Artificial Intelligence-Based Text Classification: Separating Human Writing from Computer Generated Writing

Luis Enrique Morales-Márquez [1], Erick Barrios-González [1], David Eduardo Pinto-Avendaño [1]

[1] *Autonomous University of Puebla, San Claudio Av., 14 Sur Blvd., Puebla, ZIP Code: 72592, México*

Abstract

In recent years, linguistic computational models have advanced to the point where they can generate stories and hold conversations. Despite being a useful tool, computer generated texts can be used for unethical purposes, which underscores the need to identify these texts. This paper presents a model that classifies human- or computer-generated texts, using vocabulary richness metrics and POS label ratios to train a simple artificial neural network for spanish classification, and some others features to build a Naïve Bayes Model for english classification. The objective is to classify texts in both English and Spanish. The results show a Macro F_1 of 0.67 for the texts in English and 0.6441 for the texts in Spanish. Therefore, the performance of this model is consistent with that achieved in previous research.

Keywords

Computer Generated Texts, Texts Classification, Machine Learning

1. Introduction

Artificial intelligence (AI) models have undergone rapid development and evolution in the last decade. Not surprisingly, an increase in computing power, fueled by today's infrastructure, has been a key for a rapid growth in the building and application of AI models to fulfill their purpose of mimicking human behavior and reasoning.

One of the fields that has received special attention is the development of Linguistic Models (LM), and in particular, those responsible for the automatic generation of texts, known as Text Generative Models (TGM). These seek to reproduce the writing patterns of humans, imitating the same grammatical structures, fluency, terminology, and contextual support.

TGM applications range from automatic story generation to complex tasks that require logical reasoning, such as holding a conversation or completing source code for programming. However, these models can also be used for unethical purposes, such as the generation of false news, offensive comments or incitement to violence, as well as the generation of spam texts that can benefit or harm personalities or commercial entities [1].

Given the speed and ease with which information is disseminated today, the detection of synthetic and false texts is of the utmost importance. This task is very difficult, so current efforts are mainly focused on the detection of documents generated automatically by TGM models, such as Chat-GPT or Google Bard. The fundamental premise of detection is that the generating models create texts with grammatical errors, contradictions, and redundancy or repetition of ideas.

However, the identification of these errors in writing continues being a difficult task, which is why the behavior of TGMs reflected in the generated texts has been studied to determine patterns that can characterize non-human texts [2].

This document aims to propose a model for the detection of computer generated texts. The structure of the article is as follows: Section 2 presents related works with the same purpose; Section 3 details

Proceedings IberLEF, September 2023, Jaén, Spain
EMAIL: luise.morales@viep.com.mx (A. 1); erick.barrios@viep.com.mx (A. 2); david.pinto@correo.buap.mx (A. 3)
ORCID: 0000-0001-7699-5372 (A. 1); 0000-0002-2077-7541 (A. 2)


IberLEF 2023, September 2023, Jaén, Spain

the proposed method and the associated theoretical concepts; Section 4 presents the results obtained and their analysis; and finally, Section 5 contains the conclusions.

2. Related Work

Below are some recent proposals for the classification of texts generated by humans or by some TGM.

Gehrmann, Strobelt & Rush (2019) developed Giant language Model Test Room (GLTR), a computer-generated text detector based on text statistics. The central idea is that the generated texts are written based on a very limited set of language distributions. The tests use the probability of occurrence of a word given that other words are already written, the rank of a word, and the entropy of the distribution model predicted by the text generator. By calculating the rank and entropy of each text, they showed that these metrics are higher in human texts, thus obtaining a 72% hit rate in text detection [3].

Ippolito et al. (2020) collected 250,000 texts generated by the GPT-2 model and an additional 5,000 for testing. They proposed a refined variant of the BERT linguistic model and obtained approximately 70% accuracy with prior knowledge of how GPT-2 generates texts. By using decoding strategies, which consist of selecting the next word to write based on probability distributions, they determined that knowing the generation process has a large impact on the performance of the classifier [4].

