UNIVERSIDADE FEREDAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA - CIN
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

JOSÉ MATHEUS LACERDA BARBOSA

IMPROVING BINARY CLASSIFIERS ON IMBALANCED DATA USING LARGE LANGUAGE MODELS

Recife
2023

JOSÉ MATHEUS LACERDA BARBOSA

IMPROVING BINARY CLASSIFIERS ON IMBALANCED DATA USING LARGE LANGUAGE MODELS

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Banco de Dados
Orientador: Luciano de Andrade Barbosa

Recife
2023

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

B238i Barbosa, José Matheus Lacerda
Improving binary classifiers on imbalanced data using large language models / José Matheus Lacerda Barbosa. – 2023.
53 f.: il., fig., tab.

Orientador: Luciano de Andrade Barbosa.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2023.
Inclui referências.

1. Banco de dados. 2. Aprendizado por indução. I. Barbosa, Luciano de Andrade (orientador). II. Título.

025.04 CDD (23. ed.) UFPE - CCEN 2023-173

JOSÉ MATHEUS LACERDA BARBOSA

"**IMPROVING BINARY CLASSIFIERS ON IMBALANCED DATA USING LARGE LANGUAGE MODELS**"

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Aprovado em: 12/07/2023.

Orientador: Luciano de Andrade Barbosa

BANCA EXAMINADORA

Prof. Dr. Tsang Ing Ren
Centro de Informática / UFPE

Prof. Dr. Cláudio de Souza Baptista
Departamento de Sistemas e Computação / UFCG

Prof. Dr. Luciano de Andrade Barbosa
Centro de Informática / UFPE

I want to dedicate this work to my family and friends to thank them for their help and patience through this journey.

ACKNOWLEDGEMENTS

Firstly, I would like to thank Professor Dr. Luciano de Andrade Barbosa for all the guidance and teachings that made the completion of this work possible.

To my friends and family for their understanding and patience during important moments when I couldn't be present.

I would like to thank CAPES for the financial support, which was undoubtedly a key factor in the completion of this work.

ABSTRACT

In the realm of real-world classification tasks, the challenge of imbalanced data frequently hinders the efficacy of machine learning models in performing accurate binary classifications. To address this issue directly, this study introduces "BALANCE," a novel framework designed to rectify data imbalance in text datasets for binary classification. BALANCE leverages prompt-based learning to efficiently generate synthetic data that mimics the characteristics of the minority class. This is achieved by optimizing the decoding parameters of a specific natural language generation model and tailoring text generation to the minority class. A customized prompt is subsequently employed to generate instances using the fine-tuned language model. We conducted a comprehensive experimental evaluation using three imbalanced real-world text classification datasets. The findings of our study reveal that BALANCE consistently outperforms existing methods for data creation and imbalance correction in the majority of scenarios. These results underscore the high quality of the generated instances and the potential of BALANCE to significantly enhance the performance of text classification models when dealing with imbalanced data.

Keywords: data augmentation; prompt-based learning; data balaning; large language model.

RESUMO

No âmbito das tarefas de classificação do mundo real, o desafio de dados desequilibrados frequentemente prejudica a eficácia dos modelos de aprendizado de máquina na realização de classificações binárias precisas. Para abordar esse problema diretamente, este estudo introduz "BALANCE", um novo *framework* projetado para corrigir o desequilíbrio de dados em conjuntos de dados de texto para classificação binária. O BALANCE utiliza a aprendizagem baseada em *prompt* para gerar eficientemente dados sintéticos que imitam as características da classe minoritária. Isso é alcançado otimizando os parâmetros de decodificação de um modelo de geração de linguagem natural específico, adaptando a geração de texto à classe minoritária. Em seguida, é empregado um *prompt* personalizado para gerar instâncias usando o modelo de linguagem ajustado. Realizamos uma avaliação experimental abrangente usando três conjuntos de dados de classificação de texto do mundo real desequilibrados. Os resultados de nosso estudo revelam que o BALANCE supera consistentemente os métodos existentes para criação de dados e correção de desequilíbrio na maioria dos cenários. Esses resultados destacam a alta qualidade das instâncias geradas e o potencial do BALANCE para melhorar significativamente o desempenho de modelos de classificação de texto ao lidar com dados desequilibrados.

Palavras-chaves: aumento de dados; aprendizado por indução; balanceamento de dados; grandes modelos de linguagem.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1 INTRODUCTION

1.1 PROBLEM AND MOTIVATION

Imbalanced data is a common issue in real-world classification tasks (SHAIKH et al., 2021). This problem arises in various applications, including the healthcare sector, oil spill detection, credit card fraud detection, cultural modeling, network intrusion detection, text categorization, and more (PATEL et al., 2020). Imbalanced data poses a challenge as it skews the model's focus toward the majority class, potentially affecting the model's performance on the minority class due to the insufficient number of training instances for accurate predictions.

Various strategies have been developed to tackle this issue (LIN; CHEN; QI, 2020). There are three main approaches to addressing data imbalance problems (KAUR; PANNU; MALHI, 2019): data-level, algorithm-level, and hybrid approaches.

Data-level approaches modify the training set distribution to increase or decrease the class imbalance. Examples of methods that are considered data-level approaches are Oversampling and Undersampling. Oversampling (SOMTE (CHAWLA et al., 2002)) increases the number of instances of the minority class until it reaches the number of examples in the majority class. Duplication of samples is done using random sampling to augment the training data(LIN; CHEN; QI, 2020). This method of Oversampling is prone to deal with overfitting (LIN; CHEN; QI, 2020). Undersampling is used to decrease the number of instances of the majority class until it equals the number of examples in the minority class (LIN; CHEN; QI, 2020). This method randomly selects and deletes the majority of class instances from the training set. Removing instances can cause a loss of essential information for the classifier (LIN; CHEN; QI, 2020).

Algorithm-based strategies adjust the learning or decision process to increase the importance of the minority class by, for instance, adding a penalty to examples in the majority class. In this direction, (CHEN; BREIMAN, 2004) present a modified Random Forest algorithm called Balanced Random Forest. This modification is supposed to do precisely what is algorithm-based, balancing the importance of the classes, mainly toward the minority class, and learning the weights for all points on the dataset, which will help the classification task.

The combination of these two previous approaches is called hybrid strategies. Here, a data-level approach modifies the training set distribution to learn the importance of the classes during the training process, which happens in the algorithm-level approaches. Examples of hybrid methods are described in (SEIFFERT et al., 2010) and (HIDO; KASHIMA; TAKAHASHI, 2009).

Previous work has demonstrated that such techniques are helpful for tabular and

image data (KOZIARSKI, 2021; RUPAPARA et al., 2021). Different from these data types, text can contain grammatical structure, context, and semantic information, making it more challenging to deal with. Particularly for text, data augmentation methods have been proposed to generate synthetic examples that can be used to balance the training set (BAYER; KAUFHOLD; REUTER, 2022) by applying transformations to the original text instances. These strategies can replace words in the original text with their synonyms, add or remove random words, or use back-to-back translation (WEI; ZOU, 2019a; KARIMI; ROSSI; PRATI, 2021; ZHAO et al., 2022; LIU et al., 2020). Because it is challenging to develop generalized rules for language transformation, universal data augmentation techniques in Natural Language Processing are a big challenge in this field (WEI; ZOU, 2019b).

We propose in this work to use Prompt-based learning (PBL) (LIU et al., 2021) to improve binary text classifiers on imbalanced data. PBL leverages pre-trained language models trained on a large volume of data to perform a downstream task. It does so by passing task-tailored instructions to the model, which as a result, generates text accordingly. Our solution, BALANCE, applies PBL to generate synthetic text instances of the minority class to balance the training data. To do so, BALANCE first tunes the decoder hyper-parameters of a given pretrained language model to maximize the generation of minority class instances. Then, it uses the optimized language model to generate minority class examples, which are added to the training data if considered relevant by a filtering classifier.

We have performed an extensive experimental evaluation on three real-world datasets of binary text classification: clickbait detection, rumor discovery, and sentiment analysis. The results show that BALANCE outperforms previous imbalanced and data augmentation strategies in two of the datasets, and the BERT model fine-tuned on the original imbalanced data achieved the best result in the third scenario dataset.

1.2 RESEARCH QUESTIONS

In this work, we answer the following research questions:

1. Generating high-quality synthetic data from Large Language Models. Can Large Language Models produce high-quality synthetic text instances of a given class?

2. Enhancing Model's Performance by Balancing Datasets. Can the synthetic text instances of the minority class generated by LLMs improve the performance of text classifiers on imbalanced data?

1.3 WORK ORGANIZATION

The remainder of this work is organized as:

Chapter 2 presents the basics concepts, terminologies, and researches that helped to accomplish this work. Works and researchers in the fields of Transformers, Natural Language Processing, Natural Language Generation, Text Generation, Prompt-based Learning, and methods to deal with Imbalanced Data.

Chapter 3 discusses the directly related work, what are the state-of-the-art in the field that we are working on. We presented the current methods used to deal with imbalanced text data and the task of text classification. We discuss Traditional Methods used in the literature to tackle the imbalanced problem in text datasets as well as the Augmentation Methods, which are the newest methods applied to text datasets.

Chapter 4 presents our method, BALANCE, its architecture, and its workflow. In this chapter, we show how BALANCE works, why and all the methodologies used in the construction of this idea. From the initial model training, going through the Large Language Model (LLM) tuning, and finally explaining the Text Generation stage.

Chapter 5 shows the setup of experiments, presents the datasets used in this section, the augmentation strategies compared during the experimentation, and the metrics used to compare the methods to be fair when showing the results.

Chapter 6 discusses the contributions, and limitations, and presents some ideas and approaches to work on in the future to improve more and more this job.

2 FUNDAMENTALS

This chapter introduces the fundamental concepts utilized throughout this work. Section 2.1 introduces the Transformer Architecture and the models that employ this architecture. Section 2.2 explores the concepts of Text Generation and Natural Language Generation. Section 2.3 delves into the concept of Prompt-based Learning and its various iterations. Finally, in Section 2.4 at the conclusion of this chapter, we will examine methods for addressing imbalanced data.

2.1 TRANSFORMERS

Transformer is a network architecture proposed in (VASWANI et al., 2017) based on the attention mechanism. This architecture was proposed as an alternative to sequence-based ones such as recurrence and convolution.

Similar to previous most competitive neural models, used for sequence transduction, that have an encoder-decoder structure, here these mechanisms are also used. Figure 1 shows the Transformers' architecture.

The original Transformer model is a stack of 6 layers. The output of layer l is the input of layer $l+1$ until the original input pass through all layers and the final prediction is reached. There is a 6-layer encoder stack on the left and a 6-layer decoder stack on the right.

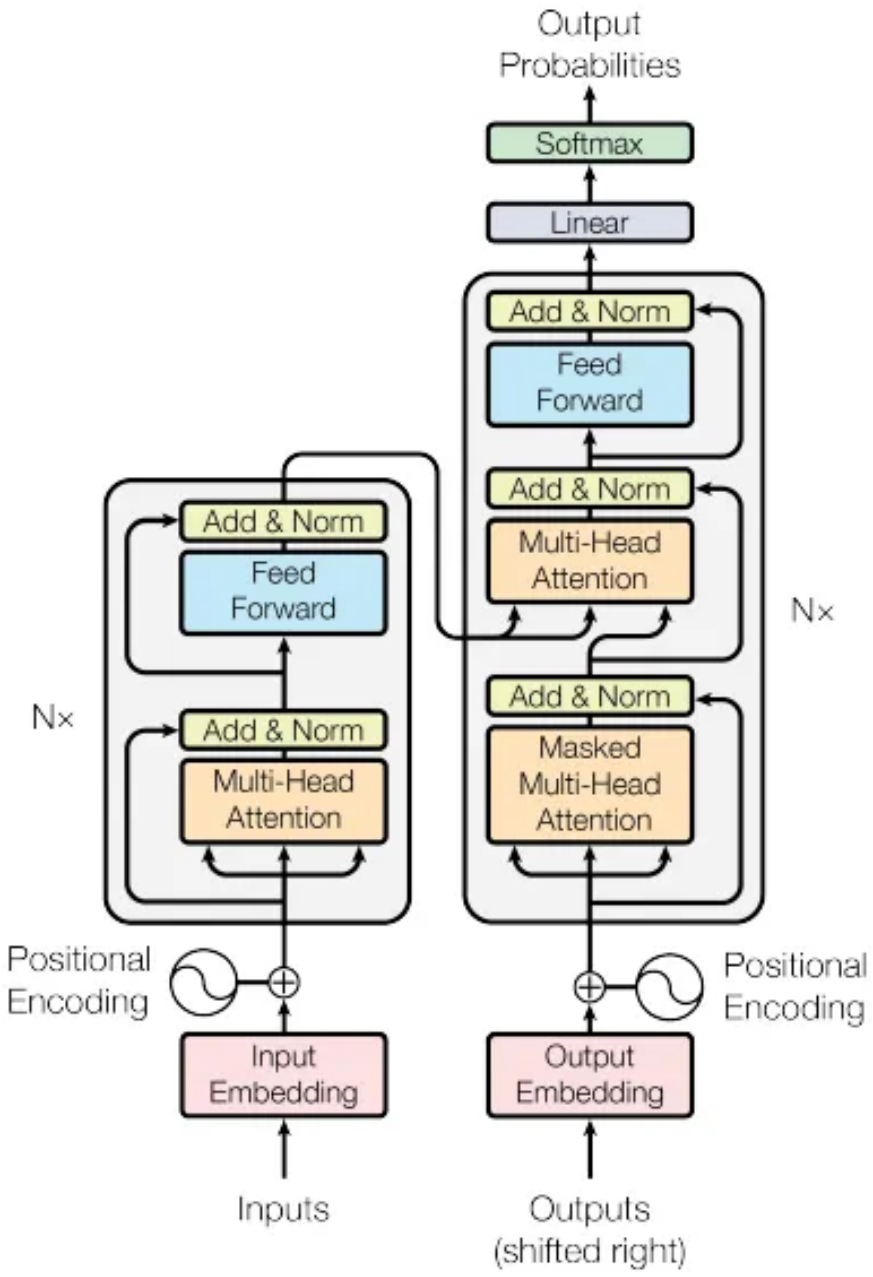
Before the whole process, the inputs are embedded to turn the words into vectors that the network understands, in the Input Embedding stage. In the Positional Encoding step is added the information of the relative position of each word to its respective vector embedding.

