INSTITUTO TECNOLÓGICO DE AERONÁUTICA

Heládio Sampaio Lopes

NATURAL LANGUAGE PROCESSING FOR TREND FORECASTING

Final Paper
2020

Course of Computer Engineering

CDU 621.38

Heládio Sampaio Lopes

NATURAL LANGUAGE PROCESSING FOR TREND FORECASTING

Advisor

Prof. Dr. Filipe Alves Neto Verri (ITA)

COMPUTER ENGINEERING

São José dos Campos
Instituto Tecnológico de Aeronáutica

2020

Cataloging-in Publication Data
Documentation and Information Division

Lopes, Heládio Sampaio
Natural Language Processing for Trend Forecasting / Heládio Sampaio Lopes.
São José dos Campos, 2020.
39f.

Final paper (Undergraduation study) – Course of Computer Engineering– Instituto Tecnológico de Aeronáutica, 2020. Advisor: Prof. Dr. Filipe Alves Neto Verri.

1. Natural Language Processing. 2. Topic Modelin. 3. Machine Learning. I. Instituto Tecnológico de Aeronáutica. II. Natural Language Processing for Trend Forecasting.

BIBLIOGRAPHIC REFERENCE

LOPES, Heládio Sampaio. **Natural Language Processing for Trend Forecasting**. 2020. 39f. Final paper (Undergraduation study) – Instituto Tecnológico de Aeronáutica, São José dos Campos.

CESSION OF RIGHTS

Heládio Sampaio Lopes
H8A St., 113
12228-460 – São José dos Campos–SP

NATURAL LANGUAGE PROCESSING FOR TREND FORECASTING

This publication was accepted like Final Work of Undergraduation Study

---

Heládio Sampaio Lopes

Author

Filipe Alves Neto Verri (ITA)

Advisor

Prof. Dr. Inaldo Capistrano Costa
Course Coordinator of Computer Engineering

São José dos Campos: November 20, 2020.

To my lovely mother, who has always supported me through this tough journey.

Acknowledgments

"Quanto mais difícil for a vitória, maior será sua felicidade em ganhar."
— Pelé

Resumo

Resumo

Abstract

Abstract

List of Figures

List of Tables

List of Abbreviations and Acronyms

AI	Artificial Intelligence
BoW	Bag of Words
CBOW	Continuous Bag of Words
NLP	Natural Language Processing
ML	Machine Learning
TF-IDF	Term Frequency Inverse Document Frequency
IT	Information Technology
LDA	Latent Dirichelt allocation
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
NIPS	Neural Information Processing Systems

Contents

1 Introduction

Recently, the growth of artificial intelligence has been helping us to solve problems in the most various areas, including in linguistics. With natural language processing techniques, computers can process text in order to extract information faster than humans.

1.1 Motivation

Every kind of expression, verbal or in writing, brings us much information to be interpreted. Whether the topic is chosen, the tone used, the choice of words, everything can be interpreted, and then generate some valuable information. Over the years, more and more knowledge is generated and we humans are unable to process such an amount of information. Natural Language Processing emerges as a technology capable of assisting us in this hard task.

Khurana *et al.* (2017) defines Natural Language Processing, abbreviated by NLP, as a branch of artificial intelligence capable of making computers comprehend and extract information from human language. NLP can perform a lot of tasks, such as identifying different topics for a set of documents, classifying texts on predefined subjects, and beyond that extract the sentiment to know what people are saying about something.

The topic discovery was widely explored in the literature, Hurtado *et al.* (2016) and Jelodar *et al.* (2020) use it to group, respectively, papers abstracts and comments in social media in similar clusters. In a long period, it is possible to analyze how these founded topics behave through time and make predictions about the near future.

1.2 Objective

Curious about the fast word's evolution, this work aims to implement Natural Language Processing techniques to propose a framework capable of modeling in real-time the topics' evolution over the years. With this in mind, evaluate the ability of those models to make predictions about future trends.

1.3 Organization of this work

The remaining of this work is organized as follows: Chapter 2 will cover the theory behind Natural Language Processing. Chapter 3 will describe some previous works which use topic discovery and trend forecast. Chapter 4 will better explain the problem and the methodology that will be used to handle it. Finally, Chapter **??** will present the chronological roadmap until the conclusion of the work.

2 Literature Review

In this chapter, we will introduce the general concepts and techniques behind Natural Language Processing. We will cover all the necessary steps for extracting meaningful topics from texts. Finally, a brief discussion about machine learning for tasks like classification and regression.

2.1 Natural Language Processing

In artificial intelligence, NLP appears as the field responsible for study the interactions between computers and natural human language. Thus, it is capable of the program a computer to extract significant information from the text corpus, called documents, and use this information to apply a machine learning model and use it in several applications.

2.1.1 Text Processing Techniques

The key task to several machine learning problems consists in make good data processing before applying any model. A clean data set can allow a model to increase its performance in the learning process, making a better identification in the patterns present in the variables. Therefore, in the following sections, it will be discussed a few techniques to clear the text and prepare it for machine learning algorithms.

2.1.1.1 Normalization

There is no right way to normalize text. This process is crucially important to put all text at the same level. A normalization process consists of a series of steps to be followed consecutively, all of then can be seen as 4 big tasks: stemming, lemmatization, stop words removal and everything else.

1. Stemming: is the process of reducing inflected words to a primitive form, the stem. This method can remove the word's affixes to capture its base meaning, and still

reducing the number of variations to save memory space. Figure 2.1 shows how some inflections for "connect" can be converted to its root form.

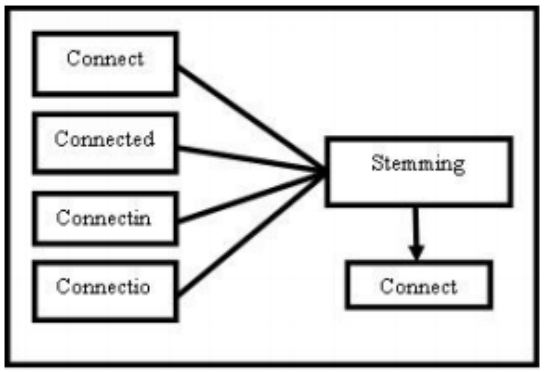


FIGURE 2.1 – Stemming process for "connect" variations, Figure from (VIJAYARANI *et al.*, 2015).