Kirchenbauer et al. (2023), instead of proposing an AI-generated text detection model, suggest the idea of watermarking TGMs without retraining them. The water-mark consists of the use of certain words that must appear when generating a text, appearance determined by existing probability models. The selected words are those that present a greater entropy in the texts generated by the model, and must have a much higher frequency of use to allow the detection of the watermark. Although this idea is subject to the weaknesses of the watermarking mechanisms of any other multimedia element, it is a potential idea that, applied correctly, may allow detection in the future [5].

Some researchers find the task so difficult that they prefer to concentrate on detecting texts written by the same model. This is the case of Gritsay, Grabovoy & Chekhovic (2022), who developed a model based on ROBERTA for the similarity detection of texts generated by the same TGM GPT-2. The results show that a large number of tokens and very wide windows are required to improve the results, although this may result in overfitting if it is decided to analyze very wide neighborhoods for the text tokens. They report accuracy of up to 97% in detection [6].

No related works have been found that have the objective of detecting computer-generated texts in Spanish, so it will be one of the tasks addressed in this work.

3. Proposed Method

The used corpus is the corpus provided in *AuTexTification* subtask 1 from IberLEF 2023 [9], this dataset contains 33845 texts in english for training and 21832 for tests and contains 32062 texts in spanish for training with 20129 for test. Two specialized models were implemented for english and spanish, the methods will be described below.

3.1. Subtask 1 - English

The implementation oriented to the English language, is based on a naive bayes model. The Naive Bayes algorithm is frequently used in binary classification, it is used as a base model to compare the results of new models [7, 8].

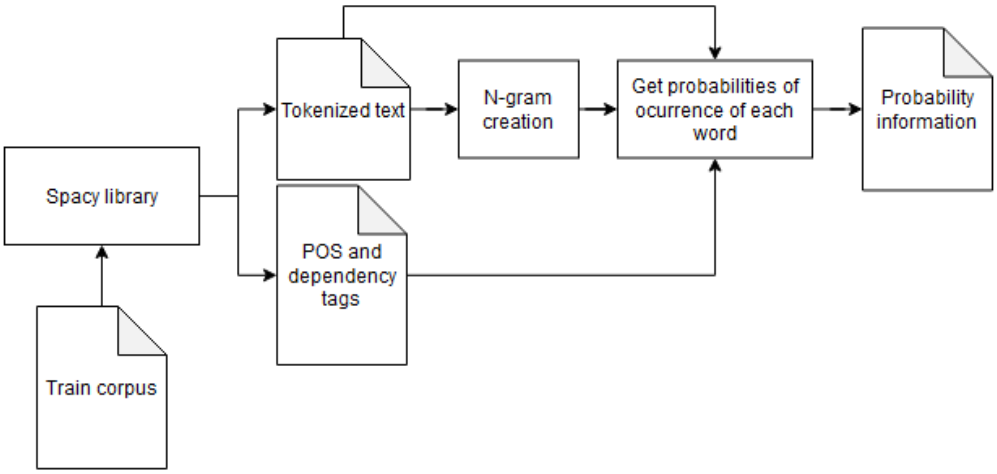

Figure 1: Pre-processing of text

To implement this model, a pre-processing is necessary. This process consists of using the library provided by Spacy to tag the texts in English. The tags that interest us are those of part of speech and the dependency. Once the tags have been created, we create n-grams of the tokenized words (not of the tags), in the end both the words, as well as the n-grams, and the tags are counted to see their occurrences by class and measure their frequency of occurrence. This information will be used to later calculate the probabilities of occurrence. In Fig. 1, this process is observed.

In the pre-processing process we obtain information related to the frequency of appearance of words, n-grams (N-gram to 2-grams, 3-grams and 4-grams), POS labels and dependency labels, this same information was evaluated implementing a naive bayes model, to choose the best characteristics a search of grid was done to explore the best combination of data.