On the encoder side, on the left side in Figure 1, the input goes through an Attention sub-layer and a Feed-Forward Network sub-layer. The Attention layer is used to calculate the relation between a current word against all other words in the same sentence, including this current word, this process is called Self-Attention. In the Transformers' architecture, all words in the sentence are passed together, which is different from a sequence-to-sequence model, thus, we need several blocks to do the calculating of Attention, and in this architecture, we call it Multi-Head Attention.

The process of Self-Attention will generate vectors to represent the words and their relationships with the other words in the same input, thus turning this information digestible to the next encoder o decoder block, we use a Feed-Forward Network.

These two components (Multi-Head Attention and Feed-Forward Network) are used in both, the encoder and decoder sides. On the decoder side, on the right side in Figure1, we have a different component, the Masked Multi-Head Attention layer is supposed to

Figure 1 – Transformer Architecture.



Source: (VASWANI et al., 2017)

mask the sentence and present only one word at a time to the decode, which will turn the model capable of learning what is the next word in the sentence and to correct the model if the next word predicted is wrong of the real next word in the sentence.

To predict this next word, at the end of the architecture, we have Linear and Softmax layers which are used to do that, predict what word has the major odd of being the next word in the sentence.

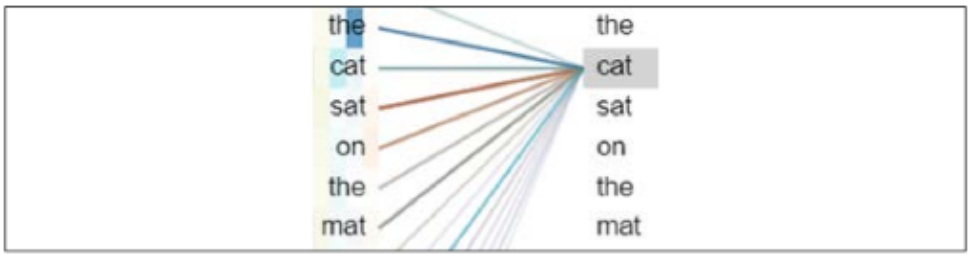
Thus, in this architecture, there is no Recurrent Neural Network, Convolutions Neural Network, or Long-Sort Term Memory, which is the purpose of the Transformer, to replace

these old methods.

These previous methods were replaced specifically by the Attention Mechanism. The Attention Mechanism is a "word-to-word" operation. It will find how each word is related to all other words in a sequence, including the word being analyzed itself.

Figure 2 shows an example of the function of the Attention Mechanism for the phrase: "The cat sat on the mat." Attention executes dot products between word vectors and determines the strongest relationships of a word among all the other words. This procedure will provide a deeper relationship between words and consequently, will produce better results.

Figure 2 – Example of the function of the Attention Mechanism.



Source: (ROTHMAN, 2021)

Many models use the Transformer architecture, like BERT (DEVLIN et al., 2018), GPT (RADFORD et al., 2018), GPT-2 (RADFORD et al., 2019a), GPT-3 (BROWN et al., 2020), T5 (RAFFEL et al., 2020), and other models.

2.2 TEXT GENERATION

Natural Language Generation (NLG) is a sub-field of Natural Language Processing (NLP), which deals with building software systems that can generate coherent text and readable (CELIKYILMAZ; CLARK; GAO, 2020).

Commonly, Natural Language Generation is considered a general term, which encompasses a wide range of tasks that take input (e.g., a structures input like a dataset or a table, a natural language prompt, or even an image) and output a sequence of text that is coherent and understandable by humans (CELIKYILMAZ; CLARK; GAO, 2020). With this general definition, Natural Language Generation can be applied to a range of different tasks in Natural Language Processing, which are: Question answering, Machine translation, Long text generation, and Summarization. Other tasks can be done by Natural Language Generation when the task is modeled in terms of an input as in NLG and the output is a sequence of tokens.

Text Generation systems usually take linguistic or non-linguistic inputs. Text generation tasks can be categorized based on the type of inputs (JIN et al., 2020): meaning-to-text, data-to-text, text-to-text, visual-to-text, and code-to-text.

Meaning-to-text (FLANIGAN et al., 2016; POURDAMGHANI; KNIGHT; HERMJAKOB, 2016; KONSTAS et al., 2017; BANARESCU et al., 2013) generation takes concepts or meaning representation as inputs and a popular meaning representation is abstract meaning representation. Data-to-text (MEI; BANSAL; WALTER, 2016; PEREZ-BELTRACHINI; LAPATA, 2018; SHA et al., 2018) generation takes structured data records or tables as an input. Text-to-text (NALLAPATI; ZHAI; ZHOU, 2017; NARAYAN; COHEN; LAPATA, 2018; NALLAPATI et al., 2016; RUSH; CHOPRA; WESTON, 2015; LI et al., 2018; GUPTA et al., 2018) generation takes natural language texts or sentences as inputs. Typical tasks include text summarization and paraphrase generation. Visual-to-text generation takes visual information such as images and videos as inputs. Examples of visual-to-text (LI et al., 2019) tasks are image captioning and visual storytelling. Code-to-text (GUPTA; EKBAL; BHATTACHARYYA, 2020) generation takes code fragments as inputs. Can exist other types of inputs to Natural Language Generation systems will characterize another type of these systems.

2.3 PROMPT-BASED LEARNING

In traditional supervised learning, a model takes an input x and predicts an output y: $P(y|x;\theta)$ (LIU et al., 2023). Thus, to model the conditional probability $P(y|x;\theta)$ and learn the parameters θ to do the predictions, we need to use a dataset containing pairs of inputs and outputs.

In the Text Classification task, a model receives a text as input and predicts the label of this text. For example, as input $x =$ "I love this movie." and the model predicts a label $y = pos$, classifying this text as a positive text.

Thus, to do traditional supervised learning is necessary to have labeled data for the target task. This is the a problem with this approach because, for many tasks, finding an amount of labeled data, where we have a pair of an input x and an output y for training a model, can be challenging (LIU et al., 2023).

In this scenario, Prompt-based learning is introduced to deal with this problem. Instead of modeling a conditional probability directly of the input text, prompt-based learning defined three steps to predict the highest scoring for the output y.

- **Prompt Addition** - Here is applied a function, (a prompting function) to modify the input text into a prompt (LIU et al., 2023). The authors explain this process in two steps:
  - Apply a template, which is a textual string that has two slots: an input slot $[X]$ for input x and an answer slot $[Z]$ for an intermediate generated answer text z that will later be mapped into y.
  - Fill slot $[X]$ with the input text x.

The authors gave an example of Prompt Addiction, when $\mathbf{x}$ = "I love this movie.", the template may take a form such as "$[X]$ Overall, it was a $[Z]$ movie.". Then, for the example, the prompt will be "I love this movie. Overall it was a $[Z]$ movie."

- **Answer Search** - this strategy searches for the highest scoring text z that maximizes the odds of the Language Model predicting correctly the text. We first define Z as a set of possible values for z. Z could range from the entire language in the case of generative tasks or could be a small subset of the words in the language in the case of classification, such as defining $Z = \{$ "*excellent*", "*good*", "*OK*", "*bad*", "*horrible*"$\}$ to represent each of the classes in $Y = \{++, +, \ , -, --\}$.

  We then define a function that fills in the slot $[Z]$ in the prompt with the potential answer z. After the slot was filled we called this prompt as *filled-prompt*.

- **Answer Mapping** - After we find the highest-scoring answer predicted by the model, we would like to go from this highest-scoring answer z to the highest-scoring output y. In some cases, this is trivial, because the answer itself is the output, as in language generation, but in other cases, we have to identify in z our output y, as in the classification task.

There are some strategies to use Prompt-based Learning, which are: few-shot and zero-shot learning.

2.3.1 Few-shot Learning

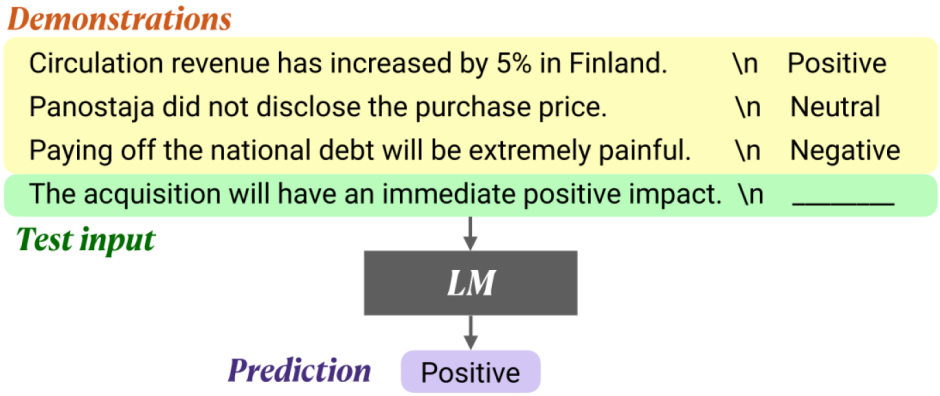
The concept of Few-shot Learning is inspired by the robust reasoning and analytical capabilities of humans (SONG et al., 2023). When humans learn by experience, after being presented with some examples of that task, this concept inspired the concept of Few-shot Learning that emulates this human behavior and teaches a machine/algorithm some task showing limited examples of this task. The definition of Few-shot learning in, (WANG et al., 2020) is a type of machine learning problem, where a limited of examples of a specific task is available.

Examples of the use of prompts with Few-shot Learning are (MIN et al., 2022; BROWN et al., 2020; TOUVRON et al., 2023), Figure 3 shows an example of prompt presented by (MIN et al., 2022).

2.3.2 Zero-shot Learning

Zero-shot learning performs the prediction task without any labeled data. Instead of giving training examples, zero-shot learning receives a high-level description of new categories so that the model can relate them to existing categories that the model has learned about.

Figure 3 – Example of the prompt with few-shot learning.



Source: (MIN et al., 2022)

This description is called a prompt, Table 1 shows some examples of prompts with zero-shot learning used in (MENG et al., 2022). The zero-shot learning strategies have been used in computer vision, machine perception, and natural language processing.

Table 1 – Examples of prompts used for Zero-shot learning. Adapted from (MENG et al., 2022).

Example	Prompt
1	Sentence 1: *input*1 Sentence 2: *input*2 Does Sentence 1 entail Sentence 2? The answer is:
2	Premise: *input*1 Hypothesis: *input*2 Does the premise entail the hypothesis? Options: Yes. No. Maybe. The answer is:
3	Premise: *input*1 Hypothesis: *input*2 What is the relation between the premise and the hypothesis? Options: Entailment. Neutral. Contradiction. The answer is:

Source: (MENG et al., 2022)

2.4 IMBALANCED DATA

Imbalanced data is a common issue in the real-world classification task (SHAIKH et al., 2021) because it biases the model toward the majority class. This can impact the model's performance in the minority class since it may not have enough training instances of this class to predict it accurately. Over the years, several approaches have been proposed to deal with this problem, in this section, we present the different strategies proposed to overcome the imbalanced data issue.

