Text Data Augmentation to Manage Imbalanced Classification: Apply to BERT-based Large Multiclass Classification for Product Sheets

Yu DU *yu.du@appvizer.com*
Cloud-is-Mine R&D
Perpignan, 66100, France

Erwann LAVAREC *erwann.lavarec@appvizer.com*
Cloud-is-Mine R&D
Perpignan, 66100, France

Colin LALOUETTE *colin.lalouette@appvizer.com*
Cloud-is-Mine R&D
Perpignan, 66100, France

Abstract

Recent studies have showcased the effectiveness of deep pre-trained language models, such as BERT (Bidirectional Encoder Representations from Transformers), in tasks related to natural language processing and understanding. BERT, with its ability to learn contextualized word vectors, has proven highly accurate for binary text classification and basic multiclass classification, where the number of unique labels is relatively small. However, the performance of BERT-based models in more ambitious multiclass classification tasks, involving hundreds of unique labels, is seldom explored, despite the prevalence of such problems in real-world scenarios. Moreover, real-world datasets often exhibit class imbalance issues, with certain classes having significantly fewer corresponding texts than others. This paper makes two primary contributions: first, it examines the performance of BERT-based pre-trained language models in handling tasks of large multiclass classification system within a specific real-world context; second, it investigates the application of text data augmentation techniques to mitigate the class imbalance problem. Through rigorous experiments in a real-world SaaS (Software as a Service) domain, the results demonstrate that: 1) BERT-based models can effectively tackle tasks of large multiclass classification system, delivering reasonable prediction performance; and 2) text data augmentation can significantly enhance prediction performance in terms of accuracy (by 34.7%) and F1-score (by 37.1%).

Keywords: Text Classification, Imbalanced Classification, Natural Language Processing, BERT, CamemBERT, Data Augmentation, Deep Learning.

1. INTRODUCTION

Transformer language modeling with pre-training and transfer learning has led to significant advancements in many natural language processing (NLP) and natural language understanding (NLU) tasks, including text classification (Negesse, 2015; Vaswani et al., 2017; Devlin et al., 2018; Liu et al., 2019; Martin et al., 2020; Althobaiti, 2020; Sedai & Houghton, 2022). Deep contextualized language models pre-trained using masked language modeling (MLM) (Sinha et al., 2021), such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018), have outperformed state-of-the-art machine learning-based classifiers like Random Forest (RF) (Parmar et al., 2018) and Support Vector Machine (SVM) (Joachims, 1998) in numerous NLP/NLU tasks. For instance, BERT-based deep learning models are highly accurate in detecting spam emails (Bhopale & Tiwari, 2021) and identifying fake news (Kaliyar et al.,

2021). These models have also demonstrated state-of-the-art performance in multiclass classification tasks with less than fifteen classes or labels (Sun et al., 2019).

However, despite their remarkable accuracy in binary text classification tasks, such as spam detection, and multiclass classification system with few classes (Zhang et al., 2015), few studies have been conducted to better understand the prediction performance of BERT-based models in dealing with larger multiclass classification tasks. This cap in research presents an opportunity for further exploration to assess the capabilities of BERT-based models in handling more complex classification problems with a higher number of classes.

Real-world multiclass text classification problems are often large and may involve hundreds of categories or class labels (Shaheen et al., 2020). For example, platforms such as Appvizer and its competitors catalog over 500 categories of Software as a Service (SaaS).Although one could simplify the classification problem by grouping semantically similar categories under a more generalized concept, this approach may not always suit real-world contexts. Consequently, it raises questions about the performance of BERT-based language models when applied to more complex multiclass text classification tasks with numerous classes. Furthermore, the number of SaaS vendors in a category often correlates with the overall popularity of the underlying business activity. For example, there are undoubtedly more Human Resource (HR) or Customer Relationship Management (CRM) software solutions than Zoo management software. This discrepancy leads to class imbalance issues in real-world datasets. To address the class imbalance problem, researchers strive to meet various goals, such as generating additional data for minority classes to balance skewed datasets. To this end, text data augmentation (TDA) has recently been explored (Bayer et al., 2021).

In this study, we investigate the performance of BERT-based language models in a real-world imbalanced multiclass classification system with many classes. The main research questions (RQ) tackled in this article are:

- **RQ1**: Do BERT-based pre-trained language models perform for classification tasks of large multiclass system?
- **RQ2**: Is text data augmentation capable of boosting the performance of BERT-based models for multiclass classification with large number of classes?

The rest of the paper is organized as follows: in Section 2, we present related works in the literature. In Section 3, we describe the research problem as well as a proposed solution. In Section 4, we present the experimentation we made and discuss the obtained results. In Section 5, we summarize and conclude the article.

2. RELATED WORKS

In this section, we will briefly review existing approaches related to our research objectives. In Section 2.1, we introduce state-of-the-art text classification methods. In Section 2.2, we introduce the concept of text data augmentation (TDA) and summarize some main TDA approaches to improve the quality of text classification methods.

2.1 Text Classification Methods

Text classification is the most fundamental and essential task in natural language processing (NLP) (Li et al., 2022). It is widely used in various NLP-related applications such as sentiment analysis (Serrano-Guerrero et al., 2015), question answering (Mishra & Jain, 2016), topic classification (Rahman & Akter, 2019), and many others. In the context of today's information explosion, it becomes time-consuming and challenging to process and classify large amounts of text data manually. Besides, the accuracy of manual text classification can also be easily influenced by human factors, such as fatigue and expertise.

Numerous models have been proposed in the past few decades for text classification (Kowsari et al., 2019). From a technical point of view, the text classification task is referred to as retrieving features from raw text data and then predicting the category of the text based on the extracted features. The essential step for most text classification methods is the feature extraction phase, which consists of identifying and extracting the most representative features (often referred to as feature vectors) from the raw text data. Different types of text classification models adopt different feature extraction strategies. To this end, one could roughly separate existing text classification approaches into two principal types: 1) traditional machine learning-based approaches and 2) recent deep learning-based approaches.

2.1.1 Traditional Models

Text data differs from other data types, such as images or signals, which can naturally be represented by numeric feature vectors. For instance, images are tensors of numerical pixel values that can be directly fitted into a convolutional neural network architecture like VGG16 (Simonyan & Zisserman, 2015) to extract relevant image features. In contrast, feature extraction from text data requires preprocessing the raw data using artificial NLP techniques (e.g., dictionary building, data cleaning, stemming, lemmatization (Balakrishnan & Lloyd-Yemoh, 2014). This preprocessing step is particularly crucial for traditional models, as classic machine learning (ML) algorithms rely heavily on numerical vectors for training. Therefore, the effectiveness of traditional ML models is largely restricted by the feature extraction procedure.

