UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

TIAGO SILVEIRA CECCON

Applying BERT language model to poem classification: a study on data imbalance issues

Work presented in partial fulfillment of the requirements for the degree of Bachelor in Computer Science

Advisor: Prof. Dr. Joel Luís Carbonera
Co-advisor: Dr. Luan Fonseca Garcia

Porto Alegre
April 2023

“The visions we offer our children shape the future. It matters what those visions are. Often they become self-fulfilling prophecies. Dreams are maps.”

— Carl Sagan

ABSTRACT

Art, and specifically poetry, has always been a very valuable resource to understand a society's identity and view of life. Recently, with the rise of the big data revolution, large datasets of the most varied subjects are starting to appear. Also, we are seeing a new wave of very powerful Artificial Intelligence systems based on deep learning, especially in the area of Natural Language Processing (NLP). It is reasonable, then, to explore how well suited these systems are to process data within the realm of poetry, since we stand to gain so much insight about human cultures from it. In this work we apply the BERT pre-trained language model to a real-world dataset of poems, in order to create classifiers to recognize the topics the poems deal with. We list some of the issues that appeared during this process and experiment with possible strategies to mitigate one of them, namely imbalance of classes. We found that it was possible to improve the baseline results by applying two of the strategies explored, those being undersampling of the majority class and the use of different weights for each class to scale the loss function.

Keywords: Artificial Intelligence. Machine Learning. Deep Learning. NLP. Large Language Models. BERT. Poetry.

Aplicando o modelo de linguagem BERT à classificação de poemas: um estudo sobre problemas de desbalancamento de dados

RESUMO

Arte, e em específico poesia, sempre foi um recurso muito valioso para a compreensão da identidade e visão de mundo de uma sociedade. Recentemente, com o crescimento da revolução de *big data*, grandes conjuntos de dados dos mais variados assuntos estão começando a aparecer. Também estamos presenciando uma nova onda de sistemas de Inteligência Artificial muito poderosos baseados em aprendizado profundo, em especial na área de Processamento de Linguagem Natural. É razoável, então, explorar o quão adequados são esses sistemas para processar dados dentro do campo da poesia, já que podemos ganhar tanto entendimento sobre as culturas humanas através deles. Nesse trabalho aplicamos o modelo pré-treinado de linguagem BERT a um dataset de poemas do mundo real, de modo a criar classificadores para reconhecer com quais tópicos os poemas lidam. Listamos alguns dos problemas que apareceram durante esse processo e experimentamos com estratégias possíveis para mitigar um deles, a saber o desbalanceamento de classes. Descobrimos que é possível melhorar os resultados iniciais ao aplicar duas das estratégias propostas, sendo estas *undersampling* da classe majoritária e o uso de diferentes pesos para cada classe escalando a função de perda.

Palavras-chave: Inteligência Artificial. Aprendizado de Máquina. Aprendizado Profundo. PLN. Modelos Linguísticos de Grandes Dimensões. BERT. Poesia.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
GPU	Graphical Processing Unit
LDA	Latent Dirichlet Allocation
LLM	Large Language Model
LSTM	Long Short-Term Memory
NLP	Natural Language Processing
RNN	Recurrent Neural Network
SVM	Support Vector Machine
TF-IDF	Term Frequency - Inverse Document Frequency

CONTENTS

1 INTRODUCTION

Pre-trained language models have been making great advances in the Natural Language Processing (NLP) field in recent years, frequently achieving state-of-the-art results (QIU et al., 2020). The fact that they are built to use a two-step framework, with the first one allowing them to build a powerful representation of language that they can then leverage when specialized to a downstream task in the second step, makes them very suited to handle problems that involve processing texts containing the type of language that is commonly used in day-to-day communication.

Poetry, being an art form, differs from standard language. A poem is at the same time a linguistic object and an aesthetic object. Both in the usage of a more strict formal structure and in the weaving of more complex meanings to particular words through metaphor and other literary techniques, the logic behind the construction of a poem is not the same as the one behind most daily interactions through language, where communication is more important than aesthetics.

We propose then that using pre-trained language models to work directly with poetry is an interesting venue to explore, as this type of text can present a challenge to these systems due to their differences from standard language. We use the BERT (DEVLIN et al., 2019) large language model as a suitable representative for this work, as it has been a reference since its publication, leading to many variants and successors. We fine-tune BERT to create models to classify poems into the topics that they talk about. Although the use of topics represent a multi-class problem, we follow the methodology of Lou, Inkpen and Tanasescu (2015) and frame the process as multiple binary classification problems. As a data source, we use the poems from the Poetry Foundation website. Since there was no dataset available that grouped together all we wanted to analyse and use for our task, one of the steps in this work was to scrape the organization's website in order to build the dataset to be used.

By choosing this approach of using a data source focused on serving its data to a human audience, instead of curated for the purposes of automated processing, we ran into some of the problems that usually appear when dealing with real-world data. We discuss these problems and our decisions on how to handle them as they appeared.

One of the problems we ran into was that of the imbalance of classes. This is a common problem for classification tasks in real world scenarios, and can cause bad performance if no explicit action is taken to mitigate it (LAKSHMI; PRASAD, 2014). We

explored further this issue and ran experiments with three possible strategies to deal with it. We experiment with one resampling technique (undersampling) and two techniques related to the loss function (different weights for each class and the use of Focal Loss as a loss function).

The questions of how pre-trained language models handle poetic texts and what are the impacts of the chosen strategies can be interesting from a purely technical perspective. However, an automated poem classification system can also have a practical significance, helping the daily work of professionals that manage poetry submission contests such as editors of poetry magazine or curators of cultural events, who would be able to optimize their time by not having to manually process all submissions. Another case in which automated classification of poems in topics can have a practical utility is by allowing the quick recovery of poems in big repositories.

The remainder of this work is organized as follows. Chapter 2 presents the necessary background information to follow the rest of the work. Chapter 3 consists of a short review of works related to ours that can be found in the technical literature. Chapter 4 describes our methodology to develop this work, detailing how we obtained the dataset we used, what was the setup for the training and the experiments. In Chapter 5, we present the results of our experiments, as well as a brief analysis of each one individually and a comparison between all of them. Finally, in Chapter 6, we share our conclusions and the possibilities we identified for future works in this topic.

2 BACKGROUND

Here we give the necessary background to understand the present work. Section 2.1 gives a brief overview of the field of machine learning. Section 2.2 describes the language model we are using, namely BERT. Section 2.3 describes some known strategies to mitigate imbalance issues. Section 2.4 details commonly used metrics to evaluate the performance of classifiers that were adopted in this work. Section 2.5 presents the strategy of threshold moving, which involves finding the best threshold for classification problems. Finally, in section 2.6 we describe our main data source, the Poetry Foundation organization and its website.

2.1 Machine Learning

Machine Learning is a sub-field of Artificial Intelligence which deals with the study of computational systems that are able to learn. In Russell and Norvig (2020), the authors define that an agent *is learning if it improves its performance after making observations about the world*. Also according to Russell and Norvig (2020), approaches developed in this field can be classified broadly into three main classes: supervised learning, unsupervised learning and reinforcement learning.

Supervised learning is a form of learning in which a system receives pairs consisting of an input and a desired output, and attempts to learn a function that maps inputs to outputs based on the patterns that it finds in the pairs it saw while training. For this type of learning, it is essential to have *labeled* data that can be used to train the system. In this work, we consider the task of poem classification as a supervised learning task.

Unsupervised learning happens when there are no labels for the inputs, and the system is expected to identify patterns and structures that it finds represented in the inputs.

Reinforcement learning is an approach where the system guides its learning through the feedback it receives from the world with which it interacts, both in the form of rewards and punishments.

The field of Machine Learning has proposed several different approaches. In the last years, *Artificial Neural Networks* have being gaining a growing attention in the field. Artificial neural networks are inspired by the architecture of biological brains. They consist of nodes, or neurons, and connections between these nodes. Each connection can carry signals from its starting neuron to its ending neuron. Neurons can then process

these signals and generate an output that is sent to other neurons through the connections that start on it. Connections between nodes have weights that can modulate the strength of the signal passing through it. Training a neural network involves automatically adjusting these weights. At the end of the training process, a neural network represents a complex mathematical function that maps inputs to desired outputs. In these systems, neurons are usually organized in layers. When a neural network is composed of several layers, it is an example of *deep learning*.

Due to the highly complex architecture of sophisticated neural networks that have several layers and a large number of trainable parameters, they often need a lot of suitable data to achieve good performance (QIU et al., 2020). Due to this, applying these approaches to problems were there is not a great amount of data curated to their needs can be challenging. For scenarios like these, one concept that can help, and that will be important for the purposes of this work, is that of *Transfer Learning*. This strategy involves applying knowledge that was previously learned about one problem to solve another problem. The association between transfer learning and neural networks goes back at least to the 1970s, as mentioned by Bozinovski (2020).

2.2 The BERT language model

BERT is a language model developed by Google and described on Devlin et al. (2019). Since its first publication, it has become a major reference in the field and inspired a vast number of variants and successors that built upon it. Its name is an acronym for Bidirectional Encoder Representations from Transformers, which already suggests the technology it is built on. Namely, the Transformer architecture, which has at its core the *attention mechanism*. We briefly describe these concepts before getting into more detail about BERT.

Russell and Norvig (2020) describe language model as *a probability distribution describing the likelihood of any string*. In other words, it is a model whose intent is to represent enough knowledge about a language so as to make meaningful predictions about the probabilities of textual content in it. They can be useful in several applications, such as generating text, identifying wrong usages of words, classifying documents, and so on. Language models can be of varying degrees of complexity, and can be implemented with many technologies. Recently, there has been an explosion of language models based on artificial neural networks with a large amount of parameters. These are commonly called

large language models.