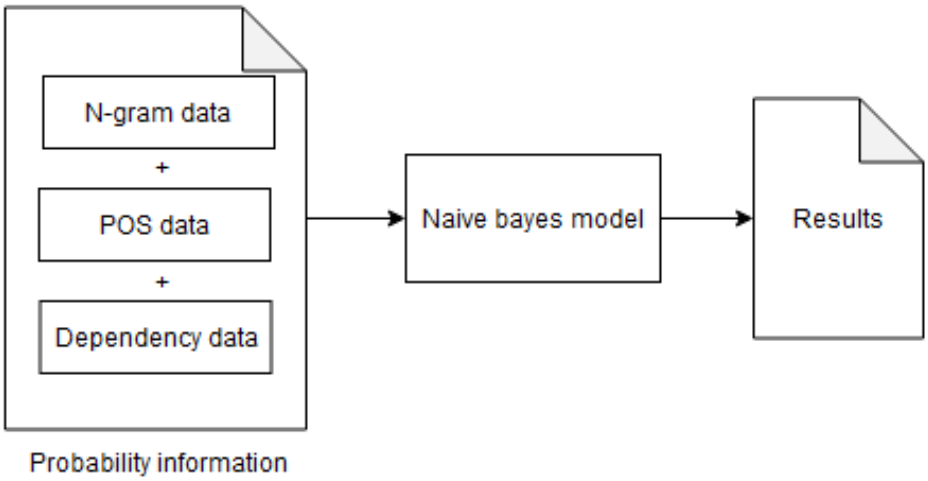

Figure 2: Input for Naive Bayes model

In Figure 2 there is an example of how the input to the naive bayes model would be, the final features selected were the frequencies of appearance of the following elements:

- N-gram data: 3-gram words (combinations of 3 words in the text).
- POS data: Individual tags for verbs (VB, VBN, VBD), adverbs (RB, RBR, RBS, WBR, RP), nouns (NNP), punctuation (LS) and spaces (_SP).
- Dependency data: Index of token head dependency (the position with respect to a sentence of the token heads of the dependency relations).

3.2. Subtask 1 - Spanish

It is decided to use statistical characteristics for the analysis of the texts. As demonstrated in [3], identification based on these metrics is possible. The method proposed here is based on the idea that a TGM will tend to use certain types of words more than others. This same rule applies to the use of bigrams, it is expected that computational models will tend to use some type of bigram mostly. In the same case with the richness of the vocabulary, naturally the generated texts could have a lower than human richness, depending on the texts with which the model that writes them has been trained.

AuTexTification task [9], belonging to the 2023 edition of IberLEF has a useful dataset, it consists of a corpus of texts in English and one of texts in Spanish, both corpus with the instances labeled as human and generated by computer. The method is applied to the texts of both languages. In total, 20,129 texts in Spanish were used for tests and 21,832 in English, for training 33,845 texts in English and 32,062 in Spanish were used.

First, punctuation marks are removed to preserve only the words or tokens that compose the text. At this point, stop words are eliminated , that is, those high-frequency words that do not add any value to the text such as articles, prepositions, pronouns, etc. [10].

Subsequently, the prevailing text is tagged using Part -of- Speech tagging (POS tagging), in this process each token is tagged within a morphosyntactic category such as noun, adjective, personal pronoun, etc. [11]. The purpose is to obtain information on the structure of the sentences that are present in the text. The SpaCy library for Python performs this procedure, recognizing 20 different tags. An example of POS tagging made in the parts-of-speech.info site can be seen in Figure 3

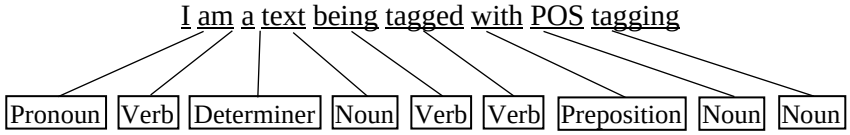

Figure 3: POS tagging of a sentence

Since the texts have different lengths in terms of the number of words, it would be incorrect to rely on the number of POS tags of each class. Instead, the proportion of each label in the text is determined using the following expression:

$$p_i = \frac{amount\ of\ POS\ tag\ i}{amount\ of\ words}, \quad (1)$$

Where p_i is the proportion of the *i*-th POS tag, the numerator is the number of occurrences of the *i*-th POS tag, and the denominator is the total number of words in the text. For each of the *i* tags considered by the Spacy library, in this case 20, we get the first 20 features.