2. Lemmatization: similar to stemming, this step also reduces words to some primitive form, but with a little improvement. Lemmatization can return the words to his dictionary form, based on its part of speech context. Hence, it is possible to discriminate words with the same spelling but different meanings depending on the context.

3. Remove stop words: Many words can occur a several time in a document without adding any meaningful information, such as *the*, *is*, *at*, *which*, and *on*. Their high frequency can be identified as an obstacle to perform good results on NLP models, (KANNAN; GURUSAMY, 2014).

   There are some types to remove stop words, most of then based on evaluating the frequency of words in a text, for more information see (VIJAYARANI *et al.*, 2015). But the classic and easier method is based on using a pre-compiled list of know words and removing then from the text.

4. Everything else: Different from the previous steps, the last one doesn't need any grammar rules or even a frequency analysis, it's purely text manipulation. It involves set all character to lowercase; remove numbers or convert then to word form; remove punctuation; expand contractions; convert special characters to ASCII form; and any other conversion needed.

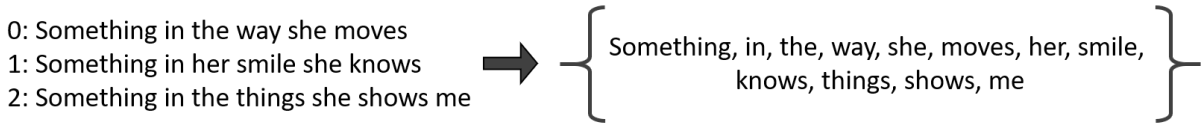
2.1.1.2 Tokenization

Once the data is normalized, we need to know how to represent it. The tokenization process consists in splitting longer strings into meaningful small pieces called tokens. The most common way to tokenize a text is chunking it the into words, i.e., given a piece of text the tokenization process will return a list of words.

2.1.1.3 Bag of Words

The machine learning algorithms take numerical features as input, hence, it will be necessary to represent the text in numerical form. With the Bag of Words model, we can represent in matrix form a set of documents.

With the tokenization output, we will have the lists representations for all documents in the data set. Those lists can be interpreted as vectors over the vector space of all unique tokens, also called by vocabulary. So, for a given sentence, we mark how many times its words appear in the list indexes where each entry corresponds to a word in the vocabulary. Figure 2.2 shows a simple example of how three sentences can be represented with the BoW model.



	Something	In	The	Way	She	Moves	Her	Smile	Knows	Things	Shows	Me
0	1	1	1	1	1	1	0	0	0	0	0	0
1	1	1	0	0	1	0	1	1	1	0	0	0
2	1	1	1	0	1	0	0	0	0	1	1	1

FIGURE 2.2 – Bag of Words example.

2.1.1.4 TF-IDF

Term Frequency Inverse Document Frequency, TF-IDF for short, it is applied to a BoW to determine the relative frequency for words in a specific document when compared to the inverse proportion of that word over all documents in the collection. So, it can be determined how important are the words in a specific document.

From BoW, for the i^{th} vocabulary's word in the j^{th} document, its TF-IDF weight is:

$$w_{i,j} = \text{tf}_{i,j} \times \log\left(\frac{N}{\text{df}_i}\right). \tag{2.1}$$

Where, the term frequency, $\text{tf}_{i,j}$, is how many time i^{th} word appears in the j^{th} document. The document frequency, df_i, is the number of documents in which th i_{th} vocabulary words is present. And, finally, N is the size of the document collection, with a large number of documents this term can explode, so the logarithmic function is applied to dampen this effect.

2.1.2 Word Embedding

The vectorization methods like BoW and TF-IDF can be very useful, but they can not represent the context of the words. This means the same words used in different contexts have the same representation, just as different words used with the same meaning are represented differently. Besides that, an one-hot encoding method, like BoW, presents a very sparse representation with high dimensionality.

Word Embedding is a technique to represent words in vectors capable of capture the words context in a document. It is also capable to smooth the high dimensionality effect by using a much more compact vector to represent the words.

There are three most known ways to perform a good word embedding. We will describe briefly each one of them below.

2.1.2.1 Word Representations in Vector Space

The first great word embedding technique emerged when Google researchers proposed two architectures to build continuous vector representations of words. Word's context can be observed as the words that surround it in a sentence. Then, using shallow neural networks, it is possible to calculate the word vector space based on the word's context, (MIKOLOV *et al.*, 2013).

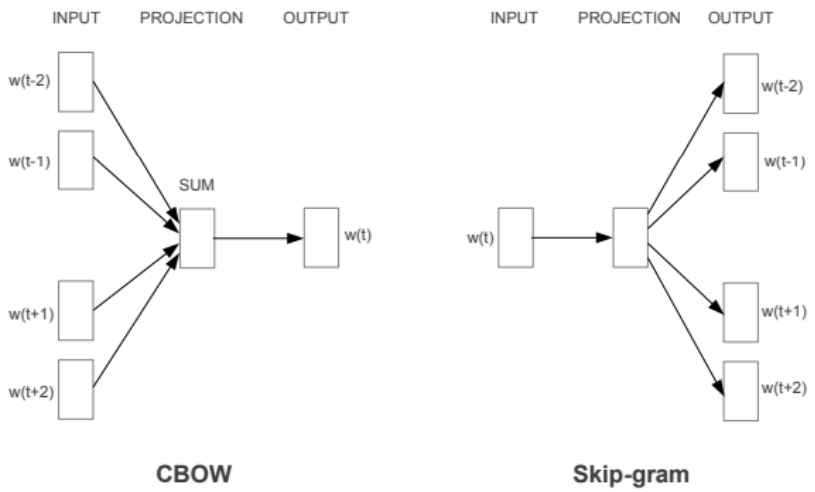


FIGURE 2.3 – Word2Vec architectures, Figure from (MIKOLOV *et al.*, 2013).

The first suggested approach is the continuous bag of words or CBOW, the left side of Figure 2.3 shows its architecture. Here the neural networks are designed to predict, given the context, which word is most likely to appear. So, words with the same probability to appear can assume a shared dimension in the words vector space.

The second approach is known by Skip-Gram, architecture at right in Figure 2.3.

Very similar to CBOW, but instead of predicting the current word the Skip-Gram uses the current word as an input to a neural network to predict its context.

After the network training process, we can use the hidden layer weight matrix as a lookup table to build the word embedding representation. The dimension for the vector space is managed by the number of neurons in the hidden layer.