BERT is such a large language model. It is an artificial neural network model based on the *Transformer architecture*. The Transformer architecture was introduced in Vaswani et al. (2017) and quickly replaced RNNs and LSTMs as the state-of-the-art architecture for NLP. It provided several advantages in comparison to those, as it eliminated their sequential nature and instead processes the entire input at the same time. This mitigates the problem of vanishing gradient that arises from sequentially processing long sentences, and also allows for a much larger level of parallelism that increases the speed of training. So, the Transformer architecture allowed for models to be trained with datasets at a scale larger than previously possible. The Transformer is able to do that because of its particular use of the attention mechanism, through self-attention. The attention mechanism is a technique used in neural networks to allow more focus to be put in certain parts of the input data than in others, contextually.

BERT has a few different versions that vary mostly in size and complexity. For instance, $BERT_{BASE}$ contains 110 million parameters, while $BERT_{LARGE}$ contains 340 million parameters. The dataset used to train BERT was comprised of a dataset of unpublished books that had around 800 million words and a significant part of the English version of the Wikipedia, with 2500 million words. It was trained with two unsupervised tasks, the first of them being to predict a masked token given its original context. The second task was to, given two sentences, predict if they appeared sequentially in the training corpus.

One of the most interesting aspects about BERT is that it is designed to work in the context of transfer learning, by following a two-step process: pre-training followed by fine-tuning. The pre-trained step was done by the original team as described above, and allowed the model to generate a powerful representation of the language it was trained in. After the end of this step, the learned weights can be loaded again and be used as a starting point in a second training phase for a more specialized downstream task, a process which is called fine-tuning. Because the loaded model can tap into the representation it learned during its pre-training phase, the second step can achieve great results using much less resources than during the pre-training phase. While this was already a known practice by the time that BERT was published, combined with the innovation of the bidirectional approach and the size of the model, this allowed BERT to become widely adopted.

2.3 Imbalance Mitigation

As mentioned by Ali et al. (2019), the issue of imbalance of classes presents a challenge for classification problems in machine learning. In this section we describe some of the strategies found in the literature to deal with this issue.

2.3.1 Undersampling

Lakshmi and Prasad (2014) present undersampling as a data-level sampling technique to deal with imbalance. Sampling techniques attempt to solve this by manipulating the usage of the existing data to fix the natural bias towards the majority class. That can be done by either artificially increasing the number of samples from the minority class (known as oversampling) or by artificially reducing the number of samples from the majority class (known as undersampling). As Lakshmi and Prasad (2014) mentions, undersampling has as some of its advantages the fact that it can be used to reduce the time and memory used by training.

2.3.2 Cost-Sensitive Learning

Another possibility to deal with imbalance issues is to manage the perceived costs of mistakes for different classes in the training process. As Ali et al. (2019) describes, the idea is to attempt to minimize the error cost of the process as a whole and to assume a higher cost for errors related to the minority class, both at the same time. Two techniques relevant to the present work are described here.

2.3.2.1 Weighted Loss Function

The first technique is to ascribe different weights to each class and to incorporate those weights when computing the value of the loss function. There are several ways to do this mathematically. One such way is described by Madabushi, Kochkina and Castelle (2020), and it consists of scaling the chosen loss function with the weight attached to the class of the sample. The formula is as follows:

$$\text{loss}(x, class) = weight[class] * \Theta$$

where $weight$ is an array with the weights for each class, $class$ indexing that array in order to get the weight for that particular class, and Θ representing the loss function used. For this technique, any loss function can be plugged in the place of Θ.

2.3.2.2 Focal Loss

Focal Loss function (LIN et al., 2018) is a variation of cross-entropy developed in the context of computer vision. The authors were trying to deal with the issue of high imbalance of foreground-background classes when attempting to detect objects.

It works by adding a new factor to the cross-entropy loss, $(1 - p_t)^\gamma$. This factor will reduce the contribution to the loss from easily and correctly classified examples, increasing the importance of hard and misclassified ones. In this factor, the term p_t is a notational convenience to mean p, that is, the model's estimated probability for the positive class, when the sample is of the positive class, and $1 - p$ when the sample is of the negative class. The term γ is a new hyperparameter they introduced that adjusts the rate at which easy examples have their contribution to the result of the loss scaled down.

The authors also incorporate the balancing factor α from Balanced Cross-Entropy, which is responsible for balancing the importance of positive and negative examples. Together, γ and α are the relevant hyperparameters for Focal Loss. Below is the definition of the formula to compute it, as presented in (LIN et al., 2018):

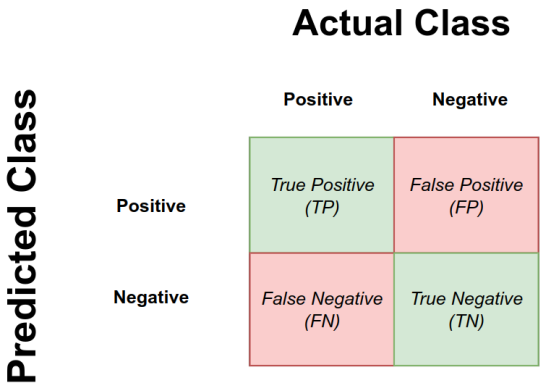
$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma log(p_t)$$

2.4 Model Evaluation

There are several metrics that can be used to evaluate binary classifiers, some of the most common ones being accuracy, recall, precision and F-measure (KADHIM, 2019). All of these are computed from elements that can be identified in a *confusion matrix*, shown in Figure 2.1.

The relevant elements are *True Positives (TP)*, *False Positives (FP)*, *False Negatives (FN)* and *True Negatives (TN)*, which can be seen in the cells of the matrix. True positives are those samples that belong to the positive class and are correctly identified, while true negatives are samples of the negative class that are correctly identified. False positives are samples of the negative class incorrectly identified as belonging to the posi-

Figure 2.1 – A confusion matrix



Source: The Author

tive class, while false negatives are samples of the positive class incorrectly identified as belonging to the negative class.

From the previously mentioned metrics, **Accuracy**, in special, is very commonly used. It is an intuitive way to think about evaluating how correct a set of predictions is. In high level terms, it just answers the question "what proportion of the total did I get right?", so it makes sense that it would be the first way of evaluating performance that comes to mind. Its formula is:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{False Positives} + \text{False Negatives} + \text{True Negatives}}$$

For problems that deal with imbalanced datasets, however, accuracy can be misleading. This happens because if one class is significantly less common than others, then a high accuracy can be achieved by just never predicting that class, which is not helpful. The other commonly used metrics for classification, Precision, Recall and F-measure can be more robust and informative in such a setting.

Precision can be thought of as a way to measure what ratio of the samples a classifier predicted as belonging to a particular class was actually correct. It is calculated as follows:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Note that this can be calculated accordingly for either the positive or the negative class when we are dealing with binary classification. The same is true for recall.

We can think of **Recall** as a way to measure what ratio of samples from a particular class in a set was correctly predicted. It is calculated as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

F-measure is a family of measures, with F1 being its most common variant. The **F1 score** gives us a way to consider both precision and recall at the same time with a single value by taking the harmonic mean of them. It is calculated using both metrics, with the following formula:

$$F1 = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

For binary classification problems such as ours, we can compute the **Macro-F1** score by calculating F1 scores for both classes (positives and negatives) and then taking the average between them. This metric provides an overall evaluation of the model, considering both classes with the same level of importance.

$$\text{Macro F1} = \frac{F1_{pos} + F1_{neg}}{2}$$

2.5 Threshold Moving

As mentioned in Lakshmi and Prasad (2014), *threshold moving* is an algorithmic level solution to deal with classification problems on imbalanced datasets. It arises from the fact that the classifier outputs a continuous value as its prediction, lending a degree of freedom for a choice on which threshold should be considered as the frontier between the negative class and the positive class.

The standard value used as the threshold for such classifiers is 0.5, which can be appropriate when dealing with a balanced problem in which there is roughly fifty percent chance of a sample being of either class. When this is not the case, however, this value might not be the most suitable.

With threshold moving, then, we define a range of thresholds we want to consider together with a step we will use to travel this range, and evaluate the classifier by setting the threshold for the interpretation of its output at each step in the range, and keeping the value that results in the best performance.

2.6 Poetry Foundation

The Poetry Foundation[1] is a literary society focused on the propagation of poetry to the widest possible audience. It was established in 2003, as a successor to the Modern Poetry Association which had existed since 1941. It is responsible for several programs revolving around poetry, such as the publication of the Poetry Magazine, the organization of events and exhibitions, the distribution of awards and an open library, as well as maintaining a digital presence through social media, mobile applications and a website.

In its website, the Poetry Foundation makes available thousands of poems for free, both from famous poets and from newer or more obscure ones. These poems are sometimes associated with certain metadata, such as the literary school of the poem, its epoch, the topics that it touches, amongst others. This makes it a useful source of data for anyone wanting to work with poetry through automated means. In specific, the metadata it has associated with poems can be harnessed for supervised learning works, such as ours.

Although it can be a good source, it must be mentioned that because its main purpose is the propagation of poetry to human readers, and not to serve as a digital platform for data processing, it also presents some challenges when being used as such. For instance, the fact that it does not offer an API, and that the presence of metadata for its poems can be somewhat erratic, with some information being present only for a few of them. Nevertheless, it is amongst the best options currently available in this context, and we will use it as our data source for this work.

[1] https://www.poetryfoundation.org/

3 RELATED WORKS

There are a few works in the literature that deal with classifying poems through machine learning methods. However, most of the ones we could find used more traditional algorithms, instead of LLMs. In this section we give a brief overview of the most relevant works we found while developing the current one. A frequent theme we noticed was that the size of the datasets used are usually small when compared with works outside of the context of poetry. Our dataset, with a little under 13 thousand samples was actually one of the largest we could find. This points to a lack of well-structured, carefully developed and easily available datasets in this field that should be addressed by an interdisciplinary effort between experts from the Literature and Computer Science fields.