In general, the process of traditional models for text classification involves the following steps: preprocessing, feature extraction and classifier training (Yang et al., 2020), as depicted in Figure 1. The raw text data is first preprocessed, followed by the application of feature extraction techniques to convert the preprocessed data into numerical feature vectors. Lastly, these extracted feature vectors are used to train a classifier with machine learning algorithms. A brief overview of each step is provided below.

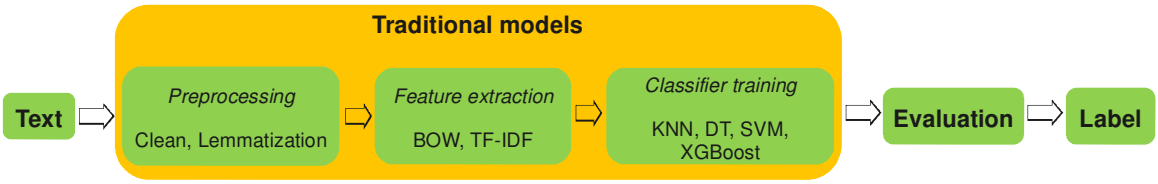

FIGURE 1: The procedure of traditional models for text classification.

Preprocessing. The preprocessing step aims to clean the raw text data in order to prepare it for feature extraction (Kannan et al., 2014). This typically involves tokenization, stop word removal and stemming/lemmatization.

Feature Extraction. The main goal of feature extraction is to generate relevant vector representations for the cleaned text data during the preprocessing phase, i.e., a form that is easier for computers to process. The relevance of the resulting vector representations stands for the minimization of information loss, in other words, the obtained feature vector needs to keep the intrinsic meaning of the text data it represents. Most of the existing feature extraction approaches are based on statistical information of the text corpus (Baeza-Yates et al., 1999; Zhang et al., 2010; Mikolov et al., 2013). For example, Bag-Of-Words (BOW) (Zhang et al., 2010) first creates a dictionary for the corpus and then represents a piece of text by a dictionary-sized vector in which the value of the i-th position refers to the frequency of the corresponding word in the input text. TF-IDF (Baeza-Yates et al., 1999) (Term Frequency-Inverse Document Frequency) is another popular feature engineering method for text data. TF refers to the frequency of a word in a specific document, and IDF represents the reciprocal of the proportion of documents containing this word to the total number of documents in the corpus. TF-IDF is the multiplication of the two. It privileges words that appear frequently in a specific document and penalizes those appearing in most of the documents. Word embedding is another technique for text representation. The

Word2Vec model (Mikolov et al., 2013) is one of the most widely used word embedding models. Unlike BOW and TF-IDF, Word2Vec uses machine learning techniques and neural networks to learn latent representations for words. It uses two types of algorithms, CBOW and Skip-gram during the training phase of the model. Word2Vec allows for the identification of relationships (semantic or syntactic) between words which is not possible with the TF-IDF and BOW approaches. Trained Word2Vec models can then be used for downstream tasks such as topic modeling, document indexing, and similarity retrieval. One can also take advantage of existing open source projects and frameworks that share pre-trained models created by researchers and practitioners, such as gensim (Rehurek & Sojka, 2010).

Classifier training. Text data is represented by numerical feature vectors at the end of the previous feature extraction phase. These data can then be used as training data to train generic classification algorithms proposed in the machine learning field, called classifiers, such as KNN (Cover & Hart, 1967), SVM (Joachims, 1998), and Random Forest (Parmar et al., 2018). Recent models like eXtreme Gradient Boosting (Chen & Guestrin, 2016) (XGBoost) and Light Gradient Boosting Machine (Ke et al., 2017) (LightGBM) have the potential to improve a lot the classification performance.

2.1.2 Deep Learning-based Models

Deep Learning (DL) is a subset of Machine Learning (ML) that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts and more abstract representations computed in terms of less abstract ones. One of the big advantages of deep learning, and also a key point in understanding why it is becoming popular, is that it is powered by massive amounts of data, which is a consequence of today's well famous information explosion.

The tremendous success of DL models in achieving human-level performance for various computer vision (CV) tasks, such as image classification and object detection, has motivated researchers of other fields like NLP to use DL in their explorations. The primary advantage of deep learning algorithms is their ability to learn high-level features from data in an incremental manner, which eliminates the need for domain expertise and manual feature extraction.

A major difference between deep learning and traditional machine learning models lies in their problem-solving strategies (González-Carvajal & Garrido-Merchán, 2020). As illustrated by Figure 2, DL techniques tend to solve the problem in an end-to-end manner. This means that DL models can learn representations directly from raw input data and map them to the desired output without the need for intermediate steps. This end-to-end learning approach enables deep learning models to capture complex and abstract patterns in the data more effectively without much manual interactions.

On the other hand, traditional machine learning approaches typically require the problem to be broken down into different parts, which are solved individually. The results of these individual components are then combined later to reach the final output, as depicted in Figure 1.

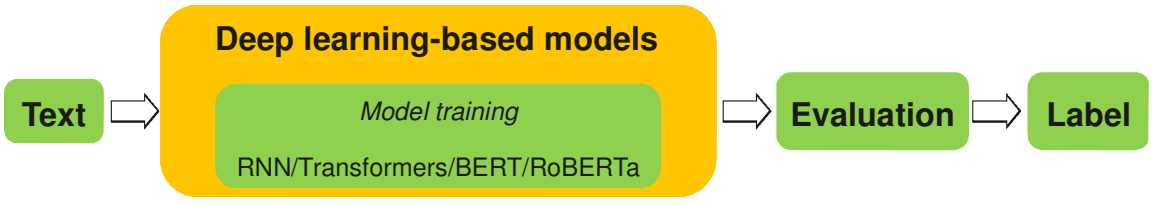

FIGURE 2: The procedure of deep learning-based models for text classification.