2.4.1 Data-level Approaches

Data-level methods aim to balance the class distribution by manipulating the training samples, including oversampling the minority class, undersampling the majority class, and combinations of the two above methods (LIN; CHEN; QI, 2020).

2.4.1.1 Oversampling

The basic idea of oversampling is to increase the size of the minority class to obtain balanced classes (KAUR; PANNU; MALHI, 2019). This process can be done in different ways. An example of the oversampling process is random oversampling, where samples of the minority class are randomly selected and duplicated, increasing their size.

Synthetic Minority Oversampling Technique - SMOTE (CHAWLA et al., 2002) is a well-known oversampling process. In SMOTE, synthetic samples are produced with the help of minority-class samples, through their nearest neighbors using Euclidean distance.

2.4.1.2 Undersampling

Undersampling is the process that draws a random set of samples from the majority class to balance the classes and the rest of the samples are ignored (KAUR; PANNU; MALHI, 2019). NearMiss (MANI, 2003) is a typical undersample method based on the nearest neighbor algorithm. When two points belonging to different classes are very close to each other in the distribution, this algorithm eliminates the point of the majority class.

2.4.1.3 Hybrid methods

Hybrid methods are those that apply both oversampling and undersampling to handle the problem of imbalanced data (KAUR; PANNU; MALHI, 2019).

To obtain a balanced training data, undersampling is used to delete the instances without useful information according to some criterion. Then oversampling is performed to replicate existing instances. Thus, the proposed method reduces the chances of losing informative instances.

2.4.2 Algorithm-level Approaches

Algorithm-level methods aim to lift the importance of minority classes by improving the existing algorithms, including cost-sensitive learning, ensemble learning, and decision threshold adjustment (LIN; CHEN; QI, 2020).

Bagging and Boosting are examples of algorithm-level methods applied to deal with an imbalanced data problem (LIN; CHEN; QI, 2020). Another example of an algorithm-level method is the Balanced Random Forest (CHEN; BREIMAN, 2004), which improves the original Random Forest algorithm to handle the imbalanced data problem. In the original

Random Forest (BREIMAN, 2001) algorithm an ensemble of decision trees is trained, where at each node of the tree a subset of all attributes is randomly selected and the best attribute on which to further grow the tree is taken from that random set. Random Forest uses the bagging method (bootstrap aggregation), thus, each tree is trained on a set bootstrapped from the original training set. Whereas in the Balanced Random Forest (CHEN; BREIMAN, 2004), each tree is trained with two bootstrapped sets of the same size, equal to the size of the minority class, being one for the minority class and the other for the majority class (DEGÓRSKI; KOBYLINSKI; PRZEPIÓRKOWSKI, 2008).

2.4.3 Hybrid Approaches

Hybrid methods refer to combining the two previous methods (data- and algorithms-level) to handle efficiently the problem of imbalanced data (KAUR; PANNU; MALHI, 2019; SEIFFERT et al., 2010; HIDO; KASHIMA; TAKAHASHI, 2009).

An example of the hybrid approach is RUSBoost(KAUR; PANNU; MALHI, 2019). During the process of training, RUSBoost weights each example, initializing these weights with $\frac{1}{m}$, where m is the number of examples in the training set. Then, T weak hypotheses are trained. In the first step of modeling the hypothesis, RUS - Random Undersampling (data-level strategies) is applied to remove the majority of class examples until $N\%$ is reached and generate a new training dataset, $S^{'}{}_{t}$. This new training set has a new weight distribution, $D^{'}{}_{t}$. In the next step, the new dataset and its new distribution are passed to the base learner to create the weak hypothesis, h_t. Then, a pseudo-loss is calculated between the train with the original dataset and the generated dataset to update the weight parameter α (algorithm-level method). The weights are updated and normalized and passed to the next iteration until the process reaches T iterations and produces the final hypothesis $H(x)$.

2.5 DECODING METHODS

With the increasing interest in open-ended text generation, different methods of decoding were proposed to that be possible to reach different goals when doing this task [1].

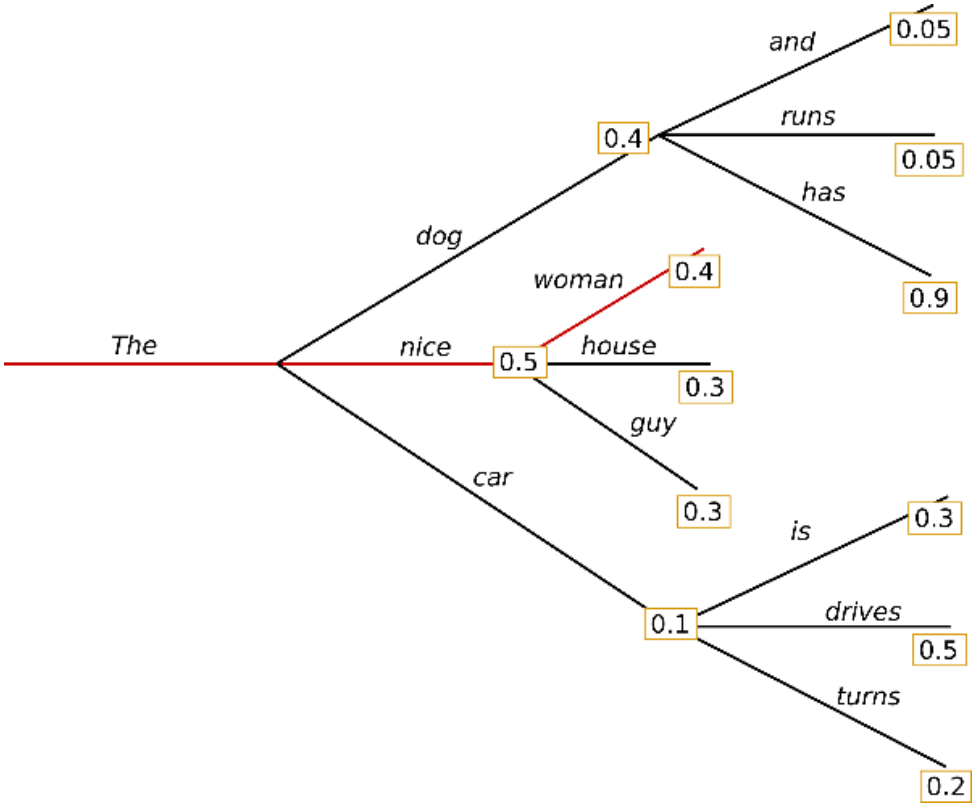
2.5.1 Greedy Search

In Greedy Search, at every time-step(t) during the generation process, we choose the word that has the highest conditional probability. Figure 4 shows an example of the function of Greedy Search.

Starting from the word "The", the algorithm greedily chooses the next word of highest probability "nice" and so on, so that the final generated word sequence is ("The", "nice", "woman") having an overall probability of $0.5 * 0.4 - 0.2$.

[1] <https://huggingface.co/blog/how-to-generate>

Figure 4 – Example of Greedy Search.



Source: (HOW..., 2020)

The major problem of Greedy Search is that it messes high probability words hidden behind a low probability words. The word "has" with its high conditional probability of 0.9 is hidden behind the word "dog", which has only the second-highest conditional probability, so that Greedy Search misses the word sequence "The", "dog", "has".

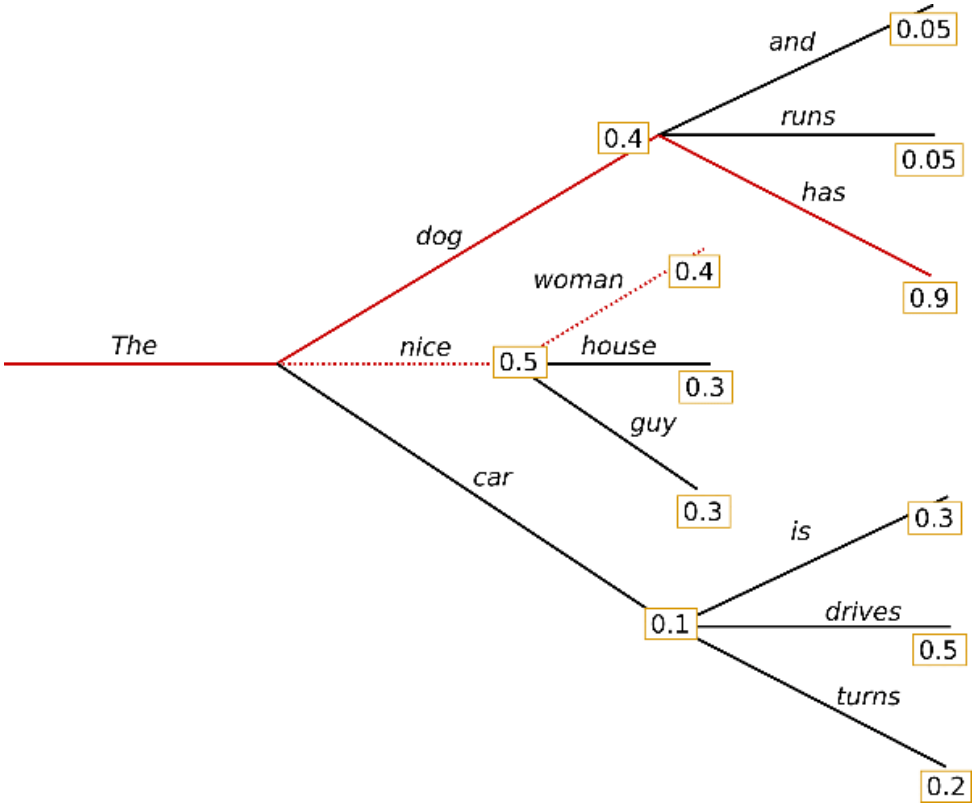
2.5.2 Beam Search

Beam Search deal with the major problem of Greedy Search. Beam Search reduces the risk of missing hidden high-probability word sequences by keeping the most likely *beams* of hypotheses at each time step and eventually choosing the hypothesis that has the overall highest probability. Figure 5 shows how Beam Search works.

At time step 1, besides the most likely hypothesis ("The", "nice"), Beam Search also keeps track of the second most likely one ("The", "dog"). At time step 2, Beam Search finds that the word sequence ("The", "nice", "woman"), which has 0.2. Beam Search will always find an output sequence with the highest probability than greedy search but is not guaranteed to find the most likely output.

There are some reasons that Beam Search might not be the best decoding option when doing open-ended generation: Beam Search can work very well in tasks where the length of the desired generation is more or less predictable as in machine translation or summarization, in the open-ended generation we do not have control of it, like in story

Figure 5 – Example of Beam Search.



Source: (HOW..., 2020)

generation (MURRAY; CHIANG, 2018; YANG; HUANG; MA, 2018); Beam Search heavily suffers from repetitive generation; and (HOLTZMAN et al., 2019), the authors argued that, high-quality human language does not follow a distribution of high probability next words. In other words, as humans, we want generated text to surprise us and not to be predictable, Figure 6 presents the authors' idea.

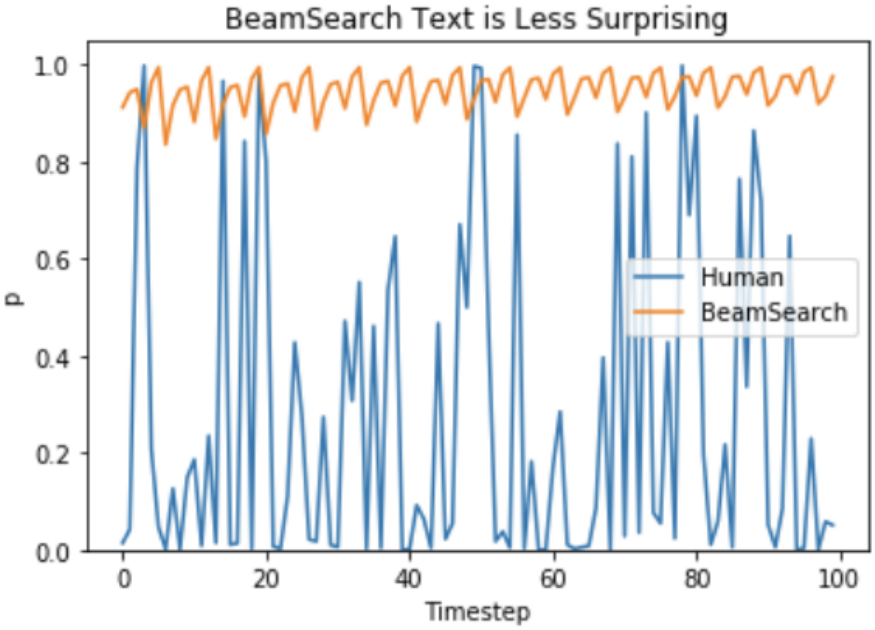
2.5.3 Sampling

A way to be less boring and predictable is to introduce some randomness. Sampling means randomly picking the next word w according to its conditional probability distribution. In Figure 7 we have an example.

