GABRIELA SOUZA DE MELO

WINOGRAD SCHEMAS IN PORTUGUESE

São Paulo
2020

GABRIELA SOUZA DE MELO

WINOGRAD SCHEMAS IN PORTUGUESE

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Ciências.

São Paulo
2020

GABRIELA SOUZA DE MELO

Winograd Schemas in Portuguese

Versão Corrigida

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Ciências.

Área de Concentração:
Engenharia de Computação

Orientador:
Prof. Dr. Fábio Gagliardi Cozman

São Paulo
2020

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 30 de agosto de 2020

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Melo, Gabriela Souza de
Winograd Schemas in Portuguese / G. S. Melo -- versão corr. -- São Paulo, 2020.
84 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Inteligência Artificial 2.Linguagem Natural 3.Análise de Texto 4.Sintaxe e Semântica da Linguagem Natural I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

RESUMO

O Desafio de Winograd se tornou uma referência em tarefas de resposta textual automatizada e processamento de linguagem natural, tendo sido desenvolvido inicialmente na língua inglesa. De forma a estimular o desenvolvimento do campo de Processamento de Linguagem Natural em português, desenvolvemos um conjunto de Esquemas de Winograd em português. Também adaptamos soluções propostas para a versão do desafio baseada em inglês, de forma a disponibilizarmos um modelo inicial para ser utilizado na versão do desafio baseada em português. Para fazê-lo, criamos um modelo de linguagem treinado sobre um conjunto de documentos da Wikipedia. De forma a avaliar o impacto da variação da capacidade do modelo de linguagem nos resultados para o desafio em português, nós testamos o treinamento do modelo utilizando-se de diversas técnicas que anteriormente levaram a resultados estado-da-arte para tarefas na língua inglesa.

Palavras-Chave – Desafio de Winograd, Aprendizagem de Máquina, Inteligência Artificial, Processamento de Linguagem Natural, Aprendizado Profundo.

ABSTRACT

The Winograd Schema Challenge has become a common benchmark for question answering and natural language processing, having been originally developed in the English language. In order to stimulate the development of Natural Language Processing in Portuguese, we have developed a set of Winograd Schemas in Portuguese. We have also adapted solutions proposed for the English-based version of the challenge so as to have an initial model for usage in its Portuguese-based version. To do so, we created a language model for Portuguese based on a set of Wikipedia documents. In order to evaluate the impact of the increase in the language model capacity in the results for the Portuguese challenge, we tested training of the model with the usage of a myriad of techniques that have previously led to state-of-the-art results for English-based tasks.

Keywords – Winograd Schema Challenge, Machine Learning, Artificial Intelligence, Natural Language Processing, Deep Learning.

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

BERT - Bidirectional Encoder Representations from Transformers

GRU - Gated Recurrent Unit

LM - Language Model

LSTM - Long Short-Term Memory

RNN - Recurrent Neural Network

WSC - Winograd Schema Challenge

CONTENTS

1 INTRODUCTION

The Winograd Schema Challenge is a reading comprehension challenge proposed in 2011 (1). Only in 2019 did a solution reach a considerable accuracy, reaching a score of 90.1% of correct answers (2). Previous to that, no solution had surpassed a 72.2% rate of correct answers[1] (3) – score that was also reached only in 2019.

The challenge consists of sentences containing an ambiguity related to a coreference problem with pronouns. Each sentence contains a pronoun that could refer to either of the two noun phrases previously mentioned in the sentence and is followed by a question about which of the noun phrases the pronoun refers to.

An example of a Winograd Schema Challenge question is:

> ***The trophy*** *doesn't fit in* ***the brown suitcase*** *because* ***it*** *is too big. What is too big?* (4)

The Winograd Schema Challenge (WSC) has been advocated as an alternative to the Turing Test, in that it would also be a measure of whether a computer is able to think or not. This is due to the fact that it contains particularly difficult coreference resolution problems that can only be solved by reasoning with the usage of commonsense knowledge. These questions are simple for humans: in a test run in 2015, humans displayed 92% accuracy (5). But the questions are challenging for computers; standard coreference resolution solvers do not work well on the challenge (6). Another difficulty is the fact that there are few examples in the original set of Winograd Schemas (based on the rules proposed by Levesque, Davis and Morgenstern (2012) (4)); hence solutions that depend on training with very large datasets of Schemas do not work if working solely based on that original set.

Interest in the WSC has also emerged as an evaluation tool for Language Models (a type of model in the field of Natural Language Processing), such as in the work by

[1] https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WS.html

Radford et al. (2019), where Winograd Schemas are used as one of the evaluation sets used for comparing the model's performance to others available in the literature (7); and also as part of language model evaluation benchmarks, namely the GLUE (8) and the SuperGLUE (9) criteria. The motivation for this might be more pragmatic than that which based the initial proposal for the task: pronoun resolution as needed in the WSC is very relevant to the development of Natural Language Processing (NLP), as pronoun resolution happens as an underlying task of many others that require natural language processing and understanding. One such task is machine translation, which depends on resolving ambiguous sentences as displayed in the WSC to be able to properly translate them (10). As such, it is sensible to utilize the score in this challenge as yet another measure of a program's ability in tasks involving natural language processing and understanding.

There is a significant disparity, however, in the development of natural language processing techniques for the English language and that for other languages. The literature is scarce and assets that are quite commonplace for English-based problems, such as a corpus to train your language model with and to compare your results with benchmark ones, are sometimes not available in other languages.

In order to stimulate the development of NLP research in Portuguese, we have created a set of Winograd Schemas in Portuguese. This task is not as simple as translating each Schema in English word by word, as there are several rules to consider when developing a Schema, and their effect changes from language to language. We have also developed a system for solving Winograd Schemas in Portuguese; this system should serve as an initial attempt for solving the task, in the hopes that future work will build upon these results. It also provides the possibility of cross-linguistic comparisons with this challenge, which might bring interesting insights into differences between languages, in what relates to tasks and solutions to such tasks for each of them.

1.1 Justification

With the lack of benchmarks available for the Natural Language Processing field in Portuguese, the measurement and evaluation of work in the field for this language gets slowed down. Given this, our project developed a set of Winograd Schemas in Portuguese and replicated a solution for the English language for it. This is based on the hypothesis that the same sort of solution that has shown good results for the English version would be an ideal initial way to attempt to solve a different language version of the challenge.

It is also worth noting that the Winograd Schema Challenge is a specific type of the Pronoun Disambiguation Problem, a problem in natural language processing (11) related to coreference resolution (12). Hence, the ability to correctly solve Winograd Schema Challenge questions can be expanded further into other problems regarding language models and natural language processing, helping contribute to the hypothesis that having this collection available in Portuguese would be beneficial to the field of Natural Language Processing as a whole.

1.2 Objectives and Contribution

Our main objective was to develop a Portuguese set of Winograd schemas, making it fully available online, and to investigate whether methods that have been successful for the English version of the challenge could also be successful in the Portuguese one.

The output of our work is both the Portuguese set of schemas and a trained model capable of answering these schemas – with this, we are then establishing an initial model for the Portuguese version of the challenge. The corpus that we developed to train our Portuguese-based language model was also made available online.

We hope to contribute not only with a Winograd Schema Challenge in the Portuguese language but also more broadly, in the development of the field of Natural Language Processing in Portuguese itself, based on the understanding that the WSC is a relevant task in the NLP field. This work also provides an example for systematic translation of the Winograd Schema set to other languages and for comparing results of solvers for the challenge in different languages, making publicly available all the code that was developed for being able to do so. Consequently, we also hope to impact the replication of similar work for other languages.

During our research a paper was published in a national conference on artificial intelligence:

MELO, G. S. de; IMAIZUMI, V.; COZMAN, F. Winograd Schemas in Portuguese. In: *XVI Encontro Nacional de Inteligência Artificial e Computacional*. 2019.

This paper has been listed on the official page for the Winograd Schema Challenge[2], as one of the few translations to other languages, and the only translation to the Portuguese language.

[2]https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WS.html

1.3 Structure

This text is organized as follows. We begin in Chapter 2 with a thorough review of some background information that is useful – and somewhat required to a better understanding – for the project in hand. We then move on to explain related work in the original, English-based, version of the Winograd Schema Challenge, in Chapter 3. In Chapter 4 we formally introduce the scope and structure of our project. We then explain our experiments and the results that were obtained in Chapter 5. We finish by presenting our conclusions in Chapter 6.

2 BACKGROUND

In this chapter, the background topics related to our work are presented. These are mainly the Winograd Schema Challenge – what it is, why it was proposed, what makes a Winograd Schema; Machine Learning and how it relates to language problems; and Language Models, followed by a description of the BERT pre-trained model. We finish the chapter discussing the corpora that gets used for training such language models and, lastly, the evaluation of language models.

2.1 The Winograd Schema Challenge

The Winograd Schema Challenge evaluates a machine's performance in a task related to coreference resolution. In a coreference resolution problem a machine must figure out which entity an expression (usually a pronoun) refers to (13). In many cases, this can be done simply by looking at gender or number agreement. However, this is not enough for Winograd Schemas. Rahman and Ng (2012) call the type of coreference problem present in Winograd Schemas "complex cases of definite pronouns" (14).

A set of rules has been established for a sentence to be considered a Winograd Schema (4). In short, the rules determine that:

1. The possible antecedents are noun phrases of the same gender;
2. A pronoun or possessive adjective refers to one of these antecedents, but is also of the correct type for the other possible antecedent;
3. Answer 0 is always the first party mentioned in the sentence, answer 1 is the second;
4. There is a *special word*, that, when changed to the *alternate word*, the sentence is still perfectly valid, but the answer switches;
5. A Schema cannot be too obvious, in the sense that a simple statistical check of

whether the special word happens more frequently with one of the possible answers than the other must not be able to solve the Schema;

6. The sentences must not be too ambiguous, that is, fluent speakers of the language must be able to correctly answer the sentence without doubts.

If we take the following Winograd Schema as an example:

> ***Joan*** *made sure to thank* ***Susan*** *for all the help* ***she*** *had given. Who had given help?*

The correct answer would be Susan. The special word in this case is *given*, which can be substituted by *received*, as in the example below:

> ***Joan*** *made sure to thank* ***Susan*** *for all the help* ***she*** *had received. Who had received help?*

After this change, the correct answer becomes Joan.

It is worth noting that even though the original challenge statement mentions the presence of a single special word, the set of Schemas is slightly more flexible in the sense that there might be more than a special word – perhaps the expression special snippet would be more precise. An example of that situation could be the following:

> *Frank was upset with Tom because the toaster he had [**bought from/sold**] him didn't work.*

Where one of the sentences contains the snippet *bought from* and the other, the single word *sold*.

We see another example where there is more than one special word in:

> *Jim signaled the barman and gestured toward his [**empty glass/bathroom key**].*

There are also situations where the alternative to the special word is actually the absence of any words at all. The following exemplifies it:

> *There is a pillar between me and the stage, and I can't see [**around/–**] it.*

Also, the initial proposal mentions pairs of Winograd Schemas – that is, the combination of each sentence with each of its versions of the special word. There are, however, some sentences that produce more than one variation, forming triplets, instead of pairs. Take the following for example:

> *George got free tickets to the play, but he gave them to Eric,* ***even though*** *he* ***was*** *particularly eager to see it.*

This sentence actually produces two different variations:

> *George got free tickets to the play, but he gave them to Eric,* ***because*** *he* ***was*** *particularly eager to see it.*

> *George got free tickets to the play, but he gave them to Eric,* ***because*** *he* ***was not*** *particularly eager to see it.*

All of the examples above were extracted from the challenge's official website [1]. The collection of Schema in the official website contains only 285 sentences — a fact that added extra difficulty for the initial solvers to the challenge.

This challenge was proposed as a new Turing Test, in that it would be a way of assessing if a machine is thinking. The argument for this is that the only way of answering Winograd Schema Challenge questions is through the use of commonsense knowledge - there are no linguistic elements that would make it possible to solve the questions without the usage of previous knowledge. These questions are simple for humans: in a test run in 2015, humans displayed 92% accuracy (5). But the questions are challenging for computers and standard coreference resolution solvers do not work well on it (6). Levesque, Davis and Morgenstern (2012) argue that it is exactly the ability to bring together background knowledge that we informally refer to as thinking (4). Therefore, if a machine was able to correctly answer these schemas, it would be said to be thinking. Rahman and Ng (2012) agree with this proposal and states that "this is an easy task for a subject who can 'understand' natural language but a challenging task for one who can only make intelligent guesses" (14).

The main arguments for using this new test instead of the Turing Test are that the latter involves the machine needing to pretend to be somebody and that a conversation

[1] https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WSCollection.html

might not be the best manner of evaluating a machine's intelligence. The Winograd Schema Challenge also has the advantage of not depending on a judge to decide whether the machine is intelligent or not (4).