In Lou, Inkpen and Tanasescu (2015), the authors present a research that has several points of intersection with ours. They used a dataset built from the same data source we use, the Poetry Foundation website, which they also scraped to gather the information they would need. Their choice of targets for the classification also mostly overlaps with ours, that is, both of the works create classifiers to recognize the topics that the poems deal with, based on the textual content of the poems. The exception to this is that they also performed an experiment with some of the sub-topics of the main topics, while we constrain ourselves to work only with the main topics. We applied the same methodology of creating binary classifiers for each topic as they did. Our work differs from theirs, however, firstly because the database of the Poetry Foundation has enlarged since then, and while they built a dataset consisting of 7214 total poems, we have almost twice as many as that, with 12970 total poems after cutting off samples that were either too small or too large for our purposes. Secondly, because we are using a pre-trained language model, while they used traditional machine learning algorithms. In specific, they worked with tf-idf, LDA and SVMs. Finally, the main issue that our work attacks is that of the imbalance of the classes in the dataset, which in their work they acknowledge as a problem but do not perform any experiment in order to mitigate.

In Barbado, González and Carrera (2021), the authors applied models based on Transformers to work with poetry classification. They applied a semi-supervised approach in which they used transformers mostly to extract word embeddings. They also work with datasets comprised of poems in Spanish, and so they use multi-lingual version of models, while we use only the English variant as our dataset is in this language. Actually, they have two datasets, one smaller, with 270 labeled samples, and one larger with

4572 unlabeled samples. They also applied a larger number of models, such as DistilBERT, XMLRoberta and others. We focus only on BERT Base, due to the resource and time constraints that define the scope of this work. They worked with two distinct set of categories, framing one of them as a series of binary classification problems, similarly to what we do in the current work; and the other as multi-label problem. In the category they framed as binary classification, their F1 scores varied greatly depending on the class. Their best results achieving 0.87, in this context. For the majority of the classes, however, they reached scores below 0.75. On the multi-label problem, their best weighted F1 scores were 0.68 and their worst ones were 0.32.

In Ruma et al. (2022), the authors also adopted two different datasets. However, in this case, the datasets represented annotated collections of the poet Hafez done by two scholars of the poet. Both datasets were very small, however, one of them containing 249 poems and the other 233. They framed their problem as trying to classify at which stage of the life of the poet the poems were written, based on the knowledge that Hafez explored different themes and styles across his life. An interesting aspect of their work is that they were able to use both the poems in the original language as well as English translations. Their best results were when using the original Persian language. They compared a few traditional machine learning algorithms, namely Random Forest, Logistic Regression and SVMs with the LSTM deep learning algorithm and its Bidirectional version, BiLSTM. LSTM outperformed the traditional algorithms. At the end of their work, after several improvements, they were able to reach F1 scores around 0.85.

A larger dataset was used by Ahmad et al. (2020), with 9142 poems in English. However, their samples are usually much smaller, with the size of only a few sentences. Their goal was the classification of these poetry posts into emotional states such as "anger", "alone", "hate", and so on. There were 13 classes in total. They compared a few deep learning techniques, with the one that achieved the best performance being an Attention-based Convolutional Bi-Directional LSTM. They report 0.88 as the F1 score they were able to achieve.

With regards to works that focused on the problem of imbalance, Jayaraman et al. (2023) performed a comparison of different sampling techniques, including undersampling, with the BERT language model and found good results for all techniques explored. They were working with sentiment analyses on two diverse datasets composed of news articles and SMS messages, respectively. Also working with BERT, Zhu, Wu and Yaseen (2022) experimented with different levels of undersampling and found that a moderate

amount of it gave the best cost-benefit. While working with more traditional machine learning algorithms, namely logistic regression and XGBoost, Nguyen and Duong (2021) compared data sampling techniques with cost sensitive learning techniques within the context of the customer churn prediction problem. While they used only oversampling techniques instead of the undersampling that we use, for the cost sensitive learning they employed both focal loss and a weighted loss function. They found that the cost sensitive techniques performed better than the sampling ones.

4 METHODOLOGY

Here we describe the methodology we adopted to conduct this work. In Section 4.1 we describe how we built and consolidated our dataset. In Section 4.2 we describe how we chose to model our problem. In Section 4.3 we described the technical details of how we implemented the work, including environments and libraries used, the configuration of parameters, the adopted metrics and how we handled threshold selection. Section 4.4 contains the definitions of the experiments we conducted, including also how we split the dataset to allow both training and evaluation of models, as well as a description of our baseline implementation.

4.1 Dataset

As mentioned in Section 2.6, we use the Poetry Foundation website as the data source for this work. In this section, we describe how we consumed this source and modeled the data into a representation that allowed us to apply a supervised learning classification approach on it. All poems used are in English, either written natively in this language or in translation from other languages.

4.1.1 Data acquisition

Since the Poetry Foundation website does not offer an API, we need an alternative and scalable way to fetch the poems and their metadata in order to work with them. We found some datasets already available at the Kaggle platform that were created from the Poetry Foundation source. However none of them contained all the information we were interested in. The most promising one turned out to be the one entitled "Complete poetryfoundation.org dataset"[1], as it was amongst the most complete ones, with more than 15000 unique entries, provided the text content of the poem and also its Poetry Foundation id. With that id, we would be able to scrape the website and fetch any missing metadata we might desire. So we used that dataset as our starting point.

We proceeded to scrape the website, using the Python[2] programming language and

[1] https://www.kaggle.com/datasets/johnhallman/complete-poetryfoundationorg-dataset
[2] https://www.python.org/

the Beautiful Soup[3] library. Since we did not want to overburden the website's servers, we used a few seconds of delay after each request, varying the amount each time. We stretched this process across five days, to ensure that the amount of load we would put on the servers would not be harmful.

The most relevant information we gathered were the topics associated with each poem and the literary school they belonged to. Analysing both of these allowed us to pick a target for the classification.

At the end of this process, the full dataset we built contained the following fields for each poem:

- **ID:** The id of the poem in the Poetry Foundation's website. This is a unique information for each poem, and it exists for every poem.
- **Author:** The author of the poem. This information is present for every poem.
- **Title:** The title of the poem. This information is present for every poem.
- **Content:** The textual content of the poem, in other words, the poem itself. This information is present for every poem.
- **School:** The literary school to which the poem belong. Examples of these are "Modern", "Imagist", "Victorian", "Renaissance", among others. This information is not present for all poems.
- **Topics:** The topics that the poem has been labeled as dealing with. Examples of these are "Love", "Religion", "Relationships", and so on. Since poems can be labeled with multiple topics, this is a list containing every topic that the poem was associated with. We provide a deeper explanation about topics in Section 4.2.1. This information is not present for all poems in this initial dataset.

We do not use all of these fields to build our classifier pipeline, as explained in Section 4.2.1. However, other workflows can be constructed by starting from our methodology and branching at this point, by making a different selection of fields to be used. For instance, by proposing to identify a poem's author based on its textual content, or by trying to infer the literary school of a poem based on a combination of its textual content and the topics it deals with. We leave this ideas as suggestions for future works.

[3]https://beautiful-soup-4.readthedocs.io/en/latest/

4.1.2 Data preparation

The choice of the language model we would use, which we describe in Section 4.3.3, led us to make initial adjustments with regards to the data we would use.

The only transformation we apply to the poems themselves is to convert their textual content entirely to lower-case, since we adopted the uncased version of BERT. We do not perform any other data cleaning techniques on them, such as stop word removal or similar methods before passing the content to the preprocessing layer that is provided as a companion to the BERT model by the Google team.

With regards to the size of the samples, however, we have a technical limitation that we must take into consideration. The BERT model we are using can only handle inputs of size up to 512 tokens. Any content larger than that would be truncated automatically by the model. So we must at least consider curating the dataset to a certain size range.

Firstly we removed all poems with less than 10 tokens. There were only a few of them and they were mostly incomplete works or erroneous entries, so they would generate more noise than gain in the training process.

Then we turned to the fact that the model we use has a maximum size limit. In the end we decided to work only with poems of size 510 or less and discard those larger than that. We did this to simplify the training process and focus on other issues, but we encourage future works to explore the possible strategies to deal with these cases so as to achieve better performances. More details about the considerations we did to come up with this decision and its impact on the amount of data we have to discard are given in section 4.2.3.

4.2 Problem Modeling

Here we describe how we chose to model our problem. In Section 4.2.1 we describe what information we chose to train our classifiers to recognize, based on the content of the poems. In Section 4.2.2 we describe our choice for binary classification instead of using a multi-label approach. In Section 4.2.3 we describe how we decided to filter our dataset so as to use only information that would be useful and helpful during the training process.

4.2.1 Target for classification

Initially we considered two possible ideas to configure our workflow for the classification. Both of them were based on using only the textual content of the poem (that is, discarding any information about author or other such metadata) as the source for predictions. The two ideas then diverged on what would be the target of the classification, that is, what information would we try to predict, given the poem's content. One of the ideas was to predict the literary school of the poem, while the other was to predict the topics the poem deals with.

By analysing the dataset, we found out that, of the 15652 poems we had, 14310 of them had information about which topics they referred to, while only 2514 had the information about their literary school. Given this scenario, we chose topics as the most viable target for classification, as working with school would mean we had far less data available.

Some explanation about how topics are used in the Poetry Foundation website can help to understand better the rest of our work, so we discuss them in the following paragraph.