In the example we can be seen that the generation is not deterministic, adding more randomness. The word "car" is sampled from the conditioned probability distribution $P(w|"The")$, followed by sampling "drives" from $P(w|"The","car")$.

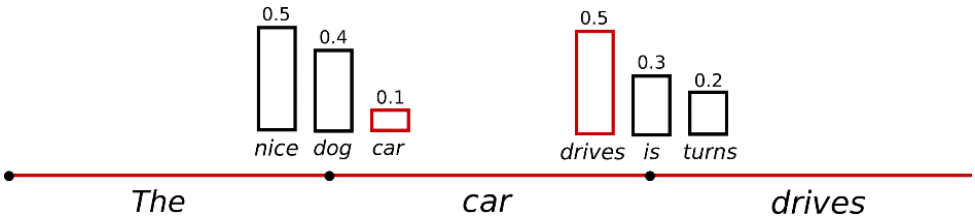
A big problem when sampling word sequences is often the generation is incoherent do not sound like written by a human (HOLTZMAN et al., 2019). A way to tackle this problem is to make the distribution sharper, increasing the likelihood of high-probability words and decreasing the likelihood of low-probability words, adding a variable called *temperature*. Figure 8 shows how it works.

Figure 6 – Beam Search X Human



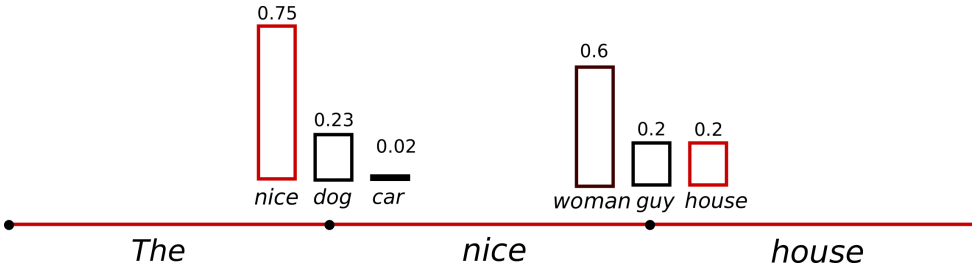
Source: (HOLTZMAN et al., 2019)

Figure 7 – Sampling Search



Source: (HOW..., 2020)

Figure 8 – Temperature - Sampling Search



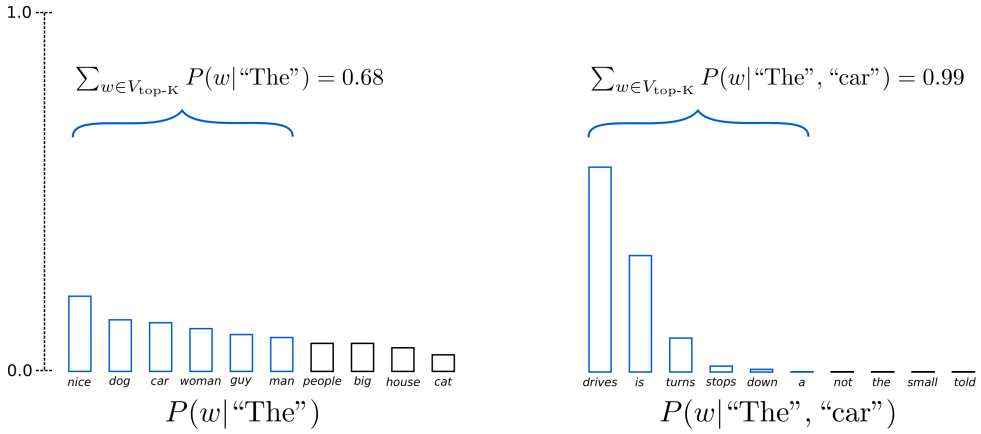
Source: (HOW..., 2020)

The conditional next word distribution becomes much sharper leaving almost no chance for the word "car" to be selected. This parameter regulates the randomness while applying *temperature* the generation can make a distribution less random, thus when *temperature = 0* the generation will work like the Greedy Search, thus finding a good number for *temperature* that fits with the goal generation is the key.

2.5.4 Top-K Sampling

Top-K Sampling was introduced in (FAN; LEWIS; DAUPHIN, 2018), here the k most likely next words are filtered and the probability mass is redistributed among only those k words.

Figure 9 – Top-K Sampling



Source: (HOW..., 2020)

In the example presented in Figure 9 we have $k = 6$, in both sampling steps, we limited our sampling pool to 6 words, dividing all probability distribution among k words.

The main concern about Top-K Sampling is that after defining a number for k it is not dynamically adapted, this can be problematic as some words might be sampled from a very sharp distribution (right side of Figure 9), whereas others from a much flat distribution (left side of Figure 9).

Step $t = 1$ eliminates the possibility of sampling words like, "people", "big", "house", and "cat", which seem like reasonable candidates. Thus, limiting the sample pool to a fixed size k could limit the model to produce gibberish for sharper distributions and limit the model's creativity for flat distribution.

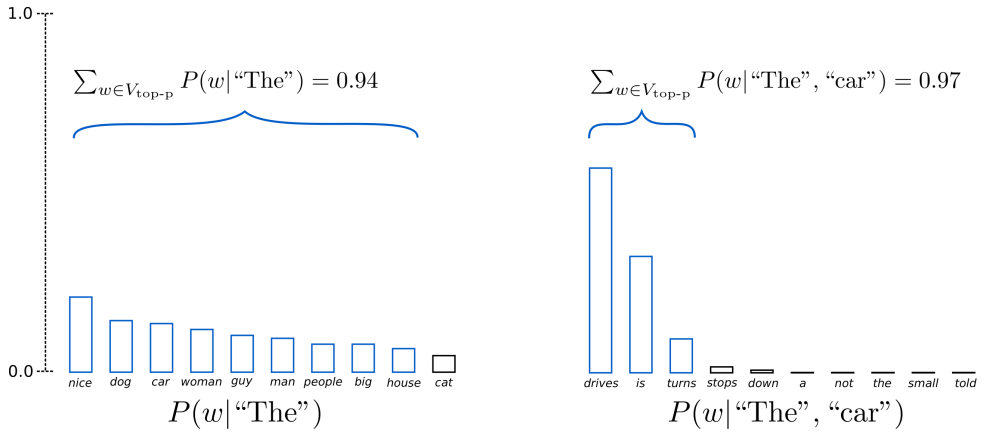
2.5.5 Top-P Sampling (Nucleus)

To tackle the problems present in Top-K Sampling, (HOLTZMAN et al., 2019) created Top-P Sampling (Nucleus).

Instead of sampling only from the most likely k words, Top-P Sampling chooses from the smallest possible set of words whose cumulative probability exceeds the probability p. The probability mass is then redistributed among this set of words. This way, the size of the set of words (the number of words in the set) can dynamically increase and decrease according to the next word's probability distribution. Figure 10 shows how Top-P Sampling works.

Setting $p = 0.92$, Top-P Sampling picks the minimum number of words to exceed together $p = 92\%$ of the probability mass. In the first example (left side of Figure 10),

Figure 10 – Top-P Sampling



Source: (HOW..., 2020)

this included the 9 most likely words, whereas it only has to pick the top 3 words in the second example (right side of Figure 10) to exceed 92%. It can be seen that it keeps a wide range of words where the next word is less predictable, and only a few words when the next word seems more predictable. This resolves the problem in Top-K Sampling of dynamically adapting the number of words in each step.

These are some decoding methods, separately they work in some way, but we can mix some of the different ways to decode open-ended generation and reach our goal when doing this task. By joining these different methods we can produce the generation that we want, altering the parameters of temperature, k, and p we can alter the way the algorithm works and produce the text generation that fits with our problem.

3 RELATED WORK

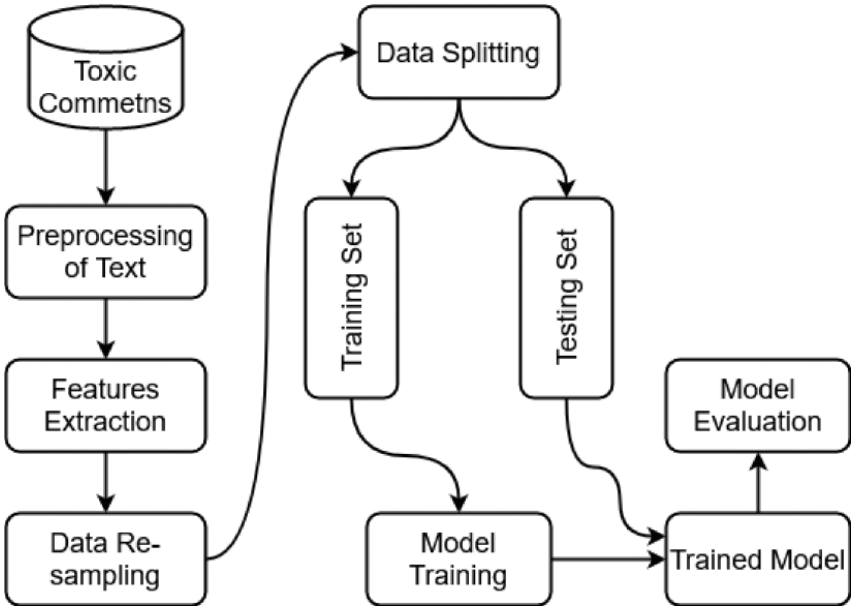
Different approaches have been proposed to tackle the problem of imbalanced datasets, from traditional methods, which alter the training set distribution, to data augmentation methods, which introduce new examples into the training set. In this chapter, we briefly discuss these methods and highlight the main similarities and differences with our work.

3.1 TRADITIONAL APPROACHES

Traditional approaches that deal with imbalanced data are divided into data-level, algorithm-level, and hybrid methods.

In (RUPAPARA et al., 2021) the authors present a work that used data-level approaches to analyze their impact on toxic comments. Figure 11 shows the methodology proposed in this work.

Figure 11 – Methodology proposed by the authors in (RUPAPARA et al., 2021).



Source: (RUPAPARA et al., 2021)

Their workflow starts with the *Toxic Comments* (TOXIC..., 2018) which is the dataset used in the experiments. This uses a multi-label dataset, where the examples were labeled into *identily_hate, insult, obscene, severe_toxic, threat, toxic, and toxicity.* The authors turned it into a binary dataset where the instances in the dataset only belong to one of the two classes: *toxic or non-toxic.*

The next step is the *Preprocessing of Text*, which performs the following tasks: tokenization, number removal, stemming, spelling correction and stopword removal. Next, their solution performs *Features Extraction* by transforming the text into vectors of

numbers. The authors use two different strategies to perform the feature extraction in their experiments: Term Frequency-Inverse Document Frequency (TF-IDF) and Bag-of-Words(BoW). The next module of the solution tries different scenarios of data resampling: **undersampling**, and **oversampling**.

Despite the similarities between our work and this one in terms of utilizing traditional approaches to handle imbalanced datasets, the primary contribution lies in the introduction of a novel ensemble algorithm known as RVVC (Regression Vector Voting Classifier). It is crucial to clarify that the experiments were conducted to assess whether this newly proposed algorithm, RVVC, when integrated into the suggested methodology (trained model) as shown in Figure 11, outperforms other algorithms.

The traditional approaches significantly enhanced the performance of the proposed ensemble, RVVC, when random undersampling was employed to balance the dataset, leading to the best results compared to other algorithms tested. Furthermore, utilizing SMOTE for dataset balancing during the oversampling experiments further improved the performance of RVVC compared to tests conducted on imbalanced and randomly undersampled balanced datasets.

(SEIFFERT et al., 2010) propose to use hybrid methods by mixing the resampling data and boosting algorithm classifiers. In the paper, the researchers proposed a new approach to tackle the imbalanced data problem, called RUSBoost. During the learning process, RUSBoost learns the weights for all instances on the training data. At the same time, RUSBoost does a Random Undersampling in the original dataset to balance the classes.