2.1.2.2 Global Vectors for Word Representation

Just a year later Pennington *et al.* (2014) arrives with a new approach to represent words in a vector space. The Global Vectors for Word Representation, or GloVe, method emerged by the need to consider some factors ignored by Skip-Gram.

Methods such as Skip-Gram learn their embedding by targeting words to their respective context, ignoring the fact that some words appear more in a context than others. Thus, this co-occurrence of words only adds more useless training examples, increasing the training complexity without adding relevant information.

GloVe, however, proposes to use the corpus statistics more efficiently. Using a weighted least squares model trained on a global word-word co-occurrence counts matrix. Thereby, it is possible to build a lookup table for the words in vocabulary and use it to represent them in a vector space.

2.1.2.3 Word Vectors with Subword Information

Both Skip-Gram and GloVe provide a good vector representation for words, but there still is an unsolved problem, What to do with unknown words? To solve this question was proposed a new embedding technique which uses subword units to build a vector space, (BOJANOWSKI *et al.*, 2017).

Similar to Skip-Gram, this new method, the FastText, train its embedding by using a target to predict the context. However, instead of using the full words FastText goes a level deeper, breaking the words in n-grams, i.e., the word becomes its own context. The Figure 2.4 shows how the word "apple" can be broken into n-grams.

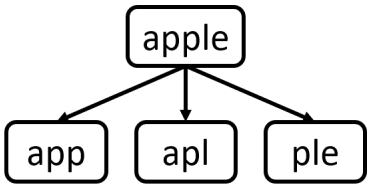


FIGURE 2.4 – Tri-gram representation for "apple" word.

There are a couple great advantages by using this method. It is now possible to

generalize new words, or unseen in training data, since they have the same characters as known ones. Although it is possible to use available pre-trained models, the FastText requires less text to be trained, it can extract much more information from small pieces of text.

2.1.3 Topic Modeling

In text-mining, we often have document collections that we want to split into similar groups or even classify then related to each other. In this way, topic modeling is an unsupervised classification tool frequently used in text-mining to identifying the hidden patterns, called "topics", in this document collection which carries consistent semantic meaning.

Clustering methods, like hierarchical clustering, can be used to group the document collection into similar clusters. However, with a large amount of data a simple document clustering is not enough, because the semantic level of topics is not taken into account. To accomplish the topic modeling task a more sophisticated method is required, so methods like Latent Dirichlet allocation.

Latent Dirichlet allocation, or LDA for short, is a probabilistic model that uses a Dirichlet distribution to model both the topics and the words. In LDA, each item in the collection is modeled as a mixture over an underlying set of topics. And the topics, in this way, is modeled as a mixture of words over the vocabulary, (BLEI *et al.*, 2003).

Without going too deep in the math behind the model, we can easily understand the two most basic principles that guide LDA. The first one said the documents are a mixture of topics which means that in a topic space each document could be interpreted as a linear combination of these topics, however, each one of the documents must be the most homogeneous as possible. The second principle said the topics are formed as a combination of words, i.e., documents about the same topics must share similar words, and also words distribution must be the most homogeneous as possible.

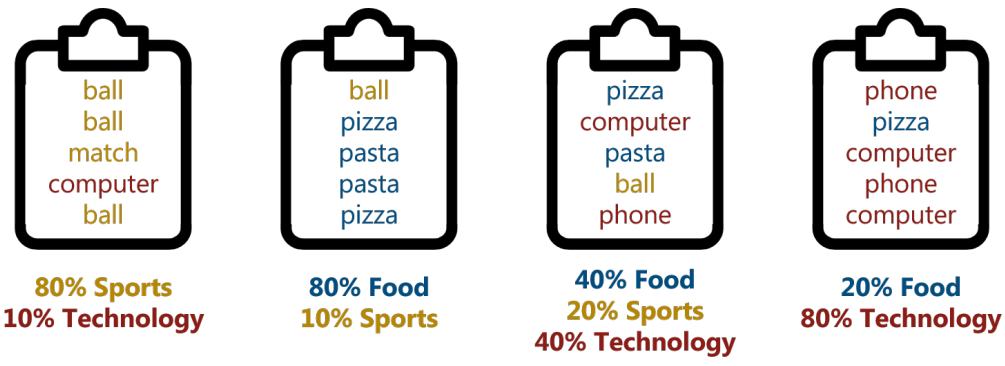


FIGURE 2.5 – Document collection modeled by topics.

Take the Figure 2.5 as an example. It shows a little fictitious document set compounded for six words “ball” and “match” which can represent the Sports Topic, “pizza” and “pasta” for te Food topic end, finally “computer” and “phone” for technology topic. In this case, each word belongs to one topic, but that is not a rule, each topic is formed by a mix of words and the documents a combination of topics.

A final remark for the topic modeling algorithms is that the topic is not discovered by names, the machine only discovers the distribution of the words by topics but not which one is. Its human work to label the topic once they are found.

2.1.3.1 Generative Process

To better understand this model, we will analyze the generative process for creating the document collection illustrated by Figure 2.6. Let’s assume we have specified who the available topics are and their respective word distributions (far left), besides to the topic proportions for each of the documents in the collection (the histogram at rigth).

