# Is it Fine to Tune? Evaluating SentenceBERT Fine-tuning for Brazilian Portuguese Text Stream Classification

Bruno Yuiti Leão Imai*, Cristiano Mesquita Garcia*†, Marcio Vinicius Rocha*,
Alessandro Lameiras Koerich‡, Alceu de Souza Britto Jr.*§ and Jean Paul Barddal*
†Instituto Federal de Santa Catarina (IFSC), Câmpus Caçador, Brazil
‡École de Technologie Supérieure (ÉTS), Université du Québec, Montréal, Canada
*Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, Brazil
§Universidade Estadual de Ponta Grossa (UEPG), Ponta Grossa, Brazil
{bruno.y.l.imai,dev.marcio.rocha}@gmail.com, cristiano.garcia@ifsc.edu.br,
alessandro.koerich@etsmtl.ca, {alceu,jean.barddal}@ppgia.pucpr.br

***Abstract*—Pre-trained language models (LMs) have been used in several scenarios and data mining tasks due to their good-quality representations and their use readiness. Although LMs constitute a significant gain in usability, they are frequently utilized statically over time, meaning that these models can suffer from concept drift and semantic shift, which correspond to changes in data distribution and word meanings. These phenomena are more noticeable when new texts become gradually available. This paper evaluates the impact of updating pre-trained SentenceBERT models overtime on a Brazilian news post classification task in text streaming fashion, a paradigm suitable for learning from data streams. While we update the SBERT model yearly with a reduced number of recent posts, we compare it with scenarios using static LMs. We used the adaptive random forest for classification and evaluated it regarding macro F1-score and elapsed time. The experimental results show that regularly leveraging sampled texts from the recent past for fine-tuning LMs can improve performance metrics over time, reaching better results than using static LMs in most years analyzed. We also evaluated the run times, which suggests that fine-tuning LMs over time provides a good trade-off between performance and run time.**

***Index Terms*—text stream, text classification, incremental learning, language models, fine-tuning**

## I. Introduction

Continuously learning from texts has become a hot topic, as it can correspond to advantages for companies and institutions. Several works applied machine learning (ML) methods to incorporate patterns and knowledge from specific domains into models [1], [2]. However, in given domains, such as news classification and sentiment analysis of $\mathbb{X}$ (former Twitter) posts, the texts become available over time, and a model trained using past data may become obsolete. Therefore, the data stream paradigm emerges as a suitable and practical strategy for (nearly) real-time scenarios [3], [4].

Data streams are "an abstraction that allows real-time analytics" [3]. Streams have distinct characteristics, which include [4]: (a) data arrive one by one or in small batches, (b) data are presented sequentially, and (c) the stream is potentially infinite. To learn in this kind of scenario, a given ML model must satisfy criteria such as [4]: (a) to be able to learn one by one (or in small batches); (b) to consume modest resources; and (c) to perform single-pass operations. A class of ML methods that can learn respecting these aspects is generally called *incremental* or *adaptive* [4].

When handling textual data streams, or simply text streams, new challenges emerge, including natural language processing (NLP) and the maintenance of vocabulary and representations [5]. A frequent phenomenon in traditional and text streams is the concept drift [3]–[5]. Concept drift constitutes changes in the underlying data distribution [3], [4]. ML systems that do not account for concept drift may experience adverse effects on performance [4]. In particular, in text streams, concept drift can take different forms, such as [5]: changes in the sentiment distribution [6], and feature drift [7], for example.

Handling text representations is also challenging in text streams. Most ML methods demand numeric vectors as input. Using simple traditional methods such as bag-of-words (BOW) [8] and term frequency-inverse of document frequency (TF-IDF) [9] can be problematic because they can generate high dimensional representations. Additionally, these text representation methods, particularly TF-IDF, require processing a set of texts to score the importance of the words. Furthermore, because each dimension represents a word in both BOW and TF-IDF, applying them would vectorize the texts into variable-dimension vectors. Therefore, several works have leveraged pre-trained language models (LMs) since they are easy to use and generally offer good-quality representations [10]. However, using LMs statically can lead to semantic shift that regards changes in word meaning over time [11].

According to Garcia et al. [5], one of the manners to overcome concept drift in text streams is by updating the model occasionally after arbitrary periods. Considering the presented aspects, this paper aims to evaluate the effects of fine-tuning pre-trained LMs in text stream scenarios. In particular, we will analyze a stream of news posts in Brazilian

Portuguese from the financial domain to classify news. We use SBERT [12] combined with BERTimbau [13] as a base LM and fine-tune, over time, considering 2000 news from the past year, having in mind the text stream paradigm and the temporal order, e.g., fine-tuned SBERT with news from 2018 and applied on news from 2019. We also evaluate the effect of LMs older than one year, using the text stream's instances' timestamp as a reference, e.g., fine-tuned SBERT with news from 2018 and applied to news from 2020 or posterior. Specifically, our research question RQ1 is: "How effective is recurring fine-tuning LMs for continuous use in Brazilian Portuguese text stream classification?". To illustrate this scenario, we aim to classify the news category; however, it could be relevant for any other class prediction, e.g., sentiment analysis.

The contributions of this paper are three-fold: (1) an analysis of the effects of fine-tuning LMs over time in text stream classification; (2) a set of annual (between 2018 and 2023) fine-tuned SBERT models in Brazilian Portuguese, a language that is mostly overlooked in LM research, directed to the financial domain[1]; and (3) a variation of WordpieceToken ratio sampling [14] that accounts for stratified sampling.

The paper is organized as follows: Section II presents background concepts for a thorough understanding of this paper. Section III describes the experimental protocol, including dataset, classifier, and metrics. Section IV presents the experimental results. Finally, Section V concludes this paper and provides potential future works.

## II. Background

This section presents concepts on text stream mining and SentenceBERT (SBERT), including its fine-tuning process.

### A. Text Stream Mining

Bifet et al. [3] stated that data streams are "an abstraction that allows real-time analytics". In data streams, the instances arrive over time sequentially, one by one or in small batches, and the stream itself may be infinite [3], [4]. Learning in streaming environments demands ML methods abilities that include [4]: (a) learning on an instance-basis or in batches; (b) using single-pass operations; and (c) consuming reasonable resources.

Text streams are a subcategory in data streams [5], which exhibit additional challenges, including, for instance, text cleaning, representation maintenance, and vocabulary maintenance. Representation maintenance regards keeping updated representations over time [5], preventing semantic shift effects on the performance of the ML method. Vocabulary maintenance refers to keeping a vocabulary concise without storing stale tokens [5]. Therefore, overlooking semantic shifts can lead to a decrease in the subsequent ML method's performance.