With the advances in neural network performance, however, the above argument might have gotten somewhat outdated. Language model-based solutions have achieved around 90% accuracy, but it would be hard to argue that such machines are capable of thinking or even that they demonstrate commonsense reasoning. Rather, it seems like they are able to exploit statistical information on text documents in order to provide answers to the challenge, but without any actual reasoning being applied to the commonsense situations being proposed in the sentences. Given these advances, to accurately measure commonsense reasoning capabilities of machines, it might be necessary to design new tests and datasets focused on this sort of task (15).

The central problem being presented in the challenge, however, is a specific type of coreference resolution (a specifically difficult one, as linguistic structure of the sentences does not provide the answer to them), and, therefore, the challenge still shows value, as these types of coreference resolution can be present in text; hence, being able to correctly resolve this type of pronoun might be a required ability to machines performing tasks related to natural language processing.

There is one extra issue with the official collection of Winograd Schemas, which is the fact that there are very few schemas contained in it — in the official website [2], only 285 sentences are listed. This not only provides a difficulty in the lack of training data to solvers for the challenge, but also makes results quite prone to lucky-draw variations. In an attempt to resolve this issue, a new dataset — WinoGrande — was recently proposed (2), which might provide a more consistent assessment of results due to its size; 44 thousand sentences are contained in this dataset. We will describe this dataset in Section 3.3.

2.2 Machine Learning

Machine learning is a subset of the artificial intelligence field in which a computer can learn by itself — based on data provided to it — instead of having to be explicitly programmed to perform such task (16). It was defined by Mitchell et al. (1997) as follows: "a computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured

[2] https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WSCollection.html

by P, improves with experience E." (17)

Developing a machine learning solution usually includes selecting a model (e.g. linear regression, decision tree, neural network, amongst many others) and then fitting the model's parameters by feeding the dataset that is available as training data. By doing this, the goal is for the model, with the adjusted parameters, to learn to represent the problem as accurately as possible.

Machine Learning models are especially valuable in situations where a vast amount of data is available. In fact, a common trick for achieving better results from a machine learning model is to use more data to train the algorithm with, as this may improve the model's generalization capabilities (generalization is the ability to correctly predict the response for some data observation that has not occurred during model training time).

In the remainder of this session, we will discuss how Machine Learning relates to Natural Language Processing and introduce Deep Learning — a type of model that has been very frequently used for Natural Language Processing and Language Models.

2.2.1 Machine Learning and Natural Language Processing

The Machine Learning field has an intersection with the Natural Language Processing field. Machine Learning, although not the only tool available for solving NLP tasks, is one of the possible tools for performing them. Natural Language Processing (NLP) is a field concerned with automatic methods for handling language-related tasks — these tasks can be simple, such as spell-checking or finding synonyms of words, but can also be much more intricate, such as machine translation, question answering, or coreference resolution; the latter being a superset of the task that is asked of the machine on the Winograd Schema Challenge.

We notice in the available literature that machine learning solutions for text problems are mostly applied to the English language. The Winograd Schema Challenge, for instance, had only been previously translated to Japanese, Chinese [3] and French (18). The English version of the challenge, on the other hand, has seen numerous recent publications just in the past two years (3, 7, 19–23). This not only makes the field too focused on English but also creates a vicious cycle, in that, the very fact that there is plenty of work in the literature regarding the English language makes it so that the entire field is able to build on top of each other's work and progress faster. Such collaboration is not

[3] https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WS.html

very much present in the Portuguese language, and the lack of benchmarks for Natural Language Processing in Portuguese might be another contributing factor for that.

2.2.2 Deep Learning

A subset of Machine Learning models — and one that has been particularly useful for NLP tasks in recent years — are those that are based on Deep Learning. This is an area of machine learning comprised of Neural Networks, which are models that work with representation learning.

2.2.2.1 Neural Networks

These models consist of multiple layers of neurons, each of which learns a different level of representation of the input data (24). This representation learning happens by, in each layer, multiplying the input data by a set of weights (and adding some bias value). Each layer consists of some neurons, which are also frequently referred to as units. The final layer is referred to as *output layer*, whereas the intermediary layers are called *hidden layers*. Equation (2.1) demonstrates the operation happening at each of these units, where $\mathbf{x}$ is the input vector, $\mathbf{w}$ the weight vector and b a bias value. Both the weights and the bias value are learned at each of the units during the training process.

$$f(\mathbf{x}; \mathbf{w}, b) = \mathbf{x}^T \mathbf{w} + b. \tag{2.1}$$

This means each unit will apply vector operations on top of the input vector that it received and will output a scalar value. The layers have multiple units, and each unit has a vector of weights. Therefore, if we look at the operation that is happening at each of the layer's units, the function applied at each layer can be seen as a matrix multiplication over the input data, where the matrix is of dimension $number\_of\_units$ x $input\_data\_dimensionality$ and the bias vector is of size $number\_of\_units$. The choice of how many units and layers to utilize is part of the design decisions involved in the use of neural networks.

Figure 1 is an example of a neural network. It consists of an input of dimension 3, with 2 hidden layers and one output layer. The first hidden layer has 4 units, while the second one has 3. The final layer is of a single unit. Each arrow represents a multiplication operation by a weight value.

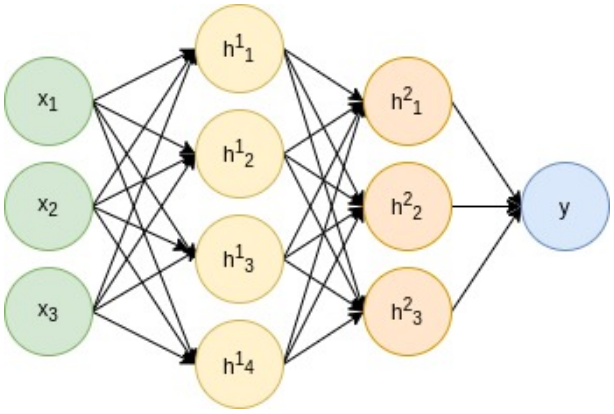


Figure 1: An example neural network.

A non-linear transformation is applied to the output of each layer, before passing those values on to the next layer. This is necessary, otherwise, the sequential linear transformations would be just equivalent to a single linear transformation, so there would not be any value added by having multiple layers. Some of the common non-linear transformations are the ReLU (Rectified Linear Unit) (having z as the layer's output, $g(z) = max\{0, z\}$), the logistic sigmoid ($\sigma(z) = \frac{1}{1+e^{-z}}$) and the hyperbolic tangent functions ($tanh(z) = 2\sigma(2z) - 1$). These non-linear transformations are commonly called activation functions. The choice of activation function is another design decision when using neural networks. The ReLU function is a common recommendation for starting, and the usage of the softmax one has shifted to not being encouraged anymore, as it might make gradient-based learning more difficult due to the widespread saturation of the units (25).

For the output layer, an activation function might also be used. The choice for this one, however, is related to the task the network will be used for. If it is a regression problem, linear functions will serve well, so an activation function might not be necessary. For classification tasks, the usual choices are either the sigmoid, for binary classification, or the softmax ($\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$ for $i = 1, \ldots, K$ and $\mathbf{z} = (z_1, \ldots, z_K) \in^K$), for multiclass classification.

2.2.2.2 Parameter Learning

The parameters that the model learns from the training data are the weights and bias for each of the units. Neural Networks are learned by repeatedly updating the parameters at each of the units, based on the cost result from the output obtained from the usage of the current parameters. This update happens via the back-propagation algorithm, which builds on top of gradient descent, and with an optimization algorithm. The basic idea is

to calculate the derivative of the cost with respect to each of the parameters and use that to update the value of the parameter.

To understand that, we first need to mention how the cost of a neural network is calculated. This value is what measures how far we are from obtaining the expected results from our neural network, and is calculated based on the outputs of the final layer in the network. The most common choice for cost function is the cross-entropy between the learned model distribution (in the equation below, $\mathbf{y}$) and the training data (below, $\mathbf{x}$), which is equivalent to the negative log-likelihood — used for maximum likelihood training. This function is defined as

$$J(\theta) = -\mathbb{E}_{\mathbf{x},\mathbf{y}\sim\hat{p}_{data}} \log p_{model}(\mathbf{y}|\mathbf{x}). \tag{2.2}$$

To empirically calculate such cost in our training set, we calculate the loss for each example (notated as L), and, having m be the size of our training set and θ the model's parameters, we can then obtain:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{i=m} L(f(\mathbf{x}^{(i)}, \theta), \mathbf{y}^{(i)}). \tag{2.3}$$

Based on the cost results obtained with the equation above, the weights can then be updated. This is done by calculating the derivative of the cost function with respect to each parameter. This calculation is done with the usage of the back-propagation algorithm. It receives such a name in opposition to the process of forward propagation. Forward propagation is the passage of input values from the input layers until the output one, with the respective operations that happen in the network. The back-propagation process, on the other hand, is the passage of the derivative of the cost function flowing backward, from the end of the network until its beginning. Such calculations can be done by using the chain rule of calculus.

After gathering the cost of our model, and the derivatives of the loss with respect to each parameter, we can then get to the update of our model's parameters. For that, we use gradient descent. The idea of gradient descent is to find the weights and bias values that minimize such cost function. We can describe the gradient descent algorithm as follows (25):

Algorithm 1: Gradient descent algorithm

Result: Updated parameters θ

initialization of parameter values and configuration of learning rate α ;

while *stopping criteria not met* **do**

  $\theta := \theta - \alpha * \frac{\partial J(\theta)}{\partial \theta}$

end

There are a few variations of the algorithm for using the gradient descent method in order to learn the best parameters for the network. One way to do it is to update the parameters at the end of a training epoch — that is, after the full set of training data has been passed into the model — a method frequently referred to as the batch gradient descent algorithm. Other ways would be with the stochastic gradient descent algorithm — where the parameters get updated after each training observation passes through the model — and the mini-batch method of training, which can be seen as a method in between the batch and the stochastic one, where the training data gets split into mini-batches, and the weight updates happen at the end of each mini-batch.

To be able to train the network, the weight vectors need to be initialized to small random values (if they were initialized to the same value we would end up with the same weight values in all units, making the usage of multiple units have no effect than it would already have with a single unit). The bias values may be initialized to zero or to small positive values.

Many improvements to the operation of Neural Networks have been proposed since they appeared. There are a variety of different optimization algorithms that are used to improve the search for the best parameters for the network; different regularization techniques, which aim to increase a model's generalization capabilities by decreasing test error without necessarily decreasing training error; and multiple types of layers and mechanisms with which to connect the layers. Many of these different ways of configuring a neural network also come with some hyperparameters to be set. Hyperparameters are some parameter values that control some of the functionality of the models and the algorithms being used in its training process, and, as opposed to the parameters that are learned by the network, these must be manually set. The choice of the number of layers, units per layers, and values for the hyperparameters are all of great impact in the network's performance; hence there is usually effort put into trying to find the best values for these.

There are many usages of Neural Networks and Deep Learning in NLP tasks, for instance, in connection with Word Embeddings (26), in Language Models, which we will

describe in the upcoming sections, and in the more recent Transformer-based architectures (27), which are based on attention mechanisms and have brought new advances to results in the NLP field.

2.3 Language Models

Language models are used to predict the probability of the next word in a sentence, or, in other words, finding which word is the most probable one following a sequence of words. That also means that it is a type of model capable of calculating the probability of a sequence of words happening together. Formally, a language model's output is a probability distribution over all tokens in the vocabulary. Language models can be useful in applications such as dialogue systems, machine translation, or text summarization, for instance.

Take the following sentence as an example:

The Winograd Schema Challenge is difficult

The language model then would be able to calculate the probability of this sentence, which can be expressed by the following equation:

$$P(w_1, ..., w_m) = \prod_{i=1}^{i=m} P(w_i | w_1, ..., w_{i-1}), \tag{2.4}$$

where w_1 corresponds in this sentence to the word *The*, and w_m to the word *difficult*.

We can also express the probability of the word *difficult*, in that sentence, as

$$P(w_{difficult} | w_{The}, w_{Winograd}, w_{Schema}, w_{Challenge}, w_{is}). \tag{2.5}$$

Note that by using the Bayes Rule, the above equation could be evaluated as

$$\frac{P(w_{The}, w_{Winograd}, w_{Schema}, w_{Challenge}, w_{is}, w_{difficult})}{P(w_{The}, w_{Winograd}, w_{Schema}, w_{Challenge}, w_{is})}. \tag{2.6}$$

If the model has a good performance, we would expect that probability to be higher than that of a random word, for instance, $P(w_{ocean} | w_{The}, w_{Winograd}, w_{Schema}, w_{Challenge}, w_{is})$.

If we had the sentence *The Winograd Schema Challenge is* and wanted to predict which word would be the next most probable one, from a fixed-size vocabulary, we could

then predict the probability distribution for the words in that vocabulary, following the equation above, and then select the one with the highest probability as the next word to complete that sentence. After the next word has been selected, we can keep going with that word selection process, and have a full text derived from this. Note that this does not necessarily yield the most probable sentence, but only a selection of the most probable word given the previous ones, at each point in the sequence.

There are a few different ways of developing models that work as Language Models, and these will be introduced in the sections below.