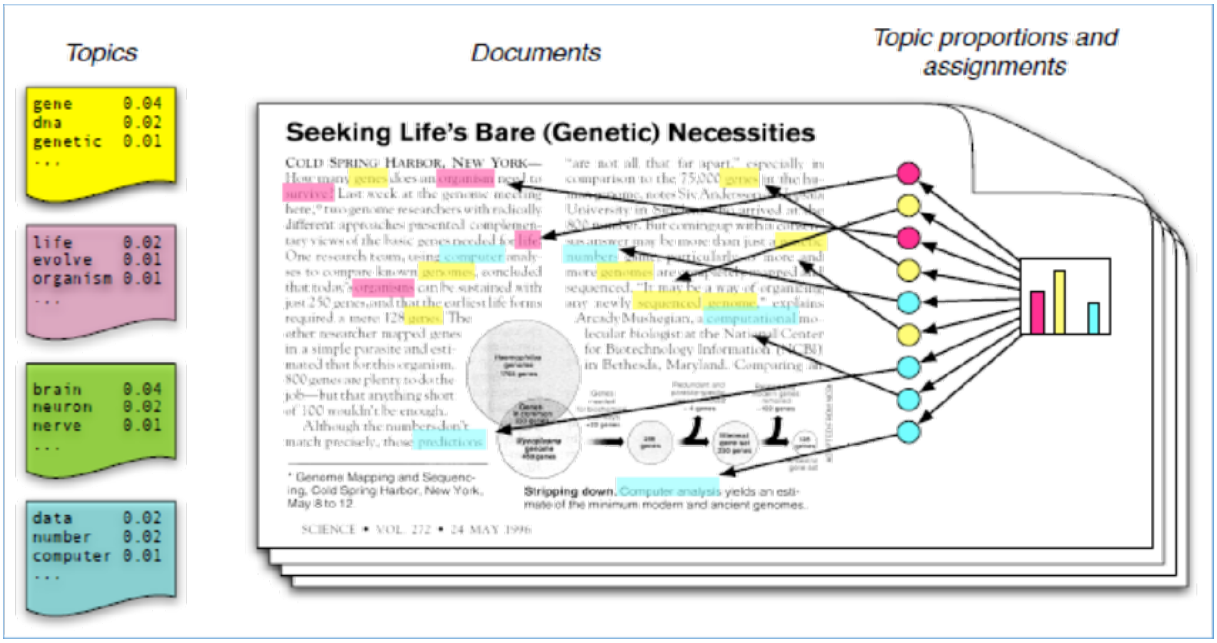


FIGURE 2.6 – The intuition behind LDA Generative process, Figure from (BLEI, 2012).

To generate a single document we choose its words as follow. Randomly choose a topic assignment from the topics distribution, then we choose the word from the corresponding topic. As shown by Figure 2.7 we can describe the generative process for a given α and β parameters.

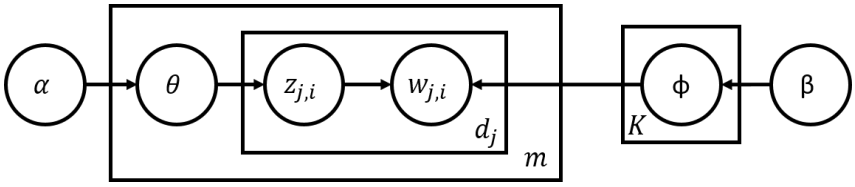


FIGURE 2.7 – Mathematical representation for LDA Generative process.

With that we can model the generative process by:

$$p(w|\alpha,\beta) = \prod_{k=1}^{K} p(\phi_k|\beta) \prod_{j=1}^{M} p(\theta_j|\alpha) \left(\prod_{i=1}^{V} p(z_{j,i}|\theta_j) p(w_{i,j}|z_{i,j}, \phi_{z_{j,i}}) \right). \tag{2.2}$$

Where ϕ is the distribution of V words over the K topics and θ is the topic distribution over the M documents. The term inside the parentheses represent the choice of a word w for a document given the assignment of topics z for the specific document.

2.1.3.2 Model Inference

The inference process for LDA consists on findind the best fit of the distribution ϕ, for words over topics, the distribution *theta*, for topics over documents, and the assignment z for each document in the collection given the words vectors representation, usually bag of words, and the hyperparametes α and β, who are the Dirichelet paramentes for respective θ and ϕ distributions.

Given the expensive computational cost, we can use some techniques to infer the LDA, the most common one is the Gibbs Sampling. This method is an algorithm for obtaining a sequence of observation which are approximated from a specified multivariate probability distribution. In easy words, showed by Faleiros *et al.* (2016), the Gibbs sampler will iterate over the documents and its words to compute the relations:

$$\theta_k = \frac{C_{w,k}^{NWZ} + \beta_w}{\sum_{w_{-i}} \left(C_{w_{-i},k}^{NWZ} + \beta_{w'} \right)}, \tag{2.3}$$

$$\phi_d = \frac{C_{d,k}^{NZM} + \alpha_k}{\sum_{k'}^{K} \left(C_{d,k'}^{NZM} + \alpha_{k'} \right)}. \tag{2.4}$$

Where $C_{w,k}^{NWZ}$ is the counter of times a word w is assigned to topic k; and $C_{d,k}^{NZM}$ is the counter of times a topic k is assigned to a word in a document d. The Algorithm 1 shows in a simple way how the Gibbs sampling works to infer the LDA distributions.

Algorithm 1 Gibbs sampler

input: number of topics K, document collection, α and β
begin
    Initialize the counters randomly;
    while Did not converge **do**
        for document $m \in [1, D]$ **do**
            for word $n \in [1, N_m]$ in document m **do**
                $nwz[w_{m,n}, z_{m,n}] - -$; $nz[z_{m,n}] - -$; $nzm[z_{m,n}, m] - -$;
                Sample a topic $z_{m,n}$ according to the previous equation
                $nwz[w_{m,n}, z_{m,n}] + +$; $nz[z_{m,n}] + +$; $nzm[z_{m,n}, m] + +$;
            end for
        end for
        if converged L since the last sampling **then**
            calculate the distributions θ_k and ϕ_d;
        end if
    end while
end

2.2 Machine Learning

The basics behaviour for a Machine Learning (ML) algorithm consists in building mathematical models based on data, usually known by training data, to provide predictions without being explicitly programmed to this, i.e., just knowing the training data, the model learns to make new predictions.

When the entries for our training data have a label, used to guide the learning algorithm, we are facing a supervised learning method. Digging a little deeper, our problem could consist in classification, when the labels are restricted to a limited set of values; or regression, when the labels could have any numerical value in range.

2.2.1 Classification

As a usual ML problem, a classification task consists in building a model to fit some training data set. In a classification model we make predictions for the class of given data points, when we have only two classes, let's call then by the positive and negative, we are facing a binary classification problem.

There are a several algorithm that we can use to build a classifier, since the simple k-nearest neighbor until the more sophisticated ones like decision trees, naive bayes, support vectors machines and artificial neural networks. Although each of these methods has its particularities in their formulation, they are capable of fitting the training data to make new predictions based on them.

2.2.2 Evaluating a classifier

After the training step we need to evaluate our classifier to verify how good our model is fitting the training data. The easiest way to evaluate our model is by reserving a split of the training set, the holdout set, exemplified in Figure 2.8. So, we use this unseen data to test the model and compute a metric.