Another characteristic of both traditional and textual data streams is their susceptibility to concept drift. Concept drift can negatively impact the dependent ML model, especially if it does not account for drifts. Section II-B briefly describes concept drift and semantic shift.

### B. Concept Drift

Concept drift regards distribution changes over time [4]. In a classification setting, these changes regard relationship changes between the input and output. More specifically, concept drift can be represented as $p_t(y|X) \neq p_{t+\Delta}(y|X)$, in which $p$ is the conditional probability of $y$ given an input $X$, $t$ is an arbitrary point in time, and $\Delta \in \mathbb{N}$.

In text stream scenarios, concept drift can appear in different forms [5], which include changes in class distribution, e.g., sentiment distribution [6], changes in relevant words (corresponding to feature drift [15] in Bag of Words, BoW, and term frequency-inverse document frequency, TF-IDF), and changes in word's meaning, i.e., *semantic shift* [11], [16]. Although the semantic shift phenomenon is studied across long periods, e.g., decades, it can occur in shorter periods, e.g., hours and weeks [6], [17]. However, the semantic shift problem is out of the scope of this paper since it is generally related to linguistic studies.

Considering the challenges of concept drift and representation maintenance, several papers leverage pre-trained LMs, such as BERT [18] and SBERT [12]. These LMs are generally pre-trained over massive data and are easy to couple to ML systems, making them a good choice for fast deployment.

### C. SentenceBERT (SBERT)

SBERT [12] is a siamese architecture that uses pre-trained models such as BERT [18] and RoBERTa [19] to generate semantically meaningful embeddings. Initially developed for natural language inference (NLI) and semantic textual similarity (STS) tasks, SBERT also provides good quality embeddings for classification tasks [10]. In particular, SBERT's siamese component, i.e., a bi-encoder, accelerates the cosine similarity estimation compared to the BERT's original cross-encoder [12].

This paper uses SBERT encapsulating BERTimbau [13], which is a pre-trained model in Brazilian Portuguese, trained using the brWaC dataset [20]. We favored using SBERT because of its capability of encoding entire texts, and its mean pooling strategy demonstrated higher performance than other aggregating strategies [12].

Fig. 1 represents the scenario addressed in this paper, where a stream consisting of news posts ($X$), their categories ($y$) (available after a classifier prediction), and a date stamp. The posts arrive sequentially, and $X$ and $y$ are stored in the buffer. The ordinary process uses the LM, i.e., SBERT, to encode the news post into a vector, which is used as input for the adaptive random forest (ARF) classifier, equipped with a concept drift detector. The ARF then outputs a category prediction and updates itself using the actual $y$ as soon as it becomes available. After, the news post stream continues to be processed over time. At each iteration, the date stamp

[1]Models are available at Huggingface - See Section III-D

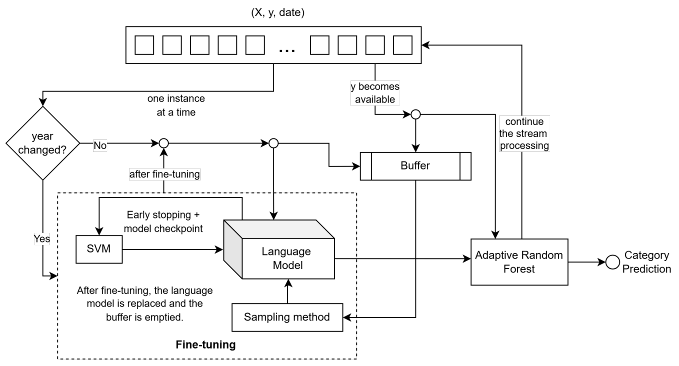

Fig. 1. Diagram of the scenario considered for text stream processing.

from the news is checked, and if the year changes, the fine-tuning process is triggered using the news posts stored in the buffer. Instead of incorporating a fully connected layer on top of the LM, we leverage an SVM. The fine-tuning of the LM uses early stopping, model checkpoint, and hold-out (2,000 news posts sampled from the buffer for training and 200 for validation). After the fine-tuning, if there is an improvement in the performance metrics, the previous LM is replaced by the fine-tuned one. Afterward, the stream processing continues, as described above. We highlight that two classifiers are leveraged during this process, i.e., SVM and ARF, however, the former is used during fine-tuning, whereas the latter is used for actual classification.

## III. Experimental Protocol

This section describes the experimental protocol utilized in this paper, including the datasets, classifier, metrics, LM fine-tuning procedure, and evaluation scenarios.

### *A. Dataset*

We collected news posts written in Brazilian Portuguese from websites specialized in economics covering the period from January 2018 to April 2024 using scraping methods to extract each piece's title, date, and content. All news items were processed to remove certain elements, such as sentences related to advertising and words that were irrelevant to the context or could potentially confuse the model, including emojis, accents, and quotation marks.

The news posts are distributed into varying numbers of categories across the years, as illustrated in Fig. 2. At the beginning, there were only eight categories cataloged in the collected dataset, i.e., '*minhas-financas*', '*mercados*', '*politica*', '*colunistas*', '*negocios*', '*onde-investir*', '*consumo*', '*carreira*'. However, new categories were introduced over the years, i.e., '*business*', '*mundo*', '*economia*', '*stock-pickers*', and '*do-zero-ao-topo*', alongside an increasing volume of news annually, offering a comprehensive perspective on the evolution of news over time.

Table I summarizes the collected data. Again, we highlight the change in the number of classes, which varied from 8 to 13 over time. Additionally, the variation in the number of news posts can indicate an increasing interest over time in topics related to financial aspects. On the other hand, the number of non-unique tokens per news post increased until 2021, changing the trend after that. We considered a token a single character or a sequence of characters surrounded by blank space. These observations suggest that the news producers tended to write and publish news posts more frequently but kept them shorter than in earlier observed moments. However, it is impossible to state that these changes represent an intentional trend.

TABLE I
Statistics of the collected news posts by year.

| Year | # of tokens (mean $\pm$ std) | # of categories per year | # of news posts |
|---|---|---|---|
| 2018 | 786.49 $\pm$ 1550.82 | 8 | 13,932 |
| 2019 | 829.48 $\pm$ 1314.86 | 10 | 10,693 |
| 2020 | 897.27 $\pm$ 945.73 | 10 | 11,994 |
| 2021 | 967.94 $\pm$ 1169.81 | 10 | 14,369 |
| 2022 | 717.64 $\pm$ 605.75 | 10 | 17,856 |
| 2023 | 641.97 $\pm$ 470.67 | 13 | 24,328 |
| 2024 | 573.92 $\pm$ 458.85 | 12 | 7,944 |

### *B. Adaptive Random Forest (ARF) Classifier*

The ARF classifier [21] was selected for its suitable performance compared to incremental competitors on text stream classification tasks subject to concept drift [22]. Although the Support Vector Machine was the best in [22], our work leverages ARF due to its ability to natively handle concept evolution, i.e., the appearance of new classes in the stream, which is characteristic of our problem.