2.3.1 N-Gram Models

The first language models that were developed followed an approach that is referred to as *n-gram models*. This approach is based on the method of approximating probabilities as the count of occurrences of words in a text. Using this method, Expression (2.6) could then be estimated as:

$$P(w_{difficult}|w_{The}, \cdots, w_{is}) \approx \frac{count(w_{The}, w_{Winograd}, w_{Schema}, w_{Challenge}, w_{is}, w_{difficult})}{count(w_{The}, w_{Winograd}, w_{Schema}, w_{Challenge}, w_{is})}. \tag{2.7}$$

However, in situations where your object of interest involves long sentences, it would become very difficult to do so if we always kept calculating the count of occurrence of the entire text up to the word we are trying to predict. From this difficulty came the idea behind the naming of this type of model — instead of using the full text to find the next most probable word, we would only consider the previous n words. That is, we would look at the words as *n-grams*, and track the counts of those.

Following the previous example, if we were to look at bigrams, we would then reduce that equation to:

$$P(w_{difficult}|w_{is}) \approx \frac{count(w_{is}, w_{difficult})}{count(w_{is})}. \tag{2.8}$$

If we were to use trigrams instead, it would then become:

$$P(w_{difficult}|w_{challenge}, w_{is}) \approx \frac{count(w_{challenge}, w_{is}, w_{difficult})}{count(w_{challenge}, w_{is})}. \tag{2.9}$$

The challenge with this model is to find the best value for n. If it is too small, it

will not capture the full meaning related to the upcoming word. Make it too large on the other hand and the calculation of the n-grams' counts of occurrences will result in a large and sparse matrix and will take an increasingly long time to get calculated. Moreover, the probability of any *n-gram* that has not occurred yet would be set to zero, making it so that the model cannot generalize to unseen *n-grams*.

2.3.2 Neural Networks-based Models

Recently, most of the successful language models have been those based on artificial neural networks. Their usage makes it possible to avoid the difficulties related to N-Gram models, mentioned above: the sparsity of the matrix of counts (an issue referred to as *curse of dimensionality*) and the lack of generalization (that is, the lack of ability of the model to set the probability of an unseen word or *n-gram* to some value other than zero). The initial approaches to the usage of neural networks for language modeling would use multilayer perceptrons, where the first layer would consist of a word embedding layer which would have its weights shared between the sentence's words and a hidden layer which would receive as input the concatenation of the sentence's word embeddings. There would then be an output layer calculating a softmax distribution of probabilities for all the words in the vocabulary. The network was trained by maximizing the penalized log-likelihood, where the penalty would be added based on the weight decay procedure (28).

However, this method has a restriction regarding the sentence length: it can always only receive the same number of words as input. To tackle this restriction of having to pass as input a fixed-length sentence, Recurrent Neural Networks (RNNs) started being used for language modeling (29).

2.3.2.1 Recurrent Neural Networks

Recurrent Neural Networks are a type of neural network layer where some information can be persisted between subsequent input data. This information is sometimes referred to as cell state or cell memory. In the context of text data, this means that we can make the network pass forward some memory information between words.

Furthermore, in this type of network, instead of updating the weights of the layers after each input is fed to the model, the weights can be shared by a sequence of inputs. This means that we can have the same weights being applied to each word in a sentence, and the weights get updated afterward. The weight updates for RNNs happen following

the back-propagation through time algorithm — it works just as regular back-propagation regarding the calculation of the gradients, but the gradients of all words in the sentence get summed before being used for updating the weights.

There are many benefits of using RNNs for language modeling. The first one, as already stated, is the possibility of feeding it differently sized inputs. Additionally, the use of longer inputs has no effects on model size. Another significant benefit is the fact that calculations use information from many steps back in time. Lastly, the fact that weights get shared between time steps means that features learned across different positions of text can get shared.

However, it is necessary to mention a few disadvantages of RNNs as compared to other models: recurrent computation is slow, and it is difficult to use information from many steps back, due to vanishing and exploding gradients.

There are different variants of Recurrent Neural Networks, sometimes referred to as gated RNNs; namely the Long Short-Term Memory (LSTM) (30) and the Gated Recurrent Unit (GRU) (31) layers. These layers are better than vanilla RNNs at modeling long-term dependencies between each input observation, as they provide for a way of reducing the impact of vanishing and exploding gradients. This is achieved due to these layers containing mechanisms for allowing them to choose what to store and what to forget, rather than just always keep multiplying the gradients. GRUs are more recent than LSTMs and were presented as a simplification of the LSTM layer type.

2.3.2.2 Recurrent Neural Network Language Models

RNNs are used for language modeling in a similar way as the non-RNN-based neural network models. That is, there is usually an input and an output layers which act as a mapping between the vocabulary size and a reduced-dimensionality embedding size. In between these layers are the RNN layers. These layers receive a sequence of words and store a cell state which helps pass forward some memory information on the previously seen words. The final output of each RNN layer then gets passed on to the following layer.

More specifically, the network receives a sentence's words, sequentially, as input; each word being represented as a sparse vector with its size matching the vocabulary size of the corpus being used to train the model. The sparsity of this vector means that, for each word, a single input layer unit is activated, with a value of 1 (all the other units receive a zero-valued input) — this representation is also known as one-hot encoding. This means

that the values received at the first hidden layer correspond to the weights leading from that single unit to that hidden layer. For this reason, this input layer can be seen as an embedding layer, as it makes the weights that represent a single word reach the first hidden layer.

The recurrent layers store its units' cell states until the end of the sequence is reached. Each word that gets fed to these layers makes the cell states get updated. The last recurrent layer leads to another regular layer with unit size equal to that of the vocabulary, as the input layer. Due to the similarity of these layers, the weights of them can actually be shared and learned together — a process that decreases training time and the amount of memory required for training (32, 33). Each next word is then predicted by applying the softmax operation over the output layer.

From the output layer, the resulting cost value can be calculated. The usual choice for language models is the cross-entropy loss, which has been mentioned previously in Section 2.2.2.2. For our neural network-based language model scenario, the cross-entropy at time step t is calculated between the predicted probability distribution for the next word ($\hat{y}^{(t)}$) and the true next word ($y^{(t)}$). Mathematically, it is described as

$$J^{(t)}(\theta) = CE(y^{(t)}, \hat{y}^{(t)}) = -\sum_{w \in V} y_w^{(t)} \log \hat{y}_w^{(t)}, \tag{2.10}$$

where V represents the vocabulary in use.

We can note, however, that the value of $y_w^{(t)}$ is zero for all words but the true next word, and it is 1 to the true next word ($y_{x_{t+1}}$). With this in mind, the calculation of the cost gets reduced to

$$J^{(t)}(\theta) = -\log \hat{y}_{x_{t+1}}^{(t)}. \tag{2.11}$$

The cost for the entire training set, in its turn, is described as

$$J(\theta) = \frac{1}{T} \sum_{t=1}^{t=T} J^{(t)}(\theta) = \frac{1}{T} \sum_{t=1}^{t=T} -\log \hat{y}_{x_{t+1}}^{(t)}. \tag{2.12}$$

2.3.3 Regularization of Recurrent Neural Network Language Models

Regularization is the process through which a model learns to better generalize — that is, to become better at predicting outputs for observations that it has not received at training time, even if resulting in an increased training set error. It is usually achieved by using methods that discourage a model from having large weights. There are many different regularization techniques, and we will focus on some that have been shown by Merity et al. 2017 (34) to provide good results specifically for language models.

2.3.3.1 Variable Length Backpropagation Sequences

The first of these techniques is the use of a variable-length backpropagation sequence, instead of a fixed-valued one. The motivation behind this is that with a fixed value, there will be words that, at training time, will always be used as the first one in a sentence, therefore not having any previous words to backpropagate into, and only the last words in each sentence will benefit from the full backpropagation window. This can harm the ability to correctly predict the probabilities for these words at test time.

This variable-length method works by receiving a sequence length *seq* as input; then, it will use that value as a base sequence length with a probability p, and will use a value of $\frac{seq}{2}$ with a probability of $1 - p$. From there, having *base_seq* be this base sequence length, the actual sequence length to be used, is drawn from a distribution described as $\mathcal{N}(base\_seq, s)$, having s be the standard deviation (which is manually set to some value — Merity et al. utilized a value of 5 (34)).

To compensate for the impact of varying the sequence length, the learning rate is rescaled based on the ratio between the selected sequence and the original sequence length that was provided.

2.3.3.2 Weight-Dropped LSTM

The Weight-Dropped LSTM is based on the usage of DropConnect, which was proposed by Wan et al. (2013) (35). The idea behind DropConnect is to drop weights in the network instead of dropping activations. This means that instead of dropping full neurons' outputs, it will instead just drop specific weight connections between layers. Figure 2 helps elucidate this difference.

The Weight-Dropped LSTM consists of using DropConnect in the hidden to hidden

connections of the LSTM layers, which prevents overfitting on the recurrent connections of the LSTM layers. Given that LSTM weights are shared between timesteps, the dropped out weights remain dropped for the full forward and backward passes.

This is also proven to be a computationally efficient application of dropout as it does not require modifications to the inner workings of the recurrent units of the networks, which would be computationally expensive to do (34).

2.3.3.3 Variational Dropout and Embedding Dropout

Having been initially proposed by Gal & Ghahramani (2016) (36), Variational Dropout is a technique that consists of using the same dropout mask for all timesteps, instead of sampling a new one for each input in a sequence. With variational dropout, a new dropout mask is sampled for each mini-batch example, and used for all steps in its forward and backward pass.

Embedding dropout is a technique for applying dropout on the embedding matrix. It is essentially the same as applying variational dropout on the connection between the embedding layer and the initial recurrent layer; this means that each dropped out word will be removed from the forward and backward pass, hence disappearing from a sentence being passed to the network if it is present in it.

2.3.3.4 Activation Regularization and Temporal Activation Regularization

These forms of regularization act by reducing the size of the weights of the network, and are forms of L_2 regularization.

Activation Regularization (AR) is the application of L_2 decay on the LSTM units

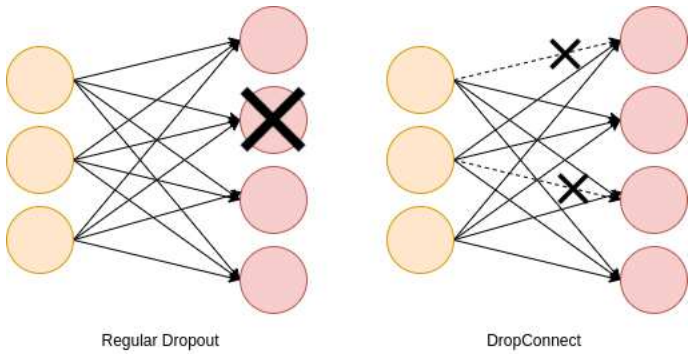


Figure 2: Visualization of the effects of Regular Dropout and DropConnect. The X marks indicate dropped units or connections.

after dropout has been applied. It contains one hyperparameter to be tuned, α, which is a scaling coefficient for the decay.

Temporal Activation Regularization (TAR) is a method for preventing large changes in the recurrent units' hidden states between timesteps. It is a sort of slowness regularizer, as it makes changes in the hidden states happen slower. TAR is L_2 decay applied to the difference in hidden state between timesteps. It contains the hyperparameter, β, which, similarly to α in AR, controls the scale of the decay.

Activation Regularization is mathematically described as

$$\alpha L_2(m \odot h_t), \tag{2.13}$$

with m being the dropout mask; and Temporal Activation Regularization, as

$$\beta L_2(h_t - h_{t-1}). \tag{2.14}$$

Both these regularization methods are applied solely to the output of the final RNN layer.

2.4 BERT - Bidirectional Encoder Representations from Transformers

BERT is a pre-trained language model that works by using a bidirectional context in order to predict words that have been omitted from the input. It can also be used for next sentence prediction — receiving two sentences and deciding whether the second sentence follows the first one or not. It is one of a family of models that provide the ability to apply transfer learning to NLP tasks.

While computer vision had already been using transfer learning for a few years — with the ImageNet model, for instance, having been released in 2009 (37) — it took some time for the field of NLP to also start being able to use pre-trained models and to collect the benefits from it. Transfer learning works by pre-training a model that can later be fine-tuned for your task at hand. With this, you can pick up on weights that have already been trained on vast amounts of data, and fine-tune it with a smaller-sized dataset.

One of the initial ways in NLP to build on top of pre-trained weights was with the usage of word embeddings, starting with the proposal of the Skip-gram model in 2013 (26). These word embeddings, however, did not take into consideration the fact that

words might have different meanings depending on the context in which they are being used. From this deficiency came the proposal of ELMo (38), which was a method for developing contextual embeddings. There was still a main issue with those contextual embeddings. This was that these embeddings were developed using either only left or right contexts for training, as an issue with bidirectional training is that words would be able to "see themselves".

BERT provides a way for solving that and actually training with bidirectionality, as stated in the full name of the model: Bidirectional Encoder Representations from Transformers. BERT achieves this by masking out a percentage of the input words so that the model will not be able to see the expected value for these words when trying to predict it.

