This type represents a group of entities of the INDIVIDUAL type that does not have a fixed name as a group (e.g. Vossas Excias, Governo Clinton, casa dos Mirandas, o Governo de Cavaco Silva). Rolling Stones is a counterexample since it is a defined name for a group composed of individuals.

A.7.4 Type GRUPOMEMBRO

Represents NEs that refer to a group of people as members of an organization or similar concept, such as a team or a religious group (e.g Mórmons, Barcelona).

A.7.5 Type INDIVIDUAL

Individual persons. Titles used in the treatment of a person should be included in the NE that delimits that entity (e.g. dr., eng., arq., Pe.). Forms of address normally used preceding a name, such as president, minister, etc. should also be included, as should degrees of relationship (aunt, sibling, grandmother, etc.) when they are part of the form of address. Diminutives, nicknames, initials, mythological names, and religious entities are tagged in this category.

Examples: Dr. Sampaio, presidente Jorge Sampaio, padre Melícias, tio Zeca, Miguel Sá, Presidente da República Jorge Sampaio.

A.7.6 Type MEMBRO

When an individual is denoted by the organization that represents him. For example, in the sentence: "O Mórmon estava na sala ao lado." the individual is denoted as Mórmon, a word that is related to the organization in which he participates.

A.7.7 Type POVO

When a given entity, usually associated with a particular location, is used to refer to the population of that location (e.g. Portugal consome muito peixe).

A.8 Category TEMPO

A.8.1 Type DURACAO

Time expressions that refer to a continuous event, expressing a temporal quantification and not a temporal localization (e.g. três meses, todo verão).

A.8.2 Type FREQUENCIA

Represents expressions that denote a repetition of an event at a point in time (e.g. diariamente, duas vezes por semana).

A.8.3 Type GENERICO

These are expressions that do not refer to a specific date, although the linguistic expression contains lexical elements that denote a temporal value (e.g. o Inverno, Março).

A.8.4 Type TEMPO_CALEND

Entities of type time are expressions that allow you to insert or locate the predicate they modify on a time axis (such as a point or an interval). They correspond to the following subtypes:

1. Subtype DATA: Absolute dates, containing information about the day, month, and year (e.g. no dia 08 de Abril de 1997). It can also be referential dates, which implies

that it uses another date as a temporal reference (e.g. João chegou ontem);

2. Subtype HORA: Entities that represent a time of the day (e.g. às 17:00);
3. Subtype INTERVALO: Represents a complex expression that has two temporal boundaries (the beginning and the end of the event, e.g. entre 2002 e 2006, de Abril a Setembro de 2006).

A.9 Category VALOR

A.9.1 Type CLASSIFICACAO

Values that designate classification, ranking, or scoring (e.g. 2-0, 15').

A.9.2 Type MOEDA

Monetary values. The NE must include the unity of the value (e.g. 30 milhões de reais, U$20,00).

A.9.3 Type QUANTIDADE

Percentages, loose numbers, and, if a quantity has units, the unit itself (e.g. 23%, 2.500, pH 2,5).

A.10 Category OUTRO

This category should cover other references that are relevant but not included in the other categories. For example, in the sentence "Eu recebi o Prêmio Camões no ano passado", "Prêmio Camões" is a NE of the OUTRO type.