In recent decades, numerous deep learning models have been proposed for text classification (Minaee et al., 2021). A notable breakthrough is Google's Bidirectional Encoder Representation

from Transformers (BERT) model (Devlin et al., 2018), which enables the generation of contextualized word vectors. Researchers have explored BERT-based text classification models (Jin et al., 2020), which have shown superior performancein multiple NLP tasks, including text classification, compared to traditional machine learning models. As a pre-trained language model (Qiu et al., 2020), BERT employs unsupervised methods to learn deep bidirectional representations from a large amount of unlabeled text through joint conditioning on both left and right context in every Transformer layer (Vaswani et al., 2017). This allows BERT to use contextual information to predict masked words in a given input sentence. To utilize a pre-trained BERT model for tasks like text classification, one needs to fine-tune the model by adding an additional output layer.

Recently, several BERT-based works have been proposed, such as RoBERTa (Liu et al., 2019), which is considered an enhanced version of BERT. Specifically, RoBERTa was trained on a larger dataset of unlabeled text for a longer period of time and employed a dynamic masking method during training, resultingin a more robust model. RoBERTa-based models have also been developed for languages other than English, such as CamemBERT (Martin et al., 2020) for French.

The advances in the NLP field have led to a wide range of practical challenges that must be addressed in order for these models to be widely used. Both the research and industry communities have made significant efforts to address these challenges, such as *Hugging Face's transformers* (Wolf et al., 2020), an open-source Python library that is designed to be extensible for researchers, simplefor practitioners, and fast and robust in industrial deployments.

2.2 Text Data Augmentation

In the field of machine learning, specifically in supervised learning, class imbalance in terms of the number of instances per class is a well-known problem and challenge. Imbalanced datasets can decrease the performance of machine learning algorithms as the overall accuracy and decision-making are often biased towards the majority class(es), leading to misclassification of minority class(es) (Abd Elrahman & Abraham, 2013; Ali et al., 2019; Vo et al., 2021). A typical example is the binary classification problem that predicts whether or not a patient will have an adverse drug reaction (Santiso et al., 2019). In this research, more than 99% of the samples in the dataset belong to one class, while only a few samples belong to other, which significantly biases the classifier's learning. Therefore, simply increasing the amount of data without considering the data imbalance issue does not necessarily lead to a better solution for the learning problem. However, the quantity of data is still crucial for the quality of supervised classifiers.

Data augmentation methods have been developed to address this issue, which are originated in the field of computer vision and aim to artificially create additional data. In the NLP field, text data augmentation (TDA) involves creating or generating new text data samples from the existing ones. One goal of TDA is to address class imbalance in dataset. An imbalanced dataset is defined as one in which the distribution of classes is not approximately equal (Zhong et al., 2020). TDA can increase the amount of data for a minority class in order to balance the class distribution and improve the robustness of a classifier. Research in TDA is relatively less mature compared to the field of computer vision (Bayer et al., 2021). However, there have been advancements made since 2019, and some general approaches have been proposed recently (Wei & Zou, 2019; Sugiyama & Yoshinaga, 2019; Li et al., 2021).

In (Wei & Zou, 2019), the authors propose Easy Data Augmentation (EDA),a word-level data augmentation technique for text data. For a given input sequence of words (i.e., a sentence), the EDA approach combines four different augmentation strategies:

- Synonym replacement – Randomly replace n words with their synonyms. The WordNet ontology (Miller, 1995) can be used as the synonym dictionary.

- Random synonym insertion – Insert a random synonym of a random word at a random location. Repeat this n times.
- Word deletion – Randomly remove words from the sentence.
- Word order swaps – Randomly choose two words in the sentence and swap their positions. Repeat this n times.

Back translation (Sugiyama & Yoshinaga, 2019) is a sentence-level text augmentation approach. It takes a text input, translates it into the target language, and then translates it back to the source language. For example, to augment a piece of French text, one might first translate it to a corresponding English sentence (i.e., fr→en) and then translate hereafter the English phrase back into the French one (i.e., en→fr). Pre-trained language translation models (Itkonen et al., 2022) are typically used to perform the translations. This process introduces variations in the text while preserving its overall meaning.

Li et al. (2021) propose CLARE, a **C**ontextua**L**ized **A**dversa **R**ial **E**xample generation model that produces fluent and grammatically correct outputs using a mask-then-infill procedure. The model first applies a mask to the input around a given position, and then fills it in using a pre-trained masked language model. To produce the output, CLARE scores and ranks the actions in descending order, which are then iteratively applied to the input. Three contextualized perturbations are proposed: *replace*, *insert* and *merge*, which allow for generating outputs of varying lengths.

Rizos et al. (2019) conducted a study to evaluate the effectiveness of text augmentation techniques in the classification of hate speech. The researchers utilized deep learning-based models, such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), in conjunction with text augmentation. Their findings indicated that text augmentation contributed to improved performance. However, the number of classes in their study was small. In contrast, our research examines the impact of text augmentation on the performance of BERT-based pre-trained language model in a real-world, multiclass classification task, encompassing hundreds of distinct classes.

3. RESEARCH PROBLEM STATEMENT AND SOLUTION

In this section, we detail the context of our research problem, illustrate the difficulties we faced, and describe the solution to tackle the challenge.

As mentioned in Section 2.1, the current context of information overload, particularly in online media services like Medium, makes manual classification of vast amounts of text data both time-consuming and challenging. Furthermore, the quality of manual text classification can be easily affected by human factors. As an online multilingual media platform, Appvizer offers SaaS vendors an easy and flexible way for increasing visibility and traffic for their software. Software owners or vendors can edit a product sheet, select the main category for their software from a list of over 600 categories, and then publish it on the platform. However, this flexibility complicates manual verification of the relevance and accuracy of the categories assigned to these product sheets, which means the website's Search Engine Optimization (SEO) performance cannot be fully guaranteed without additional checks. These checks could be conducted using an accurate classification model capable of predicting the most probable categories for a given software product sheet. That is why we have developed our own classification system and present the results obtained with it in Section 4.

The main challenges of our research problem are as follows:

- *Large multiclass classification*. Unlike other text classification problems that may contain only some dozens of distinct classes, the number of classes (software categories) in our context is quite large (i.e., many hundreds).

- *Data imbalance among classes*. From a business perspective in the SaaS community, the number of instances (software product sheets) of a software category is proportional to the overall popularity of the business activity behind it. For example, the number of Human Resource (HR) or Customer Relationship Management (CRM) software is undoubtedly much more than the number of Zoo management software[6]. As a result, a data imbalance issue is present, as illustrated in Figure 3.