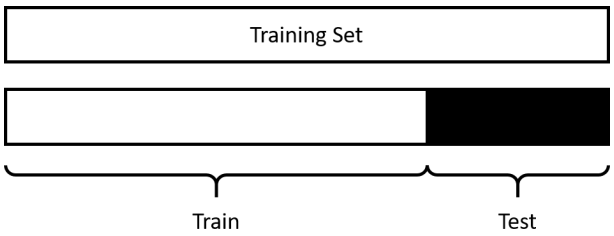


FIGURE 2.8 – Train test split for holdout method.

Another way to evaluate the model is through the k-fold cross-validation method, shown in Figure 2.9. Dividing the data set into k folds with the same size, we can train the model with $k-1$ folds and use the remaining one to test. Then we repeat this process to each fold and average the metric to evaluate the model.

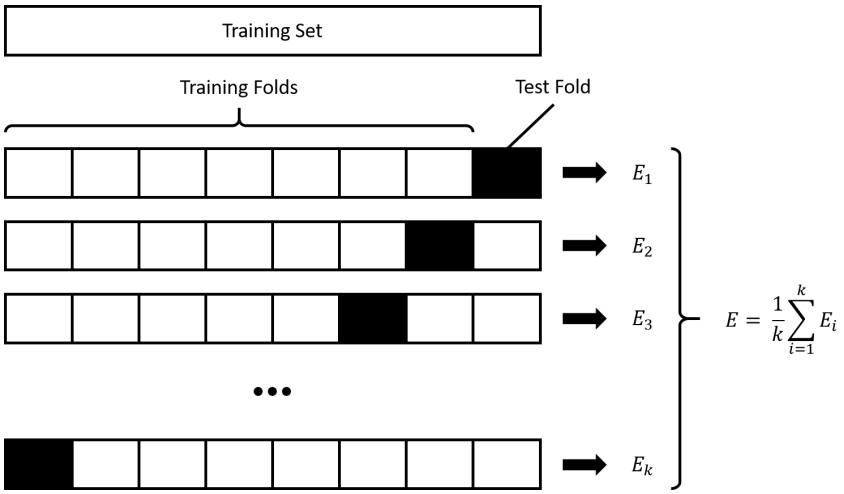


FIGURE 2.9 – Cross-validation process.

2.2.2.1 Metrics

Given the evaluation techniques we still need a metric to properly evaluate the model. To present the main metrics, let's first take a brief look in the confusion matrix. The Table 2.1 exemplifies the confusion matrix for a binary classification, each row represent the entries in a predicted class and the columns represent the actual class for the entry points.

TABLE 2.1 – Binary classification confusion matrix

		Actual Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Thus, we can have the true positives (TP) and true negatives (TN) when our model makes a good prediction about the positive and negative classes, respectively, when the model misses the prediction we call false positive (FP) or false negative (FN).

1. **Accuracy:** The simplest metric, extracted from the confusion matrix, show us the percentage of correct predictions by the relation:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}. \tag{2.5}$$

2. **Precision:** For some cases, like a very imbalanced scenario, the accuracy is not good because the even if the model predicts all the instances by the predominating class we will still have a high accuracy. We define the precision for the positive class by:

$$Precision(P) = \frac{TP}{TP + FP}. \tag{2.6}$$

   Thus, we will analyze the true predictions only by the positive predicted class.

3. **Recall:** This metric represents the fraction from a class which are correctly predicted. We define the recall for the positive class by:

$$Recall(P) = \frac{TP}{TP + FN}. \tag{2.7}$$

   Similar to precision the recall for the positive class will compare the true predictions by the total count of true actual classes.

4. **F1 Score:** In some cases we may want give the same importance to precision and recall. A popular metric to combine then is called F1-score, which is the harmonic mean of precision and recall. We define the F1-score for the positive class by:

$$F1(P) = \frac{2 \times Precision(P) \times Recall(P)}{Precision(P) + Recall(P)}. \tag{2.8}$$

3 Related Works

In the last chapter, we saw the theoretical foundation of NLP techniques, topic modeling and binary classification. In this chapter, we will review in the literature some works that employ the NLP techniques described to discover topics in a data set. In addition, we will show some applications for this type of task. And, finally, some final remarks to continue this work.

3.1 Topics Discovery

Finding meaningful topics in a document collection has been used for many authors for the most various applications. For example, Hurtado *et al.* (2016) use topic modeling to inspect research publications, patents, and technical reports aiming to model the evolution of the direction of research and forecast the near future trends in IT industry.

Using the titles and abstracts of a data set with more than six thousand academic papers between 2002 and 2010, mostly collected by Tang *et al.* (2008), they proposed a sentence-level association rule to discover meaningful topics. After categorizing the documents in topics, they were capable of building time series for each found topic, marking how many times that topic was cited in a given year. So, they were able to build an ensemble of forecasters to study the patterns and relationships among topics over the years.

For a better understanding, Figure 3.1 has a flowchart with their proposed framework for the topic discovery and forecasting.

This framework involves some well-known major steps of NLP processing. First, they convert the documents into a transactional form, i.e., the phrases in each document will be considered individually during the process. Next, they perform the basic normalization steps which include case conversion, tokenization, removing step words, part of speech tagging, stemming and lemmatization. It is also performed an additional step, specific to their application, removing verbs such as “exploiting”, “adapting” and “propose”, because they are very common in scientific publications and do not add much meaningful

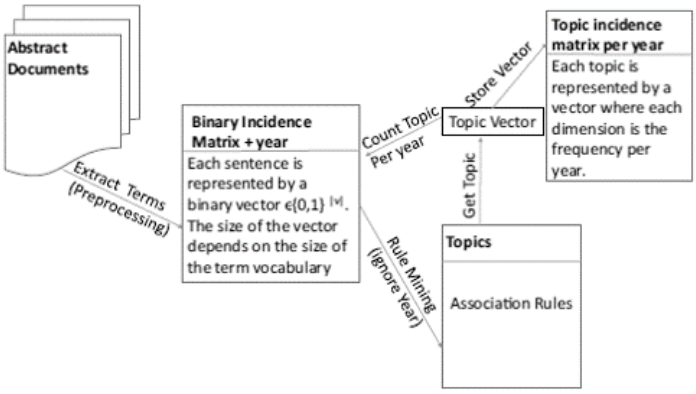


FIGURE 3.1 – Flowchart of the proposed framework, Figure from (HURTADO *et al.*, 2016).

information.