POS bigram ratio is also examined. Since we have 20 possible tags, when generating bigrams up to 400 different pairs can be established. The proportion of bigrams is obtained by the expression:

$$pb_{ij} = \frac{amount\ of\ POS\ tag\ bigrams\ (i,j)}{amount\ of\ bigrams}, \quad (2)$$

Where pb_{ij} is the proportion of tagged *i*,*j* bigrams for each pair of consecutive words. The numerator indicates the number of labeled bigrams *i*,*j* for each pair of consecutive words, and the denominator is the total bigrams of consecutive words. For a text with n words, there are n-1 bigrams, this procedure yields 400 additional features.

In addition, two vocabulary richness metrics are defined. The first of these is the Standardized Token-Type Ratio (STTR). This calculation determines the proportion of different tokens, unique words or types used by a given number of words [12]. The number of words may vary depending on the purpose; for this job the number of tokens from the text is used, see Equation 3.

$$STTR = \frac{amount\ of\ unique\ words}{N}, \tag{3}$$

Another useful measure is the indicator λ defined by [13]:

$$\lambda = \frac{\sum_{i=1}^{V} \sqrt{(f_i - f_{i+1})^2 + 1}}{N}, \tag{4}$$

Where N is the size of the text, f_i are the absolute frequencies in ascending order of each of the types, and V is the number of types. With these metrics, two more features are added.

Finally, the proportion of collocations will be added. It is expected that a TGM will have less use of this type of bigrams. Collocations are multiple word expressions, usually bigrams, that often go together, but none of them can be changed to a synonym without losing their meaning. For example, "red wine" cannot be substituted for "reddish wine" [14]. The NLTK library for Python is capable of placing collocations in a text. Therefore, the ratio of collocations to the number of bigrams in the text is calculated:

$$pc = \frac{amount\ of\ collocations}{amount\ of\ bigrams}, \tag{5}$$

In total, there are 423 features for each text, which are passed to a neural network for training as a classifier. The Neural Network was an architecture Feed Forward with 2 hidden layers with 423 neurons per layer, batch size of 200 elements, adaptive learning rate with value of 0.001 and momentum of 0.9, and 500 epochs, the activation function was ReLu. The diagram of the method can be seen in Figure 4.

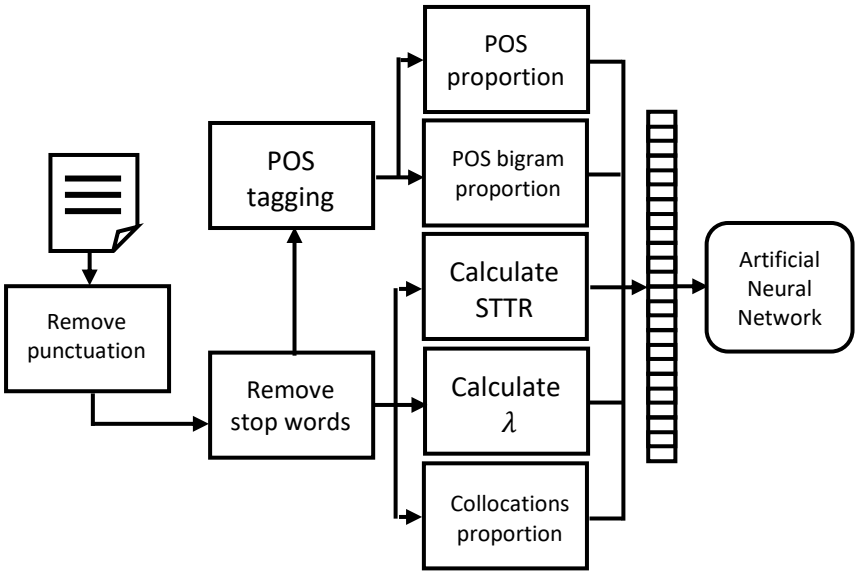

Figure 4: Diagram of the proposed method

4. Results

Two neural networks were trained, one for English texts and one for Spanish texts. The data was split into training set with 80% of the data and test set with the remaining 20%, this was applied to both languajes. The final performance metric of the classifier is the Macro F_1 Score, although precision and recall are also shown. To understand how it works, it is necessary to refer to the confusion matrix.