In addition, ARF can manage its internal trees, i.e., add new trees and remove stale ones, by using drift detectors to detect warning regions and the concept drifts themselves. ARF's internal trees are also incremental. However, using ARF's internal concept drift detectors helps keep the trees concise. We applied ARF using default parameters from River library [23], version 0.14, including the number of trees = 10.

### *C. Metrics*

To evaluate the strategies, we used macro F1 and run time. Since the addressed problem is a multiclass classification, macro F1 was chosen because accuracy, the most traditional metric for classification, is sensitive to class imbalance. To calculate macro F1, it is necessary to compute the F1-score per class. The computation of the F1-score for an arbitrary class is given in Equation 1, in which the precision and recall of such class are harmonically averaged.

$$F_1^{\text{class}} = 2 \times \frac{\text{Precision}_{\text{class}} \times \text{Recall}_{\text{class}}}{\text{Precision}_{\text{class}} + \text{Recall}_{\text{class}}} \quad (1)$$

Macro F1 equally weighs in each F1-score calculated per class. All macro F1-score values reported were calculated per subset, i.e., 2018 news, 2019 news, and so on. The values are incrementally calculated within the subset but reset for the following subset. In addition, since the approaches work in stream fashion, we measured the run times for the text stream classification and also for the fine-tuning operation, isolated.

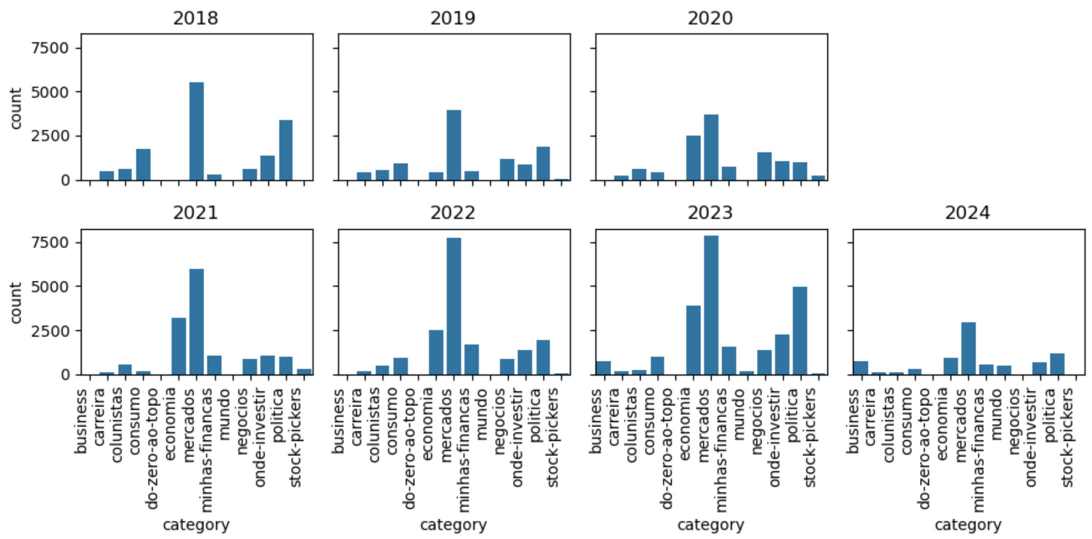

Fig. 2. News distribution according to categories across the analyzed period.

## D. Language Models (LMs)

Since the dataset used is written in Brazilian Portuguese, we used BERTimbau [13] as the base LM. BERTimbau is a BERT model pre-trained using the brWaC dataset [20], which was the most extensive open Portuguese corpus up to its development, containing 3.53 million web pages and 17.5 GB of raw text [13].

Initially, we used an SBERT with the pre-trained BERTimbau as the base model. However, there is no BERTimbau version for SBERT. Therefore, SBERT adds a mean pool layer to extract sentence embeddings. In this paper, we call this model SBERTimbau.

Considering the news posts' date stamps, at the beginning of each year, we sampled news posts from the previous year and fine-tuned the SBERTimbau model. Details on the sampling process are given in Section III-E. This model is referred to as "*SBERT<year>*", where *<year>* represents the latest year in which data has been collected and used for training. For example, when the news posts from 2018 ended, 2,200 news posts were sampled from those published in 2018, and we fine-tuned the SBERTimbau model, creating then the SBERT2018 model. The SBERT2018 model is used for encoding the news posts from 2019. At the end of 2019, again, 2,200 news posts were sampled from those published in 2019 to fine-tune the SBERT2018 model. After fine-tuning, this LM is called SBERT2019. The same procedure was used for fine-tuning the models for the subsequent years.

At the end, we have created and tested seven models:

- **SBERTimbau:** SBERT model [12] combined with BERTimbau [13] (no fine-tuning);
- **SBERT2018**[2]**:** SBERTimbau model fine-tuned with sampled news from 2018;
- **SBERT2019**[3]**:** SBERT2018 model fine-tuned with sampled news from 2019;
- **SBERT2020**[4]**:** SBERT2019 model fine-tuned with sampled news from 2020;
- **SBERT2021**[5]**:** SBERT2020 model fine-tuned with sampled news from 2021;
- **SBERT2022**[6]**:** SBERT2021 model fine-tuned with sampled news from 2022;
- **SBERT2023**[7]**:** SBERT2022 model fine-tuned with sampled news from 2023.

In all cases, embeddings are averaged element-wise if the number of wordpieces (BERT's internal tokens) of a news post is higher than 512, which is the original limit for SBERTimbau. Even though we acknowledge the existence of similar works, such as Santos et al. [24], their approach differs from ours in the following aspects: (a) our approach considers the text stream paradigm, respecting the temporal order; (b) although the authors used BERTimbau as a base LM, they fine-tuned the model using roughly 1,400,000 sentences from news between 2006 and 2022, which forbid the use of their model in news posts older than 2022; (c) their approach was fine-tuned to perform sentiment analysis; and (d) we use SBERT for news posts encoding, without appending layers for classification.

Finally, our SBERT models are available to the community on Huggingface[8].

## E. Language Model Fine-tuning Procedure

The LM fine-tuning procedure considers two steps: (a) news posts sampling; and (b) LM fine-tuning.