The model is based on the Transformer architecture (27). This is a type of model built solely upon attention mechanisms, without using recurrent layers. It consists of the multi-headed self-attention mechanism, which models the context; feed-forward layers, which compute non-linear hierarchical features; uses the techniques of layer normalization and residuals, which help with deep learning training; and uses positional embeddings, which is what makes words contain information related to their relative positioning in a sentence. BERT only uses the encoder part of the transformer architecture, as it is only interested in the generation part of the model.

BERT yielded state-of-the-art results in many NLP benchmark tasks, as, with fine-tuning, it becomes possible to use BERT to solve many different tasks. The trained model has been made publicly available so that new work in NLP can build on top of it. It was initially trained in English but soon started to be released in other languages, including a recent release of a Portuguese version of the model (39). There are also multilingual models available, which can be used in a variety of different languages.

2.5 Language Model Training Corpora

Learning in a language model happens with the usage of a corpus — that is, a large collection of texts. This corpus is usually divided into training, validation, and test pieces. The training set is used for the actual parameter learning process. The validation set is used to provide information about the model's generalization capabilities, typically used for making decisions regarding a model's architecture and hyperparameters selection. The test set is used to provide a final evaluation of the model's performance and is used, for

instance, to compare different solutions for a problem.

There are corpora in the English language that are commonly used as benchmarks for language models in this language, and that have helped to guide the development of language models for the English language. Examples of such corpora are the Penn Treebank (40) and the WikiText (41) corpus, which both have been used to track NLP progress[4]. To the best of our knowledge, there is no similar corpus, used for establishing language modeling benchmarks, for the Portuguese language.

2.6 Evaluation of Language Models

Evaluating language models is based on analyzing how closely the probabilities yielded by the model relate to the expected test values. The way in which we embed such probabilities into a metric for calculating the model's performance is through the perplexity measure, defined as follows (42):

$$PP(W) = \prod_{t=1}^{t=T} \left(\frac{1}{P(x^{(t+1)}|x^{(t)}, ..., x^{(1)})} \right)^{1/T}, \tag{2.15}$$

where W represents the test set. This equation denotes the inverse probability of the corpus, according to the language model, normalized by the number of words.

By deriving the equation above, we get that the perplexity is equal to the exponential of the cross-entropy loss we have described above with Equation 2.12:

$$PP(W) = \prod_{t=1}^{t=T} \left(\frac{1}{\hat{y}^{(t)}_{x_{t+1}}} \right)^{1/T} = \exp\left(\frac{1}{T} \sum_{t=1}^{t=T} -\log \hat{y}^{(t)}_{x_{t+1}} \right) = \exp(J(\theta)). \tag{2.16}$$

The objective of a language model, then, is to achieve the lowest perplexity as possible.

[4]https://github.com/sebastianruder/NLP-progress/blob/master/english/language_modeling.md

3 RELATED WORK

Since the proposal of the English-based Winograd Schema Challenge, there have been many attempts at building machine programs that would pass it (3,6,7,14,19–23,43–45). This chapter will present the main characteristics of these proposals, and how they relate to each other.

Recently, a discussion on the evaluation of the WSC results has been raised (46); we will also present this discussion at the end of this chapter.

We also mention a recent expansion of the schema collection, considering the impacts this will have on future work (2).

3.1 Solvers for the English-based Version of the Challenge

Among the systems that have been developed to solve the WSC, which are usually use Machine Learning, a number of them are based on understanding how the sentences are structured, and from this, to use examples or rules to resolve the ambiguity. Some of these systems derive sets of features from the sentences and use them for training the model (14). Others are based on linguistic tools for parsing sentences (22, 45) or fitting them into predicate schemas (6) and using these strategies on the Winograd sentence and on results of queries on search engines. Yet others are based on relevance theory and knowledge graphs (43) and some on logic rules based on correlation formulas (44). After many years passed from the release of the challenge (2011), the best result from this type of solver was only at 57.1% accuracy on the Winograd Schema Challenge set, a result that was obtained only in 2018 (22).

Another type of solver leverages the fact that language models trained on vast amounts of language corpora indirectly incorporate commonsense knowledge when learning word relations. These are relatively recent solutions, mostly based on Deep Learning, using neu-

ral networks based on embeddings (19), siamese networks (20), language model networks (21), and transformer architectures such as GPT-2 (7) and BERT (3, 23). Traditional linguistic tools for extracting dependency graphs in the sentences are also employed by Ruan et al. (2019) (23), who use this extracted information to complete the traditional transformer model (27).

Some of the existing solutions are unable to provide answers to all of the WSC question, and hence apply to only a subset of Schemas, restricting its usability on a more general pronoun resolution scenario (6, 43–45).

To deal with the fact that there initially were a relatively small number of instances of Winograd schemas, some of the proposals in the literature have developed their custom datasets to help with training. Rahman and Ng (2012) have developed a set of relaxed Winograd schemas, containing 941 sentence pairs (14). Following that proposal, this dataset has also been used in other works (3, 6, 20, 23). Trinh and Le (2018) (21) and Kocijan et al. (2019) (3) have also developed custom datasets, based not on WSC-like sentences, but on general text corpora, focused, for the previous work, on text corpora documents that are more closely related to commonsense reasoning tasks or, in the latter, examples for helping with the more general pronoun resolution task.

While not being able to surpass 60% of accuracy in the results for the initial years after the challenge was proposed, the more recent works have been rapidly increasing these results. There is a clear relationship between the beginning of the utilization of language models based on deep neural networks and the improvement in these results, with the current state of the art being based on a Transformer structure (27), with the usage of the BERT model (47).

3.2 Robustness of Solvers

Recently, new evaluation criteria for the challenge have been proposed by Trichelair et al. (2018) (46). These criteria are based on dividing the data into two new sets, based on associativity and switchability characteristics of the sentences.

The idea behind these subsets is that they allow for further insights into model performance and facilitates understanding the model's robustness against slight variations in sentences, which could indicate if the model is just learning on spurious statistical information on the involved entities in each sentence.

3.2.1 Switchability

The switchable set consists of sentences that can have the antecedents switched; the sentence is still valid and the answer to it switches accordingly. An example of such a sentence is as follows:

> ***Emma*** *did not pass the ball to* ***Janie*** *although* ***she*** *saw that she was open.* (4)

The switched version of it would be:

> ***Janie*** *did not pass the ball to* ***Emma*** *although* ***she*** *saw that she was open.* (4)

Both sentences are grammatically valid and semantically correct; the correct antecedent to the pronoun in the first one is Emma, in the second one, it is Janie.

An example of a non-switchable sentence is:

> ***The city councilmen*** *refused* ***the demonstrators*** *a permit because* ***they*** *feared violence.* (4)

Therefore, we can divide the full set into switchable and non-switchable sets; for this switchable subset, we can have the unswitched (original) sentences and the switched ones. A total of 47% of the sentences in the full collection have been considered to be switchable.

3.2.2 Associativity

The associative set consists of sentences where one of the antecedents relates more strongly to the special word than the other (although this is one of the rules for the Winograd Schema Challenge collection, there has not been a strong check on whether all sentences follow this rule, and therefore some can be considered as being associative). Thus, all sentences in the collection are divided between the associative and the non-associative subsets, with 13.5% of the sentences being considered associative.

An example of an associative sentence would be:

> *I'm sure that* ***my map*** *will show* ***this building****;* ***it*** *is very famous.* (4)

This classification as associative is explained for there are many buildings considered to be famous, but not many maps. The following, however, would not be considered associative:

Bill *passed the gameboy to* ***John*** *because* ***his*** *turn was over.* (4)

There is a single sentence that has been marked as both associative and switchable by Trichelair et al. (2018) (46):

The man *lifted* ***the boy*** *onto* ***his*** *bunk bed.* (4)

It is interesting to notice, however, that, when switching the candidates (the man and the boy), such associativity would not be present anymore.

3.3 An Expansion to the Collection of Schemas

Recently, a proposal for an expansion to the collection of schemas was made (2). The main motivations behind this work are that the current collection might have sentences composed of human biases that lead to language models picking up on these rather than actually needing to develop commonsense reasoning capabilities. This might be making us overestimate such machines' capabilities. This relates to the associativity of sentences brought up by Trichelair et al. (2018) (46), that we discussed above in Section 3.2.

Another problem that was brought up in both these works is that with so few questions the effect of lucky draws is significant. This might make the original collection an unstable candidate for a benchmark in commonsense reasoning.

The work by Sakaguchi et al. (2019) (2) aims to provide a new, increased, collection of schemas, that could help with the issues mentioned above. This new collection is composed of a more significant amount of schemas — 44 thousand of them — and was developed with the help of Amazon Mechanical Turk workers [1].

To decrease the effect that spurious human biases might have on the dataset, they utilized an adversarial filtering approach, which they named AfLite, that attempts to remove sentences that contain spurious biases.

They tested a RoBERTa-based model (48) on the new dataset and the original datasets, and concluded that the new one provides a much harder challenge for machines,

[1] https://www.mturk.com/

achieving a score of 90.1% on the original dataset and 79.1% on the new one. Also, they used a subset of the new collection as training data and showed that transfer learning from such a subset help solve the original collection of schemas.

4 A PORTUGUESE-BASED WINOGRAD SCHEMA CHALLENGE

In this chapter the main contributions of our work are presented. These are, namely, the construction of the Portuguese set of Schemas and the development of an initial solver for it, and will be presented in the aforementioned order.

4.1 Collection of Portuguese-based Schemas

Our Portuguese-based collection of Schemas was developed following the rules proposed by Levesque, Davis and Morgenstern (2012) (4), mentioned in Section 2.1. To develop our set, we used as a base the set of 285 original English-based Schemas that are available online[1] and manually translated each of them. Our translated set is also available both in a JSON format and in a more visually pleasing, HTML format.[2] Note that a few additional tags are present in the JSON format, and these are described in Section 5.1.3.

Three native Portuguese speakers worked on translating the sentences, all of whom were familiar with the Winograd Schemas Challenge and the rules regarding the consideration of a sentence as a Winograd Schema. Each sentence was translated by one of the speakers and validated by the other two.

At first, we kept names as they were in the original sentences. Nevertheless, the fact that many of these names are not commonly found in Portuguese speaking countries might interfere with the task. Hence, we have developed an additional collection where names have been replaced by popular names in Brazil. The only restriction for these substitutions was that gender would be kept unchanged. Also, names of famous personalities that appear in the Schemas, such as Madonna or Shakespeare, were kept as in the original

[1] https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WSCollection.xml

[2] https://github.com/gabimelo/portuguese_wsc/blob/master/data/processed/portuguese_wsc.json and https://github.com/gabimelo/portuguese_wsc/blob/master/data/raw/portugues_wsc.html

collection. This set is also available in HTML and JSON formats.[3]

Additionally, it is worth noting that there are some significant differences between Brazilian Portuguese and Portuguese in other Portuguese-speaking countries, such as Portugal. We have generated a collection of Schemas in Brazilian Portuguese, and we have not evaluated how natural they sound to Portuguese speakers from other countries.

In comparison to the original Schemas, some of the Schemas had to be adapted in relation to the gender of the noun phrases. For instance, consider the sentence:

> **The trophy** doesn't fit into **the brown suitcase** because it is too large.

Its literal translation would be:

> **O troféu** não cabe **na maleta** porque **ele** é muito grande.

In Portuguese, however, objects are not gender-neutral, and, in this case, *troféu* is of masculine gender, while *maleta* is of feminine gender. This would make the pronoun *ele* to be very easily resolved, given that it refers to a masculine object, and the only masculine object in the sentence is *troféu*. We adapted such sentences so that they would follow all the rules for being a Winograd Schema, as long as there was a plausible adaptation that would not change the meaning of the sentence. For instance, we produced:

> **A medalha** não cabe **na maleta** porque **ela** é muito grande.

Given that *medalha* is of feminine gender, it now becomes possible for the pronoun to refer to either *medalha* or *maleta*.

For eight sentences (corresponding to 4 pairs of schemas) we could not find a suitable translation, and hence these were discarded in the Portuguese set.

An example of one such schema is:

> *They broadcast an announcement, but a subway came into the station and I couldn't hear [-/over] it.*

In this case, we couldn't find a natural-sounding translation for "hearing over it". Another example is the following.

[3]These datasets can be found at https://github.com/gabimelo/portuguese_wsc/blob/master/data/raw/portuguese_wsc_portuguese_names.html and https://github.com/gabimelo/portuguese_wsc/blob/master/data/processed/portuguese_wsc.json

I couldn't put the pot on the shelf because the pot was too [tall/high].

In this case, the difficulty in translation happened as the translations for tall and high can be used somewhat interchangeably in Portuguese.

