University of Brasilia
Institute of Exact Sciences
Department of Computer Science

# BERT Self-Learning Approach with Limited Labels for Document Classification of a Brazilian Army's Administrative Documentary Set

Carlos Eduardo de Lima Joaquim

Dissertação apresentada como requisito parcial para conclusão do Mestrado Profissional em Computação Aplicada

Dissertation submitted in partial fulfillment of the requirements of Universidade de Brasília for the degree of Master of Science in Applied Computing

Supervisor
Prof. Dr. Thiago de Paulo Faleiros

Brasília
2022

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

```
        de Lima Joaquim, Carlos Eduardo
dD278b     BERT Self-Learning Approach with Limited Labels for
        Document Classification of a Brazilian Army’s Administrative
        Documentary Set / Carlos Eduardo de Lima Joaquim;
        orientador Thiago de Paulo Faleiros. -- Brasília, 2022.
           63 p.


           Dissertação (Mestrado - Mestrado Profissional em
        Computação Aplicada) -- Universidade de Brasília, 2022.


           1. Deep Learning. 2. Semi-supervised Learning. 3. Self
        learning. 4. Natural Language Processing. 5. Text
        Classification. I. de Paulo Faleiros, Thiago, orient. II.
        Título.
```

University of Brasilia

Institute of Exact Sciences
Department of Computer Science

# BERT Self-Learning Approach with Limited Labels for Document Classification of a Brazilian Army's Administrative Documentary Set

Carlos Eduardo de Lima Joaquim

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Dissertation presented in partial fulfillment of the requirements for
the degree of Master of Science in Applied Computing

April 29, 2022

# Dedication

I dedicate my dissertation work to my beloved wife Renata R. S. Joaquim, for her constant serenity, understanding, encouragement and love throughout years of study; and to my daughters, Carolina, Teresa Clara, and Maria Eduarda, the twinkling stars that make my life shine.

I also dedicate this dissertation to my parents, Luiz Carlos and Deliene, who have made countless sacrifices aiming at my personal and professional success; and to my siblings, Claudio Marcio, Rossana Carla, and Francisco Jonathan, who have always been by my side in every battle of life.

# Acknowledgements

First of all, I would like to thank the blessings that God, in his eternal benevolence, has been pouring out on me. I am grateful for the privilege of my family and the unconditional support they give me in all the challenges that life presents me with. My gratitude to my dear and beloved wife Renata, the most virtuous soul, the purest and wisest heart, a true gift from God; my daughters Carolina, Teresa Clara, and Maria Eduarda; my parents Luiz Carlos and Deliene; my brothers, Claudio Marcio, Rossana Carla, and Francisco Jonathan; and my brother-in-law Fabio de Melo Oliveira. I thank them for their joint effort to provide every opportunity and support, without hesitation, throughout my life.

Secondly, I would like to thank my supervisor, Prof. Dr. Thiago de Paulo, for his patience, wisdom and continuous support throughout the efforts to produce this work. His leadership skills and balance in the course of his guidance, coupled with his vast knowledge, were fundamental to the development and achievements associated with this research, including our joint paper publication. His mentorship was a true blessing.

Nevertheless, it is imperative to express the most authentic gratitude in recognition of the support of Prof. Dr. Marcio Victorino, of whom I had the distinct privilege of being a student, and who dedicated part of his precious time to support the CCOp Mv project and the students of the 2020/1 class. I also thank Prof. Dr. Edison Ishikawa, for his support and efforts in making our joint paper publication a reality.

Knowing that friends are the family that God allowed us to choose for ourselves, I would like to thank the friends whom neither time nor distance have been able to weaken the true bond of friendship that still endures. I thank the support of my dear childhood friend, Claudio Prata Santiago, who has been by my side in all moments of my life, whatever they may be. The best moments sometimes occur at a great distance.

I also thank my dear friends Denny Rule, and Marsha Rule, noble and enlightened souls who have been instrumental in extending the linguistic resources I enjoy today, and who have always had the right words to make my days brighter.

I also wish to thank my dear friends Carlos Felipe da Rosa and Alessandro José de Oliveira, who accomplished missions by my side in the Brazilian Army, for their heartfelt long time camaraderie and long standing brotherhood, and Sérgio Holanda, noble soul and outstanding professional, for his endless encouragement and loyal friendship.

I would also like to thank my friends Hércio Pina, Carlos Henrique B. dos Santos, Diego R. da Silva, Kirus Dias, Pedro Reis, Marcio Valeriano, and Marcelo Lima for providing remarkable moments and great laughs throughout all these years. Their company was invigorating in the most grueling times.

Finally I would like to thank the Brazilian Army for the scientific initiative and support that made this personal achievement possible, giving this author the opportunity to contribute to the accomplishment of the Brazilian Army's doctrinal missions. I especially thank the Commanding General of the Systems Development Center, General Wolski, for all the support provided throughout the development of this research.

This work was supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES) as well, as they granted the author access to the CAPES' Portal de Periódicos.

# Resumo

O considerável aumento na velocidade de produção documental e, consequentemente, no volume de dados não estruturados armazenados nas instalações do Exército Brasileiro, especificamente na forma de documentos administrativos, acrescido da necessidade de consciência situacional por parte dos Comandos, além da observação da legislação arquivística vigente, impõe a execução de processos capazes de classificar documentos.

Neste diapasão, o Processamento de Linguagem Natural (NLP) surge como um importante recurso na persecução dos objetivos relativos à classificação documental, mostrando-se meio adequado para o desenvolvimento de pesquisa que vise à classificação de documentos considerando a realidade da produção documental atual, onde sobeja considerável número de amostras documentais não rotuladas.

Observado o fato de que os mais poderosos modelos NLP desenvolvidos baseiam-se em técnicas de aprendizado supervisionado, as quais exigem considerável número de amostras rotuladas, resta o desafio de encontrar modelo capaz de classificar conjunto de dados de uma Organização Militar (OM), parcialmente rotulado, de acordo com o Modelo de Requisitos para Sistemas Informatizados de Gestão Arquivística de Documentos (e-ARQ Brasil), alcançando performance equivalente ao nível humano.

Objetivou-se desenvolver, durante a condução da presente pesquisa, a expansão do modelo BERT, com a substituição do estágio supervisionado de ajuste fino por um método de autoaprendizagem, realizando-se a mensuração da performance resultante para porcentagens específicas do conjunto de dados, inicialmente compreendidas entre 3% e 30% do total de amostras rotuladas.

Os resultados obtidos permitiram vislumbrar a aplicabilidade do método proposto nas bases de dados de documentos do Exército Brasileiro. Concomitantemente, no estudo de caso em tela, foi possível verificar performance compatível com as necessidades existentes, sendo o método proposto capaz de classificar de forma equivalente à capacidade humana, apresentando melhores resultados que os experimento de referência, com ganhos maiores à medida em que o número de amostras rotuladas disponíveis decresce.

**Palavras-chave:** BERT, Processamento de Linguagem Natural, Autoaprendizado, Aprendizado Semissupervisionado

# Abstract

The remarkable acceleration in the production speed of documents and, consequently, in the volume of unstructured data stored at the Brazilian Army facilities, specifically in the form of administrative documents, plus the need of situational awareness by the Commanders, in addition to the observation of the archival legislation, requires processes that enable the capacity of classifying documents.

In this sense, Natural Language Processing (NLP) stands as an important asset in the pursuit of objectives related to document classification, proving to be an adequate means for developing research that aims to classify documents considering the reality of current document production, where there is a considerable number of unlabeled document samples.

Given the fact that the most powerful NLP models are based on supervised learning techniques, which require a considerable number of labeled samples, the challenge remains to find a model capable of classifying a partially labeled set of data from a Military Organization (OM), according to the Requirements Model for Computerized Document Management Systems (e-ARQ Brazil), reaching a human-level performance.

It was intended to develop, during the course of this research, the expansion of the BERT model, with the substitution of the supervised fine-tuning stage by a self-learning method, analyzing the resulting performance for specific percentages of the dataset, initially ranging from 3% to 30% of the total labeled samples.

The achieved results allowed us to perceive that the proposed method is applicable to the Brazilian Army's document databases. Concomitantly, in the case study in question, it was possible to verify that the performance of the proposed method is compatible with the existing needs, being able to perform classifications equivalent to the human capacity, presenting better results than the experiments of reference, with greater gains as the number of available labeled samples decreases.

**Keywords:** BERT, Natural Language Processing, Self-Learning, Semi-supervised Learning

# Contents

# List of Figures

# List of Tables

# Acronyms

**BERT** Bidirectional Encoder Representations from Transformers.

**CDS** Systems Development Center.

**CNN** Convolutional Neural Network.

**CONARQ** National Council of Archives.

**DNF** Disjunctive Normal Form.

**e-ARQ** Requirements Model for Computerized Archival Document Management Systems.

**IR** Information Retrieval.

**LM** Language Model.

**LSTM** Long Short-Term Memory.

**MLM** Masked Language Model.

**NLP** Natural language Processing.

**NSP** Nest Sentence Prediction.

**OM** Military Organizations.

**PTM** Pre-trained Language Models.

**RNN** Recurrent Neural Network.

**SCPAD** Permanent Subcommittee on Document Assessment.

**SIGAD** Computerized Archival Document Management System.

| | |
|---|---|
| **SIGADEx** | Army's Computerized Archival Document Management System. |
| **SSL** | Semi-supervised learning. |

# Chapter 1

# Introduction

*"Things won are done; joy's soul lies in the doing."*

— William Shakespeare

With the advent of the information age, where information and communication technologies became an essential asset, the potential use of applications exploring the possibilities of obtaining useful and timely knowledge became real. Brazilian institutions can potentialize its document's collection value, transforming it into a valuable asset, while following technical publications from National Council of Archives (CONARQ).

There is an appreciable amount of information being produced in a daily basis, with a significant part of this collection becoming records related to legal and historical matter. The first value regards the merit a document has to produce evidence before the law, and the second one concerns documents related to institutional origin, rights and objectives, its organization and development [1].

Taking into consideration that valuable data can be found within the produced set of information, existing in the documents, it is necessary to emphasize how important it is to identify and effectively use the correct information.

Azemi *et al.* [2] state that inadequate information management is listed as an issue of information quality, and in [3], it is shown that the defects arising from poor or inadequate information management are commonly noticeable in the failures of company personnel to respond to intelligence in a timely way.

Additionally, Azemi *et al.* [2] declare that large volumes of information available and stored make it difficult to access for the right information at the right time. Stating that this situation might lead to the information explosions, in accordance with [4]. Along the same line of thought, Redman [5 as cited in 2] affirms that, at the tactical level, poor information quality compromises decision making.

In the same vein, a related subject, the concept of document classification, arises as the prominent idea regarding definitions set by CONARQ in its publications. More specifically, the Requirements Model for Computerized Archival Document Management Systems (e-ARQ).

In the Brazilian Army the current registered document traffic between Military Organizations (OM), synonym for Army Bases, surpasses one million three hundred ninety thousand documents a year, as registered in 2021 inner statistics, not considering the documents which are created, processed, and archived in a OM, never leaving it. Thus, the aforementioned inner statistics show that more than 22 million documents need to be evaluated and classified according to e-ARQ in order to fulfill the requirement of setting a proper destination to them.

The documentary set produced in the several OM across the country contains structured and unstructured information. The needed classification of this corpus, according to the e-ARQ, is the first step to identify and address valuable information while in compliance with technical regulations. That massive amount of data would take too long to be processed and assessed, if considering the possibility of scrutiny carried out exclusively by human hands, specifically the several Permanent Subcommittee on Document Assessment (SCPAD) existing in every OM throughout the country.

This document production started to increasingly grow after the Army's Computerized Archival Document Management System (SIGADEx) initiative. A system created by Systems Development Center (CDS), in 2006, as a strategic issue aiming at the integration and standardization. This system brings controls to its users, related to the archival process, increasing productivity while providing flexibility and traceability over existing data.

The above-mentioned e-ARQ was chosen to be the classification model given that this archetype is the standard reference to Computerized Archival Document Management Systems (SIGAD), being intrinsically linked to SIGADEx's main goals as a project.

Thus, this intended study is related to the pressing need to properly classify documents produced by the Brazilian Army, allowing correct treatment and full compliance with the requirements established by the CONARQ, the aforementioned e-ARQ.

The persecution of state-of-the-art results in the classification activity lead us to consider researching and applying specific algorithms to the documentary set. In this regard, we found language models to suit our needs, considering Petroni *et al.* [6], where it is stated that language models have many advantages, once they require no schema engineering, allow practitioners to query about an open class of relations, are easy to extend to more data, and require no human supervision to train.

In this setting, language models come with other attractive properties: they do not need human annotations, and they support an open set of queries. Moreover, some models show a surprisingly strong ability to recall factual knowledge without any fine-tuning [6].

In the present scenario, the shortage of labeled samples shall be considered as a premise. With this information being known, one major question when classifying documents, while following the supervised learning paradigm, is the existing need of a substantial number of labeled samples, in order to adequately generalize the model and make predictions of unseen samples, the supervised learning approaches does not become suitable as a deemed solution.

As the foregoing restriction regarding the number of labeled samples emerges as a limitation in several scenarios, the opposite turns out to be true. While labeled data is expensive to obtain, unlabeled data is essentially free in comparison [7]. It can be seen that creating large datasets to certain supervised learning problems requires a great deal of human effort, pain and/or risk or financial expense [8, 9]. This need for supervision poses a major challenge when we encounter critical scientific and societal problems where fine-grained labels are difficult to obtain [10].

As text classification is considered a classic and fundamental task in Natural language Processing (NLP) with a wide spectrum of applications such as question answering [11–13], spam detection [14–16] and sentiment analysis [17–19], many deep learning-based classifiers, including convolutional neural networks (CNN) and recurrent neural networks (RNN), have been developed and achieved great success when trained on large-scale labeled documents [20].

In this line of thought, it can be seen in Oliver *et al.* [8] that semi-supervised learning (SSL) provides a powerful framework for leveraging unlabeled data when labels are limited or expensive to obtain. The method can trace back to 1970s, and it attracts extensive attention since 1990s [21 as cited in 22].

Considering that the size of modern real world datasets is ever-growing so that acquiring label information for them is extraordinarily difficult and costly [9, 20], deep semi-supervised learning is becoming more and more popular [22].

It becomes an attractive approach towards addressing the lack of data, once, in contrast with supervised learning algorithms, SSL algorithms can improve their performance by also using unlabeled examples. Additionally SSL algorithms generally provide a way of learning about the structure of the data from the unlabeled examples, alleviating the need for labels [8].

Similar view is shared by Nigam *et al.* [23], who declare that unlabeled data do contain information about the joint distribution over features other than the class label. Because of this they can sometimes be used — together with a sample of labeled data — to significantly increase classification accuracy in certain problem settings.

Self-learning is a field within semi-supervised learning [24]. It is also known as self-training or self-labeling, and is usually called a wrapper for semi-supervised learning tasks [25].

Being first described by Tsypkin [26] as learning without reinforcement, it was later considered by Mämmelä *et al.* [27] as a misnomer, once Tsypkin was describing unsupervised learning, and in self-learning there is really a teacher that has defined the methodology for learning.

## 1.1 Objectives

Throughout this study, it is expected to find language model that dexterously classify the partially labeled dataset set according to the e-ARQ, reaching human-level performance [28] in the classification results in a set of documents belonging to the Brazilian Army.

It is envisaged to expand the use of Bidirectional Encoder Representations from Transformers (BERT) [29], and replace the fully supervised fine-tuning stage for a self-learning method, expecting that the process surrounding BERT downstream tasks come to a entirely semi-supervised process, successfully achieving suitable scores when evaluated.

Furthermore, from an organizational perspective, it is expected that the results coming from this research fulfill the objective of allowing adequate support to properly classify documents, assisting SCPADs to do their job, whereas it is a necessary activity to carry out the formal treatment of the documents once produced by the Brazilian Army, according to what is determined by its rules [30].

## 1.2 Hypotheses

With the purpose of addressing the problem presented in this Chapter, the following hypotheses are proposed:

1. A most heterodox strategy, considering the worst case scenario, in which the number of available labeled samples would be quite low, where it could be considered the need of bringing prejudice to the validation set.

2. A well rounded self-learning algorithm where the, after reaching the stopping criterions, a last classifier is trained and returned, together with the labeled set.

3. A self-learning algorithm in which an early stopping strategy is used, and the final classifier is limited to the one capable of emptying the unlabeled set, without any further new model.

As a means of assessing the proposed method performance other experiments were created in order to measure how well the self-learning approach would perform against the active learning (AL) method and the classic BERT.

## 1.3 Contributions

The main contributions of this dissertation are as follows:

1. We measured and analyzed the degree of effectiveness of the proposed method when utilizing a Brazilian Army's dataset as sample source. The results showed that the proposed method was capable of bringing excelling results, granting the applicability of it in the current scenario.
2. We proposed a self-learning approach that modified the fine-tuning procedure associated to BERT, allowing one to enhance the classification quality regarding to given dataset.
3. We created a heterodox experiment where the validation set suffered prejudice, showing that, even in the worst scenarios, where the labeled samples scarcity limits even the classic approach, it was possible to use the proposed method successfully.
4. Our proposed method achieved successful results, as a consequence of the compilation of the above-mentioned contributions, allowing us to publish the paper [31]:

   Joaquim, Carlos Eduardo de Lima and Thiago de Paulo Faleiros: *BERT Self-Learning Approach for Document Classification with Limited Labels.* The 16th Learning and Intelligent Optimization Conference, June 2022.

## 1.4 Dissertation Outline

This section describes the content of the next chapters of this dissertation. In Chapter 2, a comprehensive wide-ranging review of the existing literature is conducted, tangentiating topics that are considered important to give the reader a thorough understanding about the theoretical foundations that lead this research.

It elaborates on NLP, then specifically talks about the NLP activity named Text Classification, next presents the concepts of self-supervised learning, followed by the theory behind the concept of self-learning; and gives detailed notions of bidirectional language models and transformers, to finally present concepts of BERT.

It also presents the related works in a chronological order, allowing one to perceive how the scientific community evolved its concepts and conceptions about the state-of-the art strategies there were elaborated over time on this specific challenge of dealing with a limited number of labeled samples.

Chapter 3 explains the chosen approach and methodology, including information about the research design that is part of the experiment. This chapter includes knowledge about the dataset that was used, and the necessary procedures over the dataset, in order to make normalization be done successfully.

In the third Chapter of this work, the strategy of how this work will be developed using self-learning is also covered, succeeded by a brief section regarding BERTimbau and the downstream task, text classification, and finally addressing the experiment evaluation, presenting information about how the results are going to be assessed.

Chapter 4 is written to let one obtain satisfying information about the results of the experiments conducted using a self-learning approach on BERTimbau, in order to classify the documents pertaining to the provided dataset. In this Chapter, the results of the experiments are going to be described and evaluated.

Lastly, in Chapter 5, a concise overview of the experiment results, and noticeable abstractions about the results, are presented, exhibiting the conclusions of the proposed method of this dissertation.

# Chapter 2

# Literature Review

*"Imagination is more important than knowledge. For knowledge is limited to all we now know and understand, while imagination embraces the entire world, and all there ever will be to know and understand."*

— Albert Einstein

This chapter presents this work's theoretical foundation, with main concepts of Natural language Processing, text classification, self-supervised learning, and semi-supervised learning being portrayed; together with the outlined self-learning concept, and bidirectional language models idea; followed by the definitions of Transformers, and BERT.

## 2.1 Natural Language Processing

Natural language processing is the set of methods for making human language accessible to computers [32, 33], sharing similarities with parallel disciplines such as computational linguistics, which is concerned with modeling language using rule-based models [33]. Between all the tasks that grew into part of everyone's daily life [32], text classification is a specific class of problems that have been widely studied and addressed in many real applications over the last few decades [34].

Surely, there are other applications that has become embedded in our daily lives, as automatic machine translation, text summarization, and text generation. These diverse applications are based on a common set of ideas: drawing on algorithms, linguistics, logic, statistics, and more [32].

According to the same author, natural language processing is focused on the design and analysis of computational algorithms and representations for processing natural human language, having as goal to provide new computational capabilities around human language, *e.g.*, extracting information from texts, translating between languages, answering questions, holding a conversation, taking instructions, and so on.

Some authors say that NLP's inception can be traced back to the development of computer science in the 1940s, moving forward along with advances in linguistics that led to the construction of formal language theory, which models language on increasingly complex structures and rules to these structures [33].

Other authors declare NLP as born in the 1950s, being the intersection of artificial intelligence and linguistics. According to them, NLP was originally distinct from Text Information Retrieval (IR), which employs highly scalable statistics-based techniques to index and search large volumes of text efficiently. With time, NLP and IR have converged somewhat [35].

Natural language Processing has some close neighbors as it draws on many other intellectual traditions, from formal linguistics to statistical physics, *i.e.*, computational linguistics, machine learning, artificial intelligence, computer science, speech processing, ethics, and others.

As stated in [32], Natural language Processing covers a diverse range of tasks, methods, and linguistic phenomena; despite the diversity of topics it relates to, some general themes emerge, more specifically three themes as outlined below. Each one of those themes can be expressed as an opposition between two extreme viewpoints on how to process natural language.