In our initial tests, we observed that these two challenges—large number of classes and class imbalance—were significantly decreasing the quality of the classification and performance (cf. Section 4). Our inductive research aims to investigate whether a simple approach to reducing class imbalance can improve the usefulness of classes with few samples in the classification task, bringing their performance closer to that of classes with many samples and rebalancing their relative importance in the classification outcomings.

With the goal of tackling these challenges and investigating whether reducing the class imbalance could lead to improved performance, particularly in the context of a large number of classes, our research methodology consisted of two main steps:

1. *Data Augmentation*: Utilizing deep learning-based pre-trained language models combined with text data augmentation techniques, we aimed to enhance the quality and quantity of our dataset, particularly for the minority classes. Specifically, we employed the EDA approach (Wei & Zou, 2019) to increase the number of texts for these classes, thereby making the dataset more balanced.
2. *Model Fine-tuning*: Leveraging the power of pre-trained BERT-based models, we fine-tuned these models using the augmented dataset, allowing them to effectively learn from the enriched data and improve their classification performance, even with a large number of classes.

During the data collection phase, we gathered a sample of real-world SaaS product sheets, focusing on ensuring a diverse representation of different classes. We then applied the EDA technique to this dataset, generating an augmented version with a more balanced distribution of classes. Following data augmentation, we moved on to the data analysis stage, which involved fine-tuning the pre-trained BERT-based model using the generated more balanced dataset.

[6] https://www.capterra.com/zoo-software/

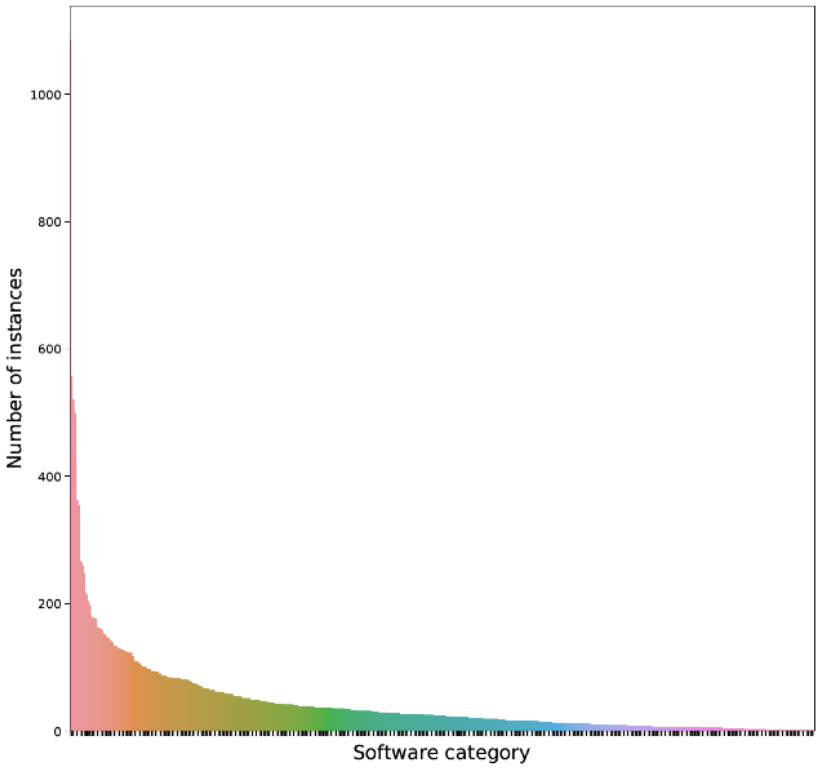

FIGURE 3: The imbalance issue among Appvizer's software categories, e.g. 1000+ instances for CRM software (the first element on the left-hand side of the figure) compared to only 1 instance for Floor Plan software (the first element on the right-hand side of the figure).

4. EXPERIMENTS AND RESULTS

In this section, we present the details of the experiments conducted for our research problem as well as the discussion of the obtained results. The subsection 4.1 presents the experimental setup and evaluation protocol, such as the used dataset, the model training procedure, etc., and the subsection 4.2 discusses the results.

4.1 Experimental Setup

4.1.1 Dataset

The experiments were based on a real-world dataset containing 13014 software product sheets in French. Each product sheet includes a title, description text, and a label representing the software's main category. Since the performance of BERT-based language models appears to decrease for long texts, we adopted the methods proposed by Sun et al. (2019) to divide software descriptions into segments of 128 tokens. To ensure the classifier has sufficient samples for learning useful features, we removed classes with fewer than 10 samples from the initial dataset. This resulted in a dataset containing 20 095 texts and a total of 223 unique classes. We then divided the dataset into train, validation, and test sets using stratified sampling, ensuring that the number of samples for each class in each subset is proportional to the class distribution in the original dataset. Table 1 summarizes the different datasets after the split. For reproducibility, the used dataset is publicly available in our HuggingFace repository[7].

[7] https://huggingface.co/datasets/appvizer/product-sheets-in-french

	Train set	Validation set	Test set
#of instances	16 076	2 009	2 010

TABLE 1: Number of samples in each of the split subset.

4.1.2 Text Data Augmentation

As illustrated in Figure 3, the classification problem exhibits a large imbalance in the distribution of the target classes, resulting in the classifier potentially being biased during training. We adopted the EDA approach (Wei & Zou, 2019) to augment the number of samples for the minority classes in the experiments. Several data augmentation frameworks and libraries have recently been proposed by practitioners in the NLP community, such as Textattack (Morris et al., 2020), TextAugment (Marivate & Sefara, 2020), and nlpaug . We adopted the Textattack Python library in our experiments. Specifically, for software categories with fewer samples than the median value of the category distribution, we used EDA to generate new text samples from existing ones in the training set. After the data augmentation process, the number of samples in the updated train set increased from 16 076 to 80 247. Table 2 illustrates an example of text data augmentation with EDA. The distribution of software category labels in the updated train set is illustrated in Figure 4.

Original text	Augmented text
Parcours d'achat avec **localisation** automatique précise du client pour **augmenter** votre taux **de** conversion	Parcours d'achat avec **place** automatique précise du client pour **élever** votre taux conversion
Créer, **gérer** et réorganiser les stratégies de **prix**	Créer, **traiter** et réorganiser les stratégies de **récompenser**

TABLE 2: Example of text augmentation using EDA (Wei & Zou, 2019).