4.2 An Initial Solver for the Portuguese-based WSC

To build an initial solver for the Portuguese-based WSC we employed a couple of different approaches that have been proposed for the English version of the challenge. The first one was an LSTM-based language model, as proposed by Trinh and Le (2018) (21). Their solution is an ensemble of models that are so large that actually running a test is a nontrivial matter. We thus pursued a much simpler language model than they did, and also a single model instead of an ensemble of models, as we were mostly interested in establishing an initial model that other researchers can easily run. Their solution is an ensemble of models with 1 to 2 billion parameters in each model of the ensemble; our best solution is a single model, containing 43 million parameters. We tested different configurations for such model, including the regularization options mentioned in Section 2.3.3. With the recent release of a Portuguese version of the BERT model (39), explained in Section 2.4, a BERT-based solution was another approach that was evaluated in terms of how well it works for answering Winograd Schema problems.

To the best of our knowledge, there is no language model corpus that is considered a widely established corpus for language model development for the Portuguese language. Due to this, we chose to develop our own corpus, which was derived from a set of Wikipedia documents, following a similar approach to the one used for generating the English Wiki-Text corpus (41).

The code for our solutions has been made publicly available, in its entirety.[4]

In the following sections, we will first explore the construction of our corpus, then we will discuss the LSTM-based solutions that we have developed, also mentioning a model that we developed in order to be able to run some comparisons for our results between the English and Portuguese versions of the challenge, and we will finish by explaining our BERT-based solution.

[4]https://github.com/gabimelo/portuguese_wsc

4.2.1 Corpus

Before we discuss the methods we used for answering the Winograd questions, we will first describe the corpus that we developed in order to train our models. Instead of utilizing a corpus that had already been prepared, we developed our own corpus for this task, as there seems to be no corpus currently established as a benchmark for language models in Portuguese. We derived our own from a Wikipedia dump which was the latest as of April 23rd, 2019 [5].

Due to the exceedingly large size of the full collection of Wikipedia documents, a subset was needed, as the extra size of the corpus would make training unfeasible and possibly without any additional gains from using this document collection in its entirety. To do so, we followed the approach used by Merity et al. (2016) (41) when developing the WikiText corpus, and applied filtering to select only the articles that are classified as Good[6] or Featured[7].

We used NLTK's[8] Portuguese tokenizer to parse the documents into tokens and kept only words appearing at least 3 times as part of the vocabulary. Words outside of the vocabulary were replaced by an <unk> token and end of sentences were represented by <eos>. We kept capitalization and punctuation in the final corpus.

This resulted in a corpus of 9.8 million tokens, where 9.4 million of them were used for training, and the remaining were split equally between test and validation sets. We have made this corpus publicly available.[9]

To evaluate the impact that the corpus size had on language model and WSC performance, we varied the size of the corpus and ran tests with those variations.

4.2.2 LSTM-based Language Models

The use of a language model for resolving Winograd Schemas goes as follows (the examples are in English but the process is the same for Portuguese schemas):

1. Each one of the candidate antecedents is substituted in place of the pronoun to be resolved. For instance, from this sentence:

[5] https://dumps.wikimedia.org/ptwiki/latest/ptwiki-latest-pages-articles.xml.bz2
[6] https://pt.wikipedia.org/wiki/Wikip%C3%A9dia:Artigos_bons
[7] https://pt.wikipedia.org/wiki/Wikip%C3%A9dia:Artigos_destacados
[8] https://www.nltk.org/
[9] https://github.com/gabimelo/portuguese_wsc/tree/master/data/processed

The trophy doesn't fit into the brown suitcase because **it** is too large.

We would then generate two sentences:

The trophy doesn't fit into the brown suitcase because **the trophy** is too large.

The trophy doesn't fit into the brown suitcase because **the suitcase** is too large.

2. Each of these generated sentences is passed on to the model. In this step, the one-hot encoding representation of all words in the sentence are sequentially sent to the language model; at each step, the generated probability for the next word in the sentence is stored.

3. The joint probability of each sentence is calculated. The sentence with the highest probability is assigned as the correct one. We employed two ways to calculate this probability, as did Trinh and Le (2018) (21). We describe these two ways of scoring the sentences later in Section 5.1.1.

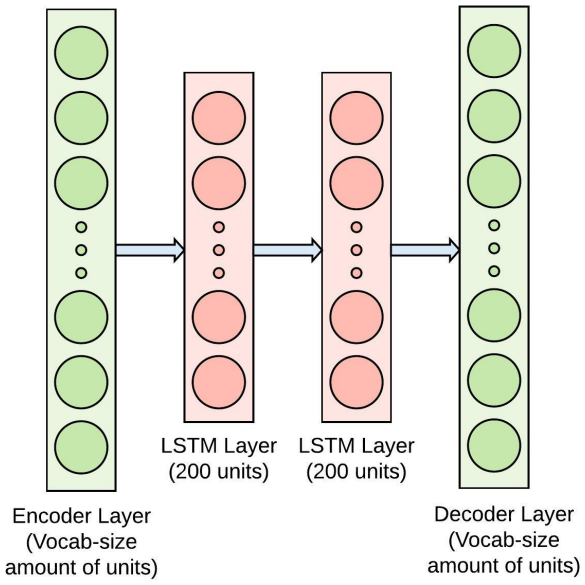
4.2.2.1 Initial Language Model

For our initial language model, we used a neural network with input and output layers with hidden unit size equal to that of the vocabulary, acting as encoding and decoding layers, using an embedding size of 200, and two LSTM layers, with 200 hidden units each; as displayed in Figure 3. We used dropout between layers, with a probability of dropout equal to 0.2.

To train the model, we used the cross-entropy loss as our cost function and optimized the parameters with the mini-batch gradient descent algorithm (using mini-batches of size 20). Sentences were organized into sequences of length 35. We used learning rate annealing and started with an initial learning rate of 20. Input and output embeddings were tied, as proposed by both Press and Wolf (2017) and Inan, Khosravi and Socher (2016) (32, 33). Gradients were clipped at 0.25 and training ran for 40 epochs.

We trained and evaluated the language model with the corpus that we developed, described in the previous section.

Figure 3: Initial language model architecture



4.2.2.2 English Comparisons

We also trained a similarly simple language model for English, and applied that to the original set of English Schemas, so as to compare how the approach works for these two different languages with similarly sized models. For this, the model was trained with the Wikitext-2 dataset (41). This dataset contains 33,278 unique tokens, having 2,075,677 training tokens, 216,347 validation tokens, and 244,102 test tokens. This model follows the same architecture as the Portuguese model. For these comparisons, we reduced the vocabulary size of our Portuguese corpus (by reducing the training set size), in order to have a similar vocabulary size and hence a similar model size.

4.2.2.3 Regularized Language Model

To analyze how much impact using a regularized model would have in the Portuguese WSC we trained a regularized neural network language model. We used a model referred to as AWD-LSTM (34), which employed a myriad of regularization techniques, described in detail in Section 2.3.3.

We started with the model that achieved state-of-the-art results for the WikiText corpus at the time of its proposal and then started experimenting with some parameter variations, in order to assess the impact that it would have in the Winograd Schema Challenge.

The main regularized model that we used — also referred to later on as Large AWD-LSTM — has 3 LSTM layers with 1150 units each. It also has the same encoding and decoding layers as our initial LSTM model but using an embedding size of 400. This model utilizes a few different types of dropout. For the dropout between layers, the dropout probability was set to 0.4 (meaning a 40% chance of being dropped out) and the one between recurrent layers was set to 0.2. Dropout for input layers was set to 0.65 (drops out weights from embedding layer), and finally, the embedding dropout was set to 0.1 (drops out full words from embedding layer). It has its weight drop parameter set to 0.5. The value of α to be used for the Activation Regularization was set to 2, and that of β, which gets used in the Temporal Activation Regularization, was set to 1.

The model was trained having the cross-entropy loss as cost function as well and was optimized for the initial epochs with the mini-batch gradient descent algorithm (with a batch size set to 80), until it would switch into using the Non-monotonically Triggered Averaged SGD algorithm (34). It uses a length of 70 tokens for the distribution of sequence lengths, as described in Section 2.3.3. It uses 30 as its initial learning rate and also clips gradients at 0.25. Input and output embedding weights were also tied. Training was set to run for 750 epochs.

We can notice that this is a substantially more complex model than ours, in terms of its size. This is reasonable to expect, given that the only regularization technique we are applying in our initial model is simple dropout between recurrent layers, whereas the AWD-LSTM model applies a plethora of regularization techniques.

We used the same corpus for training and evaluation as was used for our initial model.

4.2.3 BERT

The way we used BERT for answering Winograd Schema problems is similar to that for neural network-based language models, and we followed what was first proposed by Ruan et al. 2019 (23). We applied the same step 1 as mentioned above in Section 4.2.2, to generate the new sentences from the substitution of the pronoun. When we passed the sentences to the model, what we do is to add the separation token before the substituted pronoun. We then obtained the probability assigned by BERT to that being the sentence that follows the previous. Next, we passed the other sentence to the model, to obtain the probability of that continuation being the correct one. Finally, we compared the two probabilities that we obtained. The highest one made its respective sentence be considered the correct one. An example of the sentences that get generated to be used by the BERT

model is as follows:

> [CLS] The trophy doesn't fit into the brown suitcase because [SEP] **the trophy** is too large. [SEP]
>
> [CLS] The trophy doesn't fit into the brown suitcase because [SEP] **the suitcase** is too large. [SEP]

Although the papers that utilize BERT for the Winograd Schema Challenge use some sort of custom-made corpus for fine-tuning the pre-trained BERT model (3, 23), given that such corpora are in English, we did not utilize a similar approach for our work, as we were interested in obtaining the results from a simple initial test for the usage of the BERT model.

We used the BERT Portuguese model that was recently made available by Souza et al. (2019) (39), using Hugging Face's Transformer library (49). We tested both the large and the base sizes of the model.

5 EXPERIMENTS

We will now analyze the results obtained by our solver. The first step for this is to examine the methods that were used for this analysis – the scoring of the sentences, the metrics that were applied, how the Schemas were subdivided into further subsets, and some manual corrections that were used on the translated sentences. Then we move on to the presentation of results where we will show the results for our initial model, and for the improved version of the model, with the best hyperparameters we obtained. We will then show the results for our regularized model, comparing with those for the non-regularized versions of the model, and for our BERT-based model. We will then compare our solver applied to the Portuguese challenge and to the English challenge and the usage of our method applied to the English challenge as compared to the previous results presented on the literature for the English-based challenge (21). We also provide results on the alternate versions of the dataset that we are providing: the one with Portuguese names and the one with manual corrections.

5.1 Performance Analysis

This section presents details on how the performance of our solver was evaluated. It starts by explaining the two different scoring methods we utilized. We then indicate the metrics used when presenting the results for our models. As we mentioned in Chapter 3, new subsets for the evaluation of WSC solvers have been proposed by Trichelair et al. (2018), and have, since then, been used for reporting the performance of such solvers (46). Section 5.1.3 discusses the incorporation of this approach in our results. Lastly, we argue that grammatical mistakes introduced by the automatic substitution of candidate antecedents in place of the pronouns to be resolved — which are present in previous solutions for the English-based WSC — might impact on the performance of solvers, and explain how we measured this impact.

5.1.1 Scoring of Sentences

For the LSTM-based models, there are two approaches proposed by Trinh and Le (2018) for the scoring of the sentences (21). The first type is called full scoring. It is the ordinary joint probability of the sentence. That is:

$$Score_{full}(w_k \leftarrow c) = P_\theta(w_1, w_2, ..., w_{k-1}, c, w_{k+1}, ..., w_n),$$

where $w_k \leftarrow c$ indicates the word at position k being substituted by candidate c. The second way of scoring the model is with partial scoring, which is described as:

$$Score_{partial}(w_k \leftarrow c) = P_\theta(w_{k+1}, ..., w_n | w_1, ..., w_{k-1}, c).$$

We used the two approaches and present the results for each of them.

For the BERT-based solution, we get the score by simply applying the softmax operation to the output of the model's prediction for whether the second part of the sentence follows the first or not.

The candidate antecedent selected by the model, in all cases, will be the sentence with the highest score, out of the two sentences generated from the substitution of the candidate antecedents in place of the pronoun.

5.1.2 Metrics

The usual metric being used for measuring performance in the Winograd Schema Challenge (2, 3, 21, 23) is the accuracy metric. This metric is used as

$$accuracy = \frac{Number\ of\ Correct}{Size\ of\ Winograd\ Set}\ .$$

We should note that we considered a correct score as being a correct choice of pronoun substitution for a Winograd sentence — not for a Winograd pair — as was being done in the literature. This also means that the size of the Winograd set is the total amount of sentences in it (not the amount of sentence pairs).

We also use the consistency metric, which calculates for how many of the switchable sentences the system switches the answer accordingly between the original sentence and the switched version of it. That is, this method intends to analyze the impact of switching

the antecedents in the model's response. Ideally, the model would switch its answer accordingly; hence, if the model made a mistake on the original sentence, we would expect it to still make a mistake when switching the antecedents, and if the model had gotten the answer right in the original sentence, we would also expect that to still be the case when switching the antecedents. The consistency metric measures for how many of the sentences this actually happens.