In a diagnostic test or binary classifier, whose results can be positive or negative, we have the class labels for each instance and it is assumed that they are correct

From which the following metrics [8] are derived: Precision: Measures how many instances predicted as positive are actually positive, it is often used when seeking to reduce the number of false positives:

$$P = \frac{TP}{TP + FP}, \tag{6}$$

Recall: Measures how many positive instances are captured by positive predictions, it is used when trying to avoid false negatives:

$$R = \frac{TP}{TP + FN}, \tag{7}$$

From where the F_1 metric is calculated as a way of combining both results:

$$F1 = 2\frac{P \cdot R}{P + R}, \tag{8}$$

These metrics assume that we are forcing a class to be positive, however it is worth checking the F_1 when the positive prediction is the human class and when it is the generated class, so the F_1 Macro is used, it's applied for multi class classification:

$$MF1 = \frac{\sum F1\ scores}{Number\ of\ classes}, \tag{9}$$

Now, the confusion matrices for the English texts are presented in Fig 6. It can be seen that there are more positive than negative instances, and the system has better identified negative instances than positive ones

		Diagnoses	
Reality		9665	1525
		5385	5257

(a)

		Diagnoses	
Reality		5257	5385
		1525	9665

(b)

Figure 6: Confusion matrices of the English text, (a) positive generated class, (b) positive human class

Confusion matrix for the texts in Spanish is presented in Fig 7:

	Diagnoses	
Reality	9105	2104
	4647	4273

(a)

	Diagnoses	
Reality	4273	4674
	2104	9105

(b)

Figure 7: Confusion matrices of the Spanish text, (a) positive generated class, (b) positive human class

Results are condensed in Table 1.

Table 1
Performance of the classifiers in both languages

Languaje	Positive Class	Precision	Recall	F_1	Macro F_1
English	Generated	0.6421	0.8637	0.7366	0.6700
	Human	0.7751	0.4939	0.6034	
Spanish	Generated	0.6620	0.8122	0.7295	0.6441
	Human	0.6700	0.4790	0.5586	

It can be seen that the classifiers have a higher precision for the human class. The recall suggests that real-world cases of computer-generated text are better covered.

Also, the F1 overall is superior when evaluating the computer generated class. The results are not given in terms of accuracy, so a direct comparison with related works cannot be made. However, the overall results of Macro F_1 indicate a modest performance that is close to the work done so far and without the need to modify deep networks. The classifier in English could result in a higher score because the SpaCy library has a more extensive corpus in that language, which could make POS tagging more appropriate than that done in Spanish.

Also, we can compare our results with the published results for the test set in AuTexTification subtask 1 [9], see Table 2.

Table 2
Performance of baseline classifiers and proposed classifiers

Classifier	Languaje	Macro F_1
Logistic Regression	English	65.78
	Spanish	62.40
Symanto Brain (Few-shot)	English	59.44
	Spanish	56.05
DeBERTa	English	57.10
	Spanish	68.52
Random	English	50.00
	English	50.00
Symanto Brain (Zero-shot)	Spanish	43.47
	English	34.58
Proposed Classifiers	Spanish	67.00
	English	64.40

It can be seen that the classifiers have a higher precision for the human class. The recall suggests that real-world cases of computer-generated text are better covered.

Also, for texts in English, it can be observed that a Macro F_1 higher by 1.22 units is obtained with respect to Logistic Regression, which is the best baseline result, in addition to exceeding Symanto Brain's 59.44 (Few-shot) and above RoBERTa's and its 57.1. On the other hand, the method for texts is Spanish obtained a Macro F_1 of 64.41, which places it below the 68.52 of RoBERTa but above the 62.4 of Logistic Regression and the rest of the baseline methods. In English, a notable improvement has been obtained with respect to the baseline classifiers, while in Spanish the result of the majority has been exceeded, being only surpassed by RoBERTa.