4.1.3 Model Training and Evaluation Metrics

For the large multiclass classification task of French texts, we trained CamemBERT (Martin et al., 2020), a state-of-the-art French language model based on the RoBERTa model (Liu et al., 2019). Specifically, we made use of the "camembert/camembert-large" instance in HuggingFace, a platform that is notable for its open source Transformers library built for NLP applications.

BERT-based language models are generally pre-trained on a large amount of text data. For example, the CamemBERT instance we used is pre-trained on 135GB of unlabeled text (Wenzek et al., 2019). This allows the pre-trained model to learn an internal representation of the target language that can then be used to extract features useful for downstream tasks, a process known as language model fine-tuning or transfer learning. Furthermore, researchers have shown that the fact of further pre-training BERT on task-related domain-specific data can improve the model's performance. Therefore, we followed the suggestions in (Sun et al., 2019) by further pre-training CamemBERT with a masked language model on the original training data (cf. Table 1) and then used the further pre-trained model during the fine-tuning for the classification task.

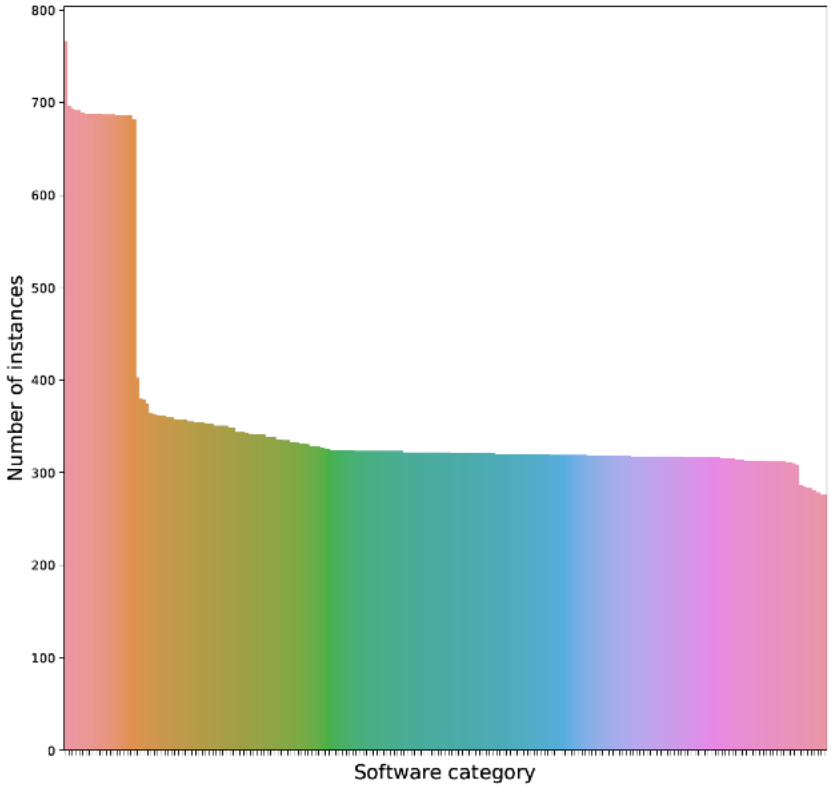

FIGURE 4: The distribution of labels in the train set after data augmentation.

Hyperparameters. We fine-tuned the further pre-trained CamemBERT model for a large multiclass classification task on a Tesla T4 GPU, using the fastbert[10] Python library for model fine-tuning. We used a batch size of 32 and the LAMB optimizer (You et al., 2019) with a warm-up with cosine annealing (Loshchilov & Hutter, 2016) learning rate schedule, a base learning rate of 6e-5, and 500 warm-up steps. We set the maximum number of the epochs to 60. We fine-tuned two instances of the further pre-trained CamemBERT with the same hyper parameter configuration on two different training sets: the original imbalanced train set without data augmentation and the augmented train set. Early stopping was used during training. Finally, we compared the two best models, that we dubbed CamemBERT-No-TDA and CamemBERT-TDA, using the same validation and test sets.

Evaluation metrics. To evaluate the performance of the fine-tuned models, the considered metrics are as follows:

- **Accuracy** on the test set – the fraction of the correct predictions (i.e., the predicted label is the same as the actual label).
- **Top-k accuracy** on the test set – the fraction of the accurate predictions (i.e., the actual label is among the top-k predicted labels ranked by predicted scores).
- **F1-score** on the test set – the harmonic mean of the precision and recall scores, which reaches its best value at 1 and worst score at 0.

In addition, we also considered other metrics observed during the fine-tuning of the models, such as the training/validation loss at the last epoch and the validation accuracy at the last epoch.

[10] https://github.com/utterworks/fast-bert

4.2 Results

In this section, we discuss the results of the conducted experiments. Figure 5 illustrates the results comparing the two models, CamemBERT-No-TDA and CamemBERT-TDA, in terms of the top-*k* accuracy metric (cf. Section 4.1.3).