To vectorize the transactions, it is used a slight variation of BoW. Instead of word counting, it is only checked whether a word belongs to a transaction, this is called the binary incidence matrix. The topic discovery step comes afterward, applying an association rule mining to the transactions and discovery their patterns. In order to avoid different topics with redundant words is applied a rule refinement process that allows similar topics to be combined.

Online forums and social media are excellent platforms for people to discuss and share information about a variety of subjects. Recently, the topic discovery technique was used to summarize different topics related to COVID-19 disease and perform a sentiment analysis on them (JELODAR *et al.*, 2020).

Reddit is a discussion website in which its users can submit posts and start discussions with other community members. The posts are organized in the called "subreddits", boards created by users to discuss a specific subject. Using over half a million comments from 10 health-related subreddits with information about COVID-19, Jelodar *et al.* (2020) performed a topic discovery to group similar comments. So, applying a sentiment analysis on each comment it was possible to summarize the average opinion about the discovered topics.

3.2 Trend Forecast

Predicting future trends can be very helpful in various applications, like to model the evolution of research. Following the topic discovery process from Hurtado *et al.* (2016), a forecast trend was used to predict the near future related to each discovered topic. With all documents belonging to at least one identified topic in the set, was created a topic incidence matrix that contains the count of times a topic is mentioned over the years.

Finally, they make an ensemble forecasting to predict the future topic counting using the framework shown in Figure 3.2.

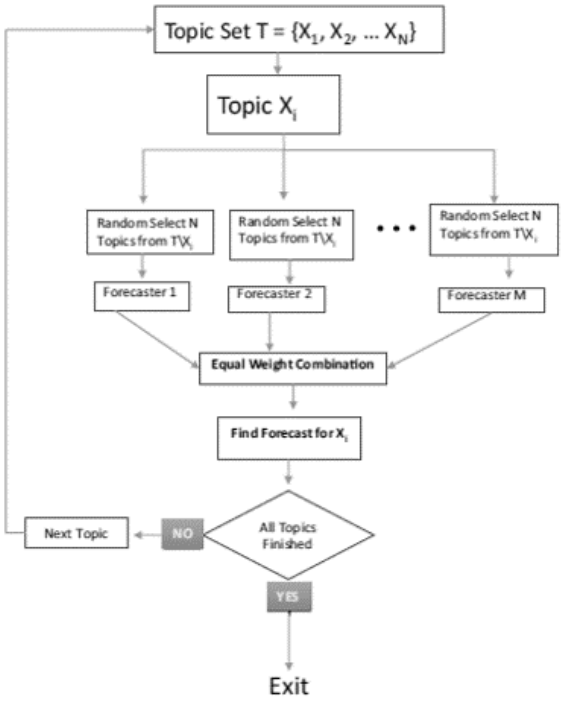


FIGURE 3.2 – Ensemble forecaster framework, Figure from (HURTADO *et al.*, 2016).

Given a specific topic, they generate M forecasters which target X_i along with N randomly chosen fields, excluding X_i. Then, the predicted value, $\hat{X}_i(t+1)$, is an average of each individual forecast, $\hat{X}_{i,F_k}(t+1)$, calculated by

$$\hat{X}_i(t+1) = \frac{1}{M}\sum_{k=1}^{M} \hat{X}_{i,F_k}(t+1). \tag{3.1}$$

By evaluating metrics like the coefficient of determination (R-squared) and mean squared error (MSE) they were able to conclude the ideal number N of variables to predict more accurately the future for all topics in the set.

Shen (2018) also predicted trends by analyzing the exponential growth in the volume of scholarly articles published over the years. However, he skips the NLP process step by using a pre-labeled data set from Springer containing the number of works above 14 subjects in 25 years. Using an ensemble forecast based on neural networks and support vector regression he was able to study the topics' growth and codependency between them.

3.3 Final Remarks

As we saw earlier topic discovery and trend forecast were subjects widely explored in the literature. With that in mind, we wish, in this work, to reproduce these techniques. However, in addition to what has been presented, we want to be able to explore some modifications.

Assuming a system that receives documents in real-time, let's assume they are news, it would be very computationally expensive to redo all the topic discovery process with each new news. It would be much simpler if there was a process that, given an input document, would be able to identify which topics were covered by it.

So, aiming at this type of application, we will propose a system capable of such activity to model the topics evolution over time.

4 Materials and Methods

In the previous chapters, we presented the motivation of this work, the literature review showing the main concepts behind work, then, we breifly described some related works. In this chapter, we will describe the problem, the whole experimentation setupp and the tools used to accomplish the proposed objectives.

4.1 Database

The first necessary task is to find a database spread over several years. For this, the Neural Information Processing Systems database was chosen, (NIPS, 2020). With 7241 entries spread in a thirty-years range, this base contains academic papers from the NIPS conference, one of the most prestigious yearly events in the machine learning community. The Table 4.1 contains the description of the main columns present in the data set.

TABLE 4.1 – Main feature description for NIPS data set.

Feature	Description	Data Type
id	Paper identifier	Integer
year	Year of publication	Integer
title	Title of publication	Text
abstract	Publication abstract	Text
paper_text	Publication corpus	Text

4.2 Pre-processing the Data

With the chosen database, we must define a data pre-processing pipeline to normalize the documents, putting all of then at the same pattern. For this, we can use the normalization techniques shown in Chapter 2 like stemming, lemmatization, stop words removal, and all necessary textual manipulations to obtain the best text representation for the documents.

4.2.1 Normalization Pipeline

To be more specific about the normalization process, let us describe it in more details. First of all, we must concatenate the title, the abstract and the paper text to treat all the paper information like a single document in our corpus. Done that, we proceed with the other normalization steps in the order as follows.

1. **Drop links:** drop all possible links, by regular expression, in the text that could might disturb the others steps;
2. **Remove numbers:** remove all numbers, by regular expression, in the text;
3. **Expand contractions:** Convert some english contractions into their full form;
4. **Remove punctuation:** Replace all punctuation marks to empty space;
5. **Convert special characters:** Replace special characters, accented letters for example, with their ASCII form;
6. **Case convertion:** Change all characthers by thei lower forms;
7. **Lemmatization:** Convert the words to their root form by lemmatizing them according their part of speech tag;
8. **Remove stop-words:** Remove the english well-known stop words.

Figure 4.1 illustrates an example of the application of the above mentioned language processing techniques to a given input.