[2]Available at: https://huggingface.co/pucpr-br/sbertimbau_news_2018/

[3]Available at: https://huggingface.co/pucpr-br/sbertimbau_news_2019/

[4]Available at: https://huggingface.co/pucpr-br/sbertimbau_news_2020/

[5]Available at: https://huggingface.co/pucpr-br/sbertimbau_news_2021/

[6]Available at: https://huggingface.co/pucpr-br/sbertimbau_news_2022/

[7]Available at: https://huggingface.co/pucpr-br/sbertimbau_news_2023/

[8]https://huggingface.co/pucpr-br

It is known that the fine-tuning process is computationally costly [25], [26]. Therefore, sampling relevant data for fine-tuning is cheaper in terms of computational resources, providing informative pieces of data for the fine-tuning process. For this paper, we leverage the WordpieceToken ratio sampling method [14], which uses the ratio between the number of wordpieces and the number of tokens to weigh the importance of each news post.

In this paper, we also propose a stratified variant, called *stratified WordpieceToken ratio* (SWPT). In practice, SWPT samples data according to their weight, i.e., the ratio between wordpieces and tokens. For each class, we proportionally sample a number of news posts, considering their weight. In particular, we sample 2,200 news posts using SWPT.

Regarding the LM fine-tuning, using the news posts previously sampled, we fine-tune the SBERT model using the *BatchAllTripletLoss* (BATL) function [27], described in Eq. 1, in which $P$ are arbitrary classes, $K$ are items from $P$, $m$ is an arbitrary margin, $d$ is the distance between positive and negative anchors, $D$ is an arbitrary distance metric and $f_\theta$ is a function that maps similar items into close points in the data space [27].

$$L_{BAT}(\theta; X) = \overbrace{\sum_{i=1}^{P}\sum_{a=1}^{K}}^{\text{all anchors}} \overbrace{\sum_{p=1,p\neq a}^{K}}^{\text{all pos.}} \overbrace{\sum_{j=1,j\neq i}^{P}\sum_{n=1}^{K}}^{\text{all negatives}} [m + d_{j,a,n}^{i,a,p}]_{+},$$
$$d_{j,a,n}^{i,a,p} = D(f_\theta(x_a^i), f_\theta(x_p^i)) - D(f_\theta(x_a^i), f_\theta(x_n^j)). \tag{1}$$

BATL maximizes the distance between inter-label instances while minimizing the distance between intra-label instances. This loss function reached relevant results when combined with the WordpieceToken ratio sampling [14].

We fine-tuned the LMs using the *early stopping* [28] and model checkpoint procedures. Specifically, we set maximum epochs as 100, used 2,000 news posts to train an offline Support Vector Machine (SVM) [29] with linear kernel and the remaining 200 posts for SVM validation. We tracked the macro F1-score for the validation set and stored the best SBERT model. In addition, we set the patience as 15, meaning that 15 epochs without improvement in macro F1-score in the validation set stopped the fine-tuning procedure. At the end of the process, we stored the best LM. Please note that we only used SVM in the fine-tuning process.

There are a few ways to evaluate the performance of a model during fine-tuning, such as monitoring loss values and performance metrics. In this paper, although we presented the BATL function, we monitored the macro F1-score in the validation set. Since we leveraged the SentenceTransformers library, we had difficulty in extracting meaningful loss values since the values we obtained did not correlate with an increase in the macro F1-score in the validation set. Therefore, we opted to use the macro F1-score in the validation set to determine the quality of a language model during the fine-tuning.

Regarding the SVM, we used it because it can find a hyperplane and maximize margins between classes. Having in mind that BATL function aims to maximize inter-label distance while minimizing the intra-label distance, we hypothesize that SVM can be used to evaluate the fine-tuning procedure. In other words, we hypothesize that SVM can reach better macro F1-scores whenever the fine-tuning generates a language model that provides better representations than the previous language model.

### F. Scenarios

In this paper, we considered two scenarios: (a) classification of news posts using the same SBERT model across the news post stream and (b) classification of news posts from the subsequent year using an SBERT model fine-tuned until the last year, hereafter dubbed *Updated SBERT*. Specifically, in scenario (a), we evaluated the LMs presented in Section III-D without any posterior update, respecting the temporal order.

The results are displayed in a table, with the LMs as rows, the datasets as columns, and the cells corresponding to the evaluated metric. In this paper, the cells contain macro F1-score values and elapsed times. In addition, the classifier, the Adaptive Random Forest, is not restarted along the news post stream.

### G. Implementation and Hardware

The implementation was developed using the following libraries: Scikit-learn [30], Seaborn [31] and Matplotlib [32] for graphics generation, River [23] for text stream simulation, ARF, and metrics, and SentenceTransformers[9] for SBERT.

The hardware employed was a 13th-generation Intel(R) Core(TM) i9-13900K with 126 GB of RAM and 2 NVIDIA 24GB GeForce RTX 4090 GPUs running Ubuntu 22.04 LTS server.

## IV. Results

Table II shows the macro F1-scores obtained for scenarios (a) and (b). Scenario (a) consists of the experiments with SBERTimbau, and SBERT2018 to SBERT2023, while scenario (b) corresponds to the values in which the model is *Upd. SBERT*.

First, we observe that, in general, even with a static LM, the macro F1 maintains acceptable levels over time due to the incremental classifier (ARF). For instance, we emphasize the SBERT2018 results, which reach 55.42% of macro F1 when classifying 2019 news, increasing to 62.86% in 2020 news and achieving similar levels in 2021 and 2022 news. Second, using the Updated SBERT leads to outstanding results. Table II shows that learning previous patterns can help reach interesting macro F1-score values, together with minimally fine-tuned and updated LMs. In all cases, the bold values are the best macro F1-score values per column obtained per subset over ten executions.

Fig. 3 shows the macro F1-score values obtained across the years using different fine-tuned LMs. *SBERTimbau* is the

[9]https://www.sbert.net/

TABLE II
LM evaluation over time, in terms of macro F1 (%), over ten executions. The best values per news subset are in bold.

| Model/News | News 2018 | News 2019 | News 2020 | News 2021 | News 2022 | News 2023 | News 2024 |
|---|---|---|---|---|---|---|---|
| **SBERTimbau** | **55.02 ± 0.47** | 61.39 ± 0.62 | 60.61 ± 0.64 | 57.64 ± 0.52 | 60.01 ± 0.87 | 55.66 ± 0.54 | 59.11 ± 0.88 |
| **SBERT2018** | | 55.42 ± 0.51 | 62.86 ± 0.9 | 59.97 ± 1.22 | 62.20 ± 0.53 | 57.76 ± 0.61 | 58.55 ± 2.68 |
| **SBERT2019** | | | 62.34 ± 0.51 | 66.20 ± 0.49 | 69.30 ± 0.54 | 59.06 ± 0.89 | 62.24 ± 2.12 |
| **SBERT2020** | | | | 67.00 ± 0.53 | **73.47 ± 0.62** | **60.63 ± 0.89** | 58.92 ± 1.86 |
| **SBERT2021** | | | | | 62.42 ± 0.62 | 58.77 ± 0.98 | 56.29 ± 2.77 |
| **SBERT2022** | | | | | | 53.79 ± 0.66 | 61.45 ± 2.69 |
| **SBERT2023** | | | | | | | 60.71 ± 0.62 |
| **Upd. SBERT** | **55.02 ± 0.47** | **61.57 ± 0.63** | **69.05 ± 0.43** | **74.52 ± 0.64** | 71.84 ± 0.75 | 60.24 ± 1.10 | **64.80 ± 2.82** |

SBERT using the regular BERTimbau, while *Updated SBERT* is the SBERT model fine-tuned with sampled news from the previous year. We observe that only in 2022 and 2023, using the Updated SBERT does not favor the macro F1-score. This behavior is correlated with the appearance of new categories (classes), as depicted in Fig. 5.