1. Learning and Knowledge

   One extreme proposes to use machine learning to train end-to-end systems that transmute raw text into any desired output structure, the other extreme understands the core work of natural language processing as being transforming text into a stack of general-purpose linguistic structures, inasmuch as these general-purpose structures should then be able to support any desired application [32].

2. Search and Learning

   As stated by Eisenstein [32], many Natural language Processing problems can be written mathematically in the form of optimization,

   $$\hat{y} = \underset{y \in \mathcal{Y}(x)}{argmax} \Psi(x, y; \theta) \tag{2.1}$$

   where,

   - $x$ is the input, which is an element of a set $\mathcal{X}$;
   - $y$ is the output, which is an element of a set $\mathcal{Y}(x)$;

- $\Psi$ is a score function (also named model), which maps from the set $\mathcal{X} \times \mathcal{Y}$ to the real numbers;
- $\theta$ is a vector of parameters for $\Psi$;
- $\hat{y}$ is the predicted output, which is chosen to maximize the scoring function.

The above formulation implies that language processing will have two different modules, *i.e.*, search and learning. It remains clear that the above defined basic structure can be applied to a huge range of problems, reflecting an implicit decision with respect to language processing algorithms, scilicet, they will have two distinct modules [32]:

(a) Search

The search module is responsible for computing *argmax* of the function $\Psi$, that is, it finds the output $\hat{y}$ that gets the best score with respect to the input $x$.

(b) Learning

The learning module, conversely, is responsible for finding the parameters $\theta$. Usually, it is done by processing a large dataset of labeled examples, $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$.

The division of natural language processing into separate modules for search and learning makes it possible to reuse generic algorithms across many tasks and models, knowing that much of the work of natural language processing can be focused on the design of the model $\Psi$ while reaping the benefits of decades of progress in search, optimization, and learning [32].

3. Relational, Compositional and Distributional perspectives

As stated by Eisenstein [32], any element of language can be described from at least three perspectives. The relational perspective on meaning is the basis for semantic ontologies, which enumerate the relations that hold between words and other elementary semantic units.

As the meaning of the words observes the principle of compositionality, which portends that a word is constructed from the constituent parts, the same line of thought can be applied to larger units as phrases, sentences, and beyond [32].

This distributional perspective, for once, makes it possible to learn about meaning from unlabeled data alone. Unlike relational and compositional semantics, no manual annotation or expert knowledge will be required. Distributional semantics is thus capable of covering a huge range of linguistic phenomena [32].

### 2.1.1 Textual preprocessing

As for Natural language Processing, regular expressions are considered the most important tool for describing text patterns. They can be used to specify strings one might want to extract from a document, and play an important role as part of a set of tasks that are collectively called text normalization [36].

Before almost any natural language processing of a text, the text has to be normalized. Text normalization stands for a set of three tasks commonly applied as part of any normalization process [36]:

1. Tokenizing (segmenting) words;
2. Normalizing word formats; and
3. Segmenting sentences.

Tokenization can be defined as the task of segmenting running text into words, which also may encompass the expansion of clitic contractions. Word normalization can trace its roots back to the concept of putting words or tokens in a standard format, choosing a single normal form for words with multiple forms. This standardization may be valuable, despite the spelling information that is lost in the normalization process. Another step of normalization is case folding, where the tokenizer maps everything to lower case [36].

As one word could appear in different forms (i.e., singular and plural noun form) while the semantic meaning of each form is the same [34], and there are advantages in consolidating different forms of a word into the same feature space, lemmatization can be applied, since it consists in the task of determining that two words have the same root, despite their surface differences [36].

The current most refined methods for lemmatization involve complete morphological parsing of the words. To make it clear how a morphological parser works, it would take a word like cats and would parsed it into the two morphemes cat and s [36].

As stated by the same author [36], sentence segmentation is the process of segmenting a text into sentences, typically using punctuation as cues. In general, sentence tokenization methods work by first deciding whether a period is part of a word or is a sentence-boundary marker. It can be done based on abbreviation dictionaries, rules or machine learning.

## 2.2 Text Classification

Text Classification[1] consists in the activity of labeling natural texts with thematic categories from a predefined set [37–39]. Historically, text classification most popular approaches in the 1980s were knowledge engineering techniques applied on expert systems manually built [37]. Such systems would typically lie on sets of logical rules of type

**if** (DNF formula) **then** (category)

where, in its most general form, Disjunctive Normal Form (DNF) formula is a disjunction of conjunctive clauses [37, 40].

Since the early 1990s, the Machine Learning approach to text classification has gained popularity and has eventually become the dominant one, at least in the research community. This approach is based in a general inductive process[2] designed to automatically build a *classifier* for a given category $c_i$ [37].

From a set of documents with known categories, the formal definition of text classification is expressed as follows:

$$Z = f(K, \Theta) + \varepsilon \tag{2.2}$$

In the Equation 2.2, as described in [38], $K$ is the text representation, $\Theta$ is the set of unknown parameters associated with the function $f$ (named *classifier* or *classification model*), the desired estimation that should be established using the training data, and $\varepsilon$ is the error of the classification.

Bearing in mind that $Z = h(K)$ has $h$ as the true and unknown function, it becomes clear that the added error $\varepsilon$ is desired to be the smallest possible, once $f$ is just an approximation of $h$. The smaller the $\varepsilon$, the more effective is the classifier $f$ [38].

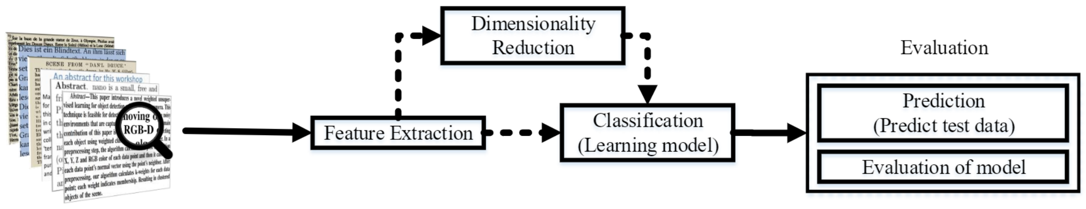


Figure 2.1: Overview of traditional text classification pipeline (Source: [34]).

As said by Kowsari *et al.* [34], in general, text classification systems are structured in terms of the pipeline illustrated in Figure 2.1, and contain four different levels of scope, being them:

[1] Also known as text categorization, or topic spotting
[2] Also called *learner*

1. Document level, where the algorithm obtains relevant categories of a full document.
2. Paragraph level, wherein the algorithm obtains relevant categories of a single paragraph.
3. Sentence level, in which relevant categories are obtained of a single sentence.
4. Sub-sentence level, where the algorithm obtains the relevant categories of sub-expressions within a sentence.

Most text classification and document categorization systems can be deconstructed into the following four phases: Feature extraction, dimension reductions, classifier selection, and evaluations [34]. In [39], preprocessing and indexing are also included as stages, while feature extraction is considered part of dimensionality reduction.

The area of text categorization is vast and finds in feature selection an important problem. In the hard version of the classification problem, a particular label is explicitly assigned to the instance, whereas in the soft version of the classification problem, a probability value is assigned to the test instance [40].

### 2.2.1 Traditional Text Classification pipelines

According to Figure 2.1, there are four main steps in the traditional text classification pipelines, of which details will be presented next.

#### Feature Extraction

As texts and documents are, in general, unstructured datasets, there is motive to convert them into a structured feature space, insomuch as mathematical modeling is used as part of a classifier. This activity is brought to scene just after the preprocessing step. The methods related to this undertaking are categorized as either word embedding or weighted word [34].

#### Dimensionality Reduction

It is not uncommon to find text or document datasets containing a large score of unique words, making data preprocessing steps to be lagged by high time and memory complexity. There is one choice that can be made and adopted as workaround, it is possible to use inexpensive algorithms [34].

But another problem emerges as a result of the inexpensive algorithm's adoption: in some datasets, these kind of cheap algorithms do not perform as well as expected. There is when dimensionality reduction may come at an opportune time. It can help reduce time and memory complexity and could be more efficient than developing inexpensive classifiers [34].

### Classification Techniques

Considered by Kowsari *et al.* [34] as the most important step of the text classification pipeline, choosing the best classifier is crucial to the endeavor. It is emphasized by the same authors that without the complete understanding of each algorithm, it is not possible to determine the most efficient model for a text classification application.

Lately, it is known that deep learning approaches have achieved surpassing results in comparison to previous machine learning algorithms on tasks such as image classification, natural language processing, face recognition, etc [34]. The success of these deep learning algorithms relies on their capacity to model complex and non-linear relationships with data [34, 41].

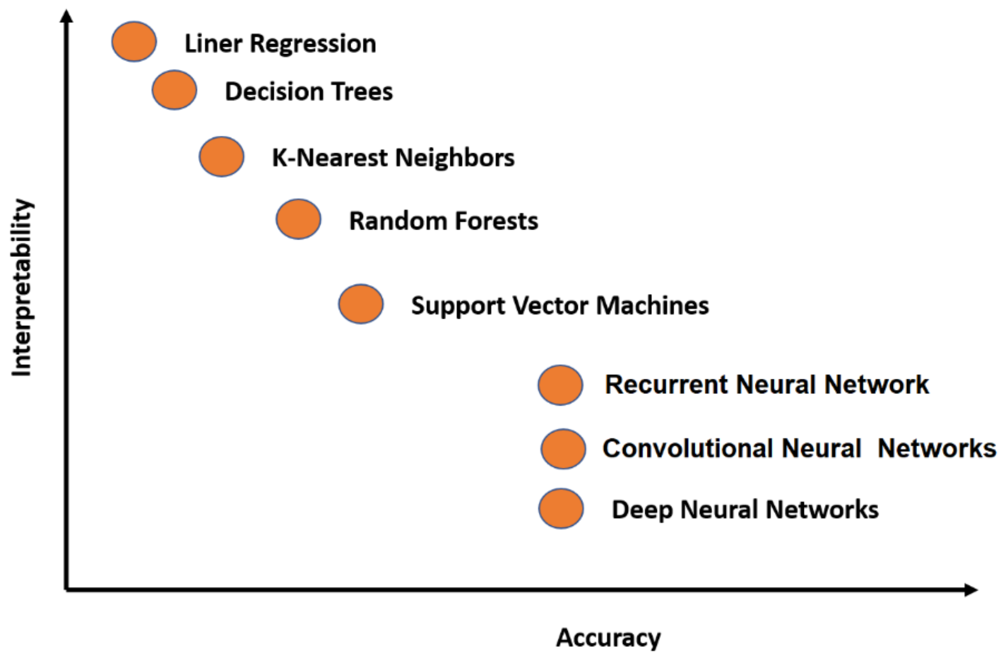


Figure 2.2: Interpretability comparison betwixt traditional and deep learning techniques (Source: [34]).

It is also important to emphasize that deep learning techniques has limitations regarding model interpretability, being a limiting factor for use cases requiring explanations of features involved in modeling. As shown in Figure 2.2, the more accurate the model, the lower the interpretability, meaning that deep learning complex algorithms are hard to understand [34].

**Evaluation**

Having evaluation as the final part of text classification pipeline, it becomes essential to understand how well a model can perform, in order to make it possible to use and develop text classification methods. There is a considerable number of methods available for evaluating supervised techniques, in this line of thought, f1-score stands out as one of the best methods of evaluation, and can be describe as [42]:

$$F_1 = \frac{2PR}{P + R} \tag{2.3}$$

where,

- Precision $P$ is the positive predictive value [42], being the fraction of true positive examples among the examples that the model classified as positive. In other words, the number of true positives divided by the number of false positives plus true positives [43].

$$P = \frac{tp}{tp + fp} \tag{2.4}$$

- Recall $R$ is the sensitivity, the fraction of examples classified as positive, among the total number of positive examples. In other words, the number of true positives divided by the number of true positives plus false negatives [43].

$$R = \frac{tp}{tp + fn} \tag{2.5}$$

- $tp$ is the number of true positives.
- $tn$ is the number of true negatives.
- $fp$ is the number of false positives.
- $fn$ is the number of false negatives.

The confusion matrix showed in Table 2.1 brings a representation of true and false positives and negatives, helping one understand their occurrence and resultant evaluation.

## 2.2.2 Training Set, Test Set, and Validation Set

As noted by Sebastiani [37], the machine learning approach relies on the availability of an initial corpus $\Omega = \{d_1, ..., d_{|\Omega|}\} \subset \mathcal{D}$ of documents preclassified under $\mathcal{C} = \{c_1, ...c_{|\mathcal{C}|}\}$. That is, given the total function $\breve{\Phi}$, the values of $\breve{\Phi} : \mathcal{D} \times \mathcal{C} \longrightarrow \{T, F\}$ are known for every pair $\langle d_j, c_i \rangle \in \Omega \times \mathcal{C}$.

|  |  | True Class |  |  |
|---|---|---|---|---|
|  |  | Positive | Negative | Total |
| Model's Decision | Positive | $tp$ | $fp$ | $tp + fp$ |
|  | Negative | $fn$ | $tn$ | $fn + tn$ |
|  | Total | $tp + fn$ | $fp + tn$ | $N$ |

Table 2.1: Confusion matrix showing the occurrence of true and false positives and negatives.

As a consequence of the above-mentioned function $\breve{\Phi}$, a document $d_j$ can be said as a positive example of the preclassified $c_i$ if $\breve{\Phi}(d_j, c_i) = T$. In turn, every negative example of $c_i$ occurs when the result of the function $\breve{\Phi}$ is $F$, $\breve{\Phi}(d_j, c_i) = F$.

The same author [37] maintain the same line of thought when state that, once a classifier $\Phi$ has been built it is desirable to evaluate its effectiveness on both research and operational settings. Then he emphasizes that, prior to the classifier construction, the initial corpus $\Omega$ has to be split in two sets, not necessarily of equal size:

1. Training and validation set

   The training and validation set is defined as $TV = \{d_1, ..., d_{|TV|}\}$, where the classifier $\Phi$, for categories $\mathcal{C} = \{c_1, ..., c_{|\mathcal{C}|}\}$, will be inductively built by observing the characteristics of these documents;

2. Test set

   The test set is specified as $Te = \{d_{|TV|+1}, ..., d_{|\Omega|}\}$. This set application is related to the effectiveness test of the classifier. Each $d_j \in Te$ is submitted to the classifier, and every classifier decision $\Phi(d_j, c_i)$ is compared with the expert decision $\breve{\Phi}(d_j, c_i)$. As a result, a measure of classification effectiveness is achieved, being based on how often the $\Phi(d_j, c_i)$ values match the $\breve{\Phi}(d_j, c_i)$ values.

According to Mitchell [44 as cited in 37], it is crucial that the documents in $Te$ do not participate by any means in the inductive construction of the classifiers. When this condition is not observed, it is likely that the results will be unrealistically good, making the evaluation lose its scientific character.

The train-and-test approach, where the classifier is re-trained on the entire corpus after evaluation has been performed, allows one to see that the previous evaluation would be a pessimistic estimation of the real performance, since the final classifier has been trained on more data than the evaluated classifier [37].

An alternative to the train-and-test approach is the k-fold cross-validation approach. According to Sebastiani [37], in this approach $K$ different classifiers $\Phi_1, ..., \Phi_k$ are built by partitioning the initial corpus $\Omega$ into $k$ disjoint sets $Te_1, ...Te_k$, and then iteratively applying the train-and-test approach on pairs $\langle TV_i = \Omega - Te_i, Te_i \rangle$.

This approach allows one to measure the final effectiveness figure, obtained by individually computing the effectiveness of each classifier, $\Phi_1, ..., \Phi_k$, and then averaging the individual results.

The cross-validation approach is an popular way of detecting the overfitting problem [45], that by definition considers a given hypothesis space $H$. In this hypothesis space, a hypothesis $h \in H$ is said to overfit the training data if there exists some alternative hypothesis $h' \in H$, such that $h$ ends having smaller error than $h'$ over the training examples, but $h'$ has a smaller error than $h$ over the entire distribution of instances [46].

Both approaches are mentioned here to allow one know that it is often the case that the internal parameters of the classifiers must be tuned, and it is done by testing which values of the parameters yield the best effectiveness. This optimization is made possible in the train-and-test approach by splitting the set $\{d_1, ..., d_{|TV|}\}$ into a training set $Tr = \{d_1, ..., d_{|Tr|}\}$, from which the classifier is built, and a validation set $Va = \{d_{|Tr|+1}, ..., d_{|TV|}\}$[3]. The set $Va$ is then used to perform parameter optimizations by repeating tests of the given classifier [37]. The same line of thought allows one to conduct parameter optimizations using the k-fold cross-validation approach.

### 2.2.3 Human Level Performance on Text Classification

Wolf, Poggio and Sinha [28] have found that subjects are able to achieve similar classification accuracy with or without syntactic information across a range of passage sizes. These results have implications for models of human text-understanding and also allows one to estimate what level of performance can be expected, in principle, from a system without requiring a prior step of complex natural language processing.

In Figure Figure 2.3, it is possible to see the results of the experiment where the green line indicates the human text classification ability after a period of exposure, showing that the human accuracy surrounds an accuracy near of 80%.

The red line presents the results regarding the same experiment, but with texts exposed in a bag of words (BOW) format, while the blue line stands for the results concerning both normal and BOW presentation structures. In total, the readers saw 100 texts, having twenty passages for each of the five classes. Ten passages were presented as Bag of Words, and the other ten were presented normally formatted [28].

The authors [28] found that subjects are able to achieve similar classification accuracy with or without syntactic information across a range of passage sizes. These results

[3]Sometimes named as a hold-out set

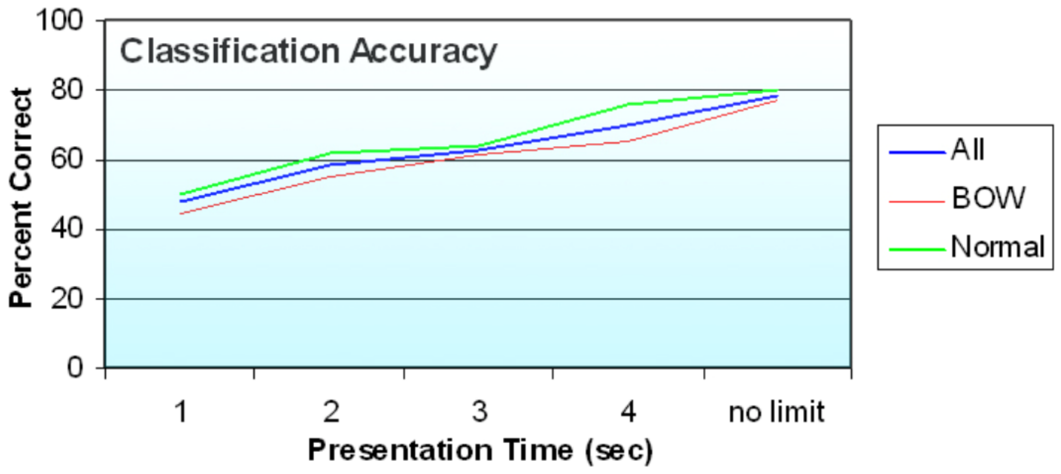


Figure 2.3: Newsgroup text classification accuracy of human readers across different presentation times (Source: [28]).

showed implications for models of human text-understanding, allowing them to estimate what level of performance could be expect from systems without requiring a prior step of complex natural language processing.

## 2.3 Active Learning

According to Ein-Dor *et al.* [47], active learning is considered a ubiquitous paradigm to cope with data scarcity, and depending on the scenario, even going through the burden of labeling a random sample, it could be possible to yield an insufficient number of positive instances to properly train a classifier.

Settles [48] states that active learning (also called "query learning," or sometimes "optimal experimental design" in the statistics literature) is a subfield of machine learning and, more generally, artificial intelligence.

The understanding of the author [48] is that, if the learning algorithm is allowed to choose the data from which it learns — to be "curious," if you will — it will perform better with less training.

Following the same reasoning, Ein-Dor *et al.* [47] declare that in the active learning paradigm, one assumes that unlabeled data are abundant, and the goal is to focus the expensive labeling process on the most informative instances. Many active learning strategies have been proposed, aiming to minimize the labeling burden, or if taken from a different perspective – maximize the value of labeling a small set of examples [47].

Active learning (also called "query learning," or sometimes "optimal experimental design" in the statistics literature) is a subfield of machine learning and, more generally, artificial intelligence.

The usefulness of an active learning strategy naturally depends on the classification scheme with which it is coupled. A successful AL approach for a Naive Bayes classifier may not be that effective for a modern deep-learning algorithm such as CNN, and vice versa [47].

The main idea in this process is to attempt to overcome the labeling bottleneck by asking queries in the form of unlabeled instances to be labeled by an oracle, as a human annotator. In this way, the active learner aims to achieve high accuracy using as few labeled instances as possible, thereby minimizing the cost of obtaining labeled data [48], as specified in Figure 2.4.