5. Conclusions

In this work, a text classification task was carried out seeking to differentiate between those generated by TGM and those generated by humans. Preprocessing was carried out on the texts in English and Spanish, which included removal of punctuation, stop words, and POS tagging. In addition, a series of statistical descriptors, including bigram usage and vocabulary richness, were determined and used to train a neural network. The results indicate that the model is moderately successful, reaching Macro F_1 scores of 0.67 for English texts and 0.64 for Spanish texts. A higher precision was obtained with the human texts, but a better recall with the generated ones. Also, the F_1 suggests better detection of generated texts.

These numbers show that the classifier can distinguish computer-generated texts from human texts with some reliability, although it is clear that there is a lot of room for improvement. Future works may include the use of additional features or the implementation of different artificial intelligence models.

Obtaining better results in English suggests that there may be language factors that could be explored further. In the same way, the use of other ways of text processing can be reviewed.

In general, it was shown that the use of statistical features instead of the use of pre-trained deep networks is feasible and can present better results with proper direction and methodology. The importance of artificial text detection will increase in the near future due to the rapid growth of generator models and the risks associated with their unethical use. This work represents an initial step in that direction.

6. References

[1] Jawahar, G., Abdul-Mageed, M., & Lakshmanan, L. V. S. (2020). Automatic Detection of Machine Generated Text: A Critical Survey. https://doi.org/10.18653/v1/2020.coling-main.208.

[2] Massarelli, L., Petroni, F., Piktus, A., Ott, M., Rocktäschel, T., Plachouras, V., Silvestri, F., & Riedel, S. (2019). How Decoding Strategies Affect the Verifiability of Generated Text. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1911.03587.

[3] Gehrmann, S., Strobelt, H., & Rush, A. M. (2019). GLTR: Statistical Detection and Visualization of Generated Text. https://doi.org/10.18653/v1/p19-3019

[4] Ippolito, D., Duckworth, D., Callison-Burch, C., & Eck, D. (2020b). Automatic Detection of Generated Text is Easiest when Humans are Fooled. https://doi.org/10.18653/v1/2020.acl-main.164

[5] Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., & Goldstein, T. (2023). A Wa-termark for Large Language Models. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2301.10226.

[6] Gritsay, G., Grabovoy, A., & Chekhovich, Y. (2022). Automatic Detection of Machine Generated Texts: Need More Tokens. https://doi.org/10.1109/ivmem57067.2022.9983964.

[7] AminiMotlagh, M., Shahhoseini, H. & Fatehi, N. (2023). A reliable sentiment analysis for classification of tweets in social networks. Soc. Netw. Anal. Min. **13**, 7 https://doi.org/10.1007/s13278-022-00998-2.

[8] Dimlioglu, T., Wang, J., Bisla, D. et al. (2023). Automatic document classification via transformers for regulations compliance management in large utility companies. Neural Comput & Applic https://doi.org/10.1007/s00521-023-08555-4.

[9] Sarvazyan, A. M., Gonzalez, J., Franco Salvador, M., Rangel, F., Chulvi, B., & Rosso, P. (2023). Overview of AuTexTification at IberLEF 2023: Detection and Attribution of Machine-Generated Text in Multiple Domains. In Procesamiento del Lenguaje Natural..
[10] Müller, A. C., & Guido, S. (2016b). Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media.
[11] Sammut, C., & Webb, G. I. (2017). Encyclopedia of Machine Learning and Data Mining. Springer.
[12] WordSmith Tools. (s. f.). https://lexically.net/downloads/version5/HTML/index.html?type_to%20ken_ratio_pr%20oc.htm.
[13] Fang, Y., & Liu, H. (2015). Comparison of vocabulary richness in two translated Honglou-meng. Glottometrics, 31, 54-75.
[14] Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.