Topics are a human-annotated information that describe the general themes that the poems talk about. They are called "Topics" on the website's main interface for browsing[4], and appear as "Quick Tags" on individual poem pages[5] - we will call them only as "topics" in this work to simplify the terminology. They range from very general ones such as "Love" to very specific ones such as "Labor Day". As far as we were able to identify, there were 123 topics in total, at the time when we started our work. Most of these appear on a very small number of poems and can be safely ignored for our purposes. Nine of them, however, play a central role and are further identified as "Subjects" in the previously mentioned main website interface. These also act as top of hierarchies. For instance the topic "Nature" can be further refined into the topics "Winter", "Trees & Flowers" and "Animals", among others. We chose to work only with these nine main topics, as they cover the vast majority of the poems and are the most general ones. They are: "Activities", "Arts & Sciences", "Living", "Love", "Mythology & Folklore", "Nature", "Relationships", "Religion" and "Social Commentaries".

In this aspect we follow the methodology adopted by Lou, Inkpen and Tanasescu

[4]https://www.poetryfoundation.org/poems/browse

[5]https://www.poetryfoundation.org/poems/42891/stopping-by-woods-on-a-snowy-evening has "Activities", "Travels & Journeys", "Nature", etc., for instance.

(2015), as they focused their work mostly to this same set of topics. They also provided as an extra section an experiment on some of the sub-topics of a couple of these, but we will not follow that particular approach.

A poem can be associated with any number of topics. Even the main ones, which we are working with, are not mutually exclusive. In other words, a poem can be associated with the topics "Activities" and "Nature" at the same time, for instance. This has direct implications for us, since it means it would not make sense to train only one classifier that would output a single topic among our chosen set when classifying a poem. We describe better our approach with regards to this in section 4.2.2.

While we consider topics to be a good choice as the target for classification, we also acknowledge that we might face a slightly larger degree of arbitrary labeling than in more simple scenarios, such as when identifying if a content is positive or negative. For instance, a human expert labeler, when faced with a poem about a person going out for a daily ritual worship of the goddess of trees, might decide to label it with the topics Activities, Mythology & Folklore, Nature or Religion - in fact, any combination of those - given their particular background. This might make it harder for a machine learning model to learn how to correctly classify samples, but it is also something to be expected in a real-world application, and thus a very interesting problem to deal with.

4.2.2 Classification approach

Since poems can be associated with any number of the target topics at the same time, we could frame the process as either a *multi-label classification* problem or as *multiple binary classification problems*. In this work, we chose to use binary classifiers for each individual topic. This is another point in which we are using a similar methodology to Lou, Inkpen and Tanasescu (2015).

We will then have a total of 9 classifiers, each one trained to give a binary response to the question "Is this poem about topic X?". With this setup, if it was desired to know all main topics relevant to a poem, it would be necessary to run it through all 9 classifiers.

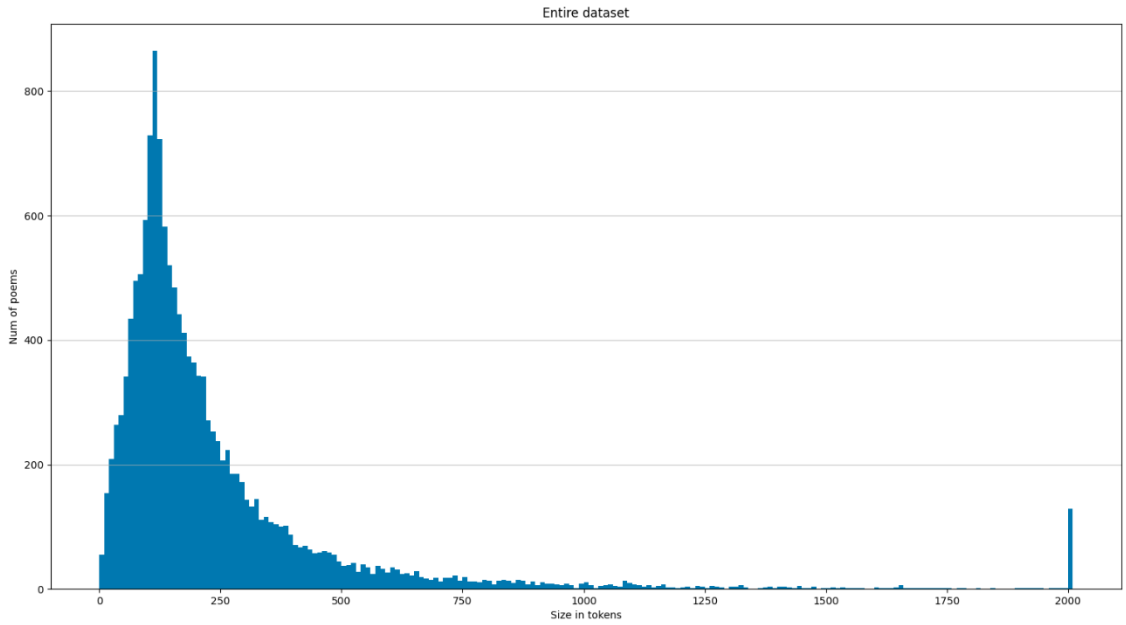
This approach has the benefit of making it very easy for us to find out and explore the specific characteristics of each topic. For instance, by specializing models so that they have the best configuration for the topic they are expected to recognize, instead of being restricted to a global configuration.

4.2.3 Data selection

One issue that becomes very clear when we choose to model the problem as multiple binary classification problems, however, is the fact that the topics do not have the same distribution across the dataset. In fact, some of them are very rare.

As we mentioned before, of the 15652 poems available, 14310 were associated with one of the main topics. We discard those that are not associated with any of the nine we are working with and keep only the 14310 as the initial unfiltered dataset.

Figure 4.1 – Poems by size in unfiltered dataset

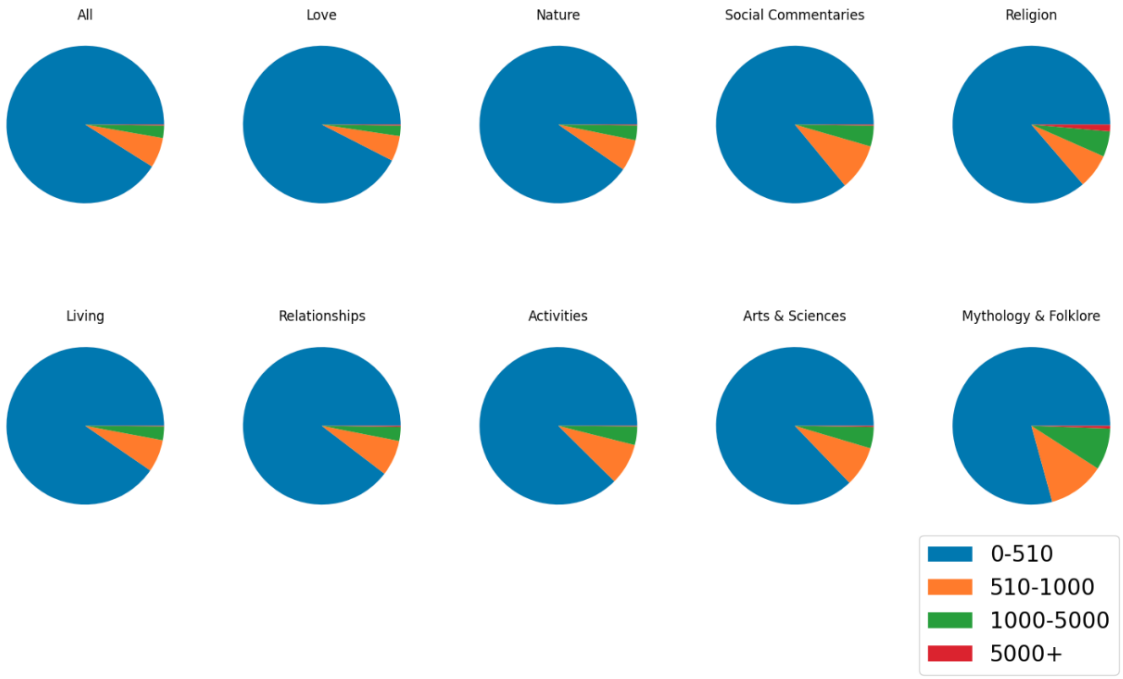


Source: The Author

Then we pick up again the discussion started in Section 4.1.2, where we mentioned that we need to make a decision about the maximum size limit that our chosen language model imposes. It forces us to make a decision between using strategies to incorporate poems larger than this limit or discarding them. Although our variant of BERT is prepared to handle 512 tokens, we consider the cutoff point at 510, in order to preserve space for the special tokens that BERT uses to mark the start and end of input sequences. Figure 4.1 shows the distribution of poems by their size in the initial dataset. We can see that despite having a long tail above the 500s, the significant majority of the poems are smaller than that. The figure uses 2000 as a cutoff point in order to keep the x axis at a reasonable scale, so all poems with more than 2000 tokens end up clumped in the rightmost position. When investigating why there were outliers with so many tokens, we found out that some of the

entries are either entire chapters of books or longer epic poems that could be considered entire books by themselves, especially ancient ones.

Figure 4.2 – Poem distribution by size ranges for all topics



Source: The Author

Figure 4.2 shows an alternative view of the distribution by using a more permissive upper bound of 5000. It also shows the distribution when considering all topics on the top left, as well as for each individual topic. The area in blue corresponds to the amount of poems that would fall within the size limit of the model, while the areas in other colors represent the poems that would need to be treated differently. We again notice that the amount of poems that would fall within our cutoff size limit is a reasonable proportion.

Table 4.1 – Poems by topic in the filtered dataset

Topic	**Number of Poems**	**Percentage of Total**
Activities	2022	15.59%
Arts & Sciences	2840	21.90%
Living	6206	47.84%
Love	2156	16.62%
Mythology & Folklore	566	4.36%
Nature	3617	27.89%
Relationships	3825	29.49%
Religion	1356	10.45%
Social Commentaries	3912	30.16%

Source: The Author

Table 4.1 shows the distribution of topics after we apply the adjustments mentioned here, by cutting off all poems with less than 10 tokens or more than 510. Since topics are not mutually exclusive, the percentages do not add up to one hundred. We are left with 12970 total poems in our filtered dataset.