3.2 DATA AUGMENTATION

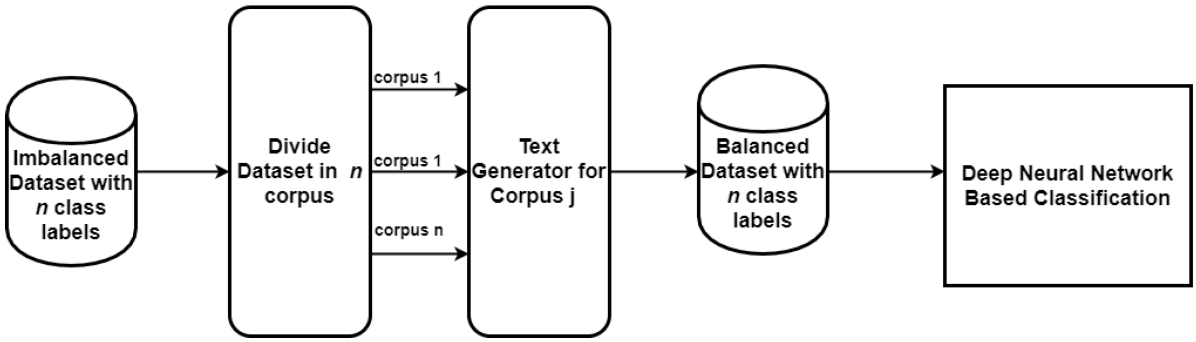
The use of Natural Language Generation (NLG) to deal with the problem of an imbalanced data set is a growing field in Natural Language Processing. This section presents a brief overview of previous approaches that have used NLG to deal with imbalanced datasets in text.

The authors in (SHAIKH et al., 2021) have used NLG to handle imbalanced data, aiming to improve classification accuracy using deep learning models. Figure 12 presents the model proposed in this work.

In the first stage, the input to the model is a text-imbalanced dataset. This dataset is divided into different sets. A set for each class. Next, each set representing a class is passed to the Text Generator to generate instances for this class. Their solution, however, does not perform any filtering on the generated sentences. They assume that the language model produces high-quality content for the task, which, as we show later, is not necessarily true. The generation process is repeated until each set reaches the number of examples of the majority class.

In (WEI; ZOU, 2019b) we have a related work here the authors present EDA: Easy Data Augmentation, a new way to do the data augmentation task in a easy way. In this work,

Figure 12 – Model proposed by the authors in (SHAIKH et al., 2021) to deal with imbalanced data.



Source: (SHAIKH et al., 2021)

Table 2 – Examples of augmented data using the method EDA.

Operation	Sentence
Original	A sad, superior human comedy played out on the back roads of life.
SR	A **lamentable**, superior human comedy played out on the **backward** road of life
RI	A sad, superior human comedy played out on **funniness** the back roads of life.
RS	A sad, superior human comedy played out on **roads** back **the** of life.
RD	A sad, superior human out on roads of life.

Source: (WEI; ZOU, 2019b)

a training example is selected to be applied to four operations, Synonym Replacement - SR, Random Insertion - RI, Random Swap - RS, and Random Deletion - RD. Table 2 shows an example given by the authors.

For each operation, the researchers defined how the augmentation would work:

- Synonym Replacement - SR: randomly choose n words from the sentence that are not stop words. Replace each of these words with one of its synonyms chosen at random.
- Random Insertion - RI: find a random synonym of a random word in the sentence that is not a stop word. Insert that synonym into a random position in the sentence. Do this n times.
- Random Swap - RS: randomly choose two words in the sentence and swap their position. Do this n times.
- Random Deletion - RD: randomly remove each word in the sentence with probability p.

The authors executed several experiments with different scenarios, changing algorithms and datasets. The results showed that, in general, the studied metric was improved before being applied EDA for training the algorithms.

Table 3 – Examples of augmented data using the method AEDA.

Original	a sad , superior human comedy played out on the back roads of life .
Aug 1	a sad , superior human comedy played out on the back roads ; of life ; .
Aug 2	a , sad . , superior human ; comedy . played . out on the back roads of life .
Aug 3	: a sad ; , superior ! human : comedy , played out ? on the back roads of life .

Source: (KARIMI; ROSSI; PRATI, 2021)

Another related work is (KARIMI; ROSSI; PRATI, 2021). In their work, they introduced a data augmentation method known as AEDA (An Easier Data Augmentation). AEDA involves the insertion of random punctuation marks into the original data to generate new examples. Specifically, they select a random number between 1 and one-third of the sequence length to determine the quantity of punctuation marks to be inserted. This approach ensures that at least one punctuation mark is added while preventing excessive punctuation insertion. The authors emphasize the importance of this step in defining the model's effectiveness because adding too many punctuation marks might negatively impact the model's performance.

Once the number of punctuation marks to be inserted is randomly determined, AEDA proceeds to randomly select positions for these marks. The insertions precisely match the number of punctuation marks determined in the previous step. Finally, AEDA randomly selects punctuation marks to be inserted into the sequence from the following set: ".", ";", "?", ":", "!", ",". Table 3 shows of augmented data created by AEDA.

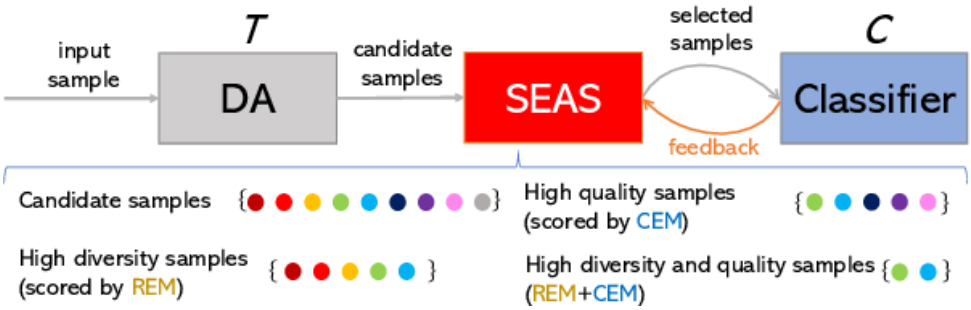
In (ZHAO et al., 2022), the authors present a method for data augmentation called EPiDA (Easy Plug-in Data Augmentation). They claim that most existing works consider only the quality or the diversity of augmentation data and thus cannot fully exploit the potential of data augmentation for Natural Language Processing. Their solution employs two mechanisms to control both the diversity and the quality of augmentation data, which are: *relative entropy maximization* (REM) and *conditional entropy minimization* (CEM).

For REM, they modify the objective function to add a term D to express the *relative entropy*, this term D calculates the difference between two distributions, the larger the difference is, the more diverse the augmented sample is. Whereas, for CEM, the authors add to the objective function a new term, H, which represents the *conditional entropy*, here they are trying to minimize the entropy of the selected sample and to maximize the mutual information with the original sample, this process gives tries to augment samples of high prediction probability and high similarity with the original sample.

Figure 13 shows the EPiDA's pipeline. It comprises a data augmentation algorithm, a *Sample Evaluation And Selection* (SEAS), and a classifier,

The data augmentation algorithm and the classifier can be customized. The *Sample Evaluation and Selection - SEAS* is where the EPiDA is applied, with the classifier trained in the available training data, and the *SEAS* uses the return of loss given by the classifier

Figure 13 – The pipeline of EPiDA.



Source: (ZHAO et al., 2022)

Table 4 – An example of the process of augmentation done by EPiDA.

	Sentence	Score Div	Score Qua.	Total
Original	Go Set a Watchman comes out Tuesday and I'm really excited for it	0.00	1.00	1.00
Aug 1	Go Set a Watchman comes out Tuesday and I'm really mad for it	0.96	0.03	0.99
Aug 2	Go Set a Watchman out Tuesday and I'm really excited for it	0.05	0.92	0.97
Aug 3	Go Set a security guard comes out Tues and I'm really excited for it	0.86	0.15	1.01

Source: (ZHAO et al., 2022)

to evaluate whether this augmented sample has diversity and quality.

The classifier calculates its loss to assess the diversity of the generated sample produced by the data augmentation algorithm. The EPiDA mechanism seeks to maximize this value since a larger loss indicates that the augmented sample differs from the ones utilized during the classifier's training process. This ensures the diversity of the generated sample.

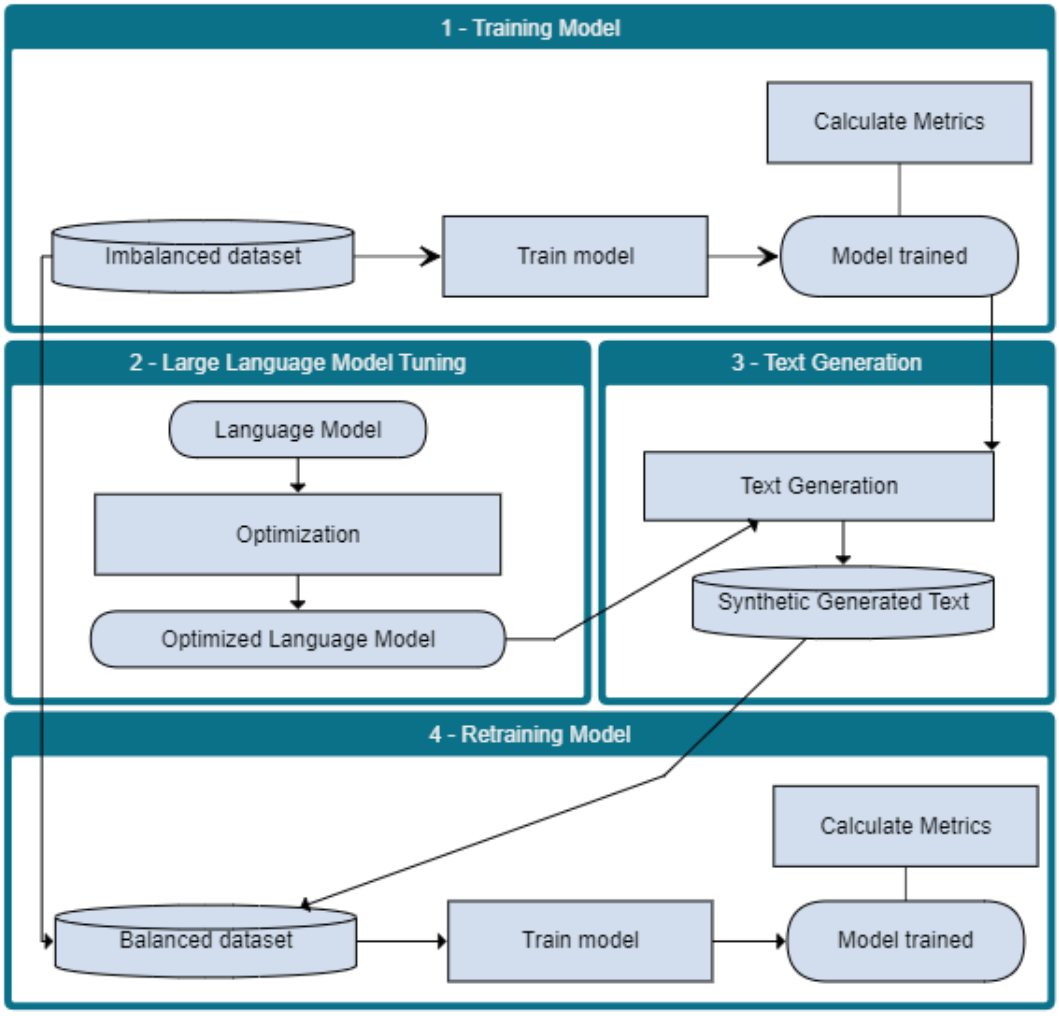
Regarding the quality of the augmented samples, the authors developed a mechanism to calculate the consistency between the original and augmented samples. This difference is compared to a given threshold, and if the condition that this difference is less than the prefixed threshold is ensured the quality of that given generated sample.

Table 4 shows examples of sentences augmented by the method presented by the paper (ZHAO et al., 2022). The authors explain that the generation of conflicting instances, whereas high diversity is more probably of low quality, and vice versa. To address this problem, they sum two scores, yielding the total score, which ensures a sample with a satisfactory balance between diversity and quality.

4 THE FRAMEWORK: BALANCE

This chapter presents BALANCE, our solution that generates text instances in the minority class using large language models (LLMs). Figure 14 shows an overview of it. First, in the Model Training module, BALANCE creates an initial classifier (C_0) from an imbalanced training set. Next, it tunes the decoding hyperparameters of a pretrained LLM using C_O, which measures the quality of the LLM's output (LLM Tuning). Then, in the Text Generation step, the tuned LLM generates the synthetic instances, which are selected by the imbalanced classifier to compose the balanced dataset along with the original training set. Finally, the balanced classifier is built using this dataset. We provide further details of each module in the following sections.