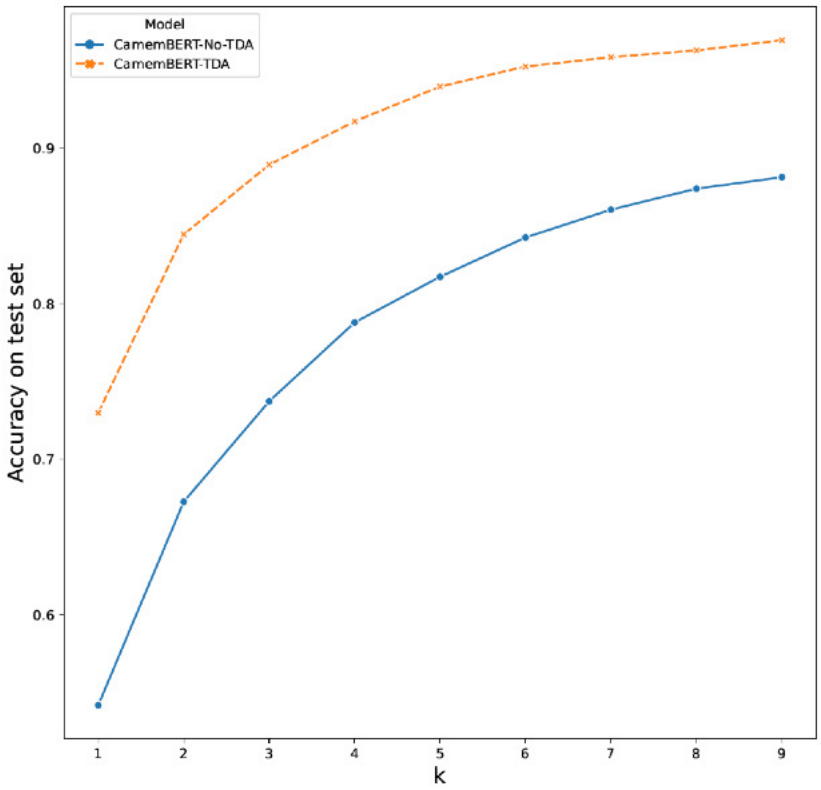

FIGURE 5: Comparison in terms of the top-k accuracy on the test set for fine-tuned CamemBERT model with and without applying TDA (text data augmentation).

Different k values (i.e., from 1 to 9) are considered. Figure 5 shows that the top-k accuracy values of the two models (curves) tend to increase as the value of k increases. This is expected because a larger value of k increases the probability of the true label being present in the top-k predicted labels. In addition, a clear improvement is observed when comparing the two models: CamemBERT-TDA clearly outperforms CamemBERT-No-TDA in terms of top-k accuracy for all considered k values. This demonstrates that applying text data augmentation techniques (TDA) to address class imbalance can significantly improve the top-k accuracy of BERT-based language models, even for a large multiclass text classification task.

Table 3 illustrates the results comparing the two models in terms of accuracy, F1-score and other considered evaluation metrics as described in Section 4.1.3. As shown in Table 3, the CamemBERT-TDA model clearly outperforms the CamemBERT-No-TDA model in terms of all the considered evaluation metrics. Specifically, the application of text data augmentation allows for a 34.7% improvement in accuracy and a 37.1% improvement in F1-score on the test set for the CamemBERT-TDA model compared to the CamemBERT-No-TDA model. In addition, during the fine-tuning phases, the CamemBERT-TDA model showed a 50.8% improvement in training loss, a 68.8% improvement in validation loss, and a 34.8% improvement in validation accuracy compared to the CamemBERT-No-TDA model.

	Model	
Metric	**CamemBERT-TDA**	**CamemBERT-No-TDA**
Accuracy	**0.730**	0.542
F1-score	**0.731**	0.533
Train loss	**0.290**	0.590
Validation loss	**0.670**	2.150
Validation accuracy	**0.790**	0.515

TABLE 3: Performance comparison on the test set for fine-tuned CamemBERT model with and without TDA (text data augmentation). Better results are bolded.

Thus, according to the obtained results, the answers to the main research questions (RQ) tackled in this paper are:

- **RQ1**: Do BERT-based pre-trained language models perform for classification tasks of large multiclass system?
  Answer: For a large real-world multiclass classification task (233 unique labels), BERT-based language models like CamemBERT can achieve considerable performance. Moreover, some simple data pre-processing can easily and significantly improve prediction performance.

- **RQ2**: Is text data augmentation capable of boosting the performance of BERT-based models for multiclass classification with large number of classes?
  Answer: Yes. Text data augmentation can effectively improve the performance of BERT-based language models in terms of various evaluation metrics in the context of the large multiclass classification. It serves as an example of simple data pre-processing that can improve prediction performance.

5. DISCUSSIONS, LIMITATIONS & IMPLICATIONS

In this section, we will discuss the limitations and the implications of our research.

The first limitation is that we excluded labels with very few instances for training during the experiments. Specifically, we used a threshold of 10 to determine whether to consider a label. As this research investigates the effectiveness of using text data augmentation (TDA) to increase the number of instances for minority classes, one might question why we did not simply use TDA to increase the amount of data for these labels instead of ignoring them. Indeed, one could opt to employ TDA to create more instances even for a label with only one instance. However, since the augmentation phase relies solely on existing data, generating more instances based on just a few data points might be less effective. In this scenario, the generalization capability of the resulting model could be questionable. Furthermore, it is not straightforward to determine the optimal threshold value, as it may depend on factors such as the language model used, dataset quality, and the downstream task itself.

The second limitation of our research is that we did not consider the relationships between labels while using TDA to augment text for minority classes. For instance, in our dataset of software categories in the SaaS domain, we may have two related labels such as “email marketing” and “marketing automation”. Both labels share common features related to marketing, such as “customer segmentation” and “campaign management”. By leveraging data from the “email marketing” label to augment the “marketing automation” label, and vice versa, we could potentially increase the amount of available data for both labels and improve our model’s

performance. However, our approach did not take these relationships into account and instead relied solely on the available data for each individual label. Future research could investigate the use of transfer learning techniques and other approaches that consider the relationships between labels to improve the effectiveness of TDA for augmenting text data for minority classes. It is worth noting that this approach may or may not yield significant improvements in model performance, and it could potentially reduce the distinction between these related classes, making it more challenging for the model to accurately differentiate between them. Nevertheless, exploring this avenue could help better understand the trade-offs and benefits of considering label relationships in the context of text augmentation.

One approach could involve using unsupervised keyword detection methods like Textrank (Mihalcea & Tarau, 2004) to locate unique keywords in the siblings' data for TDA. This can help identify and preserve the most informative and distinctive features of each class, potentially improving the performance of our models in distinguishing between related classes. In addition to keyword detection, other unsupervised methods such as clustering or topic modeling can be used to identify patterns and similarities in the data, which can then be used to guide the TDA strategy. For example, if the data contains groups of related classes sharing common features, clustering methods can be used to detect these groups and apply different TDA strategies to each group based on their specific characteristics.

Another approach to improve the effectiveness of TDA strategies is to incorporate domain-specific knowledge by means of ontologies and expert guidance. This can help ensure that the augmented data remains semantically meaningful and retains the important domain-specific features necessary for distinguishing between related classes.

Our research has significant implications for practitioners working with text data across various domains. First, our findings demonstrate the effectiveness of BERT-based pre-trained models for tasks of large multiclass classification system, which has received relatively little attention in previous studies. Few studies have explored the performance of BERT-based models for tasks involving a large number of labels, and our research contributes to filling this gap in the literature. By fine-tuning a BERT-based model on a real-world dataset in the SaaS domain, we achieved high accuracy and F1-score across numerous labels. This suggests that practitioners working with datasets containing many labels can leverage pre-trained models like BERT to perform classification tasks effectively.