We can immediately see that no topic is present in more than 50% of the dataset. So, since we are working with binary classifiers, the negative class will always be the majority class during our work.

Table 4.1 also show that two topics can be seen as outliers: Living has the highest percentage, being present in almost half of the poems in that dataset, at 47.84%. Mythology & Folklore, on the contrary, is a very rare occurrence, appearing only in 4.36% of the dataset, with a total amount of 566 poems. Since we mentioned before that one of the possibilities arising from our approach was to have classifiers configured for specific characteristics, this looks like a good point in which one would choose to do that.

Mythology & Folklore, in special, should be a particularly hard case, since it combines the fact that the majority class will completely overwhelm the minority one during training if we use all the available data, and that it inherently does not have many positive samples to teach the models.

The insights presented by Table 4.1 were the main inspiration that led us to focus this work on the issue of imbalance, as we will describe in section 4.4.

4.3 Implementation Setup

Here we give technical details of the setup we used to implement the training. Section 4.3.1 describes the cloud environment we used to have access to GPUs with the necessary power to handle the task of fine-tuning a BERT language model. Section 4.3.2 describe the libraries we used. Section 4.3.3 describes the BERT variant we used, including the architecture of the model, as well as the values we use for the main parameters of the training. Section 4.3.4 describe which metrics we chose to use, among the available ones for classification problems. Section 4.3.5 shows our approach toward threshold moving and which thresholds we chose to consider.

4.3.1 Environment

We used the Python[6] programming language to develop this work. Data preparation was done with Python scripts, while the training and validation of the models were done with Jupyter[7] notebook environments.

We use the Kaggle notebooks environment to train the models, as it provides free access to GPUs that can handle the requirements of the library used. The more common platform used for such work is Google Colaboratory (Colab), which also provides a notebook environment with access to GPUs. We performed early experiments that showed that both provided the features we would need for our purposes, meaning that either one could be used. We chose Kaggle as, at the time of the implementation of this work, it provided a better visibility of the usage of GPU resources and how much was still available, allowing for a more precise planning of the work.

4.3.2 Libraries

We use Tensorflow[8] and Keras[9] libraries for training models, as they are industry-standards with a simple API that offers the methods we needed. With them, we load a pre-trained BERT model straight from Tensorflow Hub by initializing an instance of the KerasLayer class passing as an argument the link of the pre-trained model.

We also use Pandas[10] and Scikit-Learn[11] libraries as helpers to solve tangential needs related to data manipulation. The scraping of the data from the Poetry Foundation site was done with Beautiful Soup[12].

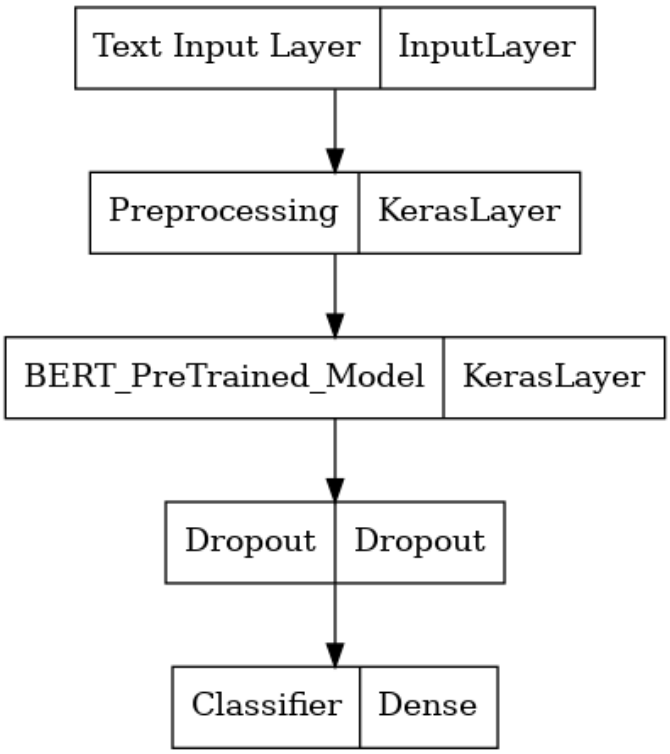
4.3.3 Experimental settings

For all reported experiments we use the BERT Base Uncased variant for English, with 12 hidden layers, hidden size of 768 and 12 attention heads.

The architecture for our model is as follows: first, there is a text input layer, to

[6]https://www.python.org/
[7]https://jupyter.org/
[8]https://www.tensorflow.org/
[9]https://keras.io/
[10]https://pandas.pydata.org/
[11]https://scikit-learn.org/
[12]https://beautiful-soup-4.readthedocs.io/en/latest/

Figure 4.3 – Model architecture



Source: The Author

which the textual content of the poems are fed. The outputs of this layer then go into the preprocessing layer, which generates the input in the format expected by BERT from the text. The resulting embeddings then are fed into the next layer, which contains the pre-trained BERT model itself. Next in line is a droupout layer with 0.1 dropout probability during training. Finally, our output layer is a dense layer with a sigmoid activation function, from which we extract the prediction of the model, which is a decimal number ranging from 0.0 to 1.0. Figure 4.3 shows a diagram representing the architecture we used.

Our standard configuration for the experiments is as follows: we use a batch size of 32, the AdamW optimizer and a learning rate of 5e-5. These are in accordance with the original paper that introduced BERT (DEVLIN et al., 2019), except that instead of the default Adam optimizer used there we use the variant proposed in Loshchilov and Hutter (2019).

We use Binary Cross-Entropy as a loss function for all experiments except for the one in which we use the Focal Loss function. The value of the binary cross-entropy loss for any given sample is calculated as:

$$Loss(y, \hat{y}) = -(y * log(\hat{y}) + (1 - y) * log(1 - \hat{y}))$$

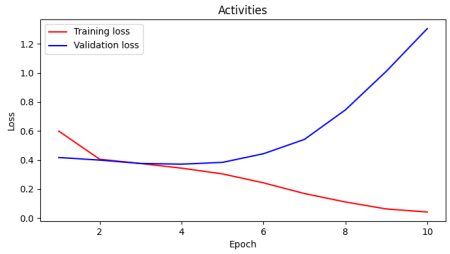
where y is the true class for that sample and is either 0 or 1, and $\hat{y}$ is the model's prediction for that sample.

We let the training run for 10 epochs. This is a far larger time than what was suggested in the original paper, which would be around 3 to 5 epochs. However, we have configured the process to evaluate the model against the validation dataset at the end of each epoch, and whenever it sees an improvement, it saves the current state of the model. We defined improvement as the value of the loss function on the validation data being reduced from the previously seen lowest value. Then, at the end of the 10 epochs, we pick the last version of the model that was saved given this criteria as the result of the training. Empirically, we found that models tended to reach their best version at around epochs 3 or 4, with some extreme cases achieving it only at epoch 6. But no model we trained ever improved after epoch 7. Given this, we are confident that letting the training run for 10 epochs is enough to catch all the improvement it could get, while not running the risk of overfitting since we discard any model that does not improve from the best seen.

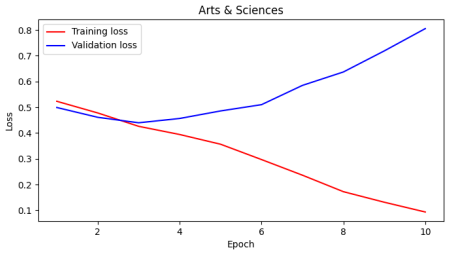
Figure 4.4 shows the training history of the baseline models for each topic, with loss on the y axis and epoch on the x axis. The training loss is displayed in red, while the validation loss is blue. From it we can confirm that at epoch 10 all models were already heavily overfitting, so it would be pointless to continue the process. We take as the final model the one saved when the blue line was at its lowest. This figure shows the histories only for the baseline models, but all later experiments showed the same characteristics.

4.3.4 Selection of evaluation metrics

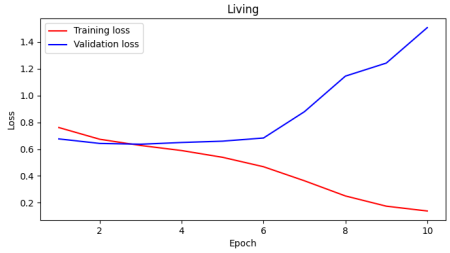
Given the imbalanced nature of our dataset, accuracy is not a good metric. For instance, considering the entire dataset that we acquired our worst imbalance case is a class which has only 714 samples out of a total of 14243. If a classifier was trained specifically to identify if a sample belonged to it or not, and that classifier chose to always answer negatively, regardless of what was fed to it, when evaluated over the entire dataset the classifier would be correct $14310 - 718 = 13592$ times out of the 14310, which would give it an accuracy of around 94.98%. In other words, it would at the same time be highly