Figure 14 – Architecture of BALANCE.



Source: This research

4.1 TRAINING MODEL

In our proposed method, we have different steps where we will need certain inputs and generate various outputs. These outputs will be used in future framework modules.

At this moment, we are limited to working with imbalanced data; thus, we will utilize the available resources. In this module, we apply hyperparameter optimization to the selected algorithm classifier using a subset of the imbalanced dataset. In our experiments, we utilized the Optuna Library for this optimization. Hyperparameter optimization is a crucial step in our solution. Even when using an imbalanced dataset to train a model, an optimized classifier at this stage ensures that the model will perform at its best given the scenario. This optimized classifier will play a vital role in our proposed framework's subsequent modules. In our experiments, we referred to this initial model, trained on imbalanced datasets, as $Classifier_0$ or C_0. To assess the impact of the selected classifier on the classification task, we experimented with three different algorithms: Bert, Random Forest, and Support Vector Machine.

Hence, it is evident that each module has distinct purposes and inputs, and they will generate varying outputs that will be employed in future steps. Ensuring the proper functioning of each previous module is essential for the smooth workflow of BALANCE.

With optimization and model training, the primary objective of this step will be achieved. In our pipeline, we have included an additional subtask for evaluating the performance of this classifier trained on the imbalanced data. This task is conducted to monitor the model's improvement at the conclusion of the entire framework.

4.2 LARGE LANGUAGE MODEL TUNING

In this module, our objective is to fine-tune the decoding hyperparameters of a pretrained Large Language Model to enhance its capacity for open-ended text generation. Consequently, this refinement will lead to the generation of high-quality synthetic instances aligned with our specific interests.

Previous approaches relied on manually configuring hyperparameter values during the decoding step of language models for text generation (SHAIKH et al., 2021). However, as our goal is to produce text within a target (minority) class to address imbalanced data, our solution focuses on tuning the following hyperparameters: *temperature*, *top-p*, and *top-k* to ensure a broader coverage of sentences within that class.

In our experiments, we employed the Generative Pretrained Transformers 2 (GPT-2) (RADFORD et al., 2019b) as our Large Language Model (LLM) for this phase. The choice of GPT-2 was primarily driven by our need to operate within the constraints of our computational resources, while still benefiting from a model known for its proficient performance in text generation tasks.

Table 5 – Prompt used to generate a new sentence. The language model generates a new sentence after the last "(label of minority class):“ text.

Each item in the following list contains a (label of minority class):
(label of minority class): $sentence_1$
...
(label of minority class): $sentence_n$
(label of minority class):

Source: This research

In each trial, we instruct the model to generate n examples within the target class (with n=100 in our experiments). To create each example, BALANCE randomly selects 10 instances from the training dataset and employs few-shot learning by applying the prompt presented in Table 5. Utilizing the imbalanced classifier, BALANCE predicts the class labels for these n generated instances. Subsequently, it calculates the percentage of samples classified as belonging to the target (minority) class. This process iterates until it reaches a predetermined number of trials.

Instead of exhaustively testing all possible combinations of hyperparameter values, we employ Bayesian Optimization (MOCKUS, 2005) to search for the optimal set of values. Ultimately, the optimization process selects the set of values that yield the highest coverage of samples classified within the target class.

After completing all subtasks within this module, we will compile a set of the optimal hyperparameter values for open-ended text generation. These values will then be applied to generate instances that will serve the purpose of balancing the dataset, which is the primary objective of the next step.

Once more, it's worth noting that in our framework, the output of each step serves as input for the subsequent one, underscoring the importance of each step functioning effectively in sequence.

4.3 TEXT GENERATION

The next step in our pipeline generates instances to balance the training set. To achieve the purpose of this step, we have as input the outputs generated in previous modules: the imbalanced classifier, previously trained on the first module, and the tuned large language model.

As in the LLM tuning process, we define a prompt to generate new examples. We construct the prompt following the same structure outlined in Table 5, and then we utilize the fine-tuned Language Model to generate a new instance. To prevent the inclusion of noisy examples in the training dataset, we use the imbalanced classifier to determine whether the generated instance belongs to the minority class.

This subtask holds significant importance within this step. With the best classifier

available at this stage (the one trained on the imbalanced dataset), we perform a classification task for each synthetically generated instance. Only if a specific condition is met, we add this new instance to the dataset.

This process continues until the number of instances in the minority class matches that of the majority class. Instead of adding instances with a default classification threshold probability of belonging to the minority class equal to 0.5, we implement the following procedure to determine the optimal threshold value: we systematically vary this value within the range of 0.5 to 0.9. For each threshold value, we create a classifier using the Model Training with the newly added instances and evaluate its performance on the validation set. BALANCE selects the classifier that achieves the highest F1 value as the final balanced classifier.

By adjusting the classification threshold, we can generate different datasets as the condition for adding new instances changes. Consequently, we can identify the best-performing classifier and, in turn, determine the optimal threshold value for adding new instances, thus ensuring the exclusion of poorly generated instances.

Upon completing this step, we will possess a balanced dataset, consisting of the original instances supplemented with meticulously filtered synthetic instances. This constitutes the output of this module.

4.4 RETRAINING MODEL

In essence, the BALANCE framework comprises the three preceding modules. This step serves as a final check to ensure that all processes have been executed correctly and that the model's performance has improved.

In contrast to the initial step, where we began with a balanced dataset, we now take the output from the previous module, which consists of a dataset balanced by the BALANCE framework, along with the model classifier. Using this new dataset, we proceed to train the classifier and then compare its performance with the classifier trained at the outset of the framework, which was initially trained on the imbalanced dataset.

5 EXPERIMENTAL EVALUATION

We developed various experiments to evaluate our proposed framework. In this section, we present the setup and the results of these experiments.

5.1 SETUP

5.1.1 Datasets

We used three imbalanced datasets to evaluate the performance of our work. Table 6 presents their distribution regarding positive and negative examples and their imbalance ratio. In the following, we provide details about each of them.

Table 6 – Datasets used in our experiments and the number of positive and negative instances.

	Positive	Negative	Imbalance Ratio
Clickbait Dataset	9,276	29,241	1:3.15
PHEME Dataset	1,972	3,830	1:1.94
Tweet Emotion Dataset	1,611	5,486	1:3.40

Source: This research

- The **Clickbait Dataset** is provided by The Clickbait Challenge [1] and is composed of examples of clickbait headlines posted on Twitter. Clickbait is a certain kind of web content advertisement that is designed to entice its readers into clicking an accompanying link. Below are some examples of this type of content:
  - A Man Falls Down And Cries For Help Twice. The Second Time, My Jaw Drops.
  - 9 Out Of 10 Americans Are Completely Wrong About This Mind-Blowing Fact.
  - Here's What Actually Reduces Gun Violence.

  The Clickbait Dataset is composed of 38,517 Twitter posts. The posts were collected from 27 major United States news publishers between November 2016 and June 2017. To avoid publisher and topical biases, a maximum of ten posts per day and publisher were sampled. All posts were annotated following a 4-point scale, starting at 0.0 and finishing at 1.0. A score of 0.0 means that the post is classified as non-click-baiting, a score of 0.33 is defined as slightly click-baiting, a score of 0.66 is regarded as considerably click-baiting, and a score of 1.0 means that the post is heavily

[1] <https://webis.de/events/clickbait-challenge/>

click-baiting. Five annotators from Amazon Mechanical Turk did these annotations. 9,276 posts were considered clickbait by the majority of annotators, and 29,241 no clickbait. Table 7 shows the division of the training and test sets that we used in our experiments.

Table 7 – Distribution of the classes of each set, training and testing.

Set	clickbait	no-clickbait
Train	4,761	14,777
Test	4,515	14,464

Source: This research

- The **PHEME Dataset**(ZUBIAGA; LIAKATA; PROCTER, 2016) comprises Twitter posts about different topics/events during breaking news. The events were:
  - Ferguson unrest - citizens of Ferguson in Michigan, United States of America, protested after the fatal shooting of an 18-year-old African American, Michael Brown, by a white police officer (ZUBIAGA; LIAKATA; PROCTER, 2016).
  - Ottawa shooting - shootings occurred in Ottawa Parliament Hill in Canada, resulting in the death of a Canadian soldier (ZUBIAGA; LIAKATA; PROCTER, 2016).
  - Sydney siege - a gunman held hostage ten customers and eight employees of a Lindt chocolate located at Martin Place in Sydney, Australia (ZUBIAGA; LIAKATA; PROCTER, 2016).
  - Charlie Hebdo shooting - two brothers forced their way into the offices of the French satirical weekly newspaper Charlie Hebdo in Paris, killing 11 people and wounding 11 more (ZUBIAGA; LIAKATA; PROCTER, 2016).
  - Germanwings plane crash - a passenger plane from Barcelona to Dusseldorf crashed in the French Alps, killing all passengers and crew. The plane was ultimately found to have been deliberately crashed by the co-pilot of the plane (ZUBIAGA; LIAKATA; PROCTER, 2016).

  Journalists annotated the posts as rumors or non-rumors. The annotation of tweets sampled for all five events led to a collection of 5,802 annotated tweets, of which 1,972 were classified as rumors and 3,830 were classified as non-rumors. Table 9 shows the distribution of these posts in the five events.

  In our experiments, we performed binary classification with the posts labeled as rumors or non-rumors. Table 10 depicts the distribution of the classes used in our experiments.

Table 8 – Distribution of annotations of rumors and non-rumors for the five events studied.

Event	Rumors	Non-rumors
Charlie Hebdo	458	1,621
Ferguson	284	859
Germanwings Crash	238	231
Ottawa Shooting	470	420
Sydney Siege	522	699
Total	1,972	3,830

Table 9 – Distribution of annotations of rumors and non-rumors for the five events studied.

Source: This research

Table 10 – Distribution of the classes of each set, training and testing (PHEME Dataset).

Set	rumours	non-rumours
Train	1,583	3,058
Test	389	772

Source: This research

Table 11 – Number of instances for each emotion in Tweet Emotion Dataset.

Emotion	Number of Instances
anger	1701
fear	2252
joy	1611
sadness	1533
Total	7097

Source: This research

- The **Tweet Emotion Dataset**[2] (MOHAMMAD; BRAVO-MARQUEZ, 2017) is a set of Twitter posts with its emotions in four categories: anger, fear, joy, and sadness. The distribution of the Tweet Emotion Dataset is presented in Table 10: We split this dataset into training, validation, and test sets as shown in Table 12.

  Similar to (IMRAN et al., 2020), In our binary classification task, we employed this dataset by converting the emotion "joy" into the positive class, while categorizing all other emotions as part of the negative class. Table 13 shows the new distribution in terms of training, validation, and test set.

All of the datasets that we worked with are imbalanced, Table 6 shows the ratio between the classes.

[2] <https://saifmohammad.com/WebPages/TweetEmotionIntensity-dataviz.html>

Table 12 – Distribution of the Tweet Emotion Dataset among different datasets.

Emotion	Train	Valid	Test
anger	857	84	760
fear	1147	110	995
joy	823	74	714
sadness	786	74	673
Total	3613	342	3142

Source: This research

Table 13 – Distribution of the Tweet Emotion Dataset after the process to turn into a binary dataset.

	Positive	Negative
Training	823	2,790
Validation	74	2268
Test	714	1,428

Source: This research

5.1.2 Augmentation Strategies

We executed the following augmentation and imbalanced strategies for this evaluation:

- Traditional imbalanced methods: We ran the implementation of SMOTE, balanced Random Forest, undersampling, and RUSBoost available on the imbalanced learn package (LEMAîTRE; NOGUEIRA; ARIDAS, 2017).

- AEDA (KARIMI; ROSSI; PRATI, 2021) is a text augmentation technique that performs random insertion of punctuation marks in sentences. We execute the implementation provided by its authors [3]. AEDA creates k instances from a single one in the training set. In this evaluation, we use the following values of k: 1, 2, and 4. Due to space limitations, during process on GPU, we only report the k value with the best result in the test set.

- EPiDA (ZHAO et al., 2022) applies different entropy strategies to enhance augmented data's diversity and semantic consistency. It receives as input a classification model (BERT in this evaluation) and a data augmentation algorithm (EDA in this evaluation). We used the implementation provided by its authors [4].