5.1.3 Subsets of Schemas

Based on the criteria established by Trichelair et al. (2018) and mentioned in Chapter 3, we have used the subsets of the datasets made available by their work (46). These subsets propose a way for better understanding the robustness of solvers and were called the switchable and the associative sets.

For the Portuguese set, we considered as associative and as switchable the same sentences that were marked as such on their work. This resulted in 35 associative sentences and 135 switchable sentences. We have two associative sentences less than the work in English (two of the sentences that were not able to be translated into Portuguese had been considered associative in the English collection). The fact that we used the same subsets as proposed by them makes it so that we can compare our results between the English and the Portuguese versions of the challenge in these subsets as well as in terms of overall performance.

From these subsets, we will be able to analyze the accuracy of our model in a few different scenarios, apart from the overall accuracy, in the full set. First, we will look at our accuracy in the associative set, that is, the accuracy considering only the sentences that are considered associative. Then, we look at the complementary set, that is, the accuracy considering only the sentences that are not considered associative. After that, we make two analysis inside the switchable set. The first one is for the accuracy on the unswitched version of the sentences — that is, the original sentences, for those that have been considered switchable. Then, we gather the accuracy for the switched version of those sentences — that is, the switched sentences, for those that have been considered switchable. Lastly, we look at the consistency score, which calculates for how many of the switchable sentences the model switched its answer when the antecedents were switched.

Our collection of Schemas with names translated to Portuguese was also subdivided into these subsets.

5.1.4 Manual Corrections

Due to the naïve algorithm that is being used, both in our solution and previous ones, for the application of automatic substitutions of the pronouns to generate the sentences to be passed to the model, some of the cases become grammatically incorrect. For instance, in the sentence:

> **Jim** signaled **the barman** and gestured toward **his** empty glass.

This has as pronoun to be resolved *his* and as possible answers *A. Jim* or *B. The barman.* The naïve automatic substitutions become:

> Jim signaled the barman and gestured toward **Jim** empty glass.
> Jim signaled the barman and gestured toward **the barman** empty glass.

These sentences are missing the "'s" after the substituted candidates to indicate the possessive pronoun being applied to these nouns. We noticed that these sentences were left without any further corrections on the dataset of sentences after substitutions released by Trinh and Le (2018) and that there has not been any work analyzing if such errors might impact on the performance of WSC solvers (21).

In Portuguese, there are even more cases in which the automatic substitution does not work than in the English ones (where these cases are restricted to the usage of his/her pronouns). In Portuguese, the first type of sentence where this sort of error occurs are sentences where the usage of possessive pronouns is similar to the one presented in English and can be exemplified in the following sentence:

> Há uma fenda na parede. É possível enxergar o jardim através **dela**.

Where the substitution for the first possible antecedent becomes:

> Há uma fenda na parede. É possível enxergar o jardim através **a fenda**.

When it should instead be:

> Há uma fenda na parede. É possível enxergar o jardim através **da fenda**.

Another case where there were issues with the automatic substitution were those where the pronoun appears before the verb:

> Fred cobriu seus olhos com os antebraços, porque o vento estava carregando muita areia. Ele **os abriu** quando o vento parou.

Resulting in:

> Fred cobriu seus olhos com os antebraços, porque o vento estava carregando muita areia. Ele **seus olhos abriu** quando o vento parou.

But when the pronoun gets substituted it should instead be:

> Fred cobriu seus olhos com os antebraços, porque o vento estava carregando muita areia. Ele **abriu seus olhos** quando o vento parou.

For the English-based collection of Schemas, we kept these substituted sentences unchanged — that is, in the same manner as they were used by previous work that utilized this set. This was done so in order to have accurate comparisons of results. For the Portuguese set, we developed, in addition to the collection of sentences after the automatic substitutions, a collection containing the manually fixed sentences, in order to assess whether this sort of treatment for the automatic substitution of the pronouns would make any difference. This helps understand how well the solution proposed here could be expanded to less structured pronoun resolution problems - where manual fixes might not be feasible.

5.2 Results

Table 1 presents the main results from our experiments. Each of the rows represents the accuracy for the entire Portuguese dataset, for each of the scoring techniques (full and partial). Shortly after the publication of these results, an updated version of the schemas was published, making the language present in them more coloquial. The results present here correspond to the previous version of the schemas.

The model we called Small LSTM is the initial, non-regularized, model we described in Section 4.2.2.1. The improved version of it is the one that had better results from tests we ran where we varied the architecture of our Small LSTM model. It differs from the

small one in that it uses batches of size 40 instead of 20 and contains 3 LSTM layers with 400 units each, with the embedding size equal to the number of hidden units, instead of 2 layers of 200 units, and embeddings of size 200. The results for the Large AWD-LSTM are based on the model we described in Section 4.2.2.3. We later show some variations of the architecture of this model, to compare with our non-regularized models. Table 1 also presents results from our test using the BERT model. We show the results from using both the base and the large sizes of the Portuguese model, and also from using the multilingual model. As we mentioned in Section 4.2.3, we only utilize one scoring method for our BERT tests, which resembles the partial scoring for the LSTM-based models.

Table 1: Main Results

	Full	Partial
Small LSTM	48.01%	50.90%
Improved LSTM	50.18%	52.35%
Large AWD-LSTM	50.54%	46.93%
BERT Portuguese Base	–	47.65%
BERT Portuguese Large	–	49.82%
BERT Multilingual	–	50.18%

The best result was obtained with the improved version of our initial LSTM model, with the partial scoring method. However, it is necessary to mention that the differences between the models are not extremely significant, and, therefore, further analysis of confidence intervals could be carried on top of these to make more decisive decisions between the approaches. Additionally, the results are still not too far from a random-chance 50% result but they resemble the initial solutions for the English-based version of the Winograd Schema Challenge, and in our final chapter, we will discuss some possible future work that might be able to still improve these results significantly.

We will now look in more detail into each of these results shown in Table 1. We will also look at some comparisons between a reduced-size Portuguese model that we developed in order to compare to results that would be obtained by using the same model and test structure for solving the English-based version of the challenge and also compare those English results with some of the results that had been previously published in the literature. Then, we will analyze the impact that the variations of schema collections that we developed — the collection with Portuguese names and the one with the manual corrections we added, which we described in Section 5.1.4 — had in the results in the Portuguese-based WSC.

5.2.1 Non-regularized LSTM Models

There are two non-regularized LSTM models that we used for our experiments. One is the small and the other, the improved version of it that we found by changing some hyperparameters of the model. The detailed results from these models are shown in Table 2. Each of the rows, except for the last, represents the accuracy for a subset of the dataset, for each of the scoring techniques (full and partial). The unswitched and switched rows represent the two versions of the switchable subset. The last row refers to the consistency, which was defined in Section 5.1.2.

Table 2: Non-regularized LSTM Results

		Small	Improved
Original (Full Dataset)	Full	48.01%	50.18%
	Partial	50.90%	52.35%
Associative	Full	51.43%	51.43%
	Partial	62.86%	62.86%
Non-Associative	Full	47.52%	50.00%
	Partial	49.17%	50.83%
Switched	Full	48.89%	46.67%
	Partial	48.15%	46.67%
Unswitched	Full	47.41%	49.63%
	Partial	48.89%	50.37%
Consistency	Full	6.67%	8.15%
	Partial	14.81%	19.26%

The main improvements in the WSC results of the improved model happen for the non-associative subset. This makes it seem like the small model had already exploited everything from the associative sentences that the improved model would be able to, and the improved one was able to expand on sentences that are not associative. The consistency of both models is still pretty low, a fact that together with the results being close to 50% indicates that there is still quite some level of chance behind the results; although the improved version of the model also showed improved consistency (even though it increased the difference in results between the switched and unswitched versions of the switchable subset). It is also interesting to note that the partial and full scoring methods yield similar results, except for two particular metrics, where there is a larger difference between the scoring methods, with the partial method achieving better results — these metrics are the accuracy on the associative subset, and the consistency score. Once again, the differences between the models are not large, and further analysis of confidence intervals could be carried on top of these.

Another way of assessing the results for our models is to look at the results that

they obtain in the language modeling task itself. As described in Section 2.6, we use the perplexity metric. The small model achieved a perplexity of 104.36 in the test set, while the improved model achieved a result of 83.82 (lower perplexity is better).

We can also take a look at a snippet of text generated by the model. The blocks of equal signs (===, for instance), are present in the Wikipedia documents, as they delineate titles of articles and sections; hence, they can also show in generated text. The first snippet was generated by our small language model:

> Conselho e serva de <unk> . Seu nome foi impresso pela Família dos Liderans gondwanatérios , numa continuação a pé , o tão cedo criado para o homem “ <unk> ” nos outros gêneros solicitar divergência no tratamento de <unk> , o que gerou os elementos baseados na lei de 10 de maio e já que não aparece o uso das formas <unk> . Durante os meses do início da década de 1950 , em todo o ano o litoral da sociedade chegou desde a atual circulação da própria Casa de Habsburgo e pela vida de seu pai . Tal como a tradição alemã , a propaganda obriga à Presidência conquistando autonomia , atraindo que mais de 3 mil trabalhadores na área e a capacidade restante das responsabilidades do vídeo se ainda afetará mais . Um dos três tratamentos estáticos que foram totalmente homogêneos , a aprovar no exemplo de 1792 o governo durou finalmente em 22 de março de 1914 , quando os dois realizaram o museu e e que as forças francesas viam seus primeiros esquadrões no interior ou ao cargo . O termo artigo “ <unk> and Child ” foi solto e deveria parecer cada vez mais e o desafio === <unk> e <unk> === Entre os capitães dos criados do Brasil , “ O Capital Branco ” (lugar da cidade de <unk>) é o sexto capítulo do “ Exército de Estudos ” . “ <unk> ” ’ é um movimento de desenvolvimento tão lento que deriva do gamão que faz com que o governo federal se juntasse a forma de realidade avançado à medida . O feminismo Weisman zomba de : “ <unk> , <unk> , barato , café , desdobramentos e tosse para o mundo ” . “ ”

And this one, by the improved model:

> Conselho . Baseado desde a composição que os sobreviventes advogavam Rio Grande do Norte para país Corinthians-Itaquera , * ” <unk> al <unk> ” (<unk> - 2008) * <unk> de la Pamplonesa , pintura dividiria muitas das informações de <unk> (“ <unk> ”) : narrativas públicas e a influência de

várias tribos já existentes . * ” <unk> === <unk> <unk> ’ , literalmente , é uma lembrança do Livro de Cores , que também é aprovado pela lei estadual . * “ Este homem tem uma designação oficial para aquela língua , de um lado , permitindo uma mulher fazer parte das outras paz que elas encontramos nela foram . Alguns homens observam do lado de fora do homem . O respeito universal e o real conhecimento da até hoje metade das contas do <unk> recebem suspeita para que um utilizará o número de unidades com o intuito de buscar qualquer modificação à natureza do dever . Não é possível encontrar a estrutura das invenções ... . De início , há múltiplas observações de tempo . Alguns destes dados foram seguramente considerados os cruzeiros de sua expedição da avaliação da sociedade europeia . Essas investigações são mais importantes , especialmente pela de novos autores , cujos detalhes aceitam o conhecimento do sistema . <unk> a letra mais pobre do que as obras de antes do , e não haja nenhum uma só ; em suas relações , de acordo com a ciência e a ciência ou até da Primeira Guerra Mundial .

There are not visible improvements between the snippets. Both texts seem to follow the main grammatical norms and are able to correctly structure sentences. However, they both struggle to keep the text to a topic and are somewhat non-sensical semantically.

5.2.2 Regularized LSTM Models

We now look in more detail into the results for our AWD-LSTM results — the regularized LSTM model. We present results for three different versions of the model. The large one is the one described in Section 4.2.2.3, and was the one to achieve the best results for the WikiText corpus (41) . There is one difference however between how we trained the large model and what we had described in that section: due to computing capability constraints, we were not able to run for the full 750 epochs that had been suggested by Merity et al. (2017) (34), but rather for only 50 epochs. There is a chance therefore that the model would have continued to have its performance increased, with the continuation of training for more epochs. The so-called medium and small versions of the models are models we trained in order to compare with the results for the non-regularized versions of the model. Therefore, we utilized the same model architecture for these medium and small models as for the improved and small non-regularized models, respectively, albeit using all regularization techniques that constitute the AWD-LSTM model, which were explained in Section 4.2.2.3.

Table 3: AWD-LSTM Results

		Large	Medium	Small
Original (Full Dataset)	Full	50.54%	49.10%	49.10%
	Partial	46.93%	48.38%	50.18%
Associative	Full	57.14%	54.29%	51.43%
	Partial	54.28%	60.00%	57.14%
Non-Associative	Full	49.59%	48.35%	48.76%
	Partial	45.87%	46.69%	49.17%
Switched	Full	47.41%	48.15%	48.15%
	Partial	48.89%	49.63%	47.41%
Unswitched	Full	48.15%	47.41%	47.41%
	Partial	45.93%	46.67%	48.15%
Consistency	Full	5.93%	5.93%	5.93%
	Partial	9.63%	14.07%	16.30%

None of these regularized models were able to surpass our improved non-regularized LSTM model in the full dataset accuracy metric (which achieved 52.35% accuracy). They did not show improved consistency but did on the other hand display a small decrease in the difference between the associative and non-associative subsets.