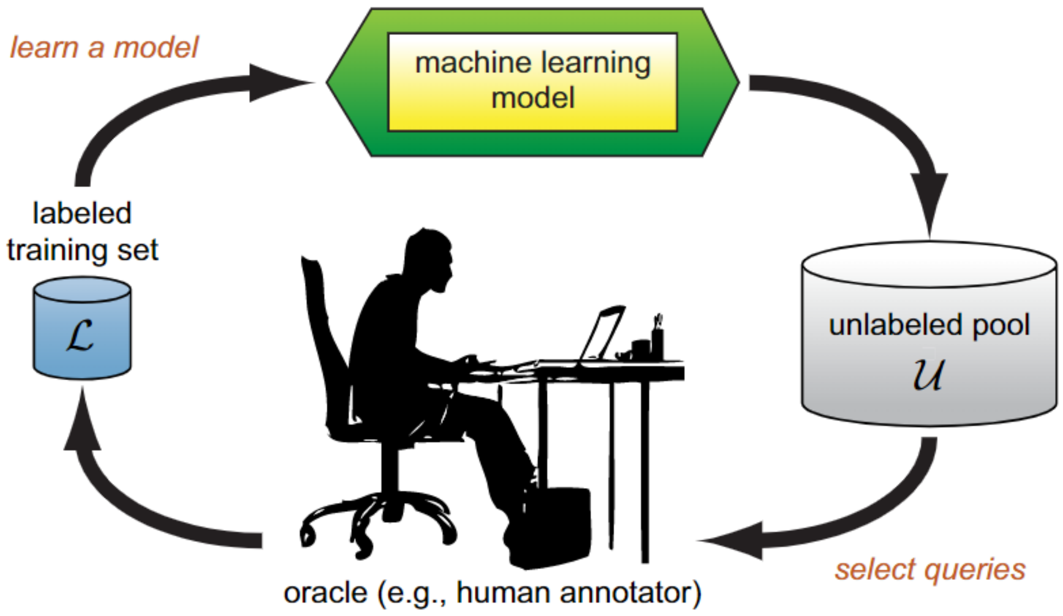


Figure 2.4: The pool-based active learning cycle. (Source: [48]).

## 2.4 Self-Supervised Learning

Before diving into other concepts, it is worth to describe self-supervised learning, in view of the fact that, although being a completely different topic, according to Liu *et al.* [49], the further presented model in Section 2.8 — BERT — is considered a generative, denoising auto-encoding model, pertaining to the NLP field of study since the masked language model (MLM), one of the most successful architectures in natural language processing, can be regarded as a denoising auto-encoding model.

Self-supervised learning can be viewed as a branch of unsupervised learning since there is no manual label involved, having labels derived from co-occurring inputs to related information. Notwithstanding the above information, unsupervised learning concentrates on detecting specific data patterns, such as clustering, while self-supervised learning targets at recovering, which still follows the paradigm of supervised settings [49]. Figure 2.5 presents useful examples demonstrating the differences between them.

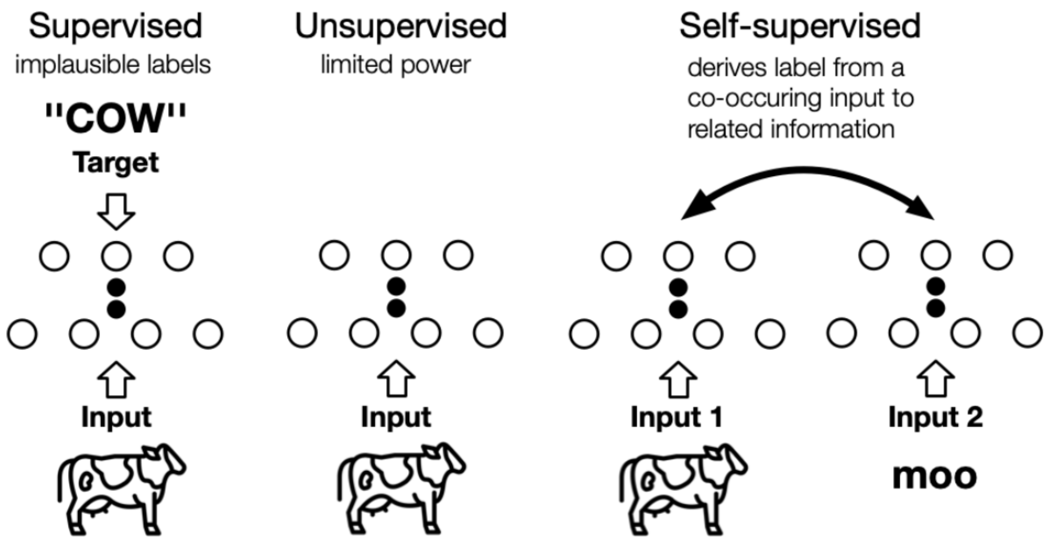


Figure 2.5: Example of differences between Supervised, Unsupervised and Self-Supervised Learning (Source: [49]).

As stated by Liu *et al.* [49], the most crucial point for self-supervised learning's success is that it figures out a way to leverage the tremendous amounts of unlabeled data that becomes available in the big data era. The mainstream self-supervision can be summarized in three general categories and detailed subsidiaries:

1. Generative: A encoder is trained to encode input $\rho$ into and explicit vector $\varrho$, and a decoder is set to reconstruct $\rho$ from $\varrho$.

2. Contrastive: A encoder is trained to encode input $\rho$ into and explicit vector $\varrho$, in order to measure similarity.

3. Generative-Contrastive (Adversarial): A encoder-decoder is trained to generate fake samples, while a discriminator is trained to distinguish the fake samples from the real ones.

The generative self-supervised learning's is successful in the self-supervised learning area because it embodies the ability to recover the original data distribution without being limited by assumptions for the downstream tasks [49].

This fact make possible the use of generative models in a wide range of applications in both classification and generation tasks, and still maintain researchers dependent on generative language models to conduct text classification tasks in the NLP domain [49].

The aforementioned auto-encoding model is based on generative models, being probably the most popular generative model, and is typically for dimension reduction. In a general way, it stands as a feed-forward neural network trained to produce its input as the output layer. The basic model is comprised of an encoder network $h_n = f_{enc}(x)$ and a decoder network $x' = f_{dec}(h_n)$, having as objective to make $x$ and $x'$ as similar as possible [49].

Amidst a set of auto-encoding model variants, the denoising auto-encoder models stands out to us, considering that this model has BERT as the most representative work in the field. The intuition here is that the representation shall be resilient to the introduction of noise. Masked language models randomly masks some of the tokens from the input and then predicts them based on their context information [49].

The problem of how to learn inductive biases is partially solved by solutions that apply self-supervised representation, once vast amounts of corpora sharing the same embedding space — language — are employed in the language model pre-training, helping to cover most language patterns and, consequently, contributing to the success of Pre-trained Language Models (PTM) in several language tasks [49].

Yet according to the same authors [49], labels are still important because there is a gap between training objects of self-supervised learning and supervised learning. Concisely explaining, no matter how self-supervised learning models improve, they are still only a powerful feature extractor, and to transfer to the downstream task, one still need some labels. Consequently, semi-supervised learning emerges as bridge to the gap betwixt self-supervised pre-training and downstream tasks, going towards the goals that are being aimed at in this research.

## 2.5 Self-Learning

Semi-supervised learning is an area in machine learning where there has been a whole spectrum of interesting ideas on how to learn from both labeled and unlabeled data instances so prediction performance could be improved [21, 22]. The semi-supervised methods are set through using unlabeled data to either modify or re-prioritize hypotheses obtained from a small amount of labeled data alone [21, 49].

As presented by Zhu [21], although not all methods are probabilistic, it is easier to look at methods that represent hypotheses by $p(y|x)$, and unlabeled data by $p(x)$. Generative models have common parameters for the joint distribution $p(x, y)$. This way, it becomes easy to see that $p(x)$ influences $p(y|x)$. It is noteworthy to know that, in semi-supervised learning, the model correctness has to be considered, since that unlabeled data may hurt accuracy if the model is wrong.

Considering the semi-supervised learning area, it is emphasized by Li and Ye [22] that self-learning is the most straightforward scheme, being the earliest semi-supervised learning method developed [49]. Self-learning, also known as self-training or self-teaching, is a wrapper algorithm that has been commonly used for semi-supervised learning [21,50]. For the purpose of this work, the terms self-training, self-labeling, and self-teaching will be considered synonyms for self-learning.

Moving forward to self-learning concepts, it is stated that a classifier is first trained on a small amount of labeled data, and then used to classify the unlabeled data. Later, the most confident unlabeled samples are selected and put into the training set using their predicted labels. The classifier is therefore retrained multiple times [50, 51].

As the central concept in self-learning is to use the labeled set $\mathcal{L}$ to build a classifier, it is evident, based on the above description, that the ensuing model will iteratively be applied to the unlabeled corpus $\mathcal{U}$ in each iteration, then will expand the training set with instances from the unlabeled corpus which were predicted with high confidence, according to a user-defined threshold $\delta$, by the classifier [9, 52].

**Algorithm 1** Self-Learning Pseudocode [52]

```
Input: $\mathcal{L}$: labeled set; $\mathcal{U}$: unlabeled set; $\delta$: confidence threshold
Result: $\mathcal{T}$: labeled set
 1: $\mathcal{T} \leftarrow \mathcal{L}$
 2: while (stopping criterion) do
 3:     $\Phi \leftarrow$ train classifier on $\mathcal{T}$;
 4:     for $i = 1$ to $|\mathcal{U}|$ do
 5:         if confidence of $\Phi$.classify($\mathcal{U}_i$) $\geq \delta$ then
 6:             $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{U}_i$, where $\mathcal{U}_i$ is the i-th instance in $\mathcal{U}$
 7:             Mark $\mathcal{U}_i$ as labeled;
 8:         end if
 9:     end for
10:     Update $\mathcal{U}$ by removing labeled instances;
11: end while
12: return $\mathcal{T}$;
```

The pseudocode shown in Algorithm 1 allows one to discern that the initial training set $\mathcal{T}$ is the labeled set $\mathcal{L}$ (line 1). With each iteration, the ensuing confident predictions obtained from $\mathcal{U}$ are then included in the training set, resulting in an expansion of this set (lines 5-8). A new classifier is built based on the new expanded training set (line 3). The process will continue unless the stopping criterion is met *e.g.*, $\mathcal{U}$ is empty, a pre-defined number of iterations are reached, or no further expansions are possible due to the given threshold $\delta$ [52].

## 2.6 Bidirectional Language Models

This section presents the definition of bidirectional language models, as a means to allow one to get the best comprehension possible about BERT, which is introduced in a dedicated section further on.

In many downstream applications one mostly cares about having access to contextual representations of words, *i.e.*, word representations that are a function of the entire context of a unit of text such as a sentence or paragraph, and not only conditioned on previous words. Formally, given the input sequence $t = (t_1, t_2, ..., t_N)$ and a position $1 \leq k \leq N$, it is sought to estimate

$$p(t_k) = p(t_k | t_1, ..., t_{k-1}, tk + 1, ..., t_N) \tag{2.6}$$

using the left and right context of that word [6].

The unidirectional language models computes the probabilities of a token sequence $(t_1, t_2, ..., t_N)$ as follows [53]

$$p(t_1, t_2, ..., t_N) = \prod_{k=1}^{N} p(t_k | t_1, t_2, ..., t_{k-1}) \tag{2.7}$$

A common way to estimate these probabilities is taking the advantage of neural language models (LM) applying *softmax* function

$$p(t_k | t_1, t_2, ..., t_{k-1}) = softmax(Wh_t + b) \tag{2.8}$$

where $h_t \in \mathbb{R}^k$ is the output vector of a neural network at position $t$ and $W \in \mathbb{R}^{|\mathcal{V}| \times k}$ is a learned parameter matrix that maps $h_t$ to unnormalized scores for every word in the vocabulary $\mathcal{V}$ [6].

In this line of reasoning, a number of neural language models then mainly differ in how they compute $h_t$ given the word history, *e.g.*, with multi-layer perceptrons, convolutional layers, recurrent neural networks or self-attention mechanisms [6].

As a way to make one clearly fathom the concepts of bidirectional LM, it is worth saying that some state-of-the-art neural language models, based on the Equation 2.7 pass token representations through several layers of Long Short-Term Memory (LSTM) to embed the history $(t_1, t_2, ..., t_k)$ into a fixed dimensional vector $\overrightarrow{h}_k^{LM}$, that is defined as the forward language model embedding of the token at position $k$, being the output of the top LSTM layer in the language model. Therefore, the language model predicts the probability of token $t_{k+1}$ using softmax layers over words in the vocabulary [53].

As a consequence, the idea that the capture of future context in the LM embeddings is beneficial, makes one consider a backward LM in addition to the classic forward LM. The backward LM predicts the previous token given the future context. So, given a sequence of $N$ tokens, it computes [53]

$$p(t_1, t_2, ..., t_N) = \prod_{k=1}^{N} p(t_k | t_{k+1}, t_{k+2}, ..., t_N) \tag{2.9}$$

After implementing the backward LM in an similar way to a forward LM, the backward LM embedding $\overleftarrow{h}_k^{LM}$ is generated for the sequence $(t_k, t_{k+1}, ..., t_N)$, being the output embeddings of the top layer LSTM [53]. The consequent result can be the concatenation of the forward and backward LM embeddings, forming the bidirectional LM embeddings, *i.e.*, $h_k^{LM} = [\overrightarrow{h}_k^{LM}; \overleftarrow{h}_k^{LM}]$, as defined in the Equation 2.6 [53].

Once understood the concepts of a bidirectional language model, it is essential to point a significant difference that occurs in BERT. Instead of a standard language model objective, BERT [29] propose to sample positions in the input sequence randomly and to learn to fill the word at the masked position, by employing transformers architecture. Additionally to the pseudo language model objective, it uses an auxiliary binary classification objective to predict whether a particular sentence follows a given sequence of words [6].

## 2.7 Transformers

As stated in [54], attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks. Since the late 2010s, these mechanisms has become popular, sometimes replacing Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), including Long Short-Term Memory (LSTM) architectures [55].

In [56], the concept of self-attention[4] was presented as part of an architecture where memory and attention are leveraged to empower a recurrent network with stronger memorization capability and the ability to discover relations among tokens. In this model, memory and attention are added within a sequence encoder allowing the network to uncover lexical relations between tokens.

[4]Originally known as intra-attention

Vaswani *et al.* [54] precisely describe the aforementioned concept, describing it as an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence, being successfully used in a variety of tasks. For them, an attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output can be computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

Transformer is the first transduction model that relies entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. The model architecture relies on a encoder that maps an input sequence of symbol representations $(x_1, ..., x_n)$ to a sequence of continuous representations $z = (z_1, ..., z_n)$. Given $z$, the decoder then generates an output sequence $(s_1, ..., s_m)$ of symbols one element at a time. At each step the model is auto-regressive, consuming the previously generated symbols as additional input when generating the next [54].

As evinced by Vaswani *et al.* Transformer observes this overall architecture by using stacked self-attention and point-wise, fully connected layers for both the encoder and decoder, as it can be see in the left and right halves of Figure 2.6, respectively.

The stack of $N = 6$ identical layers is what makes the encoder. In that configuration, each layer is composed by two sub-layers. The first one is a multi-head self-attention mechanism, and the second consists of a simple, position-wise fully connected feed-forward network [54].

Vaswani *et al.* employ a residual connection around each of the two sub-layers, followed by a layer normalization function. Scilicet, the output of each sub-layer is $LayerNorm(x+ Sublayer(x))$, where $Sublayer(x)$ is the function implemented by the sub-layer itself. As a result, outputs of dimension $d_{model} = 512$ are produced.

The decoder designed by Vaswani *et al.* is also composed by a stack of $N = 6$, but the authors added a third sub-layer in addition to the two sub-layers in each encoder layer. The third sub-layer performs multi-head attention over the output of the encoder stack. Residual connections are also observed around each of the sub-layers, followed by a layer normalization. The self-attention sub-layer was cleverly modified in order to avoid positions from attending to subsequent positions in the decoder stack. This elaborated masking, associated with output embeddings offset by one position, guarantees that the predictions for position $k$ can depend only on the known outputs at the positions less than $k$.

There are two specific structures created in the Transformer, as seen in Figure 2.7. Scaled Dot-Product Attention, showed in Figure 2.7a, consists of queries and keys of dimension $d_k$, and values of dimension $d_v$. The attention function is computed of a set of

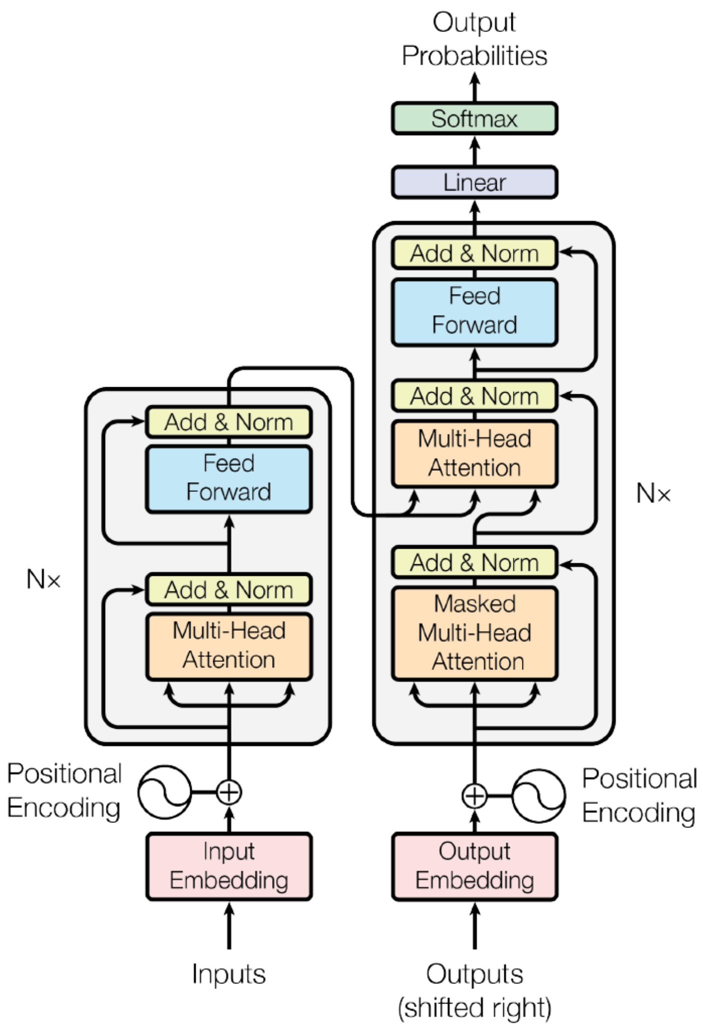


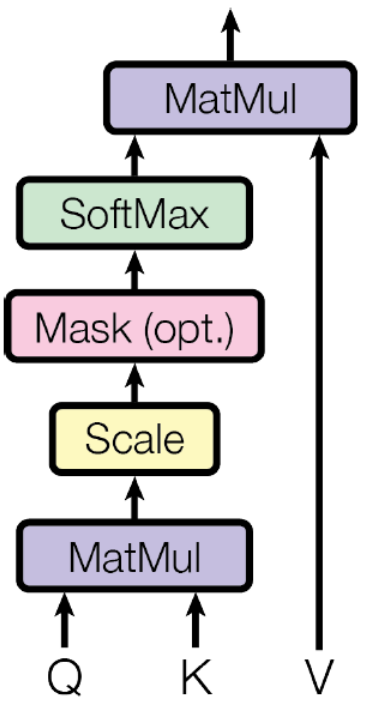
Figure 2.6: Transformer model architecture (Source: [54]).

queries, packed together in a matrix $Q$, as the keys and values, that are packed together into matrices $K$ and $V$. The resulting output is computed as [54]

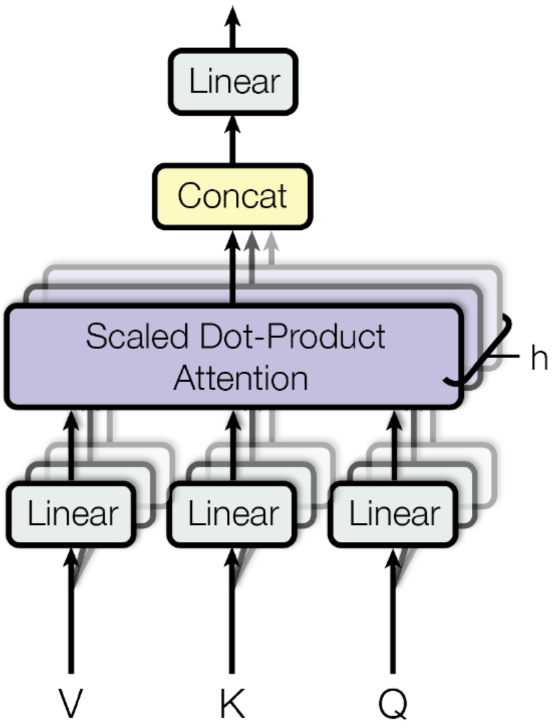
$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.10}$$

Vaswani *et al.* [54] linearly projected the queries, keys and values $h$ times with different, learned linear projections to $d_k$, $d_k$ and $d_v$ dimensions, respectively. In each projected version the attention function run in parallel, yielding $d_v-$ dimensional output values. These values are then concatenated and one more time projected, consequently resulting in the final values, as shown in Figure 2.7b.

The multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions, being needed only a single attention head to averaging inhibit this [54]. The multi-head attention is computed as



(a) Scale Dot-Product Attention



(b) Multi-Head Attention

Figure 2.7: Transformer Structures (Source: [54])

$$\begin{gathered} MultiHead(Q, K, V) = Concat(head_1, ..., head_h)W^O \\ \text{where } head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \end{gathered} \tag{2.11}$$

where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v}$.

## 2.8 BERT

BERT is a pre-trained high-capacity language model optimized to either predict the next word in a sequence or some masked word anywhere in a given sequence, (e.g. "Dante was born in [Mask] in the year 1265.") [6]. It is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning from both left and right context in all layers [29].

The beauty of this language representation model lies in the fact that the pre-trained model can be fine-tuned with just one additional output layer, enabling state-of-the-art model creation for a wide range of tasks [29]. One reason to develop such model, according to Devlin *et al.* [29], is related to the unidirectionality of other models, which incorporates sub-optimal restrictions for sentence-level tasks, and possible harmful effects in token-level tasks, where context incorporation from both directions is crucial.

The model has become so popular that, according to Raghavan [57], BERT has been used in almost every English query in Google Search, helping to get higher quality results for questions made by its users.

One of the language representation model objectives is to alleviate the unidirectionality constraint by using a masked language model (MLM) pre-training objective. This MLM randomly masks some of the tokens from the input, having as intention to predict the original vocabulary id of the masked word, based solely on its context [29].

One advantage of this approach is that the MLM enables the representation to fuse the left and the right context, allowing the deep bidirectional Transformer to be trained. Additionally, a next sentence prediction (NSP) task is also used, jointly pre-training text-pair representations as shown in the left half of Figure 2.8. Another subtle characteristic that exists and is important in large pre-trained models is the transfer learning that occurs, having in the fine-tune an effective recipe to make it happen appropriately [29].

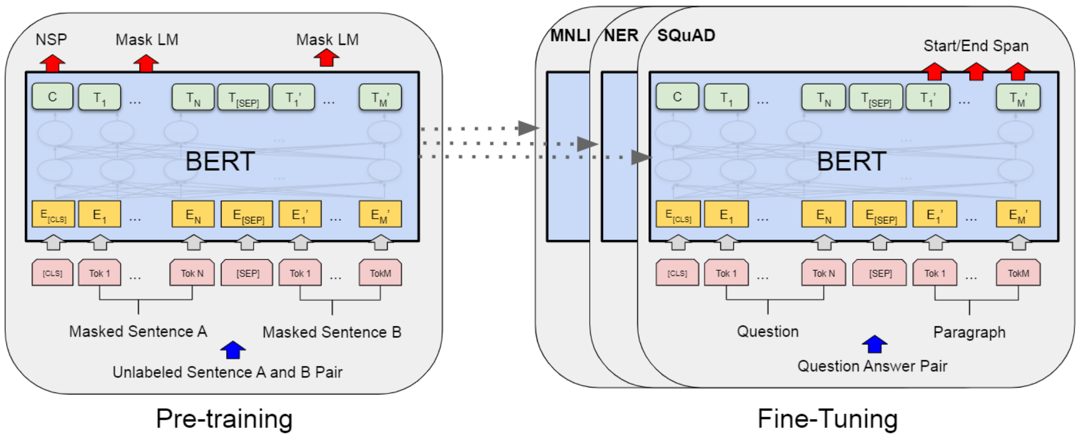


Figure 2.8: Overall pre-training and fine-tuning procedures for BERT (Source: [29]).

As one could infer from this section prior presented text, concerning BERT implementation details, there are two major steps in the framework: pre-training and fine-tuning. In the pre-training step, the model is fed with unlabeled data, then trained over different pre-training tasks. Hence, for the fine-tuning step, the BERT model is initialized with the pre-trained parameters, and series of adjustments are made in the parameters using labeled data from downstream tasks. This model results in an idiosyncratic feature, having a unified architecture across different tasks [29].

Regarding the model architecture, it is of mounting importance to know that BERT is a multi-layer bidirectional Transformer encoder based on the work of Vaswani [54] [29], distinguished as being a stack of transformer's encoder [58], as presented in the left half of Figure 2.6.

For this section, let us denote the number of layers, *i.e.*, the transformer blocks, as $L$, the hidden size as $H$, and the number of self-attention heads as $A$. The $\text{BERT}_{\text{BASE}}$ size was inspired on OpenAI GPT, being the base model for the model trained in Portuguese, and used in this work. The original model size was defined as having $L = 12$, $H = 768$, $A = 12$, feed-forward/filter size of $4H$, that is 3072, and 110M parameters.

## 2.8.1 Input-Output Representations

Focusing on making BERT be able to handle a variety of downstream tasks, Devlin *et al.* [29] developed an input representation that is able to unambiguously represent both single sentences and pairs of sentences in one token sequence.

The *sentence* term refers not to an actual linguistic sentence, but to an arbitrary span of contiguous text. As for the *sequence* word, it relates to the input token sequence to BERT, which mayhap be a single sentence or a pair of sentences packed together [29].

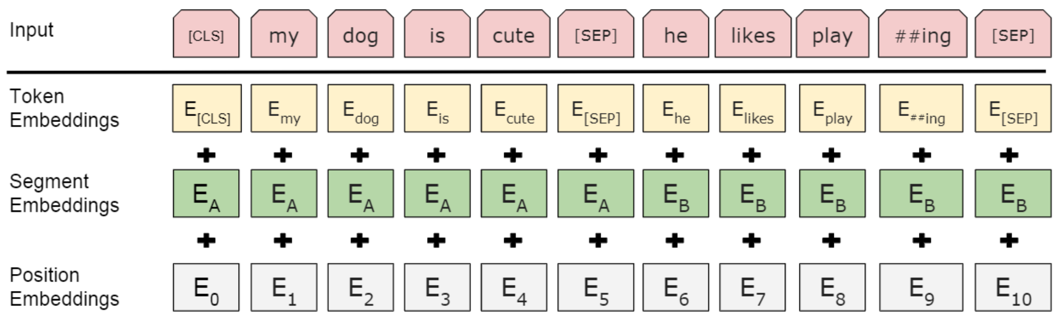


Figure 2.9: BERT input representations (Source: [29]).

In the input representation, the first token of a sequence is always the special token [CLS]. Sentence pairs in a sequence are separated by another special token [SEP]. Yet, a learned embedding indicating whether it belongs to sentence A or B is added. The input embedding is denoted as $E$, the final hidden vector of the special [CLS] token as $C \in \mathbb{R}^H$, and the final hidden vector for the $i^{th}$ input token as $T_i \in \mathbb{R}^H$. The final input representation for a given token is the sum of its corresponding token, segment, and position embeddings, as shown in Figure 2.9 [29].

A reasonable understanding about how BERT is structured, including a complete representation of the self-attention mechanism, and how the input-output representation is set on this mechanism, showing also how $L$, $H$ and $A$ are organized, can be seen in Figure 2.10.

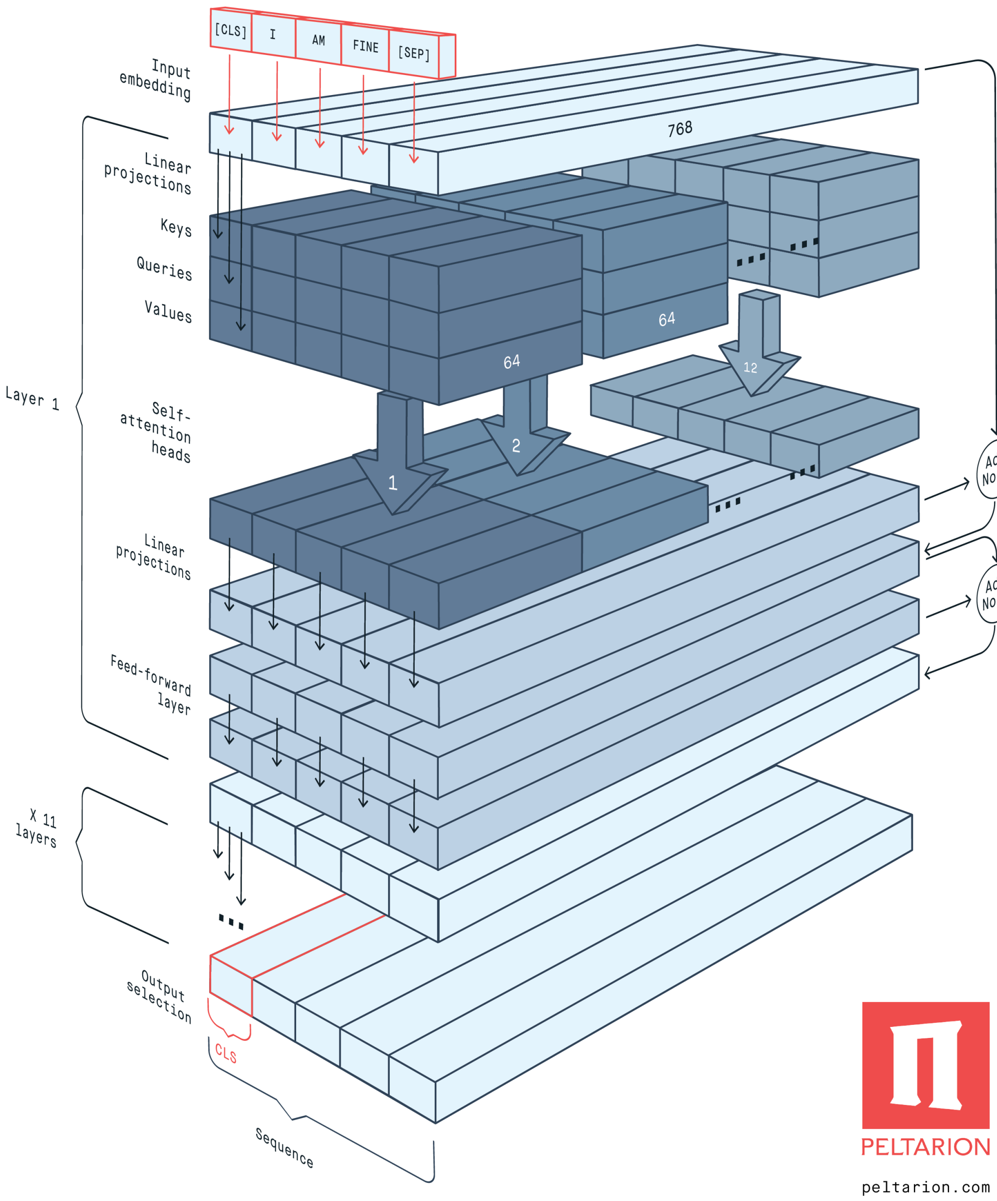


Figure 2.10: BERT model 3D representation (Source: [59]).

### BERT Encoding and Tokenization

In order to allow for a glimpse into how implementation occurs, let us firstly assume the model has 10 as the maximum length, rather that the 512 limit set in the original model. One can summarize a classification task, before submitting the data to the BERT model, as the following steps [60]:

1. Tokenization;

2. Adding the [CLS] token at the beginning of the sentence;
3. Adding the [SEP] token at the end of the sentence;
4. Padding the sentence with [PAD] tokens so that the total length equals to the maximum length; and
5. Converting each token into their corresponding IDs in the model.

As a result, one would find what is presented in Figure 2.11, where the token ids are encoded a ready to be fed into the algorithm.

```
# Original Sentence
Let's learn deep learning!

# Tokenized Sentence
['Let', "'", 's', 'learn', 'deep', 'learning', '!']

# Adding [CLS] and [SEP] Tokens
['[CLS]', 'Let', "'", 's', 'learn', 'deep', 'learning', '!', '[SEP]']

# Padding
['[CLS]', 'Let', "'", 's', 'learn', 'deep', 'learning', '!', '[SEP]', '[PAD]']

# Converting to IDs
[101, 2421, 112, 188, 3858, 1996, 3776, 106, 102, 0]
```

Figure 2.11: Simple BERT tokenization example having max length as 10. (Source: [60]).

When using the transformers package, the process needs to be developed in a different way, resulting in a tensor storing the attention mask and another tensor storing the encoded input, as shown in Figure 2.12

```
>>> input-ids
tensor([[ 101, 2421,  112,  188, 3858, 1996, 3776,  106,  102,    0,    0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
            0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
            0,    0,    0,    0]])
>>>
>>> attn_mask
tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])
```

Figure 2.12: BERT Tokenization result when using Transformers package, having max length as 64 (Source: [60]).

In this case, whilst feeding the input into the BERT model, the attention mask output in Figure 2.12 let the model knows which tokens shall be attended to and which shall not, as the zeros represent the [PAD] token. [60].

### 2.8.2 BERTimbau

BERTimbau [61] is a language representation model trained for Brazilian Portuguese. As stated by the authors, the transfer learning approach is remarkably beneficial when label data is scarce. This setting makes pretrained language models worthwhile assets where few annotated training examples are available.

The work that is the subject of this section developed three NLP experiments, comparing the produced results to the BERT, while closely replicating BERT's [29] architecture and pretraining procedures. While developing their work, Souza *et al.* [61] emphasize the advantage that of relying only on self-attention mechanisms instead of recurrence, what allows a complete view, enabling the model to see all input tokens at once, making it capable of modeling long sequences in constant time while promoting much higher parallelization.

BERTimbau [61] was trained on two sizes, being the Base model the one used in this work. The BERTimbau Base have 12 layers, 768 hidden size, 12 attention heads, and 110M parameters. The maximum sentence length also observed BERT [29], following the $S = 512$ tokens limit.

The pretraining data was extracted from brWaC corpus, which contains 2.68 billion tokens from 3.53 million documents, being considered the largest open Portuguese corpus [61]. Also, the pretraining objectives were identical the objectives set to BERT.

As a result, in [61] the performance of BERTimbau Base on NER task, when applying intermediate checkpoints, showed that the downstream task increase non-linearly with pretraining steps, having the models outperformed previously published results and multilingual model equivalents, pairing with the results of similar works on other languages.

Souza *et al.* finish their paper by registering that they have released their model expecting that others will be able to benchmark the performance of BERTimbau in other NLP tasks, this being the case in this research.

## 2.9 Related Work

This research's related work derives from different areas, here named: text classification, limited labeled data, semi-supervised learning, and self-learning. They are disposed in chronological order with the purpose of letting the reader perceive how different lines of thought, tangential to the same problem here studied, evolved over time.

As specified by Nigam *et al.* [23], the problem of learning accurate text classifiers from limited numbers of labeled examples, by using unlabeled documents to augment the available labeled documents, was addressed by showing that the accuracy of learned text classifiers can be improved by augmenting a small number of labeled training documents with a large pool of unlabeled documents. The authors introduced an algorithm for learning from labeled and unlabeled documents based on the combination of Expectation-Maximization (EM) and a Naïve Bayes classifier.

They benefited from the fact that EM is a class of iterative algorithms for maximum likelihood or maximum a posteriori estimation in problems with incomplete data [62 as cited in 23], being consistent with the scenario presented by the authors once unlabeled data are considered incomplete because they come without class labels.

Considering that unlabeled data do provide information about the joint probability distribution over words, the developed method consists of submitting a set of labeled documents to perform the initial training of the classifier, which, in turn, labels unlabeled documents based on the probabilistic results generated by the model, with specific weights assigned to them. A new classifier is then estimated using all the documents – both the originally and newly labeled. Therefore, the proceeding iterates until the classifier parameters stop improving.

Their results showed EM to perform significantly better, mainly when there was little labeled data, though unlabeled data could throw off parameter estimation when one considered that the number of unlabeled documents was much greater than the number of labeled documents. The authors modulated the influence of the unlabeled data, in order to control the extent to which EM performs unsupervised clustering, by introducing a $\lambda$ parameter into the likelihood equation, which decreased the contribution of unlabeled documents to parameter estimation.

Thus, while using the aforementioned approach to prevent the unlabeled data from degrading classification accuracy, the authors were able to show significant improvements by using unlabeled documents for training classifiers in three real-world text classification tasks, as well as showed improvements in classification, by using multiple mixture documents per class, in order to relax the assumptions of the model; though rested the belief that, when the target concept and model differ from actual distribution of the data too strongly, the algorithm using unlabeled data required a closer match between the data and the generative model than those using labeled data alone, because the use of unlabeled data would hurt instead of help performance.

Fragos, Belsis and Skourlas [63] focused on assessing the performance of two or more classifiers used in combination in the same classification task, classifying documents using two probabilistic approaches – Naïve Bayes and Maximum Entropy classification model –

then combining the results of the two classifiers to improve the classification performance, using two merging operators, Max and Harmonic Mean.

The authors applied the $X$ square test on the corpus and selected 2,000 higher ranked words for each category to be used in the maximum entropy model and evaluated the classification performance of the classifiers using micro-Recall ($\mu Re$), micro-Precision ($\mu Pr$) and micro-averaged F1 measure (*micro-F1*).

The Maximum Entropy model, presented by Fragos, Belsis and Skourlas [63], showed better performance than the Naïve Bayes classifier. Additionally, the two merging operators were used to combine results of the Naïve Bayes and SVM classifiers to improve performance, especially for the Recall rate. As results, it could be demonstrated that the merging operators do improve the performance, as indicated by the results for Micro-averaged F1 measure, that scored 0.90 and 0.91 for MaxC and HarmonicC operators, respectively.

In the same line of thought of [23], the use of self-learning and co-training is presented as a way to leverage the power of unlabeled data, together with labeled data, in [9]. The resulting work included the *TSentiment15*, an annotated Twitter dataset of 2015 comprising 228 million tweets without retweets and 275 million with retweets.

The authors evaluated the performance of self-learning and co-training and how it was affected by the amount of labeled data, the amount of unlabeled data and the confidence threshold, and, not only this, processed the available data as a batch and as a stream, showing that streaming achieved a comparable accuracy to the batch approach. The findings, in sentiment analysis, revealed that co-training performed better with limited labels, whereas self-training was best choice when significant amount of labeled data was available.

Iosifidis and Ntoutsi [9] used unigrams for self-learning, and unigram-bigrams and unigrams-SpecialF for co-training. In their experiments, although both self-learning and co-training benefited from more labeled data, when labeled data surpassed 40%, self-learning improved faster than co-training. When processing streaming, that revealed to be more efficient, history helped with the performance; notwithstanding the batch approach being better in terms of accuracy.

As avowed in [22], Li and Ye addressed issues related to generative model-based schemes, that does not naturally work on discrete data, knowing that learning a good data representation is sometimes directly opposed to the goal of learning a high-performance prediction model. To achieve their objective, the authors reformulated the semi-supervised learning as a model-based reinforcement learning problem, proposing an adversarial network-based framework. The proposed framework consisted of two networks: a predictor network for target estimation and a judge network for evaluation. Based on this framework, a recurrent neural network-based model for semi-supervised text classification was presented.

The authors bridged the idea of self-training and adversarial networks, to overcome their issues, by designing a Reinforcement Learning based Adversarial Networks for Semi-supervised Learning – RLANS – framework. The RLANS framework had the predictor $\mathcal{P}$ trained by policy gradient, where the reward signal was provided by the judge network $\mathcal{J}$. The judge network, in turn, determined whether a certain input data-label pair had a predicted label or a true label, and was trained using both real data-label pairs $\{(x_i, y_i) \in \mathcal{D}_L\}$, as positive examples, and the predicted data-label pairs $\{(x_i, \hat{y}_i) | x_i \in \mathcal{D}_U\}$ as negative examples [22].

The results of Li and Ye [22] experiments evinced that convergence and performance of RLANS were strongly affected by the pre-training of predictor and judge, but high prediction performance was always observed along with fast convergence and good stability.

Howe, Khang and Chai [64] developed a comparative study on the performance of various machine learning(“ML”) approaches for classifying judgments into legal areas, using a novel dataset of 6,227 Singapore Supreme Court judgments, and investigating how state-of-the-art NLP methods compared against traditional statistical models.

Dealing with small number of lengthy documents, the authors came to the conclusion that BERT models competed at disadvantage, because of the models’ inability to be fine-tuned on longer input texts, having LSA based linSVM outperforming both word-embedding and language models.

While Howe, Khang and Chai [64] have found limitations regarding the text length, citing it as a disadvantage, Sun *et al.* [65] conducted exhaustive experiments to investigate different fine-tuning methods of BERT on text classification tasks and provided a general solution for BERT fine-tuning. This way they were able to reach new state-of-the-art results on eight widely studied text classification datasets.

As elicited by the authors, in [65] three different ways of fine-tuning methods were explored: Fine-Tuning Strategies; Further Pre-Training; and Multi-Task Fine-Tuning. The authors yet experimented two ways of dealing with long articles: Truncation methods; and Hierarchical methods.

In [65], the problem related to the catastrophic forgetting was addressed as well. Considered a common problem in transfer learning, meaning that the pre-trained knowledge is erased while learning new knowledge [66 as cited in 65], it was managed by setting a lower learning rate, such as $2e-5$ on BERT, so it could overcome the catastrophic forgetting problem. Aggressive learning rates, as $4e-4$, led the training set to fail to converge.

The authors were able to bring off experimental findings, reporting that the top layer of BERT showed to be more useful for text classification. The appropriate decreasing learning rate allows BERT to overcome the catastrophic forgetting problem, whithin-task and in-domain further pre-training can significantly boost its performance, and the most important finding considered to this work, BERT can improve the task with small-size data.