- Zero-shot (SHAIKH et al., 2021): this method uses a zero-shot strategy to generate synthetic text instances. In this evaluation, we executed their text generation method due to its similarities to our solution.

[3] https://github.com/akkarimi/aeda_nlp
[4] https://github.com/zhaominyiz/EPiDA

- BALANCE: we use the Generative Pretrained Transformers 2 (GPT-2) as the pre-trained Language Model, provided by HuggingFace [5]. The choice for GPT-2 was for the use of a free-use LLM and at the beginning of the work the GPT-2 was the best free model available. For the LM Optimization, we use Optuna [6] to search for the best set of hyper-parameters in the following intervals: *top-k* [1-10]; *top-p* [0-1]; and *temperature* [0-1]. Table 14 shows the best values for each dataset.

Table 14 – Values of parameters after LLM tuning.

	Top_k	Top_p	Temperature
Clickbait	2	0.3915	0.7061
PHEME	2	0.3931	0.7111
Tweet Emotion	4	0.3801	0.6805

Source: This research

For each dataset, we executed the data augmentation strategies to generate the number of instances that fully balanced its training data (1:1). We also evaluated these methods on different algorithms. We used the scitkit-learn [7] implementation of Random Forest (RF) and Support Vector Machine (SVM), and the uncased BERT checkpoint available on huggingface [8]. We report the results of these algorithms trained on the original imbalanced training set as BERT_0, SVM_0, and RF_0.

5.1.3 Evaluation Metrics

We used Precision-Recall Area Under Curve (AUC) to evaluate the quality of the binary classifiers. We chose this metric because it is appropriate to imbalanced datasets (SAITO; REHMSMEIER, 2015) with no need to define a specific classification threshold.

5.2 RESULTS

Table 15 presents the Precision-Recall AUC of all approaches. BALANCE using Random Forest achieved the highest values for two datasets: 0.926 on Clickbait and 0.922 on PHEME. The closer baseline on Clickbait was BERT+SMOTE with Precision-Recall AUC of 0.79, and on PHEME was BERT+AEDA with 0.876. On the Tweet Emotion dataset, however, BERT_0 (the BERT model trained on the original imbalanced data) outperformed all the imbalanced methods (Precision-Recall AUC of 0.916). However, our approach obtained competitive results (Precision-Recall AUC of 0.885 by BALANCE-BERT).

[5] https://huggingface.co/gpt2
[6] https://optuna.org/
[7] <https://scikit-learn.org/stable/>
[8] https://huggingface.co/bert-base-uncased

Table 15 – Values of Precision-Recall AUC on the three datasets. Comparing different data augmentation strategies. The values within parentheses represent the differences or improvements observed when using the "BALANCE" method compared to the baseline model (The three first lines).

	Clickbait Dataset	PHEME Dataset	Tweet Emotion Dataset
$BERT_0$	0.772	0.812	**0.916**
SVM_0	0.624	0.852	0.793
RF_0	0.664	0.844	0.892
BERT+SMOTE	0.790	0.800	0.869
SVM+SMOTE	0.622	0.809	0.771
RF+SMOTE	0.669	0.794	0.790
BERT+Under	0.601	0.796	0.818
SVM+Under	0.618	0.795	0.705
RF+Under	0.678	0.794	0.618
Balanced RF	0.674	0.826	0.827
RUSBoost	0.623	0.783	0.808
BERT+AEDA	0.745	0.876	0.908
SVM+AEDA	0.614	0.833	0.740
RF+AEDA	0.622	0.835	0.888
EPiDA	0.735	0.840	0.904
Zero-shot	0.663	0.813	0.660
BERT+BALANCE	0.811(+0.039)	0.815(+0.003)	0.885(-0.031)
SVM+BALANCE	0.867(+0.243)	0.852(0)	0.822(+0.03)
RF+BALANCE	**0.926**(+0.262)	**0.922**(+0.074)	0.826(-0.066)

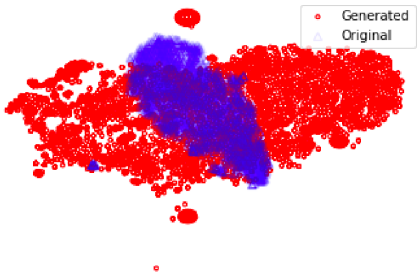
Source: This research

To investigate the possible reasons for the performance of BALANCE in the Tweet Emotion dataset, we projected, using t-SNE, the BERT vector representation of the original and the generated examples. Figure 15 shows the vector projections. The Tweet Emotion's projections of the generated examples are scattered with disjoint clusters, unlike the other two datasets, where the generated instances are more compact. Another observation from these plots is that, in all datasets, the generated examples are much more spread than the original ones, which might bring some diversity to the training set.

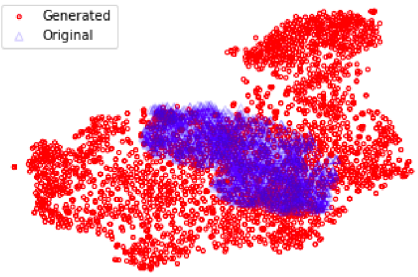
To continue this analysis, we manually inspected a random sample of 100 generated examples of each dataset to calculate the proportion of corrected instances. 69% of the instances generated for the Clickbait dataset were correct, 71% for the PHEME dataset and only 53% for the Tweet Emotion dataset. This might explain the worse performance of BALANCE compared to the imbalanced classifiers on the Tweet Emotion Dataset, even with the classifier's filter removing part of this noise, as we discuss later in this section.

Another analysis we conducted to understand the different performances of BALANCE in the datasets was to calculate the similarity between the original dataset and

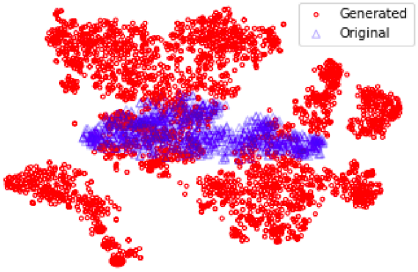
Figure 15 – Projections of the BERT vectors using t-SNE.



(a) Clickbait's projections.



(b) PHEME's projections



(c) Tweet Emotion's projections.

Source: This research

the generated one compared to the other datasets. Table 17 shows the similarities. For the Clickbait Dataset, for example, the original dataset was the most similar to the generated one (0.85), but the Tweet Emotion dataset also obtained a high similarity. This might occur because both Clickbait and Tweet Emotion are datasets from Twitter. The same happened with the PHEME dataset: the best value for cosine similarity was 0.72 when comparing the original with the generated dataset. Regarding the Tweet Emotion dataset, the similarity between the original and the generated datasets was 0.84, which is the same value to the one between the Tweet Emotion and the Clickbait datasets. Due to the proximity of these numbers, it seems that this similarity-based analysis is not very helpful to understand, for instance, the better performance of BERT_0 on the Tweet Emotion dataset, which uses the original dataset, in comparison to the models created using the generated instances.

The numbers in Table 15 also show that the choice of classification algorithm influences the Precision-Recall AUC for BALANCE. This value varied on the PHEME dataset from 0.922 using Random Forest to 0.815 with BERT. Our solution presented similar variations in the other datasets (Clickbait Dataset and Tweet Emotion Dataset).

It is also interesting to note that it is not always beneficial to balance the data since, in many cases, the models trained on the original imbalanced dataset achieved better results than the balanced methods in the three datasets. For instance, on the PHEME dataset, none of them achieved better results than the classifiers trained on balanced data with BALANCE.

To evaluate the impact of the classification filter in the text generation step, we trained

Table 16 – Values of Precision-Recall AUC for the Random Forest (bests performances on BALANCE) model trained using and not the classification filter in the text generation step to build the final training data.

Using Filter	Clickbait	PHEME	Tweet Emotion
No	0.918	0.918	0.751
Yes	0.926	0.922	0.826

Source: This research

Table 17 – Similarity between the original datasets and the generated datasets.

Original \Generated	Clickbait	PHEME	Tweet Emotion
Clickbait	0.85	-	-
PHEME	0.49	0.72	-
Tweet Emotion	0.84	0.44	0.84

Source: This research

Table 18 – Examples of generated instances for the Clickbait dataset.

Clickbait
What do the internet's biggest internet personalities think about women in general?
I don't know if it was the internet or a bad diet but it did make me feel better about myself.
Why are so many people afraid of their bank accounts

Source: This research

a random forest model without it in this phase, i.e., we considered all the synthetic examples generated by the optimized LM to train the model. Table 16 presents the results. As one can see, there is an increase in the Precision-Recall AUC values for the classifier that uses the filter to select the most promising minority class instances. This increment is higher for the Tweet Emotion dataset (from 0.751 to 0.826) than the other two datasets. This indicates that the instances generated by the LLM for this dataset are noisier than the others. Conversely, the performance of the classifiers with no filter on the Clickbait and PHEME datasets indicates that the optimized LLM in the text generation step already provides high-quality synthetic examples to balance the training set.

Tables 18, 19, and 21 show some generated instances. The Clickbait examples (Table 18) seem suitable since they are catchy headlines. The generated sentences for the PHEME dataset (Table 19) can be classified as rumors. Regarding the Tweet Emotion dataset (Table 21), the two first sentences express some emotion, but the third one only lists usernames with no text whatsoever mentioning emotions. This example shows the importance of using the imbalanced trained classifier to filter and prevent adding noisy instances to the training data generated by the LLM.

Table 19 – Examples of generated instances for the PHEME dataset.

PHEME
#Sydney siege is a tragedy, but it's not a terrorist attack. It's a war on terror.
onedog84: Over 50 people have been arrested in #Ferguson.
police have been called to the scene of the shooting, but no one has been arrested. The suspect is still at large.

Source: This research

Table 20 – Examples of generated instances for the Tweet Emotion dataset.

Tweet Emotion
a fan of the @Bryan_Bennett Twitter account. I love the way he's so funny and funny.
not sure if I'm a good person or not but I am a very good friend. I love you.
mmy_mccarthy @james_daniel @mike_bennett @brian_cameron @david_gordon

Table 21 – Examples of generated instances for the Tweet Emotion dataset.

Source: This research

6 CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSION

In this work, we presented BALANCE, a framework to create synthetic text examples using prompt-based learning to tackle data imbalance and improve classifiers in the text classification task. The process of BALANCE is composed of different stages. First, it trains a classifier with the available data, which is the imbalanced data.

In the next step, a Large Language Model (LLM) is turned to generate synthetic instances which are of our interest. In the Text Generation module, we use the previously tuned LLM to generate the synthetic examples and to balance the dataset.

With the balancing dataset, BALANCE retrains the classifier and verifies if the model's performance was improved.

Our experiments showed that BALANCE improves the performance of classifiers for text classification (Clickbait Dataset and PHEME Dataset). Furthermore, it outperformed state-of-the-art data augmentation strategies in two scenarios (Clickbait Dataset and PHEME Dataset) and has competitive results in the other (Tweet Emotion Dataset).

As demonstrated by the results of the experiments, the proposed method produces high-quality text examples and can be used to improve performance on a classification task.

6.2 FUTURE WORK

Although the experiments showed promising results, our work has some limitations. The first is related to the prompt used to generate the synthetic instances. Future work can be done to build more sophisticated prompts, more specifically for such a goal task.

Another topic that could be the target of future work is the restriction of this work to binary problems. Hence, investigating if this method can improve the models when facing a multi-class problem.

In future works toward the results' analysis, we can investigate the diversity of the generated instances, this analysis can provide the information to compare diversity with possible instances which just would add noise to the dataset.

We also can investigate different ratios of imbalanced data, collecting more datasets to experiment with or even generating datasets with different unbalanced rates.

In future investigations, one can also analyze how this proposed method would behave with other classifiers using Machine Learning packages such as pycaret[1] and LazyPredict[2] and using ensembles.

[1] <https://pycaret.org/>
[2] <https://lazypredict.readthedocs.io/en/latest/>

6.3 ATTEMPT OF USE BALANCE FOR SELF-TRAINING

In a parallel project, we investigated using BALANCE to generate synthetic instances in self-training. First, the initial model is trained, and the synthetic instances are generated using the Model Training and LLM Tuning modules. Then the self-training iterations start by predicting the positive and negative instances from the synthetic data, adding them to their respective class in the training data. A new classifier is created, and this process continues until some stop criterion (e.g., all instances added to the training data). Some experiments were executed, but the results were not promising. This can also be a direction for future work.

REFERENCES

BANARESCU, L.; BONIAL, C.; CAI, S.; GEORGESCU, M.; GRIFFITT, K.; HERMJAKOB, U.; KNIGHT, K.; KOEHN, P.; PALMER, M.; SCHNEIDER, N. Abstract meaning representation for sembanking. In: *Proceedings of the 7th linguistic annotation workshop and interoperability with discourse*. [S.l.: s.n.], 2013. p. 178–186.