Analyzing the results between these three different versions of the AWD-LSTM model, the large version did not show significant differences in results as compared to the smaller model sizes. Particularly, for the partial scoring, it had one of the lowest results for our models. It also did not perform better than its alternatives in the consistency metric. The subset in which it showed better results than the other variants was the associative subset when using full scoring. The models, however, once again did not show large differences between their results.

Looking at the results from these models in the language modeling task, the large model achieved a perplexity of 97.51 in the test set, the medium-sized model a value of 119.15, and the small one, 213.66. We can see that there is a significant difference between these models' results in the language modeling task; however, this difference did not seem to be transposed into improvements in the Portuguese-based WSC. We can also note that only the large AWD-LSTM was better than the small non-regularized LSTM, and none of the AWD-LSTM models surpassed the improved version of our non-regularized model in the language modeling task.

5.2.3 BERT Models

In this section, we present the results obtained by our BERT-based solution for the Portuguese-based WSC. It is important to remember that our solution did not apply any sort of fine-tuning to the BERT models, so these are the results of the BERT models as they are in their pre-trained format. There are two versions of the BERT model available for the Portuguese language — the base and the large versions. The large one was trained with a larger Transformer network than the base. For the multilingual model, there is only a single pre-trained set of weights.

Table 4: BERT Results

	Portuguese Base	Portuguese Large	Multilingual
Original (Full Dataset)	47.65%	49.82%	50.18%
Associative	45.71%	45.71%	60.00%
Non-Associative	47.93%	50.41%	48.76%
Switched	48.89%	48.15%	46.67%
Unswitched	46.67%	50.37%	48.89%
Consistency	37.04%	37.04%	84.44%

In comparison with the previous methods we presented, our BERT-based solution did not present improvements in results, performing worse than most of the results we presented in the previous sections. The most remarkable feature of our test with the BERT models is their consistency score — far higher than the observed in our LSTM models. The consistency result was especially significant in the multilingual model. The multilingual model, however, is the one presenting the largest difference between accuracy for the associative and non-associative subsets. A difference between the Portuguese BERT models and the ones we observed in the previous sections is that these present better results in the non-associative subset than in the associative one.

5.2.4 Comparison of our English and our Portuguese Models

In order to be able to assess how the solutions we are proposing as initial solvers for the Portuguese-based version of the WSC would perform in the English version of the challenge, we developed some methods to try to compare how similar models would perform in the English version of the challenge. In this section, we will first compare the performance of our LSTM-based solutions between the Portuguese and the English version of the challenge, and then we will do the same for the BERT-based models.

5.2.4.1 LSTM-based Models Comparisons

To be able to compare how our LSTM models would have fared in the English WSC, we developed an LSTM model in English, to apply that to the English WSC. To train that model, we used the WikiText-2 (41) dataset, which is a common corpus for language modeling in the English language. This corpus has a far lower vocabulary size than that of our corpus, used for training the models in the sections above: while WikiText-2 contains a vocabulary of 33,278 tokens, our corpus contains 98,242 tokens. This would result in an imprecise comparison between the performance of a similar model for the two versions of the challenge. We, therefore, created different versions of our corpus, to make such comparisons. Our initial idea was to develop a corpus with a similar amount of tokens in the vocabulary, while maintaining a similar amount of tokens in the training corpus, to be able to have the fairest comparison between the performance between the two languages. However, we found that a similar-sized vocabulary resulted in a much smaller training corpus size in Portuguese — a vocabulary of 33,681 tokens in Portuguese was present in a training set of only 1,329,372 tokens, while the WikiText-2 corpus contains 2,088,628 tokens in the training set. Consequently, maintaining a similar-sized training set resulted in a larger vocabulary than that used for training the English model — with 2,078,030 tokens in our training set, we had a vocabulary of 42,515 tokens. We then developed 2 different versions of this reduced corpus. One of them is the one that closely matches the vocabulary size of the WikiText-2 corpus (referred to as Pt-Same Vocab in Table 5) and the other that closely matches the training size of the WikiText-2 corpus (referred to as Pt-Same Train in Table 5).

Table 5: LSTM English and Portuguese Comparisons

		English	Pt-Same Vocab	Pt-Same Train
Original (Full Dataset)	Full	50.18%	47.65%	49.46%
	Partial	49.08%	45.85%	48.37%
Associative	Full	45.95%	37.14%	42.86%
	Partial	54.05%	45.71%	60.00%
Non-Associative	Full	50.85%	49.17%	50.41%
	Partial	48.31%	45.87%	46.69%
Switched	Full	48.09%	42.96%	42.96%
	Partial	50.38%	45.93%	47.41%
Unswitched	Full	48.09%	46.67%	48.89%
	Partial	46.56%	42.96%	45.93%
Consistency	Full	5.34%	22.22%	15.56%
	Partial	7.63%	25.93%	24.44%

The Portuguese model with similar training set size had results close to that of the

English model. The Portuguese model with the same vocabulary size had worse results, so the difference in training set size is likely an important contributing factor to it. We can note that the partial scoring in the associative subset for the model with similar training set size was a lot higher than for the English model, with a much larger difference between the associative and non-associative subsets than in English. Interestingly, our Portuguese models had better consistency scores than the English model (even better than the ones presented in previous sections).

It is also important to compare the performance of our model for the English set of Winograd Schemas to that of the work from which we based our solution on (21); this comparison is present in Table 6. Given that their publication was previous to that of the work introducing the associative and switchable subsets, we extracted the results from the latter (46). We only compare the results from their single language model solution (their better solution involves an ensemble of multiple models). We also only compare to the partial scores, as the results from full scoring were not disclosed.

Table 6: English Results - Partial Scoring

	English - Single LM (46)	English - Ours
Original (Full Dataset)	54.58%	49.08%
Associative	73.0%	54.05%
Non-Associative	51.7%	48.31%
Switched	54.20%	50.38%
Unswitched	54.96%	46.56%
Consistency	56.49%	7.63%

In terms of model size, their model consists of almost 1.8 billion parameters, while ours has 7.3 million; the difference in the capacity of the models is very significant. These results show that an improved language model can perform substantially better in the English dataset (21), which leads us to believe that reaching better performance for the Portuguese collection might also be achieved through the improvement of the language model being used.

5.2.4.2 BERT-based Models Comparisons

To compare our BERT results for the Portuguese WSC with those that BERT models would obtain in the English-based Winograd Schema Challenge, we simply used the pre-trained English models and the multilingual one for the English version of the challenge. We can compare the results shown in Table 7 with those that were presented in Table 4.

Table 7: English BERT Results

	English Base	English Large	Multilingual
Original (Full Dataset)	50.55%	50.55%	49.82%
Associative	48.65%	48.65%	48.65%
Non-Associative	50.85%	50.85%	50.00%
Switched	49.62%	50.38%	48.85%
Unswitched	50.38%	50.38%	48.85%
Consistency	29.01%	40.46%	63.36%

We can notice that the base and large versions of the English BERT model had better results in the English WSC than the Portuguese base and large models had in the Portuguese WSC; though the difference is quite small between the large models. The multilingual model, that achieved the best results for the Portuguese challenge, performed worse in the English challenge than in the Portuguese one. The associative subset had worse results for the English models than the non-associative ones, including the multilingual model, which, in the Portuguese WSC, had had better results in the associative than in the non-associative subset. Consistency for these models was also higher than for LSTM-based ones, but the English base and the multilingual models had lower consistency than the Portuguese base one, and the large one had only slightly higher consistency than that of the Portuguese large model.

5.2.5 Impact of Translation of Names and Manual Fixes

As reported in Section 5.1.4, we made some manual fixes to the automatic substitution of candidate antecedents in place of pronouns, to analyze whether this would be a necessary measure when solving the WSC utilizing language models. Table 8 shows the impact of manual fixes and also that of translating the names for some more commonly found in the Portuguese language, showing the full and partial scoring results for each of the sets: the main one (the one we have been showing results for this far), the one with the manual corrections, the one with names changed to Portuguese names, and the one with both Portuguese names and manual corrections. We made these comparisons only for our best-performant solution — the one achieved with the improved version of the non-regularized LSTM model.

For the full scoring technique, there does not seem to be much of a difference between applying these changes or not. For the partial scoring, however, there was benefit from applying the manual corrections, although maybe not a very large one. The fact that manual corrections did not imply in a much significant difference in results suggests it

Table 8: Portuguese Names and Manual Fixes

	Full	Partial
Main Set	50.18%	52.35%
Manually Fixed	49.82%	53.43%
Portuguese Names	50.54%	50.90%
Portuguese Names - Manually Fixed	50.18%	53.43%

might not be necessary to spend much effort into trying to improve the method for automatic substitution of candidates. Nevertheless, given that these aspects might have more of an influence if other approaches for solving the challenge were being used, we still find it relevant to release the collection of Portuguese Schemas with these translated names and manual fixes, in addition to the base collection.

6 CONCLUSION

We have developed a collection of Portuguese-based Winograd schemas; to the best of our knowledge, no similar collection exists today. The collection is of almost the same size as the original English collection, as only a few sentences did not have a suitable translation found. We can now have the Winograd Schema Challenge in the Portuguese language.

We have also developed an initial model for solving the Portuguese-based WSC, based on an approach that has produced good results for the English-based version of the WSC (21). We tested different network architectures and found that increasing model capacity can increase results in the WSC, even though increases in performance in the language modeling task does not always translate to improved results in the challenge. We have explored the usage of regularization techniques for the language model (34), and ran tests with the recently-developed Portuguese-based BERT model (39). The results obtained by these initial tests show just how difficult it is to solve the Winograd Schema Challenge.

It is worth noting that the solvers for the English challenge that had substantially larger performance than ours are all based on BERT or its variants (RoBERTa (48), for instance) and they all make use of additional corpus for fine-tuning the model. This seems like an interesting path for future work and possible improvements in the Portuguese-based version of the challenge.

Our code is publicly available so that future researches working on expanding on our project will be able to reproduce our results and also use it as a starting point. The collection of schemas, as well as the indication of the subsets to which each schema belongs to, is also publicly available, as well as the text corpus that we have developed for training Portuguese based language models. An article has already been published at a national conference on the field (50). This paper, presenting the only translation to the Portuguese language, has been mentioned on the official page for the Winograd Schema Challenge [1], together with the other very few translations to other languages.

[1] https://cs.nyu.edu/faculty/davise/papers/WinogradSchemas/WS.html

By providing the full code that was used for our project, we hope to also help the translation of the Winograd Schema Challenge into other languages, as many of the tools developed for handling text data to develop the translated set can be adapted to aid translations to new languages. The code developed for the evaluation of a solver on the Winograd Schema Challenge is also language and somewhat model agnostic, and therefore, can help future projects related to the Challenge — be it in Portuguese or in other languages, and using Language Models, or some other sort of solver.

REFERENCES

1 LEVESQUE, H. J. The Winograd schema challenge. In: *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*. [S.l.: s.n.], 2011. v. 46, p. 47.

2 SAKAGUCHI, K.; BRAS, R. L.; BHAGAVATULA, C.; CHOI, Y. Winogrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

3 KOCIJAN, V.; CRETU, A.-M.; CAMBURU, O.-M.; YORDANOV, Y.; LUKASIEWICZ, T. A surprisingly robust trick for the Winograd schema challenge. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019. p. 4837–4842. Disponível em: ⟨https://www.aclweb.org/anthology/P19-1478⟩.

4 LEVESQUE, H. J.; DAVIS, E.; MORGENSTERN, L. The Winograd schema challenge. In: *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*. AAAI Press, 2012. (KR'12), p. 552–561. ISBN 978-1-57735-560-1. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=3031843.3031909⟩.

5 BENDER, D. Establishing a human baseline for the Winograd schema challenge. In: *Proceedings of the 26th Modern AI and Cognitive Science Conference 2015, Greensboro, NC, USA, April 25-26, 2015.* [s.n.], 2015. p. 39–45. Disponível em: ⟨http://ceur-ws.org/Vol-1353/paper_30.pdf⟩.

6 PENG, H.; KHASHABI, D.; ROTH, D. Solving hard coreference problems. In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Denver, Colorado: Association for Computational Linguistics, 2015. p. 809–819. Disponível em: ⟨https://www.aclweb.org/anthology/N15-1082⟩.

7 RADFORD, A.; WU, J.; CHILD, R.; LUAN, D.; AMODEI, D.; SUTSKEVER, I. Language models are unsupervised multitask learners. *OpenAI Blog*, v. 1, n. 8, 2019.

8 WANG, A.; SINGH, A.; MICHAEL, J.; HILL, F.; LEVY, O.; BOWMAN, S. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 353–355. Disponível em: ⟨https://www.aclweb.org/anthology/W18-5446⟩.