Meng *et al.* [20] used the label name of each class to train classification models on unlabeled data, waiving the use of any labeled documents. Towards achieving their goals, they took advantage of pre-trained neural language models both as general linguistic knowledge sources for category understanding and as representation learning models for document classification. Their method associated semantically related words with the label names, found category-indicative words and trained the model to predict their implied categories, and generalized the model via self-training.

In [20], the problem of weakly-supervised text classification was object of study. In this abstraction only the label name of each class is provided to train a classifier on purely unlabeled data. A pre-trained language model was used as both the general knowledge source for category understanding and feature representation learning model for classification. By doing so, the authors showed that label names were an effective supervision type for text classification.

In the same field, in [67] BERT model is compared with a traditional machine learning NLP approach that trains machine learning algorithms in features retrieved by the Term Frequency - Inverse Document Frequency (TF-IDF) algorithm as a representative of traditional approaches.

The purpose of their work was adding empirical evidence to support the use of BERT as a default on NLP tasks. As stated by the authors, experiments showed the superiority of BERT and its independence of features of the NLP problem such as the language of the text, including an experiment with Portuguese news, having Portuguese as one of the chosen languages to see if BERT model would also behave well, adding empirical evidence to use BERT as a default technique in NLP problems [67].

# Chapter 3

# Methodology

*"You can, you should, and if you're brave enough to start, you will."*
— Stephen King, *On writing: A memoir of the Craft*

## 3.1 Background

Having in mind that the main goal of this experiment was to assess the possible performance improvements that originate from the use of self-learning for downstream tasks, specifically text classification, the development of the methodology initially took place with a literature review in order to find the main methodological concepts, with the purpose of basing the development of this work. After the consolidation of these concepts, a particular case study was developed beginning with the gathering of a specific dataset from a OM.

Knowing that the total amount of documents of the Brazilian Army is scattered all over the country, in more than 680 servers, and that, at some point of time, as established by the Brazilian regulations, it shall be evaluated and classified, the Brazilian Army granted access to 45,041 documents, after requested by this author. As seen in [64], BERT model proved competitive on smaller training subsets, which evokes the idea that a BERT based model could be successfully applied on the given dataset.

While dealing with this specific dataset, it is important to have in view that it was necessary to work in a limited computational environment, since the collected data contained sensitive information and thereby it was not possible to have the data hosted in any environment outside of the Brazilian Army's facilities, specifically outside the OM's facilities. Therefore, one virtual machine, having an Intel Xeon 2.2GHz Dual Core CPU, 27.3GB RAM, and a Tesla P100 GPU 16GB VRAM, was available for use throughout the development of this work, which caused some experiments to take a substantial amount of time to complete.

When it comes to the algorithm, initially a minimum set of labeled documents was trained in order to later classify the entire documents pertaining to the six chosen classes. The process of labeling the unlabeled documents was based on the confidence level established as a threshold for the classification process.

Concerning the confidence level, the first experiments with classical BERT brought in f1-score 0.83 as one of the lowest results for specific classes, then a confidence level milestone was set having 0.82 as the basic confidence level for unlabeled samples' classification. Subsequently, the threshold possibilities were expanded, and additional confidence levels were considered as an option, starting from 0.6, and reaching 0.95.

This procedure continued iteratively until there were no more documents remaining to be classified at a specific confidence level. Afterwards the results were compared to the results that stem from another classification methods, Active Learning, associated with a Logistic Regression model applied on a TF-IDF vectorized corpus, and BERT itself.

## 3.2 Data

By default, all the data made available for this work is stored in a database, compressed, and protect by encryption. Storing the raw data, there is a data model that encapsulates a unique data schema to represent every document, plus the metadata needed to run the system itself. In this work, the only necessary data is a fragment of the document schema containing the body of the document.

The process of extracting only the necessary data for this research was done by producing a Python script, using Py4J to connect the tool to a Java class, which was capable of doing the decryption and decompression. The script was created in order to access the database, extract the documents, and persist them. Once the relevant data was extracted and decrypted, it was stored in a new database and submitted to preprocessing. At this stage, corrupted, drafts and non-processed documents were removed from the dataset, and all the corpus was converted to lowercase, along with manual preprocessing following the steps below:

1. Portuguese stop words were removed from the corpus using Natural Language Toolkit (NLTK) [68] library.
2. Punctuation was removed.
3. Numbers pertaining to itemization were removed.
4. Object pronouns attached to Portuguese verbs were removed.

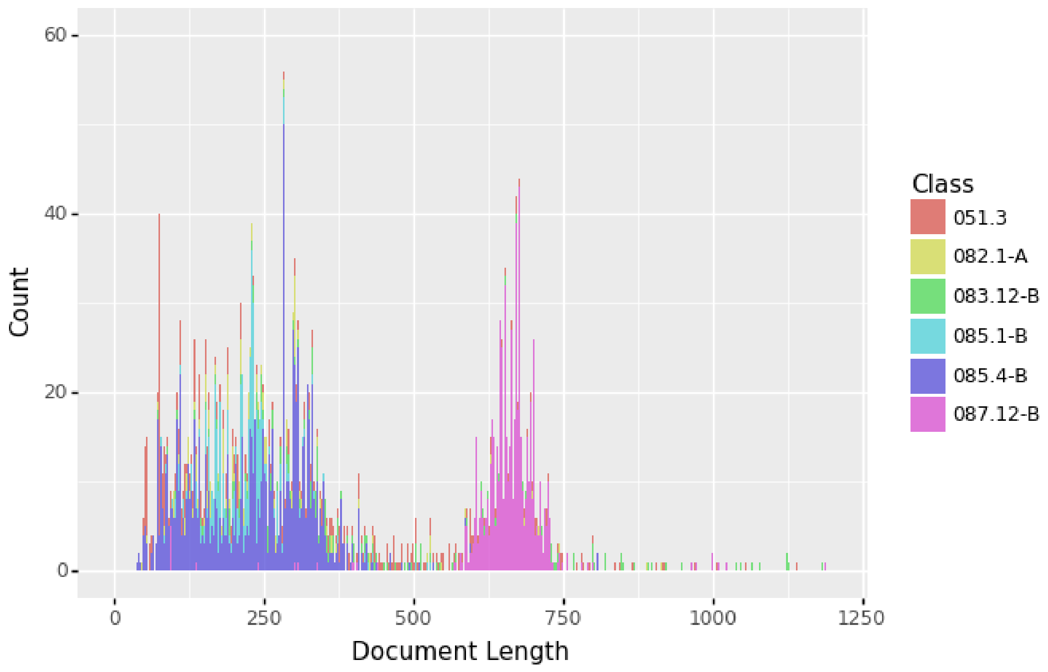


Figure 3.1: Document length distribution.

5. Lemmatization was conducted on the corpus using spaCy [69].

The corpus was then submitted to Ktrain, a lightweight wrapper for the deep learning library TensorFlow Keras (and other libraries) to help build, train, and deploy neural networks and other machine learning models [70].

The Ktrain preprocessing methods' output, which were responsible for the preprocessing steps required to transform raw data into the format expected by the model (i.e., a preprocessor instance that preprocess the validation or test set for a Transformer model), resulted in a dataset containing 5,940 documents, and 5,799 terms, with an unbalanced class distribution, dispersed over six distinct classes, as seen in Table 3.1. It is worth to mention that total number of available samples was a result of the manual labeling process carried out by this author.

Table 3.1: Class distribution.

| **Class ID** | **Class Description** | **Class Size** | **%** |
|---|---|---|---|
| 085.4-B | Vacations or Medical Leave | 2201 | 37.05 |
| 087.12-B | Verification of Medical Conditions | 1352 | 22.76 |
| 051.3 | Budget execution | 926 | 15.59 |
| 085.1-B | Honorable Mention/Service Leave | 810 | 13.64 |
| 083.12-B | Relocation | 372 | 6.26 |
| 082.1-A | Promotion | 279 | 4.70 |

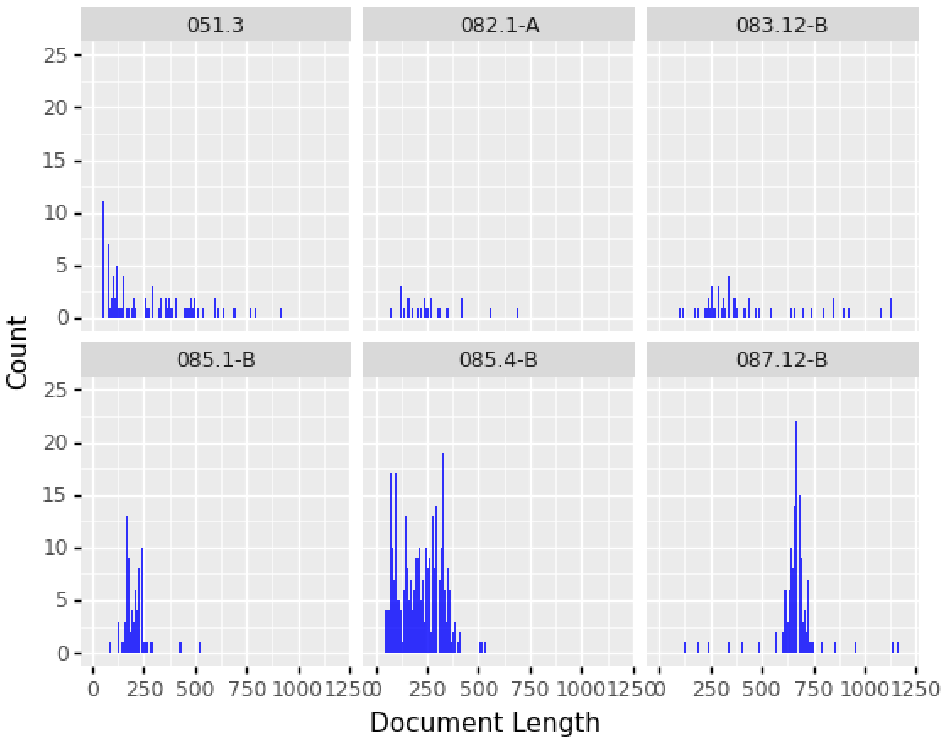


Figure 3.2: Grouped counting document length distribution per class.

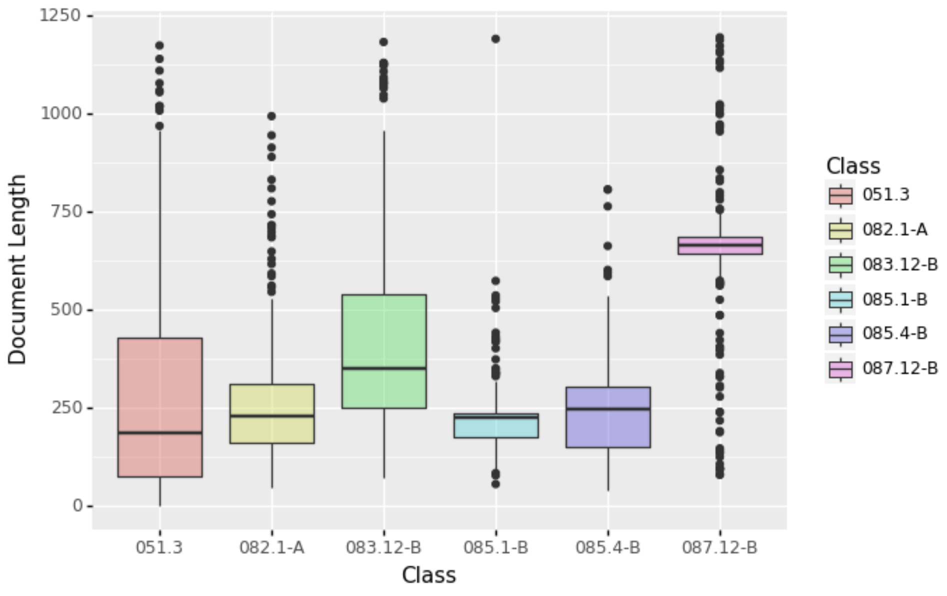
As it can be observed from Figure 3.1, a considerable number of documents has less than five hundred characters, furthermore the distribution of document lengths ranges from 39 to 2582 characters with a mean of 352.89 characters.

After data transformation, the resulting class distribution is shown in Figures 3.2 and 3.3, allowing one to see the final length distribution per class. As the class 087.12-B presents the most centered document length, with a low range, the classes 051.3, that showed to be skewed right, and 083.12-B present a considerable larger range.

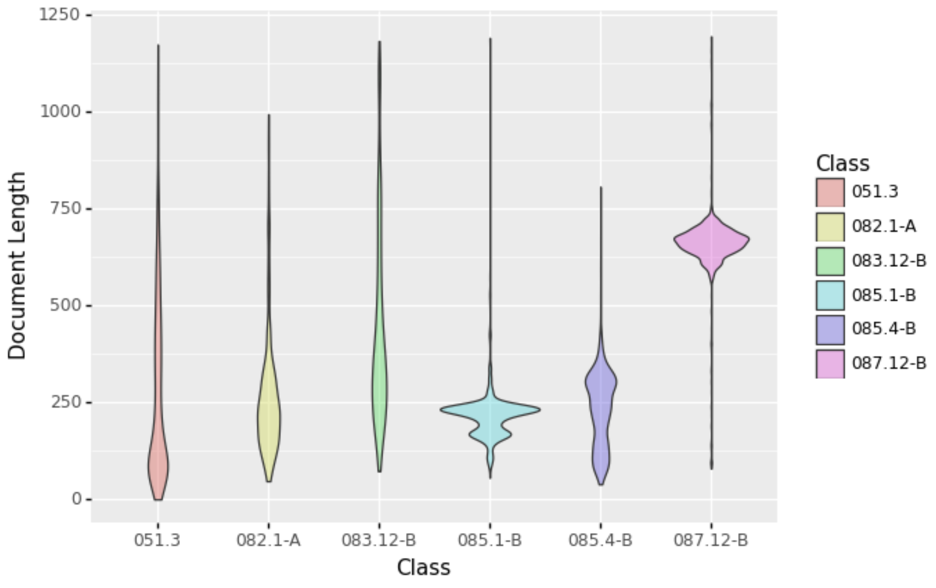
In order to successfully establish a number of labeled documents, a labeling tool, termed Document Classifier[1], presented in Figure 3.4, was developed so the process of searching and labeling documents according to e-ARQ's temporality table was conducted without further efforts regarding finding similar documents in the dataset.

The previously mentioned tool was basically designed to allow searching for specific slices of text in the corpus and then enable the manual labeling of the same through the selection of documents related to the same subject, and consequently label the documents by applying an existing code in the temporality table to the documents.

[1]Also called Classificador de Documentos in Portuguese.



(a) Boxplot distribution per class



(b) Violin distribution per class

Figure 3.3: Boxplot and Violin document length distribution per class

### 3.2.1 BERTimbau

Throughout this work, finding an algorithm that could achieve state-of-the-art performance in Portuguese was a concern due to the existing linguistic bias in favor of languages that predominate on the areas where only major companies and research centers can afford training language models with billions of parameters on massive datasets [71]. The work of Souza *et al.* [61], BERTimbau, arouse as the current answer to this need, delivering state-of-the-art results for downstream natural language processing tasks in Portuguese language, and will be simply referred to as BERT throughout this study. The pre-trained BERT [29] based model, BERTimbau, was the model of choice, applied when running the fine-tuning step using the developed self-learning approach.

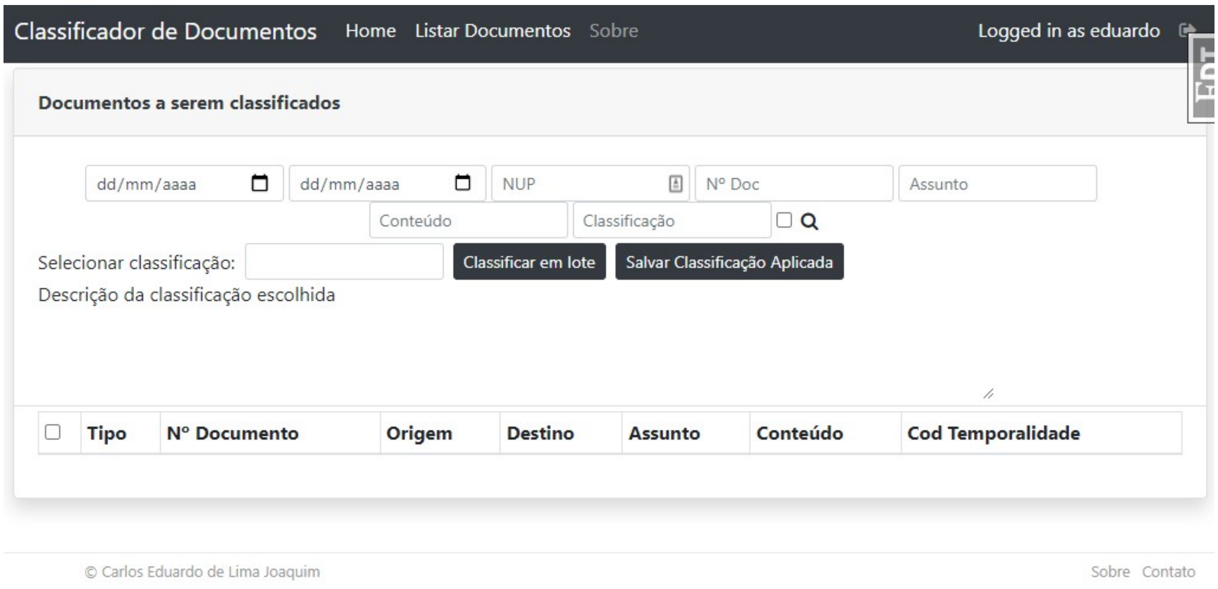


Figure 3.4: Document Classifier developed in order to make it possible to label the documents present in the dataset.

It was yet an opportunity to evaluate the model in a new downstream task, once the authors' work was done assessing results on three different downstream NLP tasks: sentence textual similarity, recognizing textual entailment, and named entity recognition.

## 3.3 Self-Learning

Bearing in mind that this research can find in self-learning a solid answer to the problem of limited number of labeled documents in a dataset, the process of document classification through self-learning starts with data preprocessing and selection. As it is intended to assess the resulting performance of the algorithm, a specific percentage of samples is selected from the total amount of labeled documents, starting with 3% and increasingly growing up this number to 30%.

The Self-Learning BERT (BERT-SL) models were fed with the test set, validated, and its results were registered, in order to be compared to classical BERT and Active Learning, following the sample distribution from this very Section 3.3.

Observing the data organization described in Tables 3.2 and 3.3, in each experiment, the selected data was submitted to the self-learning Algorithm 2, until there were no more unlabeled samples in the corresponding set. This algorithm, in line 1, creates the labeled set by receiving both, the training set and the validation set.

Hence, a loop is set in line 2 and keep iterating until the stopping criterion is reached, in this case the emptiness of the unlabeled set. After the first training procedure, where the training subset was submitted to the aforementioned algorithm, the full subset of unlabeled documents was then submitted to the recently trained model as described below.

Inside the loop, a new classifier is created, in line 3; trained, in line 4; then the entire unlabeled set is submitted to the classifier trained in line 4 and, if the threshold is met, line 6, the given sample is marked as labeled, in line 8, and placed into the training set, as showed in line 7.

Afterwards the unlabeled set is updated, and all new labeled instances are removed from it, as described in line 11. Subsequently, the final classifier is trained using the final training set, and both the labeled training set and the final classifier are returned.

**Algorithm 2** BERT Self-Learning Approach Pseudocode

**Input:** $\mathcal{L}_{tr}$: labeled training set; $\mathcal{L}_v$: labeled validation set; $\mathcal{L}_{ts}$: labeled test set; $\mathcal{U}$: unlabeled set; $\delta$: confidence threshold

**Result:** $\mathcal{T}$: labeled set; $\Phi_f$: final classifier

```
1:  $\mathcal{T} \leftarrow \mathcal{L}_{tr}, \mathcal{L}_v$
2:  while ($\mathcal{U}$ is not empty) do
3:      $\Phi \leftarrow$ new classifier
4:      $\Phi \leftarrow$ train new classifier on $\mathcal{T}$;
5:      for $i = 1$ to $|\mathcal{U}|$ do
6:          if confidence of $\Phi$.classify($\mathcal{U}_i$) $\geq \delta$ then
7:              $\mathcal{T} \leftarrow \mathcal{T} \cup \mathcal{U}_i$, where $\mathcal{U}_i$ is the i-th instance in $\mathcal{U}$
8:              Mark $\mathcal{U}_i$ as labeled;
9:          end if
10:     end for
11:     Update $\mathcal{U}$ by removing labeled instances;
12: end while
13: $\Phi_f \leftarrow$ train final classifier on $\mathcal{T}$
14: return $\mathcal{T}, \Phi_f$;
```

The resulting prediction for every sample, whose probabilistic classification was equal to or greater than the confidence level, allowed it to be incorporated in the labeled document selection of the training set for the next self-learning iteration.

The samples whose probabilistic classification did not match or exceed the confidence level remained in the unlabeled dataset, to be submitted to the new model, which would then be created in the next iteration, once this new model would have more samples available in the training step.