In all settings, performance decreases using the News 2023 subset compared to the other subsets. We attribute this decrease to the emergence of two new categories: *"business"* and *"do-zero-ao-topo"*. Furthermore, an increase in the number of news posts from the category *politica* may have influenced the results. It is also noteworthy the existence of the categories *business* and *negocios*, since the terminology *negocios* corresponds to business in Portuguese. In this situation, illustrated by Fig. 4, we noticed a gradual drift, considering the drift dynamics taxonomy proposed in Gama et al. [4]. News posts from *business* started appearing in mid-2023, co-existing with those from *negocios*, but quickly replacing them almost completely. Since their contexts are expected to be similar, the classification model would tend to confuse them during the transition of categories, also contributing to the decrease in the macro F1-score. Although both classes, i.e., *negocios* and *business*, are semantically equivalent, we opted to keep both in the experiments because, in a real-world setting, it is not possible to predict this sort of situation since a third-party manages the actual labeling of the news posts.

Using the Updated SBERT until 2022 helped reach the best macro F1-score values, demonstrating that updating the LM is beneficial. However, with the 2022 news, ARF could not reach the best macro F1-score with the updated text representations. In this case, SBERT2020 offered better representations to ARF when classifying 2022 and 2023 news, showing that concepts previously learned by ARF potentially hampered or delayed its incremental learning.

Fig. 5 (upper half) shows the macro F1-score values obtained over time using a 200-sized window. A concept drift happens in all scenarios in the middle of 2023, impacting the performance regarding macro F1. Fig. 5 (lower half) shows the number of categories over time. In 2018, the stream started with eight categories, increasing to 10 at the end of 2019 and keeping stable until 2023, when the number of categories goes to 13. Although the impact of concept evolution, i.e., change in the number of classes, is less meaningful, it is still noticeable in 2019. This is demonstrated by the slightly negative trend of macro F1 with SBERTimbau, SBERT2018, and Updated SBERT in 2019.

We evaluated statistically the results in terms of macro F1-score using the Friedman test and the Nemenyi test, following the procedure presented in [33]. The Friedman test assigns the existence of a statistical difference between the methods and, if the statistical difference does exist, we leveraged the Nemenyi test to find the differences. We used a significance, i.e., $\alpha$, of 0.05. Our null hypothesis, i.e., $H_0$, is that there is no significant difference among the macro F1-scores, while the alternative hypothesis ($H_\alpha$) indicates that there is a significant difference among the observed macro F1-scores. We used the Friedman test since it is impossible to assess data normality using 10 runs.

The scenario that evaluates the 2018 news was left out of the analysis since the models are the same: SBERTimbau and Updated SBERT, which has not been updated so far. Evaluating the scenario of 2019 news, we obtained the $p$-value = 5.00e-4, which is below 0.05; therefore, we rejected $H_0$. To visualize where the difference lies, we applied the Nemenyi test. Fig. 6 shows the average ranking considering 10 runs. The critical distance (CD) resulted in 1.05, with $k = 3$ approaches and $N = 10$ runs. The methods linked by a horizontal bar are considered statistically equivalent. Thus, in the scenario of 2019 news, Upd. SBERT was statistically equivalent to SBERTimbau. Additionally, both are statistically better than SBERT2018.

For 2020 news, we obtained the $p$-value = $5.56 \times 10^{-6}$. Again, we rejected $H_0$. Considering $k = 4$ approaches and $N = 10$, we obtained CD = 1.48. Fig. 7 shows the average ranking. Upd. SBERT occupied rank 1 in all 10 runs. However, with 10 runs, the CD obtained makes Upd. SBERT and SBERT2018 equivalent. SBERT2018 and SBERT2019 were also equivalent, and SBERT2019 and SBERTimbau were considered equivalent.

For 2021 news, we obtained the $p$-value = $6.10 \times 10^{-8}$ and we rejected $H_0$. Using $k = 5$ approaches and $N = 10$, we obtained CD = 1.93. Fig. 8 shows the average ranking. Again, Upd. SBERT occupied rank 1 in all 10 runs. However, with 10 runs, the CD obtained makes Upd. SBERT, SBERT2020 and SBERT2019 equivalent. SBERT2020, SBERT2019, and SBERT2018 were equivalent, and SBERT2018 and SBERTimbau were considered equivalent to each other.

Considering the 2022 news, we obtained the $p$-value

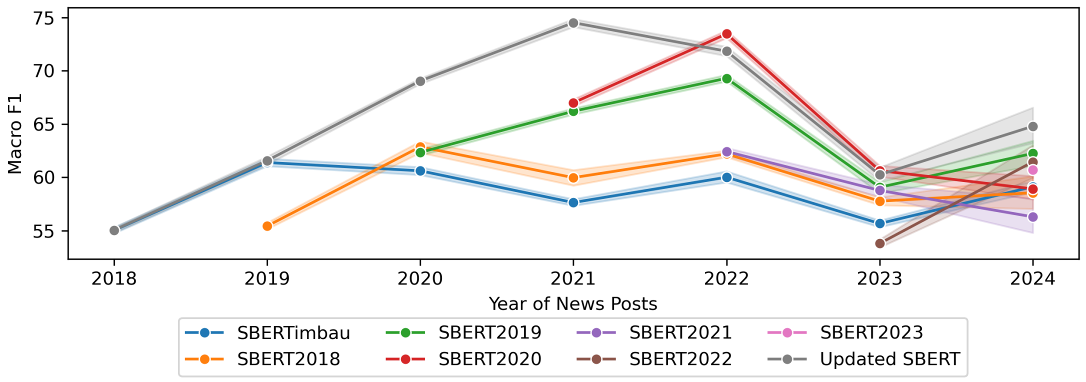

Fig. 3. Macro F1-score values obtained using different models in the news posts organized by year. *Updated SBERT* regards the stream processing using an SBERT model fine-tuned with news posts from the previous year. The updated SBERT is shown in gray. The shadows correspond to the standard deviation.