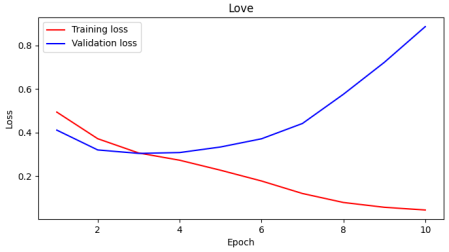
(a) Activities



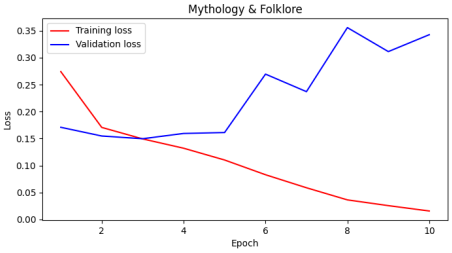
(b) Arts & Sciences



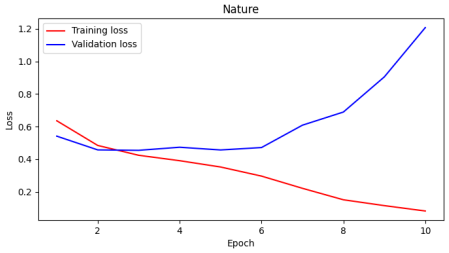
(c) Living



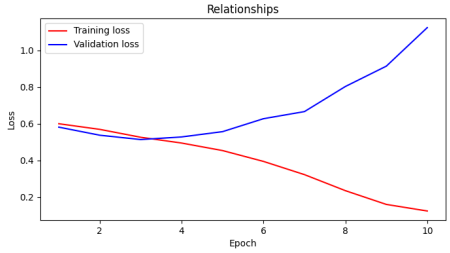
(d) Love



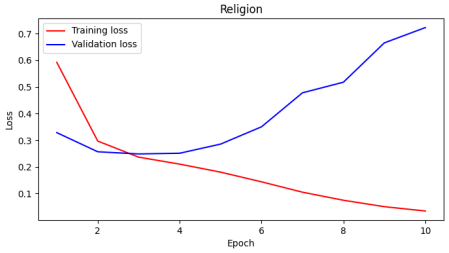
(e) Mythology & Folklore



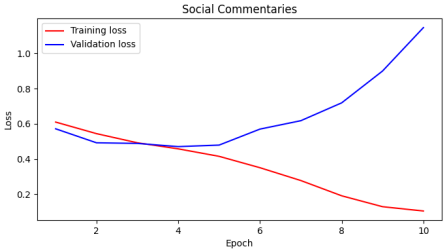
(f) Nature



(g) Relationships



(h) Religion



(i) Social Commentaries

Figure 4.4 – Training histories of baseline models

accurate, and totally useless.

For problems that deal with imbalanced datasets such as ours we need other metrics. The metrics we chose to consider are Precision, Recall and F1 score, with the macro-F1 score being our main standard to evaluate the global performance of the models.

We still report precision and recall for both the positive and negative classes when we show the results of individual experiments, but when we make comparisons between experiments it is always based on the macro-F1 score achieved, considering both classes. When we refer just to F1 score in the text, without specifying if it is for the positive or negative classes, we are referring to the macro-F1.

4.3.5 Thresholds considered

We used threshold moving alongside every experiment we ran in this work. The threshold range used was from 0.05 to 0.95, and the step size was 0.05. We evaluate the results by using each of these thresholds with our main metric, macro-F1.

For every experiment we show tables of the metrics extracted for both a standard threshold of 0.5 and the threshold that gave the best performance for that particular model. Then, when we compare the results of different experiments, we always use only the score achieved with the best performing threshold.

We deviate briefly from the proposed range in the case of Focal Loss to investigate some peculiarities we found with it, but we do not bring the results from this deviation into comparison with the other experiments that followed rigorously our proposed methodology. We give more details about this in the specific section detailing the results for the Focal Loss experiment, section 5.4.

4.4 Experiment definitions

Here we present the rationale and configuration of each experiment we performed, then in Section 5 we present their results.

Given the scenario we found after doing this initial probing of the proposed work, we decided to focus on the problem of the imbalance of the classes in the dataset. We expect this to be one of the major issues that classifiers trained with this dataset will encounter, since they will be disproportionately rewarded for classifying a poem as not

belonging to the class they were trained to recognize.

There are several other challenges with this dataset that would be worthwhile pursuing. One of them is the fact that the amount of data is not that big (although within the confines of labeled poetry datasets it is one of the biggest we could find). Another is how to best incorporate the outliers in size so as to make them work with methods that require a limited maximum size, as BERT does. There's also the question of the choice and labeling of the topics, to what degree is it arbitrary or universal, and so on. We leave all of these as suggestions for future works.

In order to explore our chosen problem in more depth, we ran experiments with a few selected strategies found in the literature to mitigate this particular issue, and compare them with a baseline training in which no action was performed to address the imbalance. Section 4.4.2 describes our baseline of comparison for the experiments. Section 4.4.3 describes our experiment with the undersampling of the majority class, while 4.4.4 describes the experiment where we used weights for each class to scale the loss function, and 4.4.5 describes the experiment in which we used Focal Loss as a loss function.

4.4.1 Dataset split

In every experiment described here, we used a simple train/test/validation split and ran only one training for each relevant case we wanted to test. We chose this approach due to the computational cost of the process and the limitations we had for the scope of this work. For reference, each individual classifier takes about one to a few hours to be trained, even using the cloud environment described before. Since we are working with binary classifiers for each topic, and we have 9 topics, each experiment requires at least 9 training rounds, or 9 hours in the very best scenario. We understand that this approach leaves the training more susceptible to random circumstances that might impact the final performance of the model, however it was the compromise we had to make, given the constraints around the scope of this work.

We make the dataset split early on in the process, and then use the same withheld test set to evaluate every experiment. We expect this to help in making the results more comparable.

We split 10% of the data to take as a withheld test set, and leave the remaining 90% available for train and validation sets when training the models. While we tried not to deviate too much from the distribution of the full dataset, keeping it perfectly equal is a

challenging task due to the fact that topics are not mutually exclusive. That is, by picking a poem to fill a topic's quota, we might inadvertently change the distribution of another topic in the test set. Due to this, we do not attempt to keep an exact match in distributions between training and test data, but we tried to ensure that every topic is represented enough to provide a fair evaluation to its model while not distorting excessively the relative distributions between topics. So, we still wanted Living to have a disproportionately high representation, Mythology & Folklore to be rare, and so on. The only criteria we used to guide the split is the distribution of the resulting sets, other than that we choose randomly between any poems that could be picked.

Table 4.2 shows how the set available for train/validation split during training is distributed, while table 4.3 shows the same for the test set.

Table 4.2 – Poems by topic in the train+validation set

Topic	**Number of Poems**	**Percentage of Total**
Activities	1762	15.08%
Arts & Sciences	2505	21.45%
Living	5462	46.77%
Love	1876	16.06%
Mythology & Folklore	499	4.27%
Nature	3197	27.38%
Relationships	3323	28.45%
Religion	1211	10.37%
Social Commentaries	3447	29.52%
Train+Validation dataset	*11678*	*100.00%*

Source: The Author

Table 4.3 – Poems by topic in the test set

Topic	**Number of Poems**	**Percentage of Total**
Activities	260	20.12%
Arts & Sciences	335	25.93%
Living	744	57.58%
Love	280	21.67%
Mythology & Folklore	67	5.19%
Nature	420	32.51%
Relationships	502	38.85%
Religion	145	11.22%
Social Commentaries	465	35.99%
Test dataset	*1292*	*100.00%*

Source: The Author

4.4.2 Baseline

As a baseline, we train classifiers for each topic using the default configurations described on Section 4.3.3, without doing anything to address the imbalance between classes. We use the results achieved to evaluate the results of the other experiments, in which we try different strategies to mitigate that problem, so we can have an idea of their impact.

4.4.3 Experiment 1 - Undersampling

The first strategy we implemented was the undersampling of the majority class. Given our choice of binary classifiers for each topic, and the fact that there is no topic which is associated with half of the dataset or more, in our case the majority class is always the negative one. So for this experiment we always keep all of the positive samples available for training, but we discard from training a given amount of the negative samples.

We framed this as the question of how much imbalance we want to train the classifier with. We experimented with five levels of imbalance, which can be thought of as no imbalance, a small amount of imbalance, a medium amount, a large amount and full imbalance. Mathematically, we represent these levels as the values 0.0, 0.25, 0.5, 0.75 and 1.0, respectively. For each topic, we train a classifier for each of these levels, and at the end evaluate which level gave the best performance for each topic.

The way we applied this factor to define the specific number of negative samples to use during training is as follows: we always pick the full amount of positive samples available in the training set. We also always pick as many negative samples as positive samples. This leaves an excess of negative samples we could pick. We then use the ratio of imbalance desired to modulate how much of the excess we pick. The formula we apply is the following:

$$N = \lfloor P + (T * R * F) \rfloor$$

Where N is the total number of negative samples to pick, P is the number of positive samples available in the training set, T is the total number of samples in the training set, R is the ratio of the excess of negatives in the training set and F is the

factor we are using, for which we tested the values mentioned before. This is a heuristic formulation we came up while developing the work, amongst several possible ones.

We do not apply any specific consideration of which negative samples to pick and which to discard, other than the amount to pick.

4.4.4 Experiment 2 - Weighted Loss Function

For this experiment we use a weighted loss function. Following the description on section 2.3.2.1, we scaled our default loss function (binary cross-entropy) with different weights for the positive and the negative classes.

In order to compute the weights for each class, we use the heuristic provided by the scikit-learn helper function compute_class_weight[13]. In practice, the heuristic generates the weight for each class as follows:

$$\text{Weight For Class C} = \frac{\text{Total Samples}}{\text{Samples Of Class C} * \text{Number Of Classes}}$$

From the formula, we can see that the weight of a class C is inversely proportional to the proportion of the samples of C in the dataset. Thus, we can see that a class which is overabundant in the dataset will receive a smaller weight, since the *Samples Of Class C* factor in the denominator will be large. Conversely, a class that is underrepresented will receive a larger weight due to the same factor. The *Number Of Classes* factor is used only to stabilize the magnitude of the values that the loss function will generate.

4.4.5 Experiment 3 - Focal Loss Function

For our final experiment, we replace our loss function with the Focal Loss function, described in section 2.3.2.2.

For this work we use the values mentioned as a good pair in the paper, setting γ to 2 and α to 0.25.

[13] https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html

5 RESULTS

In this section, we present the results for each experiment we ran. We compare each result against the baseline individually, discussing its performance and more interesting characteristics. At the end, we present a full comparison between all of the results.

For all comparisons, we select the results achieved by selecting the best possible threshold for that particular case.

Tables detailing the results of each individual experiment contain the scores for precision, recall and F1 considering both the positive and negative classes. Tables that show a comparison between different experiments show only the F1 score.