This process iteratively repeated its steps until all the documents in the $\mathcal{U}$ dataset were considered labeled – to expound, the whole dataset received a classification equal to or greater than the defined confidence level.

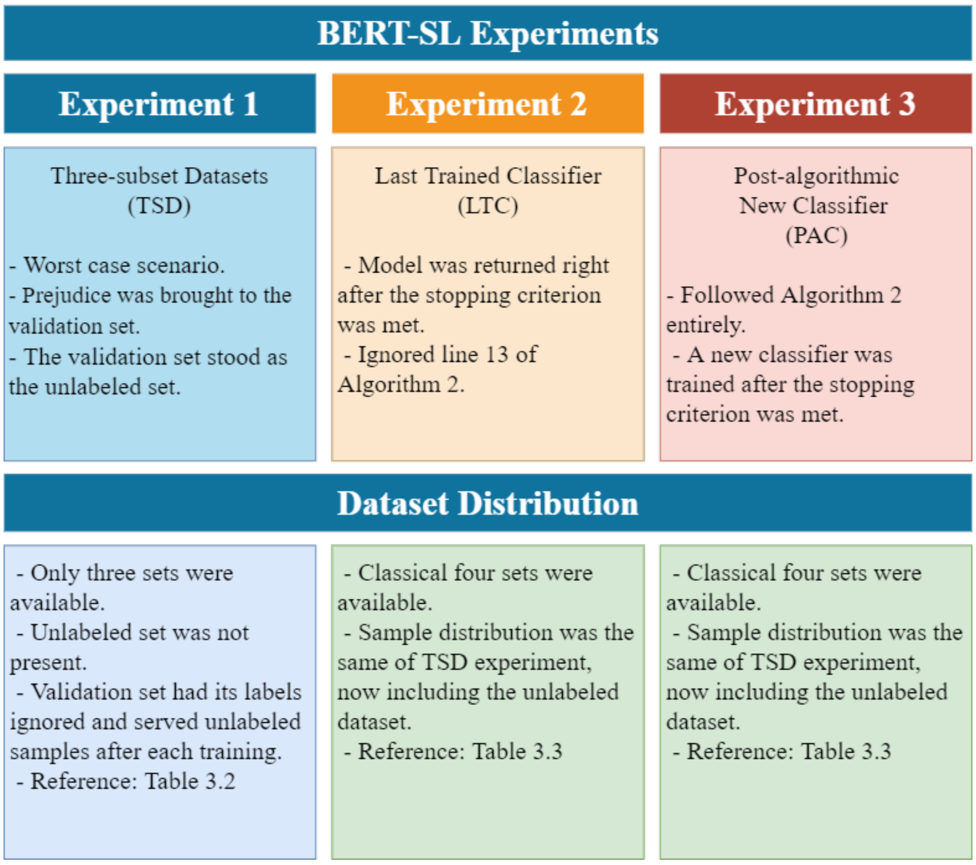
**Algorithm 3** BERT Self-Learning Approach Variant Pseudocode

```
Input: $\mathcal{L}_{tr}$: labeled training set; $\mathcal{L}_v$: labeled validation set; $\mathcal{L}_{ts}$: labeled test set; $\mathcal{U}$: unlabeled set; $\delta$: confidence threshold
Result: $\mathcal{T}$: labeled set; $\Phi_f$: final classifier
 1: $\mathcal{T} \leftarrow \mathcal{L}_{tr}, \mathcal{L}_v$
 2: while ($\mathcal{L}_v$ is not empty) do
 3:     $\Phi \leftarrow$ new classifier
 4:     $\Phi \leftarrow$ train new classifier on $\mathcal{T}$;
 5:     for $i = 1$ to $|\mathcal{L}_v|$ do
 6:         if confidence of $\Phi$.classify($\mathcal{L}_{v_i}$) $\geq \delta$ then
 7:             $\mathcal{L}_{tr} \leftarrow \mathcal{L}_{tr} \cup \mathcal{L}_{v_i}$, where $\mathcal{L}_{v_i}$ is the i-th instance in $\mathcal{L}_v$
 8:             Mark $\mathcal{L}_{v_i}$ as to be removed from $\mathcal{L}_v$;
 9:         end if
10:     end for
11:     Update ($\mathcal{L}_v$) by removing labeled instances;
12:     $\mathcal{T} \leftarrow \mathcal{L}_{tr}, \mathcal{L}_v$
13: end while
14: return $\mathcal{T}, \Phi_f$;
```

### 3.3.1 Experiments

As a means to better achieve the objectives of this research, and looking for trying out a plethora of scenarios and results, the experiments had samples and models available in three different ways, as described presented in Figure 3.5, and described below.

Classical BERT experiments were conducted as well, in order to let one compare the benchmark of BERT-SL against BERT in two different dataset setups of BERT, as described in this Subsection. In this fashion, during this work one will observe, as exhibited in Table 3.3, the four sets being used, notably the training set, validation set, unlabeled set, and test set.



| BERT-SL Experiments | | |
|---|---|---|
| **Experiment 1** | **Experiment 2** | **Experiment 3** |
| Three-subset Datasets (TSD)<br><br>- Worst case scenario.<br>- Prejudice was brought to the validation set.<br>- The validation set stood as the unlabeled set. | Last Trained Classifier (LTC)<br><br>- Model was returned right after the stopping criterion was met.<br>- Ignored line 13 of Algorithm 2. | Post-algorithmic New Classifier (PAC)<br><br>- Followed Algorithm 2 entirely.<br>- A new classifier was trained after the stopping criterion was met. |
| **Dataset Distribution** | | |
| - Only three sets were available.<br>- Unlabeled set was not present.<br>- Validation set had its labels ignored and served unlabeled samples after each training.<br>- Reference: Table 3.2 | - Classical four sets were available.<br>- Sample distribution was the same of TSD experiment, now including the unlabeled dataset.<br>- Reference: Table 3.3 | - Classical four sets were available.<br>- Sample distribution was the same of TSD experiment, now including the unlabeled dataset.<br>- Reference: Table 3.3 |

Figure 3.5: Summary table presenting a short description of the experiments and the respective dataset distributions.

### Three-subset Datasets

The experiment named Three-subset Datasets (TSD) processed an organization of the datasets that allows one to assess the BERT-SL performance when the validation set suffered prejudice by having its labels removed, and being classified according to the established threshold, having its samples moved from the validation set to the training set, as presented by Algorithm 3. This experiment followed the sample distribution indicated in Table 3.2: training set; validation set, that was processed as the unlabeled set; and test set.

### Last Trained Classifier

The Last Trained Classifier (LTC) experiment was developed following the sample distribution based on Table 3.3: training, validation, unlabeled and test set.

Table 3.2: Sample distribution for the Three-subset Datasets experiment. In this configuration, one can verify the sets splitting according to the dataset size, considering that the number of samples related to the unlabeled dataset was discarded.

| Dataset Size | Training Size | Validation Size | Test Size |
|---|---|---|---|
| 200 | 100 | 50 | 5740 |
| 594 | 269 | 146 | 5346 |
| 1188 | 540 | 291 | 4752 |
| 1782 | 810 | 437 | 4158 |

This experiment ignored line 13 of Algorithm 2 and returned then model to be tested right after the stopping criterion was met, having a successful model that was able to deplete the aforementioned dataset.

#### Post-algorithmic New Classifier

The Post-algorithmic New Classifier (PAC) experiment also incorporated the training, validation, unlabeled and test set referenced in Table 3.3.

The experiment carried out observing this configuration included a new classifier after meeting the stopping criterion. Hence the new classifier was used in the test step so the algorithm efficiency could be evaluated.

Table 3.3: Sample distribution for the Last Trained Classifier, and Post-algorithmic New Classifier experiments. In this configuration, one can see the sets division according to the dataset size.

| Dataset Size | Training Size | Validation Size | Unlabeled Size | Test Size |
|---|---|---|---|---|
| 200 | 100 | 50 | 50 | 5740 |
| 594 | 269 | 146 | 179 | 5346 |
| 1188 | 540 | 291 | 357 | 4752 |
| 1782 | 810 | 437 | 535 | 4158 |

### 3.3.2 Experiments that worked as performance benchmarks

The main objective of these experiments is to allow one observe the resulting performance of BERT-SL compared to the active learning method, and BERT in two different ways: BERT having the same number of samples BERT-SL initially had, named BERT'; and BERT having the same number of samples that BERT-SL had after incorporating the samples classified by itself, termed BERT".

Table 3.4: Sample distribution for the active learning experiment.

| Dataset Size | Training Size | Validation Size | Test Size |
|---|---|---|---|
| 200 | 150 | 50 | 5740 |
| 594 | 448 | 146 | 5346 |
| 1188 | 897 | 291 | 4752 |
| 1782 | 1345 | 437 | 4158 |

### Active Learning

With regards to the active-learning approach, the method follows this configuration, and receives the same number of samples the training set of BERT-SL has available to itself, videlicet, the number of samples classified by the algorithm itself plus the original samples from the training set, as seen in Table 3.4.

It is worth noting one major difference between the final training sets from BERT-SL and the active learning method. While the total of samples in the final training set of the self-learning method includes samples classified by the algorithm itself, the final training set of active learning includes samples that have been classified by an oracle, since this is the premise of the method itself. The BERT-SL method was expected to be exposed to some drawbacks, since classification by the oracle occurs in the active-learning method, while BERT-SL performs its own classification, based on the confidence level.

### BERT'

In this configuration, the number of samples available for the BERT training set is the same number of samples as in BERT-SL. Thus, reflecting a factual scenario, which commonly occurs in everyday life, one will be capable of assessing if the proposed solution offers benefits in its use. The sample distribution of this experiment is presented in Table 3.5

The specific goal of this experiment is to measure the self-learning algorithm's effectiveness as it finds new samples on its own, and how strongly this identification of new samples can improve its own performance when compared to classical BERT.

Table 3.5: BERT' sample distribution. In this distribution, the unlabeled set was merged with the test set.

| Dataset Size | Training Size | Validation Size | Test Size |
|---|---|---|---|
| 200 | 100 | 50 | 5790 |
| 594 | 269 | 146 | 5525 |
| 1188 | 540 | 291 | 5109 |
| 1782 | 810 | 437 | 4693 |

**BERT”**

In this setting, the total number of samples provided to the training of classical BERT is equivalent to the total number of samples classified and trained by BERT-SL, as a result of the classification process performed by the self-learning algorithm applied to the unlabeled dataset. The BERT” data distribution can be seen in Table 3.6

Table 3.6: BERT” sample distribution. This configuration included a distribution where twice as many samples were made available for BERT classic considering the minimal chosen dataset size of two hundred samples. It is possible to visualize that number of samples of the unlabeled set was merged with the training set.

| **Dataset Size** | **Training Size** | **Validation Size** | **Test Size** |
|---|---|---|---|
| 200 | 150 | 50 | 5740 |
| 200 | 200 | 5740 | |
| 594 | 448 | 146 | 5346 |
| 1188 | 897 | 291 | 4752 |
| 1782 | 1345 | 437 | 4158 |

### 3.3.3 Downstream Task

As the classification techniques evolved, coming from traditional methods, passing through ensemble-based learning techniques, non-parametric techniques, and tree-based classifiers, the choices made for this research, regarding classification techniques, followed the current understanding that deep learning approaches have achieved surpassing results in comparison to previous machine learning algorithms [34].

With the purpose of finding the best possible parameters for this experiment, several subsets were produced, with distinct number of labeled documents, composing the self-learning initial training process. Intending to explore the best setup regarding the deep learning model, the first experiment had as objective nothing more than the discovery of hyperparameters that would be expected to best perform when classifying the available dataset, bringing off satisfying results.

Thus, a set of basic experiments were produced using two hundred samples, once dealing with label scarcity is the main concern and challenge of this research, using the classical BERTimbau model, where several adjustments were made while pursuing the appropriate benchmark to be applied to the Self-Learning BERT model.

Considering that a small number of samples would be available for some classes in some subsets, and continuing in this line of thought, further effort was put into finding the optimum hyperparameters, *i.e.*, learning rate, number of epochs, and batch size.

Before trying to discover the best hyperparameters, it was of paramount importance to find the order of magnitude of the learning rate, which was defined after running some epochs and analyzing the loss generated from that training.

The model experiment design comprised momentum values ranging from 0.85 to 0.95, as outlined in [72], with BERTimbau using AdamW optimizer, and the Leslie Smith's 1cycle learning schedule used in the training.

It was possible to find the benchmark after having thoroughly tested all possible combinations of the values of the hyperparameters defined in Table 3.7. The found benchmark and related results will be presented in Chapter 4.

Table 3.7: Benchmark model hyperparameters.

| **Learning Rate** | **Batch Size** | **Nº of Epochs** |
|---|---|---|
| 3e-5 | 2 | 4 |
| 4e-5 | 6 | 5 |
| 5e-5 | 8 | 6 |
| 6e-5 | 10 | - |
| 7e-5 | - | - |

Then a series of new experiments were conducted with the aim of producing results that could initially be compared to Active Learning. However, the initial observations of the first experiments allowed the hypothesis of measuring the effectiveness of the self-learning method against the results of BERT itself.

Experiments were then produced with classical BERT, following the organization and distribution of the aforementioned data. The first experiment, called BERT', had the data distributed as presented in Table 3.5, once it is considered the most realistic scenario, once in real life there would be no more samples to feed classical BERT with.

Yet another experiment related to classical BERT was carried out, this time considering the number of documents classified by BERT-SL. This experiment could then be compared to the model produced by the proposed self-learning process. Data distribution for this experiment, called BERT", can be seen in Table 3.6.

Finally, considering only the dataset with the smallest number of samples, an experiment delivering two hundred samples to the training set was also performed, in order to deliver an advantage to BERT", so one can measure the results and how well BERT-SL can perform, even with fewer samples available. This experiment corresponds to the record in the second row of Table 3.6.

With respect to the Self-Learning BERT method, the confidence level, which is the threshold to a sample be considered as a labeled sample and included in the training set to be trained in the next iteration, was initially attached to values around 0.82, but it was thought to be valid to expand the possible confidence levels, for the sake of assessing how well they could perform with different number of samples.

Thereby, the confidence levels were specified in a range from 0.50 to 0.95, having exploratory variations of its values ranging from 0.81 to 0.84 in experiments where the results around 0.82 were considered satisfying.

## 3.4 Evaluation

The main idea of using BERTimbau, a BERT [29] based language model, is related to the possibility of finding adequate solutions to existing problems in the Portuguese language. It includes developing an entirely semi-supervised method by replacing the classical fine-tuning step, which is essentially a supervised process, with a self-supervised process.

By doing that, it covered both an accurate way of text representation regarding BERT embeddings, and the application of renowned methods for performing natural language processing tasks in a distinctive and effective approach.

Several observations came from the experiments performed in this research. In this line of thinking, evaluation harks back to the idea that it is imperative to use celebrated performance measures to gauge the efficiency of the proposed method. Given this reasoning, weighted f1-score was the measure of choice and was used to evaluate how good the method performed.

The established classification process was then set through the self-learning language model, which, for every subset of distinct size, after running the initial training, was assigned to classify the other unlabeled samples. After labeling the entire subsets, the f1-score will be estimated for each specific subset and the weighted f1-score will be determined by submitting the remnant samples, the test set, to the resulting model.

Conscious that some datasets would have a really limited number of labeled samples, the challenge here was going to find the best setup to make the self-learning process efficient.

The resulting weighted f1-score was finally compared to the active learning process f1-score and to classical BERT f1-score, using the same subsets created before. With the intention of being able to reproduce the experiments and having the same samples selected in every situation, the self-learning, the active learning, and classical BERT methods, a seed was defined in order to avoid negative impacts with different results coming as a consequence of randomness, allowing one to obtain reproducible output across multiple experiments.

# Chapter 4

# Results

*"You may never know what results come of your actions, but if you do nothing, there will be no results."*

— Mahatma Gandhi

The results of the experiments conducted using the self-learning approach during the fine-tuning step of BERT training are reported in this chapter. It begins by showing the loss plot, the first step of the first experiment, a chart used to help identify the maximal learning rate according to the presented falling loss.

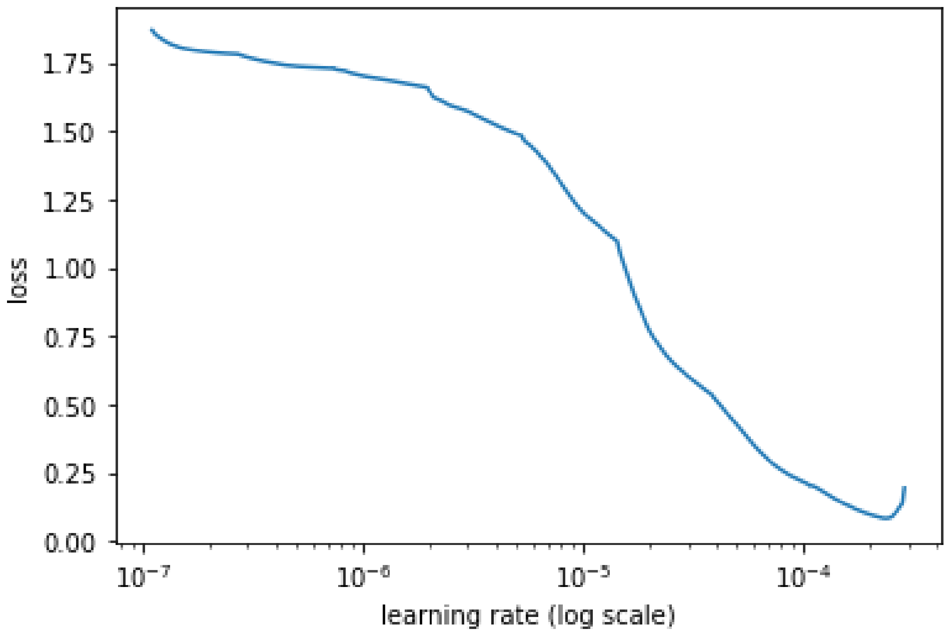


Figure 4.1: Loss plot used to identify the maximal learning rate associated with falling loss.

Subsequently, the chosen magnitude was applied to a variety of learning rates and then, the first experiment took place by developing a benchmark where several combinations are plotted in order to discover the possible best combination to be used in the next experiments.

Afterwards, the best results of series of experiments, using the chosen combination of hyperparameters, are shown, where the iterative self-learning method was used to allow one to evaluate the performance of the proposed method.

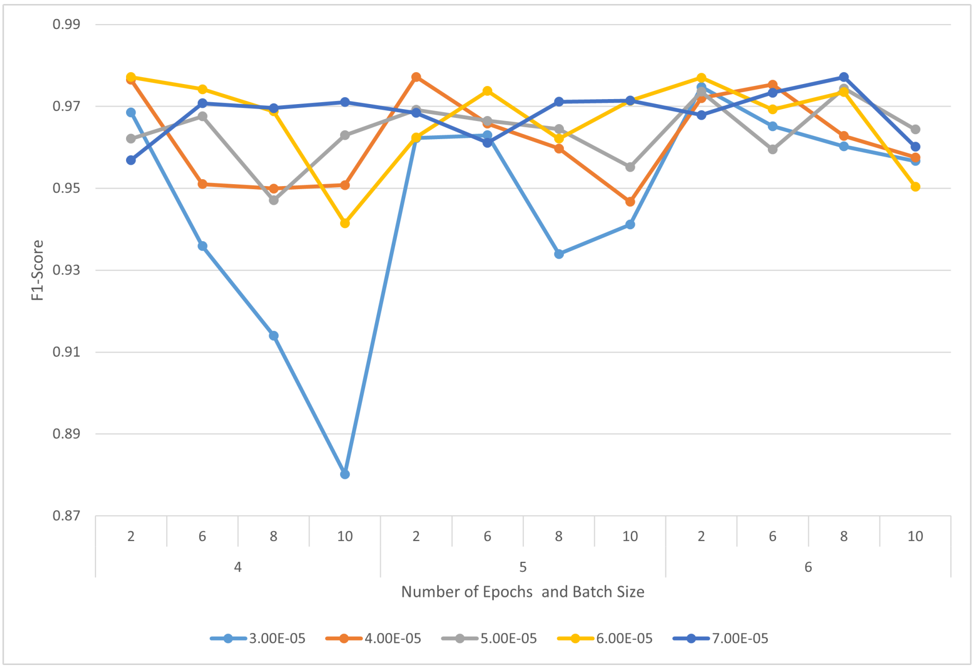


Figure 4.2: Hyperparameter benchmark shows the most performing hyperparameter combination.

## 4.1 Best Hyperparameters

With regards to carry out the best hyperparameter experiment, where hyperparameter combinations would show the most prominent benchmark, it can be seen in Figure 4.1 that $10^{-5}$ was the magnitude of choice after plotting the falling loss, as the loss begins to diverge at higher learning rates.

After choosing the learning rate magnitude, a variety of associations between the hyperparameters ran using the classical BERT, and the resulting chart, illustrated in Figure 4.2 allows one to notice the performance variation through the assortment of hyperparameters, and infer that the best performance is associated to the point in the chart that registered a 4.00E-5 learning rate, five epochs and batch size of two combination.