9 WANG, A.; PRUKSACHATKUN, Y.; NANGIA, N.; SINGH, A.; MICHAEL, J.; HILL, F.; LEVY, O.; BOWMAN, S. R. Superglue: A stickier benchmark for general-purpose language understanding systems. *CoRR*, abs/1905.00537, 2019. Disponível em: ⟨http://arxiv.org/abs/1905.00537⟩.

10 DAVIS, E. Winograd schemas and machine translation. *CoRR*, abs/1608.01884, 2016. Disponível em: ⟨http://arxiv.org/abs/1608.01884⟩.

11 MORGENSTERN, L.; DAVIS, E.; ORTIZ, C. L. Planning, executing, and evaluating the Winograd schema challenge. *AI Magazine*, v. 37, n. 1, p. 50–54, 2016.

12 WEBSTER, K.; RECASENS, M.; AXELROD, V.; BALDRIDGE, J. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, v. 6, p. 605–617, 2018. Disponível em: ⟨https://www.aclweb.org/anthology/Q18-1042⟩.

13 SOON, W. M.; NG, H. T.; LIM, D. C. Y. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, MIT Press, v. 27, n. 4, p. 521–544, 2001.

14 RAHMAN, A.; NG, V. Resolving complex cases of definite pronouns: The Winograd schema challenge. In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Jeju Island, Korea: Association for Computational Linguistics, 2012. p. 777–789. Disponível em: ⟨https://www.aclweb.org/anthology/D12-1071⟩.

15 KOCIJAN, V.; LUKASIEWICZ, T.; DAVIS, E.; MARCUS, G.; MORGENSTERN, L. *A Review of Winograd Schema Challenge Datasets and Approaches*. 2020.

16 SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, IBM, v. 3, n. 3, p. 210–229, 1959.

17 MITCHELL, T. M. et al. Machine learning. 1997. *Burr Ridge, IL: McGraw Hill*, v. 45, n. 37, p. 870–877, 1997.

18 AMSILI, P.; SEMINCK, O. A Google-proof collection of French Winograd schemas. In: *Proceedings of the 2nd Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2017)*. Valencia, Spain: Association for Computational Linguistics, 2017. p. 24–29. Disponível em: ⟨https://www.aclweb.org/anthology/W17-1504⟩.

19 LIU, Q.; JIANG, H.; LING, Z.-H.; ZHU, X.; WEI, S.; HU, Y. Combing context and commonsense knowledge through neural networks for solving Winograd schema problems. *CoRR*, abs/1611.04146, 2016. Disponível em: ⟨http://dblp.uni-trier.de/db/journals/corr/corr1611.html\#LiuJLZWH16⟩.

20 OPITZ, J.; FRANK, A. Addressing the Winograd schema challenge as a sequence ranking task. In: *Proceedings of the First International Workshop on Language Cognition and Computational Models*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, 2018. p. 41–52. Disponível em: ⟨https://www.aclweb.org/anthology/W18-4105⟩.

21 TRINH, T. H.; LE, Q. V. A simple method for commonsense reasoning. *CoRR*, abs/1806.02847, 2018. Disponível em: ⟨http://arxiv.org/abs/1806.02847⟩.

22 EMAMI, A.; CRUZ, N. D. L.; TRISCHLER, A.; SULEMAN, K.; CHEUNG, J. C. K. A knowledge hunting framework for common sense reasoning. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 1949–1958. Disponível em: ⟨https://www.aclweb.org/anthology/D18-1220⟩.

23 RUAN, Y.; ZHU, X.; LING, Z.; SHI, Z.; LIU, Q.; WEI, S. Exploring unsupervised pretraining and sentence structure modelling for Winograd schema challenge. *CoRR*, abs/1904.09705, 2019. Disponível em: ⟨http://arxiv.org/abs/1904.09705⟩.

24 LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *nature*, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015.

25 GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. ⟨http://www.deeplearningbook.org⟩.

26 MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. [s.n.], 2013. Disponível em: ⟨http://arxiv.org/abs/1301.3781⟩.

27 VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L. u.; POLOSUKHIN, I. Attention is all you need. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., 2017. p. 5998–6008. Disponível em: ⟨http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf⟩.

28 BENGIO, Y.; DUCHARME, R.; VINCENT, P.; JANVIN, C. A neural probabilistic language model. *J. Mach. Learn. Res.*, JMLR.org, v. 3, p. 1137–1155, mar. 2003. ISSN 1532-4435. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=944919.944966⟩.

29 MIKOLOV, T.; KARAFIáT, M.; BURGET, L.; CERNOCKý, J.; KHUDANPUR, S. Recurrent neural network based language model. In: KOBAYASHI, T.; HIROSE, K.; NAKAMURA, S. (Ed.). *INTERSPEECH*. ISCA, 2010. p. 1045–1048. Disponível em: ⟨http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html\#MikolovKBCK10⟩.

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

31 CHO, K.; MERRIENBOER, B. van; GÜLÇEHRE, Ç.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014. Disponível em: ⟨http://arxiv.org/abs/1406.1078⟩.

32 PRESS, O.; WOLF, L. Using the output embedding to improve language models. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Valencia, Spain: Association for Computational Linguistics, 2017. p. 157–163. Disponível em: ⟨https://www.aclweb.org/anthology/E17-2025⟩.

33 INAN, H.; KHOSRAVI, K.; SOCHER, R. Tying word vectors and word classifiers: A loss framework for language modeling. *CoRR*, abs/1611.01462, 2016. Disponível em: ⟨http://arxiv.org/abs/1611.01462⟩.

34 MERITY, S.; KESKAR, N. S.; SOCHER, R. Regularizing and Optimizing LSTM Language Models. *arXiv preprint arXiv:1708.02182*, 2017.

35 WAN, L.; ZEILER, M.; ZHANG, S.; CUN, Y. L.; FERGUS, R. Regularization of neural networks using dropconnect. In: DASGUPTA, S.; MCALLESTER, D. (Ed.). *Proceedings of the 30th International Conference on Machine Learning*. Atlanta, Georgia, USA: PMLR, 2013. (Proceedings of Machine Learning Research, 3), p. 1058–1066. Disponível em: ⟨http://proceedings.mlr.press/v28/wan13.html⟩.

36 GAL, Y.; GHAHRAMANI, Z. A theoretically grounded application of dropout in recurrent neural networks. In: LEE, D. D.; SUGIYAMA, M.; LUXBURG, U. V.; GUYON, I.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems 29*. Curran Associates, Inc., 2016. p. 1019–1027. Disponível em: ⟨http://papers.nips.cc/paper/6241-a-theoretically-grounded-application-of-dropout-in-recurrent-neural-networks.pdf⟩.

37 DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A Large-Scale Hierarchical Image Database. In: *CVPR09*. [S.l.: s.n.], 2009.

38 PETERS, M. E.; NEUMANN, M.; IYYER, M.; GARDNER, M.; CLARK, C.; LEE, K.; ZETTLEMOYER, L. Deep contextualized word representations. In: *Proc. of NAACL*. [S.l.: s.n.], 2018.

39 SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. Portuguese named entity recognition using bert-crf. *arXiv preprint arXiv:1909.10649*, 2019. Disponível em: ⟨http://arxiv.org/abs/1909.10649⟩.

40 MIKOLOV, T.; DEORAS, A.; KOMBRINK, S.; BURGET, L.; CERNOCKÝ, J. Empirical evaluation and combination of advanced language modeling techniques. In: *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy, August 27-31, 2011*. ISCA, 2011. p. 605–608. Disponível em: ⟨http://www.isca-speech.org/archive/interspeech_2011/i11_0605.html⟩.

41 MERITY, S.; XIONG, C.; BRADBURY, J.; SOCHER, R. Pointer sentinel mixture models. *CoRR*, abs/1609.07843, 2016. Disponível em: ⟨http://arxiv.org/abs/1609.07843⟩.

42 JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing (2nd Edition)*. USA: Prentice-Hall, Inc., 2009. ISBN 0131873210.

43 SCHüLLER, P. Tackling Winograd schemas by formalizing relevance theory in knowledge graphs. In: *Proceedings of the Fourteenth International Conference on Principles of Knowledge Representation and Reasoning*. AAAI Press, 2014. (KR'14), p. 358–367. ISBN 1-57735-657-8, 978-1-57735-657-8. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=3031929.3031973⟩.

44 BAILEY, D.; HARRISON, A.; LIERLER, Y.; LIFSCHITZ, V.; MICHAEL, J. The Winograd schema challenge and reasoning about correlation. In: *Working Notes of the Symposium on Logical Formalizations of Commonsense Reasoning*. AAAI Press, 2015. Disponível em: ⟨http://www.cs.utexas.edu/users/ai-lab/?wsc15⟩.

45 SHARMA, A.; VO, N. H.; ADITYA, S.; BARAL, C. Towards addressing the Winograd schema challenge: Building and using a semantic parser and a knowledge hunting module. In: *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, 2015. (IJCAI'15), p. 1319–1325. ISBN 978-1-57735-738-4. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=2832415.2832433⟩.

46 TRICHELAIR, P.; EMAMI, A.; CHEUNG, J. C. K.; TRISCHLER, A.; SULEMAN, K.; DIAZ, F. On the evaluation of common-sense reasoning in natural language understanding. *CoRR*, abs/1811.01778, 2018. Disponível em: ⟨http://arxiv.org/abs/1811.01778⟩.

47 DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

48 LIU, Y.; OTT, M.; GOYAL, N.; DU, J.; JOSHI, M.; CHEN, D.; LEVY, O.; LEWIS, M.; ZETTLEMOYER, L.; STOYANOV, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

49 WOLF, T.; DEBUT, L.; SANH, V.; CHAUMOND, J.; DELANGUE, C.; MOI, A.; CISTAC, P.; RAULT, T.; LOUF, R.; FUNTOWICZ, M.; BREW, J. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

50 MELO, G. S. de; IMAIZUMI, V.; COZMAN, F. Winograd schemas in portuguese. In: *ENIAC 2019*. [s.n.], 2019. p. 787–798. Disponível em: ⟨https://sol.sbc.org.br/index.php/eniac/article/view/9334/9236⟩.

APPENDIX A – EXAMPLE OF TRANSLATED WINOGRAD SCHEMAS - WITH PORTUGUESE NAMES

1. Os vereadores recusaram a autorização aos manifestantes porque eles [temiam a/eram favoráveis à] violência.
   Answers: [Os vereadores/Os manifestantes]

2. A medalha não cabe na maleta porque ela é muito [grande/pequena].
   Answers: [A medalha/A maleta]

3. Jéssica certificou-se de agradecer Vanessa por toda ajuda que ela havia [recebido/oferecido].
   Answers: [Jéssica/Vanessa]

4. Paulo tentou ligar para o Gabriel, mas ele não [foi bem sucedido/estava disponível].
   Answers: [Paulo/Gabriel]

5. O advogado fez uma pergunta ao acusado, mas ele estava relutante em [repeti-la/respondê-la].
   Answers: [O advogado/O acusado]

6. O caminhão de entregas passou rapidamente pelo ônibus escolar porque ele estava indo muito [depressa/devagar].
   Answers: [O caminhão de entregas/O ônibus escolar]

7. Felipe sentiu-se [vingado/fracassado] quando seu rival de longa data André revelou que ele era o vencedor da competição.
   Answers: [Felipe/André]

8. O homem não conseguia erguer seu filho porque ele era muito [fraco/pesado].
   Answers: [O homem/O filho]

9. A grande bola atravessou a mesa pois ela era feita de [aço/isopor].
   Answers: [A grande bola/A mesa]

10. João não conseguia enxergar o palco com Arthur na sua frente porque ele é muito [baixo/alto].
    Answers: [João/Arthur]

11. Vinícius jogou sua mochila lá embaixo para o Raimundo depois que ele atingiu [o topo/a parte inferior] da escada.
    Answers: [Vinícius/Raimundo]

12. Apesar de ambas correrem aproximadamente na mesma velocidade, Sandra derrotou Marcia porque ela começou muito [bem/mal].
    Answers: [Sandra/Marcia]

13. A escultura caiu da prateleira porque ela não estava bem [acomodada/nivelada].
    Answers: [A escultura/A prateleira]

14. O desenho do Samuel estava pendurado imediatamente acima do da Tina e ele de fato ficava muito melhor com outro [abaixo/acima] dele.
    Answers: [O desenho do Samuel/O desenho da Tina]

15. Ana se saiu muito [melhor/pior] do que sua amiga Luciana na prova porque ela tinha estudado muito.
    Answers: [Ana/Luciana]

16. Os bombeiros chegaram [depois/antes] dos policiais porque eles estavam vindo de muito longe.
    Answers: [Os bombeiros/Os policiais]

17. Felipe estava chateado com o Vinícius porque a torradeira que ele havia [comprado dele/vendido para ele] não funcionava.
    Answers: [Felipe/Vinícius]

18. Guilherme [gritou com/acalmou] Luiz porque ele estava muito transtornado.
    Answers: [Guilherme/Luiz]