BAYER, M.; KAUFHOLD, M.-A.; REUTER, C. A survey on data augmentation for text classification. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 55, n. 7, dec 2022. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3544558>.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, p. 5–32, 2001.

BROWN, T.; MANN, B.; RYDER, N.; SUBBIAH, M.; KAPLAN, J. D.; DHARIWAL, P.; NEELAKANTAN, A.; SHYAM, P.; SASTRY, G.; ASKELL, A. et al. Language models are few-shot learners. *Advances in neural information processing systems*, v. 33, p. 1877–1901, 2020.

CELIKYILMAZ, A.; CLARK, E.; GAO, J. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*, 2020.

CHAWLA, N. V.; BOWYER, K. W.; HALL, L. O.; KEGELMEYER, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, v. 16, p. 321–357, 2002.

CHEN, C.; BREIMAN, L. Using random forest to learn imbalanced data. *University of California, Berkeley*, 01 2004.

DEGÓRSKI, L.; KOBYLINSKI, L.; PRZEPIÓRKOWSKI, A. Definition extraction: improving balanced random forests. In: IEEE. *2008 International Multiconference on Computer Science and Information Technology*. [S.l.], 2008. p. 353–357.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

FAN, A.; LEWIS, M.; DAUPHIN, Y. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

FLANIGAN, J.; DYER, C.; SMITH, N. A.; CARBONELL, J. Generation from Abstract Meaning Representation using tree transducers. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, 2016. p. 731–739. Disponível em: <https://aclanthology.org/N16-1087>.

GUPTA, A.; AGARWAL, A.; SINGH, P.; RAI, P. A deep generative framework for paraphrase generation. In: *Proceedings of the aaai conference on artificial intelligence*. [S.l.: s.n.], 2018. v. 32, n. 1.

GUPTA, D.; EKBAL, A.; BHATTACHARYYA, P. A semi-supervised approach to generate the code-mixed text using pre-trained encoder and transfer learning. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, 2020. p. 2267–2280. Disponível em: <https://aclanthology.org/2020.findings-emnlp.206>.

HIDO, S.; KASHIMA, H.; TAKAHASHI, Y. Roughly balanced bagging for imbalanced data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, v. 2, n. 5-6, p. 412–426, 2009. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/sam.10061>.

HOLTZMAN, A.; BUYS, J.; DU, L.; FORBES, M.; CHOI, Y. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

HOW to generate text: using different decoding methods for language generation with Transformers. 2020. <https://huggingface.co/blog/how-to-generate>.

IMRAN, A. S.; DAUDPOTA, S. M.; KASTRATI, Z.; BATRA, R. Cross-cultural polarity and emotion detection using sentiment analysis and deep learning on covid-19 related tweets. *IEEE Access*, v. 8, p. 181074–181090, 2020.

JIN, H.; CAO, Y.; WANG, T.; XING, X.; WAN, X. Recent advances of neural text generation: Core tasks, datasets, models and challenges. *Science China Technological Sciences*, Springer, v. 63, p. 1990–2010, 2020.

KARIMI, A.; ROSSI, L.; PRATI, A. AEDA: An easier data augmentation technique for text classification. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021. p. 2748–2754. Disponível em: <https://aclanthology.org/2021.findings-emnlp.234>.

KAUR, H.; PANNU, H. S.; MALHI, A. K. A systematic review on imbalanced data challenges in machine learning: Applications and solutions. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 52, n. 4, aug 2019. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3343440>.

KONSTAS, I.; IYER, S.; YATSKAR, M.; CHOI, Y.; ZETTLEMOYER, L. Neural AMR: Sequence-to-sequence models for parsing and generation. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, 2017. p. 146–157. Disponível em: <https://aclanthology.org/P17-1014>.

KOZIARSKI, M. Csmoute: Combined synthetic oversampling and undersampling technique for imbalanced data classification. In: IEEE. *2021 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2021. p. 1–8.

LEMAîTRE, G.; NOGUEIRA, F.; ARIDAS, C. K. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, v. 18, n. 17, p. 1–5, 2017. Disponível em: <http://jmlr.org/papers/v18/16-365.html>.

LI, S.; TAO, Z.; LI, K.; FU, Y. Visual to text: Survey of image and video captioning. *IEEE Transactions on Emerging Topics in Computational Intelligence*, v. 3, n. 4, p. 297–312, 2019.

LI, Z.; JIANG, X.; SHANG, L.; LI, H. Paraphrase generation with deep reinforcement learning. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, 2018. p. 3865–3878. Disponível em: <https://aclanthology.org/D18-1421>.

LIN, E.; CHEN, Q.; QI, X. Deep reinforcement learning for imbalanced classification. *Applied Intelligence*, Springer, v. 50, n. 8, p. 2488–2502, 2020.

LIU, P.; YUAN, W.; FU, J.; JIANG, Z.; HAYASHI, H.; NEUBIG, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021.

LIU, P.; YUAN, W.; FU, J.; JIANG, Z.; HAYASHI, H.; NEUBIG, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, ACM New York, NY, v. 55, n. 9, p. 1–35, 2023.

LIU, R.; XU, G.; JIA, C.; MA, W.; WANG, L.; VOSOUGHI, S. Data boost: Text data augmentation through reinforcement learning guided conditional generation. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 2020. p. 9031–9041. Disponível em: <https://aclanthology.org/2020.emnlp-main.726>.

MANI, I. knn approach to unbalanced data distributions: a case study involving information extraction. In: . [S.l.: s.n.], 2003.

MEI, H.; BANSAL, M.; WALTER, M. R. What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, 2016. p. 720–730. Disponível em: <https://aclanthology.org/N16-1086>.

MENG, Y.; HUANG, J.; ZHANG, Y.; HAN, J. Generating training data with language models: Towards zero-shot language understanding. *arXiv preprint arXiv:2202.04538*, 2022.

MIN, S.; LYU, X.; HOLTZMAN, A.; ARTETXE, M.; LEWIS, M.; HAJISHIRZI, H.; ZETTLEMOYER, L. Rethinking the role of demonstrations: What makes in-context learning work? In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022. p. 11048–11064. Disponível em: <https://aclanthology.org/2022.emnlp-main.759>.

MOCKUS, J. The bayesian approach to global optimization. In: SPRINGER. *System Modeling and Optimization: Proceedings of the 10th IFIP Conference New York City, USA, August 31–September 4, 1981*. [S.l.], 2005. p. 473–481.

MOHAMMAD, S. M.; BRAVO-MARQUEZ, F. Emotion intensities in tweets. *arXiv preprint arXiv:1708.03696*, 2017.

MURRAY, K.; CHIANG, D. Correcting length bias in neural machine translation. *arXiv preprint arXiv:1808.10006*, 2018.

NALLAPATI, R.; ZHAI, F.; ZHOU, B. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2017. v. 31, n. 1.

NALLAPATI, R.; ZHOU, B.; SANTOS, C. dos; GULÇEHRE, Ç.; XIANG, B. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. Berlin, Germany: Association for Computational Linguistics, 2016. p. 280–290. Disponível em: <https://aclanthology.org/K16-1028>.

NARAYAN, S.; COHEN, S. B.; LAPATA, M. Ranking sentences for extractive summarization with reinforcement learning. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018. p. 1747–1759. Disponível em: <https://aclanthology.org/N18-1158>.

PATEL, H.; RAJPUT, D. S.; REDDY, G. T.; IWENDI, C.; BASHIR, A. K.; JO, O. A review on classification of imbalanced data for wireless sensor networks. *International Journal of Distributed Sensor Networks*, SAGE Publications Sage UK: London, England, v. 16, n. 4, p. 1550147720916404, 2020.

PEREZ-BELTRACHINI, L.; LAPATA, M. Bootstrapping generators from noisy data. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, 2018. p. 1516–1527. Disponível em: <https://aclanthology.org/N18-1137>.

POURDAMGHANI, N.; KNIGHT, K.; HERMJAKOB, U. Generating English from Abstract Meaning Representations. In: *Proceedings of the 9th International Natural Language Generation conference*. Edinburgh, UK: Association for Computational Linguistics, 2016. p. 21–25. Disponível em: <https://aclanthology.org/W16-6603>.

RADFORD, A.; NARASIMHAN, K.; SALIMANS, T.; SUTSKEVER, I. et al. Improving language understanding by generative pre-training. OpenAI, 2018.

RADFORD, A.; WU, J.; CHILD, R.; LUAN, D.; AMODEI, D.; SUTSKEVER, I. et al. Language models are unsupervised multitask learners. *OpenAI blog*, v. 1, n. 8, p. 9, 2019.

RADFORD, A.; WU, J.; CHILD, R.; LUAN, D.; AMODEI, D.; SUTSKEVER, I. Language models are unsupervised multitask learners. In: . [S.l.: s.n.], 2019.

RAFFEL, C.; SHAZEER, N.; ROBERTS, A.; LEE, K.; NARANG, S.; MATENA, M.; ZHOU, Y.; LI, W.; LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, JMLRORG, v. 21, n. 1, p. 5485–5551, 2020.

ROTHMAN, D. *Transformers for Natural Language Processing: Build innovative deep neural network architectures for NLP with Python, PyTorch, TensorFlow, BERT, RoBERTa, and more*. [S.l.]: Packt Publishing Ltd, 2021.

RUPAPARA, V.; RUSTAM, F.; SHAHZAD, H. F.; MEHMOOD, A.; ASHRAF, I.; CHOI, G. S. Impact of smote on imbalanced text features for toxic comments classification using rvvc model. *IEEE Access*, IEEE, v. 9, p. 78621–78634, 2021.

RUSH, A. M.; CHOPRA, S.; WESTON, J. A neural attention model for abstractive sentence summarization. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, 2015. p. 379–389. Disponível em: <https://aclanthology.org/D15-1044>.

SAITO, T.; REHMSMEIER, M. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, v. 10, n. 3, p. e0118432, 2015. Https://journals.plos.org/plosone/article/file?id=10.1371/journal.pone.0118432type=printable. Disponível em: <https://app.dimensions.ai/details/publication/pub.1012273932>.

SEIFFERT, C.; KHOSHGOFTAAR, T. M.; HULSE, J. V.; NAPOLITANO, A. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, v. 40, n. 1, p. 185–197, 2010.

SHA, L.; MOU, L.; LIU, T.; POUPART, P.; LI, S.; CHANG, B.; SUI, Z. Order-planning neural text generation from structured data. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. [S.l.: s.n.], 2018. v. 32, n. 1.

SHAIKH, S.; DAUDPOTA, S. M.; IMRAN, A. S.; KASTRATI, Z. Towards improved classification accuracy on highly imbalanced text dataset using deep neural language models. *Applied Sciences*, v. 11, n. 2, 2021. ISSN 2076-3417. Disponível em: <https://www.mdpi.com/2076-3417/11/2/869>.

SONG, Y.; WANG, T.; CAI, P.; MONDAL, S. K.; SAHOO, J. P. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, feb 2023. ISSN 0360-0300. Just Accepted. Disponível em: <https://doi.org/10.1145/3582688>.

TOUVRON, H.; LAVRIL, T.; IZACARD, G.; MARTINET, X.; LACHAUX, M.-A.; LACROIX, T.; ROZIÈRE, B.; GOYAL, N.; HAMBRO, E.; AZHAR, F. et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

TOXIC Comment Classification Challenge. 2018. <https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge>.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, Ł.; POLOSUKHIN, I. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017.

WANG, Y.; YAO, Q.; KWOK, J. T.; NI, L. M. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 53, n. 3, jun 2020. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3386252>.

WEI, J.; ZOU, K. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

WEI, J.; ZOU, K. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, 2019. p. 6382–6388. Disponível em: <https://aclanthology.org/D19-1670>.

YANG, Y.; HUANG, L.; MA, M. Breaking the beam search curse: A study of (re-) scoring methods and stopping criteria for neural machine translation. *arXiv preprint arXiv:1808.09582*, 2018.

ZHAO, M.; ZHANG, L.; XU, Y.; DING, J.; GUAN, J.; ZHOU, S. EPiDA: An easy plug-in data augmentation framework for high performance text classification. In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, 2022. p. 4742–4752. Disponível em: <https://aclanthology.org/2022.naacl-main.349>.

ZUBIAGA, A.; LIAKATA, M.; PROCTER, R. Learning reporting dynamics during breaking news for rumour detection in social media. *arXiv preprint arXiv:1610.07363*, 2016.