Second, our research demonstrates the effectiveness of text data augmentation (TDA) in improving the performance of models trained on imbalanced text data, which can be particularly useful in many real-world applications.

Third, our research highlights the importance of considering imbalanced datasets in text classification tasks. As we demonstrated, using text data augmentation (TDA) techniques can be effective in addressing the issue. Our findings suggest that practitioners should carefully consider the class distribution in their text datasets and adopt appropriate strategies to mitigate the impact of class imbalance on their models' performance.

Overall, our research provides valuable insights and practical guidance for practitioners working with text data in various applications.

6. CONCLUSIONS & FUTURE WORKS

Deep language models such as BERT have proven to be accurate and effective for many natural language processing and understanding tasks, such as binary classification or text multiclass classification system with few classes (Devlin et al., 2018; Liu et al., 2019; Marivate & Sefara, 2020; Bhopale & Tiwari, 2021; Kaliyar et al.,2021). However, few studies have focused on understanding the performance of these models in large multiclass problems with hundreds of unique labels. Another challenge addressed in this study is the class imbalance issue within the

classification task, which can significantly impact and bias the training of the model. Our research contributes to fill the gap in the literature by investigating the performance of BERT-based pre-trained language models in tasks of large multiclass classification system, which has been relatively underexplored. In this article, we investigate a large, imbalanced multiclass text classification problem using BERT-based pre-trained language models in a real-world scenario within the SaaS domain. Text data augmentation (TDA) approaches, such as EDA (Wei & Zou, 2019), are applied to the training set to generate new data for minority classes and reduce the imbalance in the data.

The findings of our research are as follows: 1) pre-trained BERT-based models, when further pre-trained on domain-specific data, can achieve considerable performance in large multiclass text classification tasks, yielding reasonable accuracy and F1-measure scores; 2) augmenting the text data for minority classes with EDA (Wei & Zou, 2019) significantly improves the prediction performance of BERT-based language models; and 3) combining BERT-based pre-trained models with text data augmentation techniques proves advantageous when addressing large imbalanced multiclass classification tasks.

Building upon these findings, our research may have some practical implications for various industries and applications, particularly in domains that may involve numerous classes, such as SaaS software categorization, document categorization, and content recommendation systems. The benefits of our research include improved classification accuracy and the potential for more robust and reliable text classification systems. The target audience for this study includes practitioners working for industries that rely on text classification algorithms for various applications. By implementing the methodology proposed in this study, one can enhance the effectiveness of the text classification systems and ultimately improve the overall performance of their applications.

In future work, we plan to explore the performance of our approach using multilingual NLP models such as XLM (Conneau & Lample, 2019) and XLM-R (Conneau et al., 2020).

7. REFERENCES

Abd Elrahman, S. M., & Abraham, A. (2013). A review of class imbalance problem. *Journal of Network and Innovative Computing*, 1, 332–40.

Ali, H., Salleh, M. N. M., Saedudin, R., Hussain, K. & Mushtaq, M. F. (2019). *Imbalance class problems in data mining: A review. Indonesian Journal of Electrical Engineering and Computer Science*, 14(3), pp. 1560-1571.

Althobaiti M. J. (2020). Automatic Arabic Dialect Identification systems for Written Texts: A survey. *International Journal of Computational Linguistics (IJCL)*. Vol (11), Issue (3).

Baeza-Yates, R., Ribeiro-Neto, B. et al. (1999). *Modern information retrieval*, volume 463. ACM press New York.

Balakrishnan, V., & Lloyd-Yemoh, E. (2014). Stemming and lemmatization: A comparison of retrieval performances. *Lecture Notes on Software Engineering*, 2, 262–7. doi:10.7763/LNSE.2014.V2.134.

Bayer, M., Kaufhold, M.-A., & Reuter, C. (2021). A survey on data augmentation for text classification. *ACM Computing Surveys*, 55. doi:10.1145/ 3544558.

Bhopale, A. P., & Tiwari, A. (2021). An application of transfer learning: Fine- tuning bert for spam email classification. In *International Conference on Machine Learning and Big Data Analytics* (pp. 67–77). Springer. doi:10. 1007/978-3-030-82469-3_6.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acmsigkdd international conference on knowledge discovery and data mining* KDD '16 (pp. 785–94). doi:10.1145/2939672. 2939785.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 8440–51). doi:10.18653/v1/2020.acl-main.747.

Conneau, A., & Lample, G. (2019). Cross-lingual language model pretraining. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (p. 7059–7069).

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13, 21–7. doi:10.1109/TIT.1967. 1053964.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805,. doi:10.48550/arXiv.1810.04805.

González-Carvajal, S & Garrido-Merchán, E. C. (2020). Comparing BERT against traditional machine learning text classification. arXiv preprint arXiv:2005.13012.

Itkonen, S., Tiedemann, J., & Creutz,M. (2022). Helsinki-nlp at semeval- 2022 task 2: A feature-based approach to multilingual idiomaticity detection.*In Proceedings of the 16th International Workshop on Semantic Evaluation* (SemEval-2022) (pp. 122–34). doi:10.18653/v1/2022.semeval-1.14.

Jin, D., Jin, Z., Zhou, J. T., & Szolovits, P. (2020). Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 8018–25). volume 34. doi:10.1609/aaai.v34i05.6311.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning* (pp. 137–42). Springer. doi:10.1007/BFb0026683.

Kaliyar, R. K., Goswami, A., & Narang,P. (2021). Fakebert: Fake news detection in social media with abert-based deep learning approach. *Multimedia tools and applications*, 80, 11765–88. doi:10.1007/s11042-020-10183-2.

Kannan, S., Gurusamy, V., Vijayarani, S., Ilamathi, J., Nithya, M., Kannan, S., & Gurusamy, V. (2014). Preprocessing techniques for text mining. *International Journal of Computer Science & Communication Networks*, 5, 7–16.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* NIPS'17 (p. 3149–3157).

Kowsari, K. JafariMeimandi, K., Heidarysafa, M. Mendu, S. Barnes, L. & Brown, D. (2019). Text classification algorithms: A survey. Information; 10(4), pp. 150.