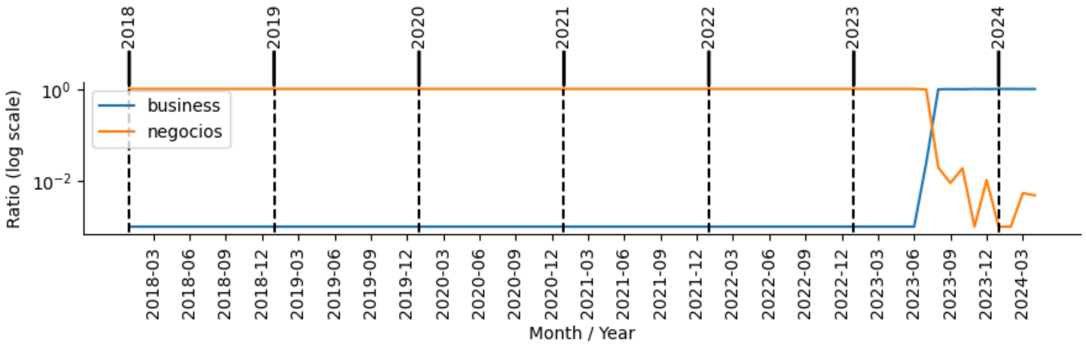

Fig. 4. Ratio between news posts from the *business* and *negocios* categories.

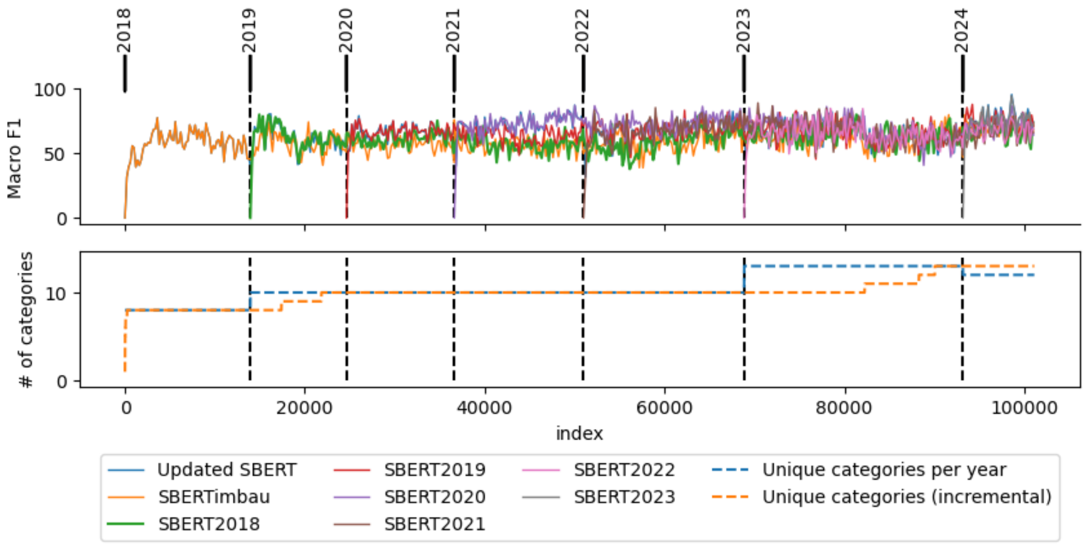

Fig. 5. Macro F1 considering each LM applied in this study and the number of categories over time.

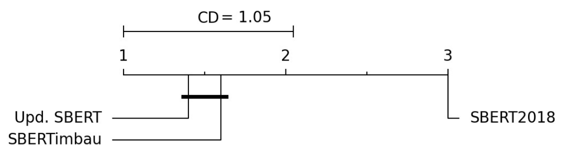

Fig. 6. Average ranking of the methods for 2019 news, considering 10 runs.

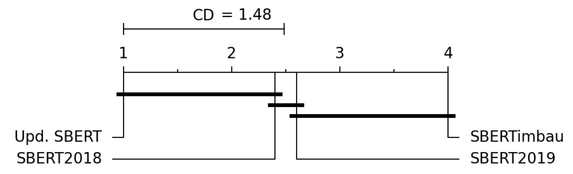

Fig. 7. Average ranking of the methods for 2020 news, considering 10 runs.

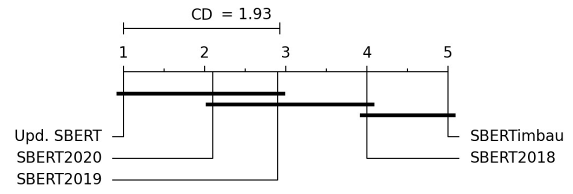

Fig. 8. Average ranking of the methods for 2021 news, considering 10 runs.

= $3.11 \times 10^{-9}$ and we rejected $H_0$. Using $k = 6$ approaches and $N = 10$, we obtained CD = 2.38. Fig. 9 shows the average ranking of the approaches. In this case, SBERT2020 obtained a better average rank than Upd. SBERT. Considering the CD, SBERT2020, Upd. SBERT, and SBERT2019 were considered statistically equivalent; SBERT2019, SBERT2021, and SBERT2018 were also considered equivalent, and SBERT2021, SBERT2018, and SBERTimbau were considered equivalent, although SBERTimbau occupied the worst rank in all runs.

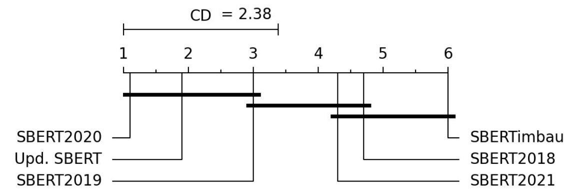

Fig. 9. Average ranking of the methods for 2022 news, considering 10 runs.

Focusing on the 2023 news, we obtained the $p$-value = $1.51 \times 10^{-9}$, thus rejecting $H_0$. Using $k = 7$ approaches and $N = 10$, we obtained CD = 2.85. Fig. 10 shows the average ranking of the approaches. Again, SBERT2020 obtained a slightly better average rank than Upd. SBERT. Considering the CD, SBERT2020, Upd. SBERT, SBERT2019, and SBERT2021 were considered statistically equivalent; Upd. SBERT, SBERT2019, SBERT2021, and SBERT2018 were also considered equivalent, and SBERTimbau and SBERT2022 were considered equivalent, although SBERT2022 occupied the worst rank in all runs. The fact that SBERT2022 was the worst rank is interesting since it is temporally the closest to 2023 news and, therefore, expected to obtain better results.