Whenever relevant, we highlight the better result in comparison by printing it in **bold**.

5.1 Baseline

Table 5.1 shows the results of the baseline experiment considering 0.5 as the threshold to classify a sample as a positive. Table 5.2 shows the result of the same experiment, but when we consider the threshold that gives the best F1 score. Comparing both tables we can see that we can improve the results just by adjusting our interpretation of the classifier's response without retraining. With this, we can compensate the pessimism, so to speak, of the classifier.

Table 5.1 – Baseline results with threshold set to 0.5

Topic	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	63.04%	22.31%	83.17%	96.71%	32.95%	89.43%	61.19%
Arts & Sciences	73.91%	25.37%	78.76%	96.87%	37.78%	86.88%	62.33%
Living	74.66%	51.88%	53.81%	76.09%	61.22%	63.04%	62.13%
Love	77.61%	37.14%	84.8%	97.04%	50.24%	90.51%	70.37%
Mythology & Folklore	25.0%	0.43%	63.94%	99.27%	0.85%	77.78%	39.31%
Nature	66.67%	62.38%	82.42%	84.98%	64.45%	83.68%	74.07%
Relationships	66.27%	33.27%	67.79%	89.24%	44.3%	77.05%	60.67%
Religion	71.15%	25.52%	91.29%	98.69%	37.56%	94.85%	66.21%
Social Commentaries	79.09%	44.73%	75.02%	93.35%	57.14%	83.19%	70.17%
Average	*66.38%*	*33.67%*	*75.67%*	*92.47%*	*42.94%*	*82.93%*	*62.94%*

Source: The Author

One thing that becomes immediately clear, and that will be a constant theme across all experiments, is the outlier nature of the topic Mythology & Folklore. We already saw that this is rarest topic in the dataset, appearing in less than 600 of the poems. So we expect classifiers to have a much harder time when dealing with it than with the other

Table 5.2 – Baseline results with best thresholds

Topic	**Threshold**	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	0.3	52.34%	47.31%	87.04%	89.15%	49.7%	88.08%	68.89%
Arts & Sciences	0.25	56.71%	50.45%	83.3%	86.52%	53.4%	84.88%	69.14%
Living	0.4	70.51%	68.15%	58.64%	61.31%	69.31%	59.95%	64.63%
Love	0.3	67.58%	52.86%	87.7%	92.98%	59.32%	90.26%	74.79%
Mythology & Folklore	0.05	33.54%	11.83%	63.65%	86.82%	17.49%	73.45%	45.47%
Nature	0.45	65.33%	65.95%	83.53%	83.14%	65.64%	83.33%	74.49%
Relationships	0.25	57.35%	70.72%	78.16%	66.58%	63.34%	71.91%	67.62%
Religion	0.2	43.66%	42.76%	92.78%	93.03%	43.21%	92.9%	68.05%
Social Commentaries	0.25	66.74%	65.59%	80.84%	81.62%	66.16%	81.23%	73.69%
Average	-	*57.09%*	*52.85%*	*79.51%*	*82.35%*	*54.17%*	*80.67%*	*67.42%*

Source: The Author

topics. In the context of the baseline setting, looking at the Table 5.2 with the best thresholds, we see that its F1 score is much worse than the rest, being almost 20% lower than the next lowest score. Also, the best threshold for it was the lowest in the range we check, indicating that the classifier is extremely pessimist when trying to identify if a poem deals with this topic.

Although no other of the best thresholds is as low as the one for Mythology & Folklore, we notice that they are all located below the standard 0.5 one. That is, if we do nothing to address the imbalance of the dataset, all classifiers will tend toward giving lower scores.

We also note that with the best thresholds a general pattern appears of the precision of the positive class being lower than with the standard threshold, while the positive recall gets higher. The situation is the opposite for the negative class, with precision increasing and recall decreasing. This is actually a natural consequence of the fact that all of the best thresholds for the baseline classifiers were lower than the standard 0.5. By lowering the threshold, we are increasing the range of outputs that make us interpret the prediction of a sample as putting it in the positive class. This is guaranteed to not *decrease* the number of samples being predicted as belonging to the positive class, while making it likely that the number of such predictions *increases*. By predicting more samples as belonging to the positive class, we will naturally tend to incorrectly classify more of the members of the negative class that were previously near the borderline and were previously correctly classified as negative, which decreases precision. Also, by increasing the number of samples we classify as belonging to the positive class, we tend to correctly identify positive samples that were previously just below the line of the threshold, which increases the recall. The exact same logic can be analogously applied to the negative class, by inverting the impact on precision and recall for that class. Hence, the pattern is exactly what we would expect to find, given that the best threshold is always lower than the standard one.

This is still something worth noticing, however, since it can have implications depending on the use case we want for the classifier. In our case, we were just interested in making a general exploration of the performance and behavior of such classifiers in the proposed context, so we use the macro-F1 score as a metric that strives to balance the performance of both precision and recall, for both the positive and negative classes. But if we were working under a more specific scenario in which we would want to prioritize either the amount of samples we classify as positive, or the strictness of our predictions, then it would make more sense to look at the other metrics. For instance, if our goal was to perform information retrieval and we were willing to tolerate some false positives as long as we could retrieve as many correct instances of a class as possible, we might prefer increasing the recall, and thus either make that metric the target for maximization when applying threshold moving, or just generally favoring lower thresholds. Conversely, if we had an use case where we would want to be as strict as possible in cutting off poems that do not belong to the desired class, such as for instance if we needed to filter out as many poem submissions to a contest as possible, so that our human judges would be handled a smaller set of poems to examine manually, we might opt for the opposite approach, by either making precision our target metric or generally preferring higher thresholds.

Table 5.3 shows a cleaner view of the results, displaying only the F1 score that the baseline implementation achieved for each topic. It is against those values that we will compare the results achieved by all other experiments.

Table 5.3 – Baseline F1 scores

Topic	**Baseline**
Activities	68.89%
Arts & Sciences	69.14%
Living	64.63%
Love	74.79%
Mythology & Folklore	45.47%
Nature	74.49%
Relationships	67.62%
Religion	68.05%
Social Commentaries	73.69%
Average	*67.42%*

Source: The Author

5.2 Experiment 1 - Undersampling

Table 5.4 shows a summary of the results achieved with undersampling and fixing the threshold at 0.5, while Table 5.5 shows the analogous results when picking the best threshold. The most interesting point here is in the column "Imbalance", which shows the imbalance ratio that produced the best results. A ratio of 0.0 would mean that the positive and the negative classes had the same amount of samples in the training set, meaning the most severe undersampling of that class and consequently the least amount of data used in training. Conversely, a ratio of 1.0 of imbalance would mean that we do not undersample the negative class in order to balance the training set, and use the highest amount of data in training.

Table 5.4 – Summary of undersampling results for threshold 0.5

Topic	**Imbalance**	**Threshold**	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	0.5	0.5	71.08%	42.14%	85.61%	95.26%	52.91%	90.18%	71.55%
Arts & Sciences	0.25	0.5	60.67%	51.79%	87.18%	90.71%	55.88%	88.91%	72.39%
Living	0.75	0.5	73.18%	63.44%	57.96%	68.43%	67.96%	62.76%	65.36%
Love	0.25	0.5	59.01%	59.64%	88.80%	88.54%	59.32%	88.67%	74.00%
Mythology & Folklore	0.25	0.5	34.48%	14.92%	95.49%	98.45%	20.83%	96.94%	58.89%
Nature	0.5	0.5	72.48%	56.43%	81.04%	89.68%	63.45%	85.14%	74.30%
Relationships	0.5	0.5	62.96%	60.96%	75.68%	77.21%	61.49%	76.44%	69.19%
Religion	0.5	0.5	61.04%	32.41%	91.93%	97.38%	42.34%	94.58%	68.46%
Social Commentaries	0.0	0.5	69.60%	59.57%	78.97%	85.37%	64.19%	82.04%	73.12%
Average	-	-	*62.72%*	*49.03%*	*82.52%*	*87.89%*	*54.26%*	*85.07%*	*69.70%*

Source: The Author

Table 5.5 – Summary of undersampling results for best threshold

Topic	**Imbalance**	**Threshold**	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	0.5	0.25	62.29%	54.29%	87.79%	90.91%	58.01%	89.32%	73.67%
Arts & Sciences	0.5	0.25	57.75%	58.57%	88.49%	88.14%	58.16%	88.32%	73.24%
Living	0.75	0.4	69.29%	77.96%	63.96%	53.10%	73.37%	58.03%	65.70%
Love	0.75	0.35	62.12%	58.57%	88.71%	90.12%	60.29%	89.41%	74.85%
Mythology & Folklore	1.0	0.1	27.62%	43.28%	96.80%	93.80%	33.72%	95.27%	64.50%
Nature	1.0	0.3	64.74%	69.52%	84.78%	81.77%	67.05%	83.25%	75.15%
Relationships	0.5	0.45	60.73%	66.53%	77.36%	72.66%	63.50%	74.93%	69.22%
Religion	0.5	0.6	72.58%	31.03%	91.87%	98.52%	43.48%	95.08%	69.28%
Social Commentaries	1.0	0.25	69.19%	62.80%	80.11%	84.28%	65.84%	82.14%	73.99%
Average	-	-	*60.70%*	*58.06%*	*84.43%*	*73.7%*	*58.16%*	*83.97%*	*71.07%*

Source: The Author

An interesting fact about Table 5.4 is that, when fixing the threshold at 0.5, no topic achieved the best performance by using the most of the available data. We will see in the next paragraph that this is not the case when we allow the threshold to move and pick the best one. In fact, for the standard threshold there was even one topic, Social Commentaries, that had its best performance by being perfectly balanced, and dismissing all of the exceeding samples of the negative class. It is interesting that it was not Mythology & Folklore, the most imbalanced one, but likely the fact that Mythology & Folklore only has 566 poems in total in the whole dataset, while Social Commentaries has 3912, is

playing a role in here, as Mythology & Folklore has just too little data to be able to afford the luxury of discarding all of the excess.