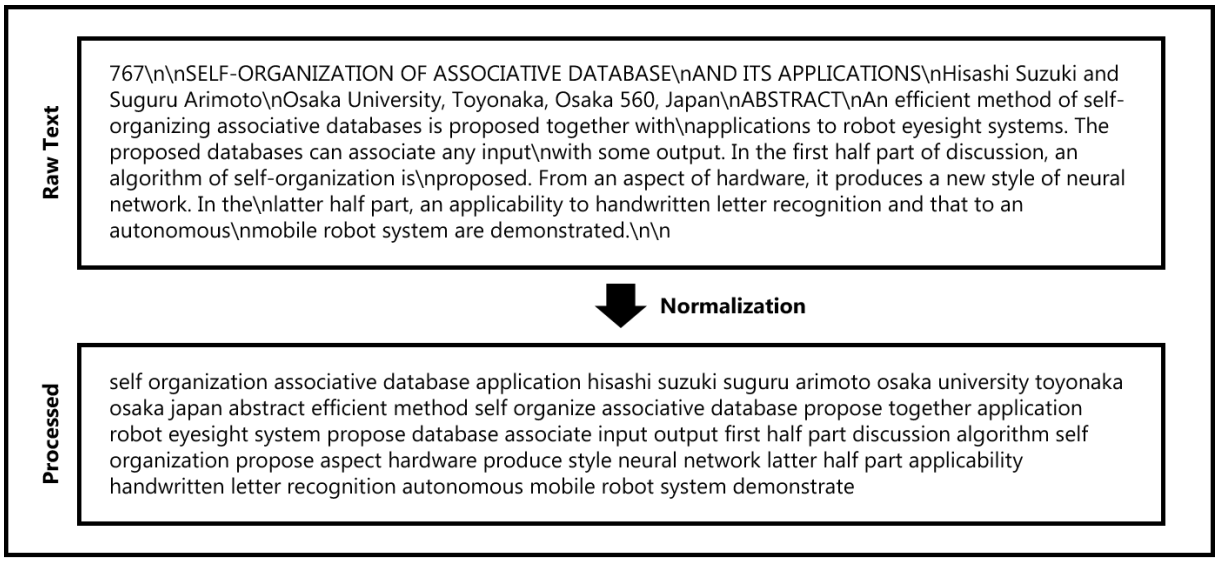


FIGURE 4.1 – Demonstration about the normalization process.

4.2.2 Segmentation

After processing the data, we need to index then over the time and, then, split the full treated data set in three subsets. They must be time ordered, as shown in Figure 4.2, this means that for each document in T_i its time index $t_x^{(i)}$ must be lower then any index $t_x^{(m)}$ in a document contained in T_m, and so on.

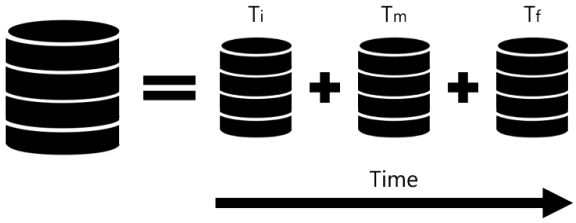
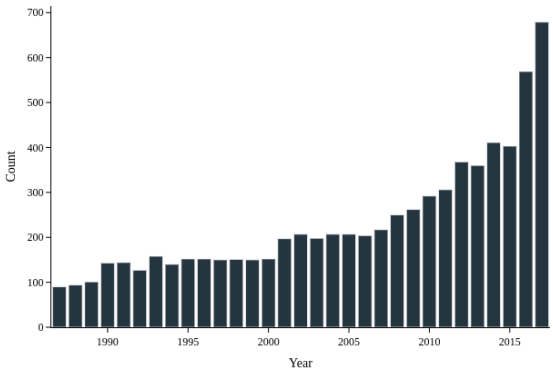
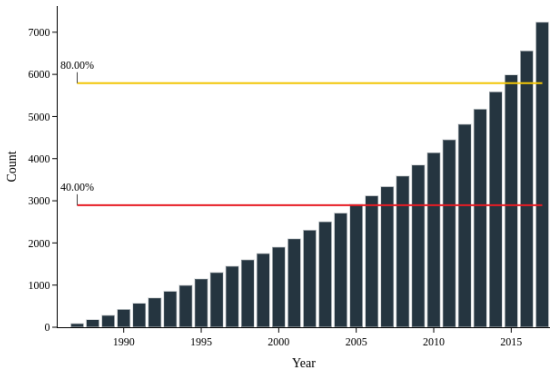


FIGURE 4.2 – Database time split representation.

Before proceeding, let us give a name to the subsets. The initial subset will be called *Past* from now on, the middle and final ones will be respectively designated by *Present* and *Future*. To segment these three subsets, let us first analyze the yearly distribution of documents. Figure 4.3a shows how the documents are spread over the years, and Figure 4.3b shows the cumulative distribution of documents with the percentiles marks of 40% and 80%.



(a) Histogram for year distribution.



(b) Cumulative histogram.

FIGURE 4.3 – Yearly distribution of documents.

After this division, shown in Figure 4.3, we approach the limits to 2004 and 2014. Then, the temporal division for the subsets was done as indicated in the Table 4.2.

TABLE 4.2 – Subsets description after segmentation.

Subset	No. of documents	Years
Past	2713	1987 - 2004
Present	2877	2005 - 2014
Future	1651	2015 - 2017

4.3 Topic Modeling

With the *Past* set, techniques of topic modeling will be applied to identify the discussed subjects in the documents. Figure 4.4 illustrates the sequence of steps for this task. Next to the topic identification we will obtain a topic set, then each document contained in *Past* can be labeled with at least one topic from the set, this new database will be called by *labeled Past*.