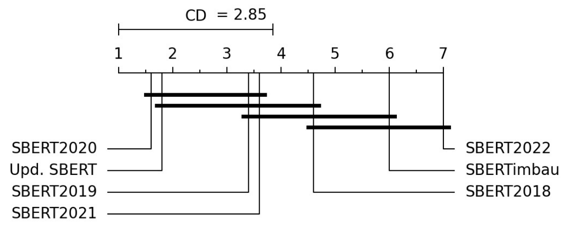

Fig. 10. Average ranking of the methods for 2023 news, considering 10 runs.

Finally, with the 2024 news, we obtained the $p$-value = $1.32 \times 10^{-6}$, rejecting $H_0$. Using $k = 8$ approaches and $N = 10$, we obtained CD = 3.32. Fig. 11 shows the average ranking of the approaches. In this scenario, Upd. SBERT obtained the best average rank again, followed by SBERT2019, SBERT2022, and SBERT2023, which were considered statistically equivalent. In addition, SBERT2019, SBERT2022, SBERT2023, SBERT2020, SBERT2018, and SBERTimbau were also statistically equivalent. Finally, SBERT2023, SBERT2020, SBERT2018, SBERTimbau, and SBERT2021 were also considered statistically equivalent.

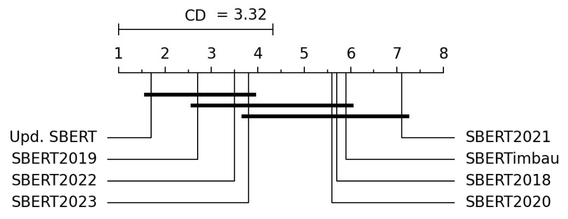

Fig. 11. Average ranking of the methods for 2024 news, considering 10 runs.

As a last comparison considering macro F1-score, we evaluate the values obtained (a) in the long run, i.e., considering the complete text stream; and (b) in the 2024 news scenario. Fig. 12 graphically shows these comparisons. The mean values in each scenario are represented by the horizontal dashed lines. Clearly, Upd. SBERT always obtained above-average values. We also highlight that the long run scenario is biased since only Upd. SBERT and SBERTimbau, in fact, were evaluated across the complete stream. The other approaches, on the other hand, have fewer data points to evaluate, meaning that it may not be a fair comparison.

Still considering Fig. 12, the comparison in the 2024 news posts scenarios might also be considered unfair since some methods had more textual data to learn from over time than others. However, this comparison provides an interesting snapshot. Only SBERT2019, SBERT2022, and Upd. SBERT had

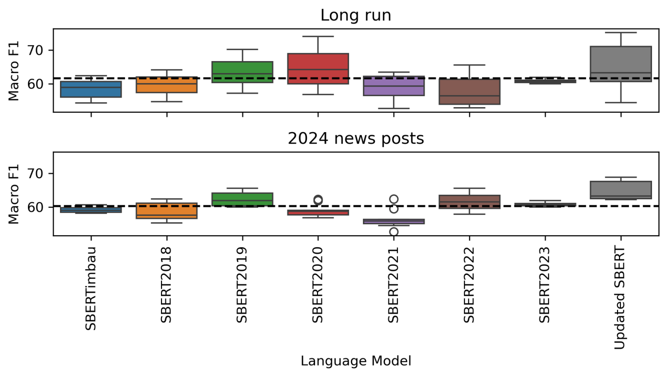

Fig. 12. Comparison between approaches considering the complete stream and 2024 news posts.

their medians above the average macro F1-scores. To sum up, we understand that updating both the SBERT and the classifier over time is generally effective across the stream, while having only the incremental classifier with a static language model can obtain, at most, average results.

We also evaluated the run times. To perform a fair comparison, we measured and averaged the elapsed times per row. Fig. 13 visually shows the elapsed time per row in seconds. SBERTimbau, with the exception of 2018 news, reached the longest elapsed times on average. Although there is not a clear reason for that, we hypothesize that optimizations in the pooling and embedding layers may result in a more efficient model for inference. In addition, we hypothesize that, since the Batch All Triplet Loss helps the class separation in the vector space, it makes the classification a bit easier for the classifier. Considering that we leveraged an online/incremental classifier, a non-fine-tuned model such as SBERTimbau may generate vector representations not so distinguishable in the vector space, making the incremental classifier perform more internal operations to identify better splits of the vector space, thus leading to a longer run time.

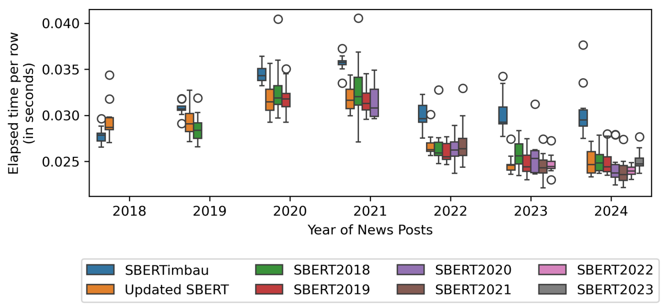

Fig. 13. Elapsed run times per approach, per year.

The elapsed times in Fig. 13 do not include the elapsed times for fine-tuning. The elapsed times for fine-tuning are listed in Table III. Considering real-time processing, the longest value, i.e., for SBERT2022, would correspond to only about 15 minutes. This is a very short time compared to a year of text processing and classification. Therefore, we consider that the trade-off is valid in this scenario.

TABLE III
ELAPSED TIMES FOR FINE-TUNING, WITH THEIR MAXIMUM NUMBER OF EPOCHS, AND THE BEST EPOCH FOUND.

| Model | # of epochs | Best epoch | Elapsed time (seconds) |
|---|---|---|---|
| SBERT2018 | 20 | 8 | 828.24 |
| SBERT2019 | 20 | 11 | 851.58 |
| SBERT2020 | 18 | 9 | 775.82 |
| SBERT2021 | 15 | 12 | 732.58 |
| SBERT2022 | 21 | 13 | 888.86 |
| SBERT2023 | 19 | 14 | 798.60 |
| Average | 18.83 $\pm$ 2.14 | 11.17 $\pm$ 2.32 | 812.61 $\pm$ 55.73 |

## V. CONCLUSION

The use of pre-trained LMs has been boosting applications and research worldwide due to their ease of use, ability to provide good quality representations, and capability of domain adaptation using modest resources. This paper evaluates the impact of using a fixed pre-trained LM across a text stream for classification tasks. In our application, the text stream provides news posts in Brazilian Portuguese from the financial domain, collected between 2018 and April 2024, for classification regarding categories. We used the Adaptive Random Forest (ARF) as a classifier, given its ability to handle both concept drift and concept evolution, i.e., learn incrementally, adjust itself to novel classes, and handle stale internal trees. For analysis purposes, we split the news posts into years. To compare with the fixed SBERT models, we evaluate the employ of the *Updated SBERT*, which regards the SBERT fine-tuned over time using 2,000 news posts from the last period. To validate the fine-tuning process and the quality of the updated representations, we used 200 news posts.