Table 4.1: Featured BERT-SL Classification Performance against benchmark experiments. Each row presents the performance comparison outlined against BERT', BERT" and active learning, showing the best results according to the dataset size (number of samples available), presenting the confidence level and the featured BERT-SL F1-score.

| **Ds Size** | **CL** | **F1-Score** | **Performance against** | | |
|---|---|---|---|---|---|
| | | | BERT' | BERT" | AL |
| 200 | 0.84 | 0.9771 | 0.0472 | 0.0157 | 0.1330 |
| 594 | 0.82 | 0.9867 | 0.0165 | 0.0146 | 0.0328 |
| 1188 | 0.90 | 0.9884 | 0.0146 | 0.0068 | 0.0110 |
| 1782 | 0.90 | 0.9933 | 0.0121 | 0.0027 | 0.0123 |

## 4.2 Self-learning

This section reports the main experiment results, encompassing all self-learning BERT variations described in Section 3.3, here termed BERT-SL TSD, BERT-SL LTC, and BERT-SL PAC. Before delving deeper into further details of the experiments, it can be seen that the results presented by BERT-SL were superior, with the vast majority of confidence levels set resulting in values that surpassed active learning and classical BERT itself, as it can be seen in Figure 4.3. The best result for each dataset is presented in Table 4.1

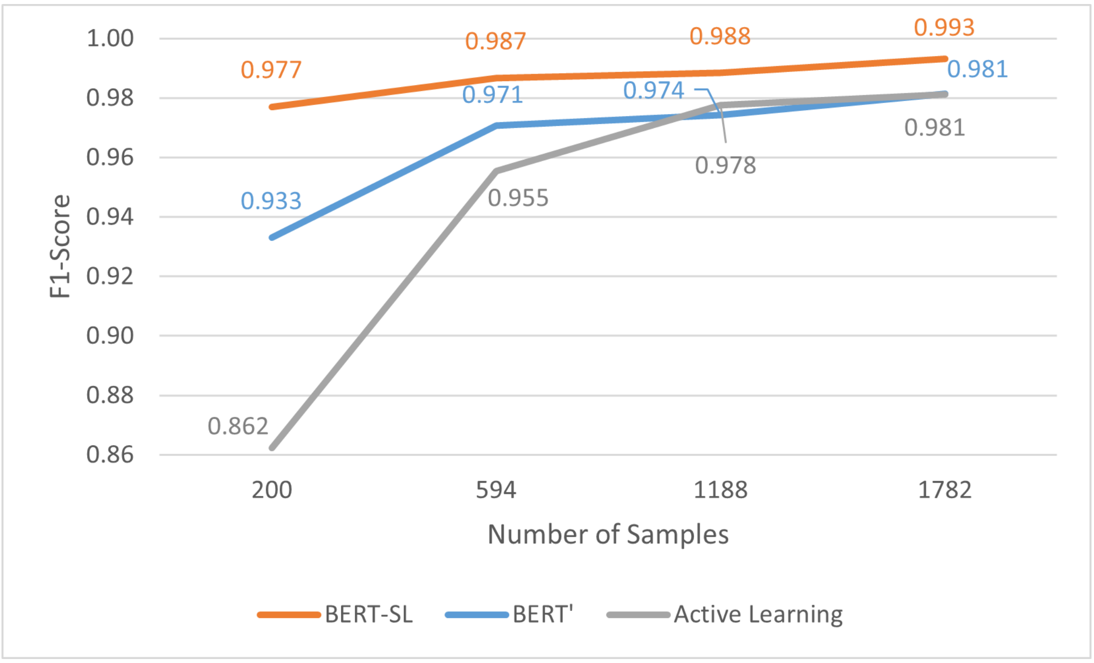


Figure 4.3: BERT-SL overall performance, for every dataset, compared to BERT' and Active Learning experiments.

As shown in Table 4.1, the use of self-learning also allowed the production of results comparable to BERT's, showing that it obtained excelling outcomes when compared to classical BERT, whose results are registered in Table 4.2, in several scenarios and confidence levels.

In Table 4.2, an additional experiment can yet be observed. In this experiment, twice as many samples are made available for BERT Classic considering the minimal chosen dataset size of two hundred samples. The two hundred samples were entirely submitted to classical BERT. Still, BERT-SL outperformed BERT 53.13% of the time, being able to reach scores 1.51% greater than classical BERT, which marked 0.9625 F1-Score weighted average.

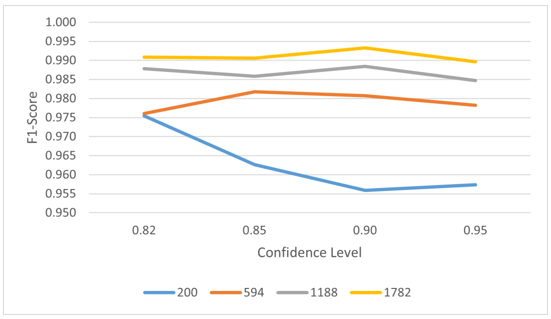
Overall, across multiple training experiments, the best classification result achieved a f1-score of 0.9933, while BERT' marked 0.9814 f1-score, BERT" achieved 0.9905 f1-score and the active learning process obtained 0.9811 f1-score.

Despite the highest score being related to the biggest dataset, the 1782-sample dataset, it is relevant to emphasize that the top gain came from the 200-sample dataset. The maximum gain coming from this experiment yielded a score 4.72% greater than BERT', 1.57% above BERT", and 13.30% beyond the active learning mark, as seen in Figure 4.3.

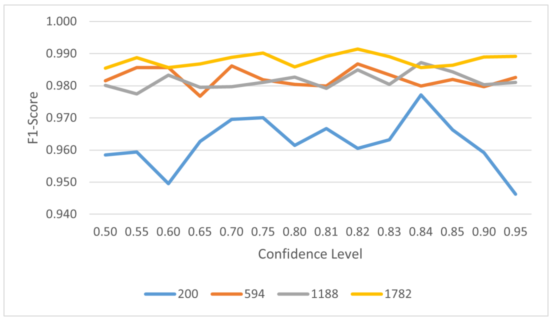
The BERT-SL, when working with the 594-sample dataset, provided a gain of 1.65% over BERT', and 3.28% over the active learning method. When dealing with 1188-sample dataset, the proposed method was able to deliver scores 1.46% better than BERT', and 1.10% greater than the active learning method. The gain got lower as the dataset size increased, reaching a gain of 1.21% over BERT' for the 1782-sample dataset, and 1.23% over the active learning method for the same dataset.

Table 4.2: Classical BERT classification results. This table presents the results stemming from the benchmark experiments. It is possible to see BERT' and BERT" results according to each number of samples, and data distribution.

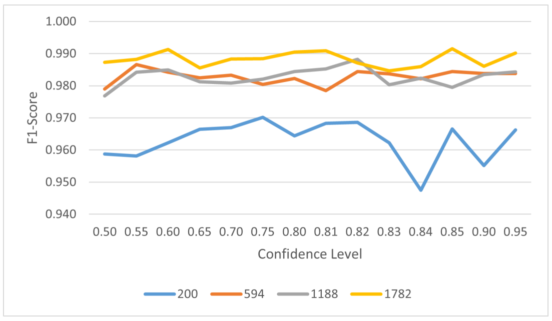
| **Dataset Size** | **Data Distribution** | **BERT' F1-Score** | **BERT" F1-Score** |
|---:|---:|---:|---:|
| 150 (200 Eq) | 100/50/5790 | 0.9330 | — |
| 200 | 150/50/5740 | — | 0.9620 |
| 200 | 200/5740 | — | 0.9625 |
| 415 (594 Eq) | 269/146/5525 | 0.9708 | — |
| 594 | 448/146/5346 | — | 0.9798 |
| 831 (1188 Eq) | 540/291/5109 | 0.9743 | — |
| 1188 | 897/291/4752 | — | 0.9817 |
| 1247 (1782 Eq) | 810/437/4693 | 0.9814 | — |
| 1782 | 1345/437/4158 | — | 0.9905 |



(a) Three-subset datasets, also referred to as BERT-SL TSD



(b) Last trained classifier, also referred to as BERT-SL LTC



(c) Post-algorithmic new classifier, also referred to as BERT-SL PAC

Figure 4.4: Performance of Self-learning BERT experiments.

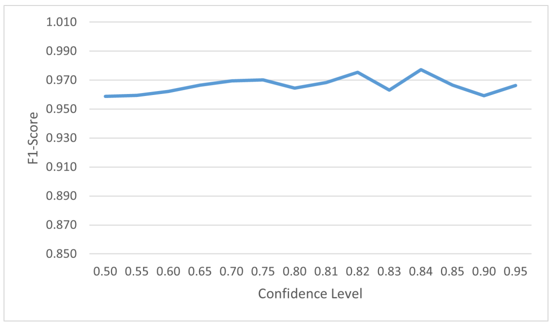
When deploying 15% or more samples from the corpus, it was possible to observe, from the 1188-sample dataset use, that BERT' and the active learning process yielded close results, while BERT-SL was getting results nearer BERT", that can be considered an exceptional case, where the advantage was given to BERT" while it had much more samples labeled by an oracle.

While still referring to BERT", it is worth noting that, for experiments with the 200-sample dataset, the results presented a f1-score of 0.9620; for the 594-sample dataset, 0.9799 f1-score; for the 1188-sample dataset, 0.9817 f1-score; and for the 1782-sample dataset, 0.9906 f1-score.

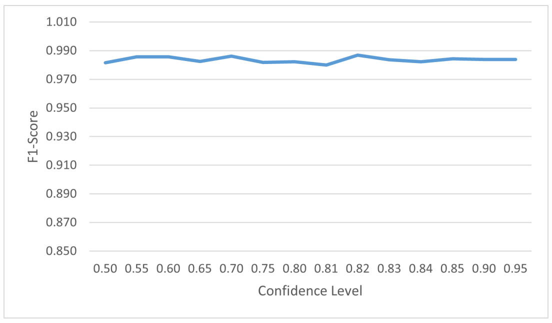
Considering the results coming from BERT-SL, presented in Figure 4.3, the proposed method has obtained the aforementioned gain of 1.57% over BERT" for the 200-sample dataset; 0.70% over BERT" for the 594-sample dataset; 0.68% over BERT" for the 1188-sample dataset; and 0.27% over BERT" for the 1782-sample dataset.

### 4.2.1 Three-Subset Datasets Experiment

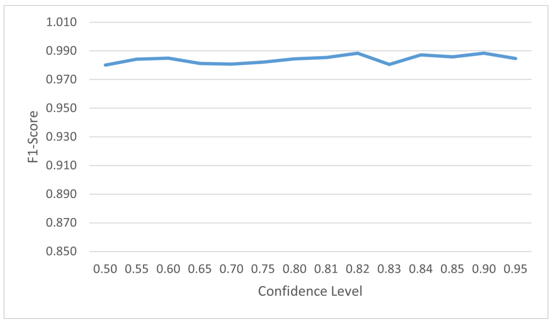
The named three-subset datasets experiment (TSD) was the first procedure using BERT-SL, and were carried out considering a strict scenario, where there would be no unlabeled



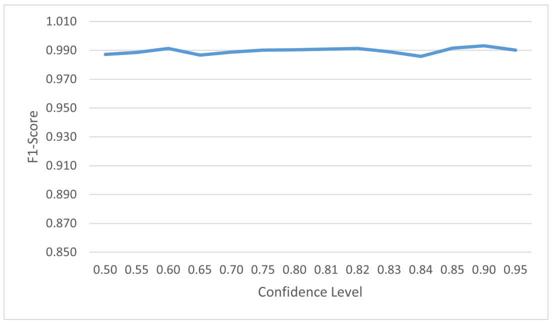
(a) 200-Sample Dataset



(b) 594-Sample Dataset



(c) 1188-Sample Dataset



(d) 1782-Sample Dataset

Figure 4.5: Best F1-score weighted average performance in every confidence level by dataset size.

dataset. The representation of this experiment can be seen on Algorithm 3. In this configuration, the validation set suffered prejudice by being classified according to the established threshold and having its samples moved from the validation set to the training set.

In this configuration, having no unlabeled dataset allowed the algorithm to work with the test set, validation set, and training set. The existing limitation imposed the use of the validation set as an alternative to the unlabeled set. This way, every BERT-SL round of classification extracted samples from the validation set, until it became empty, or the number of rounds reached ten. It was initially expected that the results would get worsen as the validation set decreased.

In this specific perspective, only a few thresholds were set. Initially, based on the fact that one of the first experiments with classical BERT brought in a 0.83 f1-score as one of the lowest records, 0.82 was set as the base confidence level. As seen in Figure 4.4a, even when bringing prejudice to the validation set, the method performed sufficiently well. As the results surprisingly showed reliable performance, the thresholds were expanded and higher values of confidence level were therefore set, reaching a threshold limit of 0.95.

One can also notice that the performance decreased as the confidence level raised, and, for the threshold of 0.82, the 200-sample dataset showed a 0.9754 f1-score weighted average score, while the 594-sample dataset presented a 0.9760 f1-score weighted average score, an honestly similar performance.

Notwithstanding that, this configuration, when processing the 200-sample dataset, allowed BERT-SL to achieve results 4.54% greater than BERT', 1.39% greater than BERT", and 13.11% greater than the active learning method, presenting a 0.9754 f1-score. Yet, the 200-sample dataset submitted to BERT-SL brought in a result 1.33% greater than the additional BERT" experiment, where two hundred samples, instead of the regular one hundred samples, were submitted to the classical BERT training step.

Whilst processing larger datasets, the performance difference has decreased, as one can perceive by looking at the results from 594-sample dataset, that top scored 0.9817, surpassing BERT' by 1.13%, BERT" by 0.18%, and the active learning method by 2.75%.

The resulting rates coming from the 1188-sample dataset submission, while achieving a f1-score of 0.9884, outperformed BERT' by 1.46%, BERT" by 0.68%, and the active learning process by 1.10%. The result at a 0.9 confidence level was the most performative result originated from the above-mentioned dataset, as observed in Figure 4.5c.

With regards to the 1782-sample dataset, the configuration obtained gains of 1.21% when compared to BERT', 0.27% against BERT", and 1.22% in comparison to the active learning method. The f1-score of 0.9933, at a 0.9 confidence level, was the most outstanding result regarding this dataset, as registered in Figure 4.5d.

### 4.2.2 Last Trained Classifier Experiment

Following, as it can be seen in Figure 4.4b, more threshold expansions were added to the last trained classifier (LTC) experiment, in order to find out if similar behavior to the three-subset experiment (TSD) would occur. A proper representation of this experiment is found on Algorithm 2, when suppressing line thirteen.

By the time this experiment was conducted, one lower threshold limit was identified, once the threshold at 0.55 registered performance poorer than the active learning method. Differently than TSD experiment, the obtained results between confidence levels of 0.82 and 0.85 showed steady results for every dataset submitted to BERT-SL.

The only exception was the 200-sample dataset, even though BERT-SL provided the most promising result of every experiment running this dataset, as shown in Figure 4.5a, when using the confidence level of 0.84, reaching a peak of 0.9771 f1-score, and then plunging until the confidence level of 0.95, where BERT-SL marked a f1-score of 0.9462.

The results registered in Figure 4.4b show that the experiment running the 200-sample dataset had a more fluctuating behavior when compared to the datasets holding higher number of samples. It is also worth pointing out that at the 0.84 threshold, the 594-sample dataset experiment reached 0.9798 f1-score, while yielded a f1-score of 0.9867 at confidence

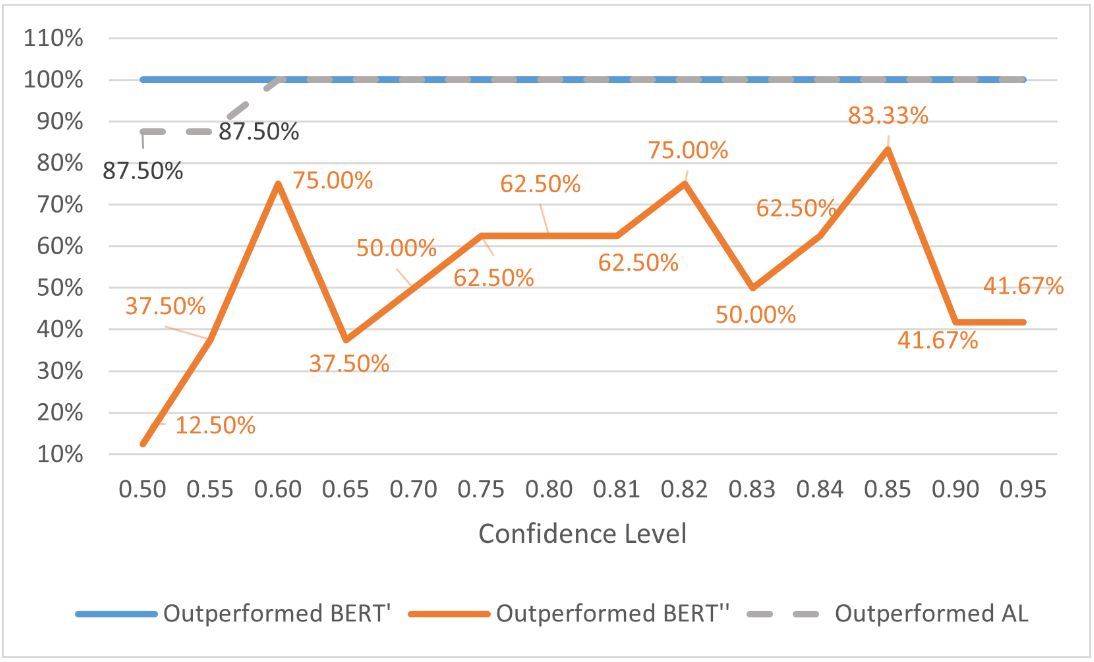


Figure 4.6: Percentage of BERT-SL experiments which outperformed BERT', BERT" and Active Learning, grouped by confidence level.

level of 0.82. When considering the results produced by BERT-SL LTC variation, with the 594-sample dataset as input, the most performing result reached the aforementioned f1-score of 0.9867, providing a gain of 1.64% against BERT', 0.70% above BERT", and 3.28% superior to the active learning method. This was the top f1-score for experiments related to the 594-sample dataset, as one can observe in Figure 4.5b.

With regards to the 1188-sample dataset, it remains evident that the results are close to the outcomes originated from BERT-SL LTC with the 594-sample dataset. As the Figure 4.4b may lead to this conclusion, the 594-sample f1-score average for BERT-SL LTC variation is 0.9823 whilst the results for the dataset in question brought in a f1-score average of 0.9815.

Moreover, BERT-SL LTC having the 1188-sample dataset as input reached the top f1-score when marked 0.9872, making the BERT-SL beat BERT' by 1.33%, BERT" by 0.56%, and the active learning experiment by 9.74%.

This experiment's highest score came from the 1782-sample dataset, reaching a f1-score of 0.9914, albeit it cannot be considered the highest gain against BERT' or the Active Learning experiment. This mark made BERT-SL LTC, while processing the 1782-sample dataset, outperform BERT' by 1.02%, BERT" by 0.08%, and the active learning method by 1.04%.

### 4.2.3 Post-algorithmic New Classifier Experiment

Thereafter, the BERT-SL post-algorithmic new classifier (PAC) experiment variation of the proposed method, whose most appropriate representation is found in Algorithm 2, and results are presented in Figure 4.4c, initially let one discern that BERT-SL, when processing smaller amounts of data showed a volatile performance after confidence level 0.82.

When associated to the 200-sample dataset, the results presented a steady growth until the confidence level of 0.75, where it started to slowly decrease its rates, showing two sharp falls, at the confidence levels of 0.84 and 0.9, that recovered immediately afterwards, reaching a f1-score of 0.9662 at the confidence level of 0.95.

Notwithstanding the fact that a recovery is presented, opposing to what happened to BERT-SL TSD variation on the highest confidence level, yet the obtained rate was not the best one. In this experiment, the confidence level of 0.75 can be highlighted as the most performing threshold, enabling the BERT-SL to reach f1-score of 0.9701, outperforming BERT' by 3.98%, BERT" by 0.84%, and the active learning method by 12.50%.

The confidence level lower limit was set as this experiment was being conducted, once the confidence level of 0.50 registered a performance below the active learning process. As the number of samples grew, the f1-score followed, but the gain rate showed a tendency to fall. The same phenomenon was observed in the other BERT-SL variations.

One peculiarity of BERT-SL PAC is that at lower confidence levels the proposed method was able to produce the best scores when dealing with the 200-sample and 594-sample datasets, getting 0.9701, when the confidence level was 0.75, and 0.9866, when the confidence level was 0.55, respectively.

The result linked to the 200-sample dataset showed that the achieved f1-score is greater than BERT' by 3.98%, BERT" by 0.84%, and the active learning method by 1.24%. The 594-sample dataset, when processed by BERT-SL brought in results 1.63% greater than BERT', 0.69% greater than BERT", and 3.26% greater than the active learning method.

With reference to the 1188-sample dataset, the BERT-SL PAC performed steadily, reaching a performance peak at the confidence level of 0.82, reaching a f1-score of 0.9883, which represented a 1.44% gain against BERT", being 0.66% greater than BERT", and 1.08% greater than the active learning method.