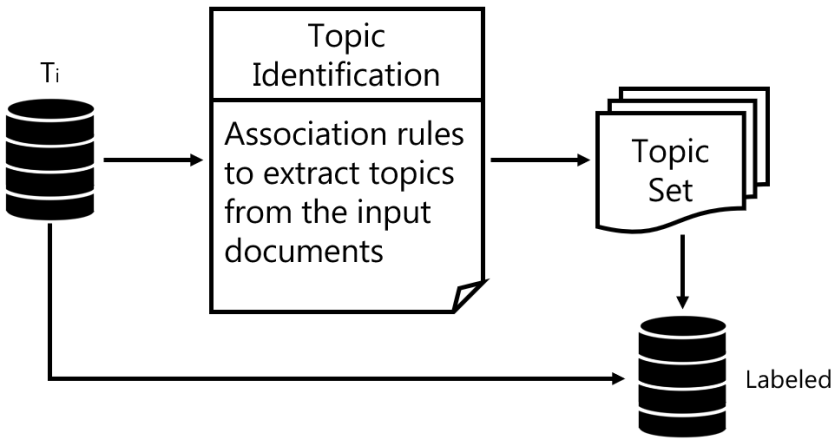


FIGURE 4.4 – Topic identification process.

4.4 Document Classification

For each topic in our set of discovered topics, we must be able to identify which topics are covered by a new document. Thus, we will build a classifier to perform this verification. Knowing that a document can talk about several topics, so we must have a multi-class classifier. We can see this as an individual binary classifier for each topic that tells us whether the document has it. Using the *labeled Past* set it is possible to build this classifier. Figure 4.5 illustrates it in detail.

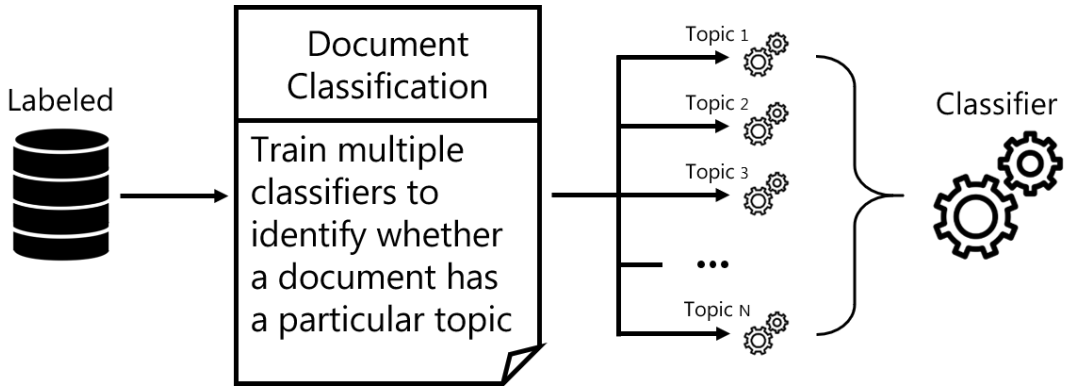


FIGURE 4.5 – Multi-class classifier from *Labeled* set.

4.5 Tools

In order to execute this steps, some packages and tools were used. Let's briefly describe these tools used so far.

5 Results and Discussions

In the last chapter we presented the methodology behind experimentation. In this chapter, we will show the main results obtained by the topic modeling and classification steps.

6 Conclusions, Recommendations, and Future Works

Bibliography

BLEI, D. M. Probabilistic topic models. **Communications of the ACM**, ACM New York, NY, USA, v. 55, n. 4, p. 77–84, 2012.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. **Journal of machine Learning research**, v. 3, n. Jan, p. 993–1022, 2003.

BOJANOWSKI, P.; GRAVE, E.; JOULIN, A.; MIKOLOV, T. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, v. 5, p. 135–146, 2017. ISSN 2307-387X.

FALEIROS, T. d. P.; LOPES, A. d. A. *et al.* Modelos probabilísticos de tópicos: desvendando o latent dirichlet allocation. São Carlos, SP, Brasil., 2016.

HURTADO, J. L.; AGARWAL, A.; ZHU, X. Topic discovery and future trend forecasting for texts. **Journal of Big Data**, Springer, v. 3, n. 1, p. 7, 2016.

JELODAR, H.; WANG, Y.; ORJI, R.; HUANG, H. Deep sentiment classification and topic discovery on novel coronavirus or covid-19 online discussions: Nlp using lstm recurrent neural network approach. **arXiv preprint arXiv:2004.11695**, 2020.

KANNAN, S.; GURUSAMY, V. Preprocessing techniques for text mining. **International Journal of Computer Science & Communication Networks**, v. 5, n. 1, p. 7–16, 2014.

KHURANA, D.; KOLI, A.; KHATTER, K.; SINGH, S. Natural language processing: State of the art, current trends and challenges. 08 2017.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

NIPS. **Neural Information Processing Systems Foundation Inc**. 2020. Accessed: 2020-10-13. Disponível em: <https://papers.nips.cc/>.

PENNINGTON, J.; SOCHER, R.; MANNING, C. Glove: Global vectors for word representation. In: **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1532–1543. Disponível em: <https://www.aclweb.org/anthology/D14-1162>.

SHEN, Q. Topic discovery and future trend prediction in scholarly networks. In: . Shanghai, China: [s.n.], 2018.

TANG, J.; ZHANG, J.; YAO, L.; LI, J.; ZHANG, L.; SU, Z. Arnetminer: extraction and mining of academic social networks. In: **Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2008. p. 990–998.

VIJAYARANI, S.; ILAMATHI, M. J.; NITHYA, M. Preprocessing techniques for text mining-an overview. **International Journal of Computer Science & Communication Networks**, v. 5, n. 1, p. 7–16, 2015.

FOLHA DE REGISTRO DO DOCUMENTO

1. CLASSIFICAÇÃO/TIPO	2. DATA	3. DOCUMENTO N°	4. N° DE PÁGINAS
TC	July 7th, 2020	DCTA/ITA/DM-018/2015	39

5. TÍTULO E SUBTÍTULO:
Natural Language Processing for Trend Forecasting

6. AUTOR(ES):
Heládio Sampaio Lopes

7. INSTITUIÇÃO(ÕES)/ÓRGÃO(S) INTERNO(S)/DIVISÃO(ÕES):
Aeronautics Institute of Technology – ITA

8. PALAVRAS-CHAVE SUGERIDAS PELO AUTOR:
Natural Language Processing; Machine Learning

9. PALAVRAS-CHAVE RESULTANTES DE INDEXAÇÃO:
Natural Language Processing; Machine Learning

10. APRESENTAÇÃO: (**X**) **Nacional** ( ) **Internacional**
ITA, São José dos Campos, 2020. Trabalho de Graduação. 39 páginas.

11. RESUMO:
Resumo

12. GRAU DE SIGILO:
(**X**) **OSTENSIVO** ( ) **RESERVADO** ( ) **SECRETO**