Li, D., Zhang, Y., Peng, H., Chen, L., Brockett, C., Sun, M.-T., & Dolan, B. (2021). Contextualized perturbation for textual adversarial attack. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 5053–69). doi:10.18653/v1/2021.naacl-main.400.

Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P. S., & He, L.(2022). A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13, 1–41. doi:10.1145/3495162.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M.,Zettlemoyer, L., & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692. doi:10.48550/arXiv.1907.11692.

Loshchilov, I., & Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983, doi:10.48550/arXiv. 1608.03983.

Marivate, V., & Sefara, T. (2020). Improving short text classification through global augmentation methods. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction* (pp. 385–99). Springer. doi:10. 1007/978-3-030-57321-8_21.

Martin, L., Muller, B., Ortiz Suárez, P. J.,Dupont, Y., Romary, L., de la Clergerie, É., Seddah, D., & Sagot, B. (2020). CamemBERT: a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 7203–19). Online: Association for Computational Linguistics, doi:10.18653/v1/2020.acl-main.645.

Mihalcea, R. & Tarau, P., "TextRank: Bringing Order into Text," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, Jul. 2004, pp. 404–411.

Mikolov, T., Chen, K., Corrado, G.,& Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, doi:10.48550/arXiv.1301.3781.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38, 39–41. doi:10.1145/219717.219748.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao,J. (2021). Deep learning–based text classification: a comprehensive review.*ACM Computing Surveys (CSUR*), 54, 1–40. doi:10.1145/3439726.

Mishra, A., & Jain, S. K. (2016). A survey on question answering systems with classification. *Journal of King Saud University-Computer and Information Sciences*, 28, 345–61. doi:10.1016/j.jksuci.2014.10.007.

Morris, J. X., Lifland, E., Yoo, J. Y., Grigsby, J., Jin, D., & Qi, Y. (2020). Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. arXiv preprint arXiv:2005.05909, doi:10.48550/arXiv.2005.05909.
Negesse F. (2015). Classification of Oromo Dialects: A Computational Approach. *International Journal of Computational Linguistics (IJCL)*. Vol. (6), Issue (1)

Parmar, A., Katariya, R., & Patel, V. (2018). A review on random forest: An ensemble classifier. In *International Conference on Intelligent Data Communication Technologies and Internet of Things* (pp. 758–63). Springer volume 26. doi:10.1007/978-3-030-03146-6_86.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pp. (1532-1543). doi:10.3115/v1/D14-1162

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., & Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63, 1872–97. doi:10.1007/s11431-020-1647-3.

Rahman, M. A., & Akter, Y. A. (2019). Topic classification from text using decision tree, knn and multinomial naïve bayes. In *2019 1st International Conference on Advances in Science*, Engineering and Robotics Technology (ICASERT) (pp. 1–4). IEEE. doi:10.1109/ICASERT.2019.8934502.

Rehurek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks*. Citeseer. doi:10.13140/2.1.2393.1847.

Rizos, G., Hemker, K., & Schuller, B. (2019). Augment to prevent: short-text data augmentation in deep learning for hate-speech classification. In *Proceedings of the 28th ACM international conference on information and knowledge management* (pp. 991 – 1000).

Santiso, S., Casillas, A., & Pérez, A. (2019). The class imbalance problem detecting adverse drug reactions in electronic health records. *Health informatics journal*, 25, 1768–78. doi:10.1177/1460458218799470.

Sedai, A., & Houghton B. (2022). Unicode-based Data Processing for Text Classification. *International Journal of Computational Linguistics (IJCL)*, vol (13), issue (2).

Serrano-Guerrero, J., Olivas, J. A., Romero, F. P., & Herrera-Viedma, E. (2015). Sentiment analysis: A review and comparative analysis of web services. *Information Sciences*, 311, 18–38. doi:10.1016/j.ins.2015.03.040.

Shaheen, Z., Wohlgenannt, G. & Filtz, E. (2020). Large scale legal text classification using transformer models. arXiv preprint arXiv:2010.12871.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. (pp. 1–14).

Sinha, K., Jia, R., Hupkes, D., Pineau, J. Williams, A. & Kiela, D. (2021). Masked language modeling and the distributional hypothesis: Order word matters pre-training for little.arXiv preprint arXiv:2104.06644.

Sugiyama, A., & Yoshinaga, N. (2019). Data augmentation using back- translation for context-aware neural machine translation. In *Proceedings of the Fourth Workshop on Discourse in Machine Translation* (DiscoMT 2019)(pp. 35–44). doi:10.18653/v1/D19-6504.

Sun, C., Qiu, X., Xu, Y., & Huang, X. (2019). How to fine-tune bert for text classification? In *China national conference on Chinese computational linguistics* (pp. 194–206). Springer volume 11856. doi:10.1007/978-3-030-32381-3_16.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Vo, M. T., Vo, A. H., Nguyen, T. Sharma, R., & Le, T. (2021).Dealing with the class imbalance problem in the detection of fake job descriptions. *Comput. Mater. Continua*, 68(1), pp. 521- 535.

Wei, J., & Zou, K. (2019). Eda: Easy data augmentation techniques for boosting performance on text classification tasks. arXiv preprint arXiv:1901.11196, doi:10.48550/arXiv.1901.11196.

Wenzek, G., Lachaux, M.-A., Conneau, A., Chaudhary, V., Guzmán, F., Joulin, A., & Grave, E. (2019). Ccnet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the Twelfth Language Resources and Evaluation Conference* (pp. 4003–12).

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M. et al. (2020). Huggingface's transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 38–45), doi:10.18653/v1/2020.emnlp-demos.6.

Yang, J., Bai, L., & Guo, Y. (2020). A survey of text classification models. *In Proceedings of the 2020 2nd International Conference on Robotics, Intelligent Control and Artificial Intelligence*, pp. 327-334.

You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., & Hsieh, C.-J. (2019). Large batch optimization for deep learning: Training bert in 76 minutes. arXiv preprint arXiv:1904.00962, doi:10.48550/arXiv.1904.00962.

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. Advances in neural information processing systems, 28.

Zhang, Y., Jin, R., & Zhou, Z.-H. (2010). Understanding bag-of-words model: a statistical framework. *International journal of machine learning and cybernetics*, 1,43–52. doi:10.1007/s13042-010-0001-0.

Zhong, Z., Zheng, L., Kang, G., Li, S., & Yang, Y. (2020). Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 13001–8). volume 34. doi:10.1609/aaai.v34i07.7000.