While considering the 1782-sample dataset, the results showed 0.9915 to be the best performance regarding this experiment, what makes the proposed method be equivalent to BERT", bringing in a gain of only 0.09%, albeit the BERT-SL PAC variation showed to be 1.03% more efficient than BERT', and 1.05% more efficient than the active learning method.

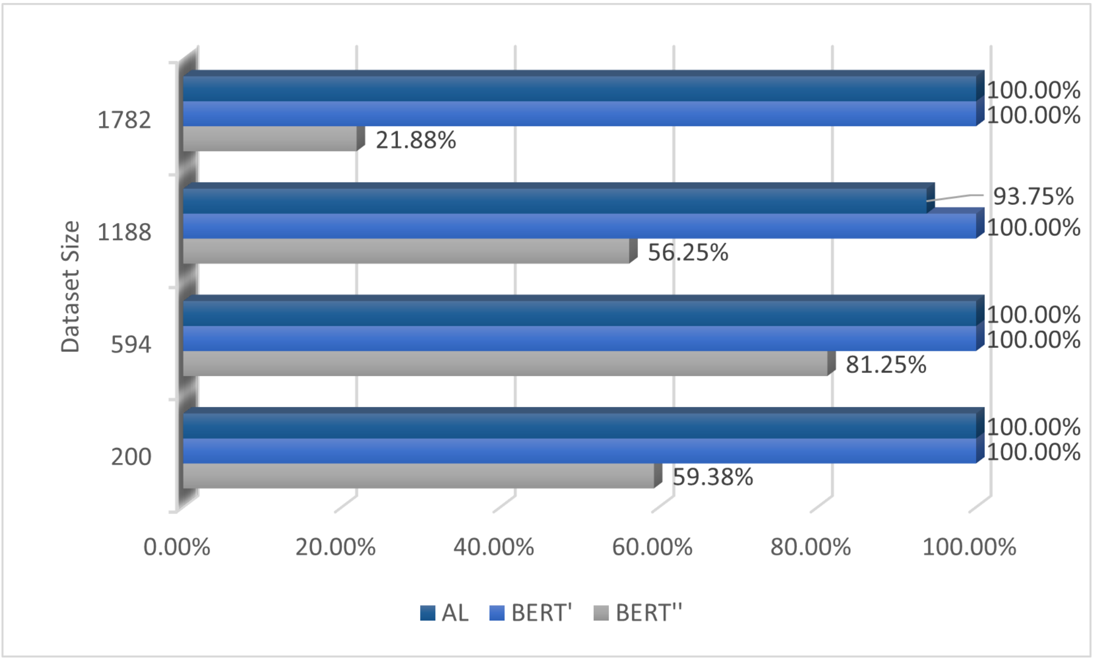


Figure 4.7: Percentage of BERT-SL experiments that outperformed BERT', BERT", and Active Learning, grouped by dataset size.

## 4.3 Classification evaluation

The outcomes originated as a consequence of the experiments running the proposed self-learning method, when grouped by the dataset size, whilst making no distinctions of what sort of experiment was applied, allows one to see that, for every dataset, the highest f1-score weighted average was reached when the defined confidence levels were above 0.81, as illustrated in Figure 4.5.

It is also worth noticing that, in spite of the 1188-sample dataset having a 0.9882 f1-score at confidence level of 0.82, the Figures 4.5c and 4.5d draw what seems to be a tendency to find best scores in the higher confidence level marks. When looking at the two largest datasets, the best results came when the confidence level was set at 0.90.

Nevertheless, one can perceive, by looking at the Figure 4.3, that the gain related to the proposed method decreases as the number of samples in the datasets increases, howbeit the f1-score increases.

This assessment leads one to infer that there is a need to find the balance between gain and f1-score. Surely, the number of labeled samples is the major problem to be considered, and by following this reasoning one can descry that, when processing the 200-sample dataset, BERT-SL was able to provide a 4.72% higher gain than BERT', and 13.30% higher than the active learning method; while the same method, when processing the 1782-sample dataset, showed a gain 1.21% superior to BERT', and 1.23 superior to the active learning method, a difference of 3.51%, and 12.07%, respectively.

On larger datasets, the difference of gain resulting from BERT-SL tends to diminish. When labeling 10% more samples, after processing the 1188-sample dataset, the gain related to BERT' results drop to 0.25%, while the gain related to the active learning method drops to 0.13%.

When looking at the gain that comes from BERT-SL, it may sound good to stick to the smaller dataset, but the change in the results can be something crucial depending on the scenario. At the same time, it is a remarkable difference, considering that labeling large numbers of documents is an expensive task, many times unfeasible.

As presented in Figure 4.6, throughout the undertook experiments, the language model resulting from the proposed method was able to outperform BERT' in every designed confidence level, surmounting the aforementioned method 100% of the time, surpassed the active learning process in every confidence level but 0.55 and 0.50, and outperformed BERT" 83.33% of times when confidence level was 0.85.

Considering the complete testing universe, Figure 4.7 delineates the results. It remains clear that BERT-SL excelled BERT' in every experiment and surpassed the active learning in almost all of them, being the only exception the tests where the 1188-sample dataset served as the input, when 93,75% surpassed the active learning method.

With regards to BERT-SL performance compared to BERT", it is exhibited, in Figure 4.7, that BERT-SL outstripped BERT" 59.38% of times when having 200 samples. The outperformance in this relation achieved 81.25% when having 594 samples, 56.26% when having 1188 samples, and 21.28% when having 1782 samples.

It has been successfully demonstrated that, even when more samples were fed to classical BERT, BERT-SL showed to be able to achieve better scores, mainly when it comes to labeled samples shortage, scenario where BERT-SL outshines BERT.

The experiment, in Table 4.1, showed that BERT-SL was capable of achieving up to 4.72% better performance than the classical BERT, and 13.30% than the Active Learning.

As described above, it was observed that, for experiments involving a set of two hundred samples, BERT-SL obtained a superior performance, capable of, considering the aforementioned 22 million documents, estimating in 1,039,272 the number of documents adequately classified, when compared to BERT'; in 2,926,811 the number of documents adequately classified, when compared to the Active Learning method; and, in the worst case performance gain, adequately classifying 59,644 documents, when compared to BERT".

The average gain for this setting would be of 314,524 documents properly classified by BERT-SL, in comparison to classical BERT, number that can be of significance, considered the scenarios where the proposed method could be applied.

# Chapter 5

# Conclusion

*"The gift of fantasy has meant more to me than my talent for absorbing positive knowledge."*

— Albert Einstein

This chapter presents the conclusions and last thoughts of the developed work, the dissertation, providing considerations, and an interpretation concerning the performed research, including the outcomes of the conducted experiments. The substantial and remarkable findings, as well as the major contributions of this dissertation are presented, followed by future directions for further research.

Once considered the importance of appropriately evaluating the documents, given that the existing document set is intrinsically related to the Army's capacity of pursuing the fulfillment of its doctrinal missions, the ensuing experiments showed convincing results regarding the initial objective of this study, which was to find a self-learning method that could reach the human classification capacity while exploring possibilities of mitigating the labeled samples shortage.

By observing the presented results from Section 4, and knowing that in real world large amounts of data are available without any sort of label, it is possible to infer, with the results of this study, that the introduction of the self-learning method in the fine-tuning stage yielded evidences of improvement of BERT's performance in the available dataset, which properly reflects the targeted scenarios of this study, that aims to treat the problem of scarcity of labels in the document sets of the Brazilian Federal Government.

The self-learning approach, associated with BERTimbau, showed to be a promising method regarding NLP classification tasks, showing better results than the classical BERT when the same number of samples is available to both methods, surpassing the traditional method in every single experiment following this setup, outshining when the available labeled samples are smaller.

Even when considering the minimal gain, almost 60,000 documents would receive the adequate classification, hence receiving the proper destination in the future, and by that allowing the Brazilian Army and the Federal Government to accurately give the right destination to every existing document in the system.

The method showed outstanding results associated with datasets whose labels reach only 3% of the samples, showing an increasing performance, when compared to classical BERT, as the number of available labeled samples decreases. It was therefore possible to achieve suitable results while carrying out the experiment, making it possible to consider applying the proposed method to other datasets in the Brazilian Federal Government, once it has been provided enough evidence that self-learning could be an asset while addressing the labeled document scarcity.

## 5.1 Future Directions

Regarding future works, this experiment could be developed following some directions, including the proposed bellow:

Once the considered method was developed following a restrict scope, in order to deal with a problem related to the Federal Government, and the directions of this work were limited to the fulfillment of this need, it would be interesting to expand the scope and explore how the proposed method would perform dealing with other databases, mainly noisy ones, like messages based on tweets, and posts from Facebook. This scope expansion could be done in the future, in order to assess the method's efficiency to a diversity of scenarios, and how well it would generalize for each one of them.

# Bibliography

[1] Arquivo Nacional: *Gestão de documentos: curso de capacitação para os integrantes do Sistema de Gestão de Documentos de Arquivo SIGA, da administração pública federal*. Course packet, January 2019. Electronic Data (1 file: 993 kb). 1

[2] Azemi, Nor, Hazlifah Zaidi, and Norhayati Hussin: *Information Quality in Organization for Better Decision-Making*. International Journal of Academic Research in Business and Social Sciences, 7, January 2018. 1

[3] Swash, GD: *The information audit*. Journal of managerial psychology, 1997. 1

[4] Strong, Diane M, Yang W Lee, and Richard Y Wang: *Data quality in context*. Communications of the ACM, 40(5):103–110, 1997. 1

[5] Redman, Thomas C: *Improve data quality for competitive advantage*. MIT Sloan Management Review, 36(2):99, 1995. 1

[6] Petroni, Fabio, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel: *Language Models as Knowledge Bases?*, 2019. 2, 3, 22, 23, 26

[7] Liang, Percy: *Semi-supervised learning for natural language*. PhD thesis, Massachusetts Institute of Technology, 2005. 3

[8] Oliver, Avital, Augustus Odena, Colin Raffel, Ekin D. Cubuk, and Ian J. Goodfellow: *Realistic Evaluation of Deep Semi-Supervised Learning Algorithms*, 2019. 3

[9] Iosifidis, Vasileios and Eirini Ntoutsi: *Large Scale Sentiment Learning with Limited Labels*. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 1823–1832, New York, NY, USA, 2017. Association for Computing Machinery, ISBN 9781450348874. `https://doi-org.ez54.periodicos.capes.gov.br/10.1145/3097983.3098159`. 3, 21, 33

[10] Jean, Neal, Sang Michael Xie, and Stefano Ermon: *Semi-supervised Deep Kernel Learning: Regression with Unlabeled Data by Minimizing Predictive Variance*, 2019. 3

[11] Antol, Stanislaw, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh: *Vqa: Visual question answering*. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015. 3

[12] Chen, Danqi and Wen tau Yih: *Open-Domain Question Answering.* In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 34–37, Online, July 2020. Association for Computational Linguistics. `https://aclanthology.org/2020.acl-tutorials.8`. 3

[13] Sharma, Yashvardhan and Sahil Gupta: *Deep learning approaches for question answering system.* Procedia computer science, 132:785–794, 2018. 3

[14] Crawford, Michael, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter, and Hamzah Al Najada: *Survey of review spam detection using machine learning techniques.* Journal of Big Data, 2(1):1–24, 2015. 3

[15] Alom, Zulfikar, Barbara Carminati, and Elena Ferrari: *A deep learning model for Twitter spam detection.* Online Social Networks and Media, 18:100079, 2020. 3

[16] Makkar, Aaisha and Neeraj Kumar: *An efficient deep learning-based scheme for web spam detection in IoT environment.* Future Generation Computer Systems, 108:467–487, 2020. 3

[17] Zhang, Lei, Shuai Wang, and Bing Liu: *Deep learning for sentiment analysis: A survey.* Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4):e1253, 2018. 3

[18] Araque, Oscar, Ignacio Corcuera-Platas, J Fernando Sánchez-Rada, and Carlos A Iglesias: *Enhancing deep learning sentiment analysis with ensemble techniques in social applications.* Expert Systems with Applications, 77:236–246, 2017. 3

[19] Do, Hai Ha, PWC Prasad, Angelika Maag, and Abeer Alsadoon: *Deep learning for aspect-based sentiment analysis: a comparative review.* Expert systems with applications, 118:272–299, 2019. 3

[20] Meng, Yu, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han: *Text Classification Using Label Names Only: A Language Model Self-Training Approach*, 2020. 3, 35

[21] Zhu, Xiaojin Jerry: *Semi-supervised learning literature survey.* 2005. Last modified on July 19, 2008. 3, 20, 21

[22] Li, Yan and Jieping Ye: *Learning adversarial networks for semi-supervised text classification via policy gradient.* In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pages 1715–1723, 2018. 3, 20, 21, 33, 34

[23] Nigam, Kamal, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell: *Text classification from labeled and unlabeled documents using EM.* Machine learning, 39(2):103–134, 2000. 3, 32, 33

[24] McEntee, Eamon: *Enhancing Partially Labelled Data: Self Learning and Word Vectors in Natural Language Processing.* 2019. 4

[25] Fazakis, Nikos, Stamatis Karlos, Sotiris Kotsiantis, and Kyriakos Sgarbas: *Self-trained LMT for semisupervised learning.* Computational intelligence and neuroscience, 2016, 2016. 4

[26] Tsypkin, Ya: *Self-learning–What is it?* IEEE Transactions on Automatic Control, 13(6):608–612, 1968. 4

[27] Mämmelä, Aarne, Jukka Riekki, Adrian Kotelba, and Antti Anttonen: *Multidisciplinary and Historical Perspectives for Developing Intelligent and Resource-Efficient Systems.* IEEE Access, 6:17464–17499, 2018. 4

[28] Wolf, Florian, Tomaso Poggio, and Pawan Sinha: *Human Document Classification Using Bags of Words.* August 2006. 4, 16, 17

[29] Devlin, Jacob, Ming Wei Chang, Kenton Lee, and Kristina Toutanova: *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, 2019. 4, 23, 26, 27, 28, 31, 41, 49

[30] Exército Brasileiro: *Instruções Gerais para Avaliação de Documentos do Exército*, October 2019. EB10-IG-01.012. 4

[31] Joaquim, Carlos Eduardo de Lima and Thiago de Paulo Faleiros: *BERT Self-Learning Approach for Document Classification with Limited Labels.* The 16th Learning and Intelligent Optimization Conference, June 2022. 5

[32] Eisenstein, Jacob: *Natural language processing*, 2018. 7, 8, 9

[33] Beysolow II, Taweh: *Applied natural language processing with python.* Springer, 2018. 7, 8

[34] Kowsari, Kamran, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown: *Text classification algorithms: A survey.* Information, 10(4):150, 2019. 7, 10, 11, 12, 13, 47

[35] Nadkarni, Prakash M, Lucila Ohno-Machado, and Wendy W Chapman: *Natural language processing: an introduction.* Journal of the American Medical Informatics Association, 18(5):544–551, 2011. 8

[36] Jurafsky, Daniel and James H. Martin: *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* 2021. `http://www.web.stanford.edu/~jurafsky/slp3/`, 3rd edition draft. 10

[37] Sebastiani, Fabrizio: *Machine learning in automated text categorization.* ACM computing surveys (CSUR), 34(1):1–47, 2002. 11, 14, 15, 16

[38] Kobayashi, Vladimer B, Stefan T Mol, Hannah A Berkers, Gábor Kismihok, and Deanne N Den Hartog: *Text classification for organizational researchers: A tutorial.* Organizational research methods, 21(3):766–799, 2018. 11

[39] Vijayan, Vikas K, K. R. Bindu, and Latha Parameswaran: *A comprehensive study of text classification algorithms.* In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1109–1113, 2017. 11, 12

[40] Aggarwal, Charu C and ChengXiang Zhai: *A survey of text classification algorithms.* In *Mining text data*, pages 163–222. Springer, 2012. 11, 12

[41] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton: *Deep learning.* nature, 521(7553):436–444, 2015. 13

[42] Sasaki, Yutaka: *The truth of the F-measure.* Teach Tutor Mater, January 2007. 14

[43] Thomas, Wood: *F-score*, May 2019. `https://deepai.org/machine-learning-glossary-and-terms/f-score`. 14

[44] Mitchell, Tom M: *Machine Learning.* 1996. 15

[45] IBM: *What is overfitting?* IBM Cloud Learn HUB. `https://www.ibm.com/cloud/learn/overfitting`. 16

[46] Mitchell, Tom M: *Machine Learning.* 1997. 16

[47] Ein-Dor, Liat, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim: *Active Learning for BERT: An Empirical Study.* In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online, November 2020. Association for Computational Linguistics. `https://aclanthology.org/2020.emnlp-main.638`. 17, 18

[48] Settles, Burr: *Active learning literature survey.* 2009. 17, 18

[49] Liu, Xiao, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang: *Self-supervised Learning: Generative or Contrastive.* IEEE Transactions on Knowledge and Data Engineering, pages 1–1, 2021. 18, 19, 20, 21

[50] Li, Fan, David A. Clausi, Linlin Xu, and Alexander Wong: *ST-IRGS: A Region-Based Self-Training Algorithm Applied to Hyperspectral Image Classification and Segmentation.* IEEE Transactions on Geoscience and Remote Sensing, 56(1):3–16, 2018. 21

[51] Mihalcea, Rada: *Co-training and self-training for word sense disambiguation.* In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 33–40, 2004. 21

[52] Iosifidis, Vasileios and Eirini Ntoutsi: *Sentiment analysis on big sparse data streams with limited labels.* Knowledge and Information Systems, pages 1–40, 2019. 21, 22

[53] Peters, Matthew E., Waleed Ammar, Chandra Bhagavatula, and Russell Power: *Semi-supervised sequence tagging with bidirectional language models.* arXiv preprint arXiv:1705.00108, 2017. 22, 23

[54] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin: *Attention Is All You Need*, 2017. 23, 24, 25, 26, 27

[55] Padmanabhan, Arvind: *Attention Mechanism in Neural Networks.* Available at `https://devopedia.org/attention-mechanism-in-neural-networks` Accessed on: Mar. 28, 2021, 2019. Version 3, November 2016. 23

[56] Cheng, Jianpeng, Li Dong, and Mirella Lapata: *Long Short-Term Memory-Networks for Machine Reading.* In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas, November 2016. Association for Computational Linguistics. `https://aclanthology.org/D16-1053`. 23

[57] Raghavan, Prabhakar: *How AI is powering a more helpful Google*, Oct 2020. `https://blog.google/products/search/search-on/`. 27

[58] Kulshrestha, Ria: *Keeping up with the Berts.* BERT Explained: What it is and how does it work?, Nov 2020. `https://towardsdatascience.com/keeping-up-with-the-berts-5b7beb92766`. 28

[59] Brinne, Björn: *High-res 3D representation of a transformer (BERT)*, Apr 2020. `https://peltarion.com/blog/data-science/illustration-3d-bert`. 29

[60] Yeung, Albert Au: *BERT - Tokenization and Encoding.* Github Pages, Jun 2020. `https://albertauyeung.github.io/2020/06/19/bert-tokenization.html`. 29, 30, 31

[61] Souza, Fábio, Rodrigo Nogueira, and Roberto Lotufo: *BERTimbau: pretrained BERT models for Brazilian Portuguese.* In *9th Brazilian Conference on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear)*, 2020. 31, 41

[62] Dempster, Arthur P, Nan M Laird, and Donald B Rubin: *Maximum likelihood from incomplete data via the EM algorithm.* Journal of the Royal Statistical Society: Series B (Methodological), 39(1):1–22, 1977. 32

[63] Fragos, Kostas, Petros Belsis, and Christos Skourlas: *Combining Probabilistic Classifiers for Text Classification.* Procedia-Social and Behavioral Sciences, 147:307–312, 2014. 32, 33

[64] Howe, Jerrold Soh Tsin, Lim How Khang, and Ian Ernst Chai: *Legal Area Classification: A Comparative Study of Text Classifiers on Singapore Supreme Court Judgments*, 2019. 34, 36

[65] Sun, Chi, Xipeng Qiu, Yige Xu, and Xuanjing Huang: *How to Fine-Tune BERT for Text Classification?*, 2020. 34

[66] McCloskey, Michael and Neal J Cohen: *Catastrophic interference in connectionist networks: The sequential learning problem.* In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 34

[67] González-Carvajal, Santiago and Eduardo C. Garrido-Merchán: *Comparing BERT against traditional machine learning text classification*, 2021. 35

[68] Bird, Steven, Ewan Klein, and Edward Loper: *Natural Language Processing with Python*. 2009. `http://nltk.org/book`. 37

[69] Honnibal, Matthew, Ines Montani, Sofie Van Landeghem, and Adriane Boyd: *spaCy: Industrial-strength Natural Language Processing in Python*. `https://doi.org/10.5281/zenodo.1212303`, 2020. 38

[70] Maiya, Arun S.: *ktrain: A Low-Code Library for Augmented Machine Learning*. CoRR, abs/2004.10703, 2020. `https://arxiv.org/abs/2004.10703`. 38

[71] Castro, Nádia Félix Felipe da Silva and Anderson da Silva Soares: *Multilingual Transformer Ensembles for Portuguese Natural Language Tasks*. 2020. 41

[72] Smith, Leslie N.: *A disciplined approach to neural network hyper-parameters: Part 1 – learning rate, batch size, momentum, and weight decay*, 2018. 48