Returning to our first research question, i.e., “How effective is recurring fine-tuning for continuous use in Brazilian Portuguese text stream classification?”, we notice that no updating at all is the worst choice in this setting. Updating the LM over time showed consistent macro F1 results across the stream, keeping the same incremental classifier. However, in 2023, the occurrence of concept drift and concept evolution negatively affected the macro F1 for all LMs evaluated. Experiments with other classifiers are demanded to check whether this observation relates to a potential slowness of ARF in recovering from the concept drifts. We also observe that using only the incremental classifier over the text stream would not be enough, since SBERTimbau could reach only average macro F1-scores, below the levels obtained by the Upd. SBERT. Therefore, we conclude that recurring fine-tuning is effective for continuous use in classification, especially when combined with an incremental classifier. In the best case (2021 news), Upd. SBERT obtained around 7.5 percentage points (pp.) above the second best approach, and in the worst case, i.e., 2022 news, Upd. SBERT obtained 1.6 pp. below the best approach for that year, showing consistently interesting results across the text stream.

We also measured the elapsed times across the text stream processing scenarios. Generating better-quality vector representations may lead to a smaller run time. In addition, in our experiments, the time invested in fine-tuning corresponds to a short time compared to a year (granularity of time used in this paper). Therefore, our approach provides a good trade-off between performance and time.

In future works, we intend to evaluate longer streams and the impact of updating the model under non-specified periods using drift-based triggers.

## Acknowledgements

Cristiano Mesquita Garcia is a grantee of a *Doutorado Sanduíche* scholarship provided by *Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES). This study is a byproduct of project #409371/2022-0 funded by the *Conselho Nacional de Desenvolvimento Científico e Tecnológico* (CNPq).

## References

[1] J. G. C. Krüger *et al.*, "An Explainable Machine Learning Approach for Student Dropout Prediction," *Expert Systems with Applications*, 2023.

[2] E. Soares *et al.*, "Evolving Fuzzy Set-based and Cloud-based Unsupervised Classifiers for Spam Detection," *IEEE Lat. Am. Trans.*, 2019.

[3] A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer, *Machine Learning for Data Streams: with Practical Examples in MOA*. MIT Press, 2023.

[4] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A Survey on Concept Drift Adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, pp. 1–37, 2014.

[5] C. M. Garcia *et al.*, "Concept Drift Adaptation in Text Stream Mining Settings: A Comprehensive Review," *arXiv:2312.02901*, 2023.

[6] ——, "Event-driven Sentiment Drift Analysis in Text Streams: An Application in a Soccer Match," in *2023 International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2023.

[7] L. Yuan *et al.*, "Addressing Feature Drift in Data Streams using Iterative Subset Selection," *ACM SIGAPP Applied Computing Review*, vol. 19, no. 1, pp. 20–33, 2019.

[8] Z. S. Harris, "Distributional Structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[9] G. Salton and C. Buckley, "Term-weighting Approaches in Automatic Text Retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.

[10] B. S. Thuma *et al.*, "Benchmarking Feature Extraction Techniques for Textual Data Stream Classification," in *IJCNN*. IEEE, 2023, pp. 1–8.

[11] W. L. Hamilton *et al.*, "Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change," in *Proc. of the 54th Annual Meeting of the Assoc. for Comp. Ling. (Volume 1: Long Papers)*, 2016.

[12] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proc. of the 2019 Conf. on Empirical Methods in NLP*. Assoc. for Comp. Ling., 2019.

[13] F. Souza *et al.*, "BERTimbau: pretrained BERT models for Brazilian Portuguese," in *9th BRACIS*, 2020.

[14] C. M. Garcia *et al.*, "Improving Sampling Methods for Fine-tuning SentenceBERT in Text Streams," *arXiv e-prints*, 2024.

[15] J. P. Barddal, H. M. Gomes, F. Enembreck, and B. Pfahringer, "A survey on feature drift adaptation: Definition, benchmark, challenges and future directions," *J. Syst. Softw.*, vol. 127, pp. 278–294, 2017.

[16] F. Bravo-Marquez *et al.*, "Incremental Word Vectors for Time-evolving Sentiment Lexicon Induction," *Cognitive Computation*, pp. 1–17, 2022.

[17] I. Stewart *et al.*, "Measuring, Predicting and Visualizing Short-term Change in Word Representation and Usage in VKontakte Social Network," in *Proc. of the Int. AAAI Conf. on Web and Social Media*, 2017.

[18] J. Devlin *et al.*, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. of the 2019 Conf. of the North American Chapter of the Assoc. for Comput. Ling.: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019.

[19] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[20] J. A. Wagner Filho *et al.*, "The brWaC Corpus: A New Open Resource for Brazilian Portuguese," in *Proc. of the 11th Int. Conf. on Lang. Resources and Evaluation (LREC 2018)*, 2018.

[21] H. M. Gomes *et al.*, "Adaptive Random Forests for Evolving Data Stream Classification," *Mach. Learn.*, vol. 106, pp. 1469–1495, 2017.

[22] C. M. Garcia *et al.*, "Methods for Generating Drift in Text Streams," *arXiv preprint arXiv:2403.12328*, 2024.

[23] J. Montiel *et al.*, "River: Machine Learning for Streaming Data in Python," *Journal of Machine Learning Research*, 2021.

[24] L. L. Santos *et al.*, "FinBERT-PT-BR: Análise de Sentimentos de Textos em Português do Mercado Financeiro," in *II Br. Workshop on AI in Finance*. SBC, 2023.

[25] S. Amba Hombaiah *et al.*, "Dynamic Language Models for Continuously Evolving Content," in *Proc. of the 27th ACM SIGKDD*, 2021, pp. 2514–2524.

[26] O. Sharir, B. Peleg, and Y. Shoham, "The Cost of Training NLP Models: A Concise Overview," *arXiv preprint arXiv:2004.08900*, 2020.

[27] A. Hermans *et al.*, "In Defense of the Triplet Loss for Person Re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[28] Y. Yao *et al.*, "On early stopping in gradient descent learning," *Constructive Approximation*, vol. 26, no. 2, pp. 289–315, 2007.

[29] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 1992, pp. 144–152.

[30] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[31] M. L. Waskom, "Seaborn: Statistical Data Visualization," *J. Open Source Softw.*, 2021.

[32] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, 2007.

[33] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *The Journal of Machine learning research*, vol. 7, pp. 1–30, 2006.