Examining the Table 5.5, in turn, we can see almost the opposite, that no topic performed the best by using 0.0 or 0.25 imbalance, which would mean a more balanced set but with less data to be trained on. It makes sense that using too little data would incur in performance problems, so that is an expected finding when we give the threshold freedom to search for the best result. However, we also notice that only 3 topics, Mythology & Folklore, Nature and Social Commentaries performed the best by using all the available data (highest imbalance, smallest undersampling). Two other topics, Living and Love, had their best performances when using a large imbalance (0.75). But the ratio that performed the best in most cases was a moderate amount of imbalance, giving the best result for 4 topics: Activities, Arts & Sciences, Relationships and Religion. This seems to indicate that finding a middle point between using the most available data and not letting one class overwhelm the other is a good strategy for this dataset.

Looking at both Tables 5.4 and 5.5 together, we can notice that picking the best threshold possible made the best level of imbalance for all topics either stay the same or increase, but never decrease. In other words, giving the threshold freedom to move allowed more data to be used to generate the best performance in some cases, while never requiring less data to be used to optimize the result.

For future comparisons in which we include the undersampling method, we always consider the level of imbalance that performed the best for the best thresholds in this experiment, in other words, those in Table 5.5, as the canonical result.

We can notice in here that, similarly as what happened with the baseline, with undersampling most of the best thresholds tend to still be below the standard 0.5. The only exception being the Religion topic, with has a slightly higher best threshold, at 0.6. This means that the impacts on precision and recall are roughly the same as mentioned in Section 5.1, with the exception of the previously mentioned topic. The impacts for thresholds above 0.5 are explained in more depth in section 5.3, where we will see that several topics have this characteristic. It makes sense that the threshold would still tend towards being lower than 0.5, since by undersampling we are still presenting the classifiers with a universe in which there is a larger amount of negative cases than positive ones during training, which pushes the prediction output values down, and not compensating this by any other means.

Table 5.6 shows the comparison between the F1 scores obtained by the models

trained with the baseline configuration and those trained using an undersampling of the negative class.

We can see that the undersampling method outperforms the baseline in every topic. While this is an interesting fact, the more interesting result was the one shown previously, that it is not always the highest level of imbalance (which means more data gets used for training) that yields the better results for this experiment.

Table 5.6 – Comparison Baseline vs Undersampling F1 scores

Topic	**Baseline**	**Undersampling**
Activities	68.89%	**73.67%**
Arts & Sciences	69.14%	**73.24%**
Living	64.63%	**65.70%**
Love	74.79%	**74.85%**
Mythology & Folklore	45.47%	**64.50%**
Nature	74.49%	**75.15%**
Relationships	67.62%	**69.22%**
Religion	68.05%	**69.28%**
Social Commentaries	73.69%	**73.99%**
Average	*67.42%*	***71.07%***

Source: The Author

5.3 Experiment 2 - Weighted Loss Function

Table 5.7 shows the class weights we used for each topic. These were computed automatically based on the distribution present in the training set of this experiment, using the helper function described in section 4.4.4. As expected, the most imbalanced topic, Mythology & Folklore, showed the largest difference between the weights, while the least imbalanced one, Living, ended up with a very small difference.

Table 5.8 shows the results achieved by using pondered class weights when setting the threshold to 0.5. Table 5.9 shows the results achieved by choosing the best possible threshold. The interesting point to note here is that the best thresholds increased by adopting the proposed weights in the weighted loss function, being now closer to 0.5. However, even with this adjustment the Mythology & Folklore topic still has a very low ideal threshold, at 0.25.

Here we see a few differences in the pattern of precision and recall of positive and negative classes than what we saw in the baseline results, but the pattern still follows the same logic. The differences are due to the fact that now, for some of the topics, the best

Table 5.7 – Weights used for each class

Topic	**Negative**	**Positive**
Activities	0.59	3.32
Arts & Sciences	0.64	2.33
Living	0.94	1.07
Love	0.60	3.11
Mythology & Folklore	0.52	11.70
Nature	0.69	1.83
Relationships	0.70	1.76
Religion	0.56	4.83
Social Commentaries	0.71	1.69

Source: The Author

Table 5.8 – Weighted loss function results with threshold set to 0.5

Topic	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	44.34%	54.23%	87.78%	82.85%	48.79%	85.24%	67.02%
Arts & Sciences	58.7%	48.36%	82.97%	88.09%	53.03%	85.45%	69.24%
Living	69.84%	69.09%	58.63%	59.49%	69.46%	59.06%	64.26%
Love	61.4%	59.64%	88.92%	89.62%	60.51%	89.27%	74.89%
Mythology & Folklore	44.44%	23.88%	95.94%	98.37%	31.07%	97.14%	64.1%
Nature	63.08%	64.29%	82.64%	81.88%	63.68%	82.26%	72.97%
Relationships	60.35%	61.55%	75.26%	74.3%	60.95%	74.78%	67.86%
Religion	50.0%	37.93%	92.39%	95.2%	43.14%	93.77%	68.46%
Social Commentaries	64.81%	72.47%	83.42%	77.87%	68.43%	80.55%	74.49%
Average	*57.44%*	*54.6%*	*83.11%*	*83.08%*	*55.45%*	*83.06%*	*69.25%*

Source: The Author

threshold is higher than the standard 0.5, which inverts the impact on precision and recall due to the same argument. The higher the threshold, the fewer the samples are predicted as positive. So, there is a tendency for fewer false positives, which leads to higher precision and lower recall, and conversely the opposite for the negative class. An interesting side effect of the best thresholds being in general closer to 0.5 that is noteworthy is the fact that, while by just tuning the threshold without addressing the issue in any other form we can improve the macro-F1 score (as we saw with the baseline results), by addressing it with the strategy of using class weights we can also later apply threshold moving and improve the macro-F1 score, but with a smaller impact on precision and recall. For specific use cases, such as the ones we mentioned in section 5.1, this might be useful as it would allow us to improve the overall performance of the classifier without sacrificing too much on our desired priorities. The present strategy might have an advantage over the undersampling strategy with regards to this, since we saw that with that strategy we are still bound to end up with lower best thresholds overall.

Table 5.10 shows the comparison between the F1 scores obtained by the models trained with the baseline configuration and those trained using class weights. The perfor-

Table 5.9 – Weighted loss function results with best thresholds

Topic	**Threshold**	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	0.6	51.08%	45.38%	86.62%	89.05%	48.07%	87.82%	67.94%
Arts & Sciences	0.55	60.47%	46.57%	82.69%	89.34%	52.61%	85.89%	69.25%
Living	0.45	68.55%	76.75%	62.31%	52.19%	72.42%	56.8%	64.61%
Love	0.55	64.29%	57.86%	88.65%	91.11%	60.9%	89.86%	75.38%
Mythology & Folklore	0.25	38.46%	29.85%	96.21%	97.39%	33.61%	96.8%	65.2%
Nature	0.55	66.41%	61.19%	81.99%	85.09%	63.69%	83.51%	73.6%
Relationships	0.45	58.55%	67.53%	77.14%	69.62%	62.72%	73.19%	67.95%
Religion	0.7	70.31%	31.03%	91.86%	98.34%	43.06%	94.99%	69.03%
Social Commentaries	0.5	64.81%	72.47%	83.42%	77.87%	68.43%	80.55%	74.49%
Average		*60.32%*	*54.29%*	*83.43%*	*83.33%*	*56.17%*	*83.27%*	*69.72%*

Source: The Author

mance on 6 of the topics improved by applying the weights, while the baseline was still better for 3 of them. Despite some of the differences between the scores being small, a combination of having the majority of the better results plus massively outperforming on Mythology & Folklore pushes the average score for this experiment more than 2 points above the baseline. The considerable advantage in that particular topic makes sense, as it is our biggest outlier in terms of imbalance.

Table 5.10 – Comparison Baseline vs Weighted loss function F1 scores

Topic	**Baseline**	**Weighted Loss**
Activities	**68.89%**	67.94%
Arts & Sciences	69.14%	**69.25%**
Living	**64.63%**	64.61%
Love	74.79%	**75.38%**
Mythology & Folklore	45.47%	**65.20%**
Nature	**74.49%**	73.60%
Relationships	67.62%	**67.95%**
Religion	68.05%	**69.03%**
Social Commentaries	73.69%	**74.49%**
Average	*67.42%*	***69.72%***

Source: The Author

5.4 Experiment 3 - Focal Loss Function

Table 5.11 shows the results of using Focal Loss as a loss function when setting the threshold to 0.5, while table 5.12 shows the results when we choose the best possible threshold within the range we defined in our methodology. Three things draw our attention in here: a) that the scores for the standard 0.5 threshold are outstandingly bad; b) that the ideal threshold is the same (0.6) for every single topic - this is not the case for any of the other experiments we ran, in all of them there is always some diversity; c) that

the small difference of 0.1 in threshold, from 0.5 to 0.6, can result in much better scores overall.

Table 5.11 – Focal Loss results with threshold set to 0.5

Topic	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	20.12%	100.0%	0%	0.0%	33.51%	0%	16.75%
Arts & Sciences	25.95%	100.0%	100.0%	0.1%	41.21%	0.21%	20.71%
Living	57.59%	100.0%	0%	0.0%	73.08%	0%	36.54%
Love	21.95%	99.64%	95.24%	1.98%	35.98%	3.87%	19.92%
Mythology & Folklore	5.59%	98.51%	99.1%	8.98%	10.58%	16.47%	13.52%
Nature	32.51%	100.0%	0%	0.0%	49.07%	0%	24.53%
Relationships	38.85%	100.0%	0%	0.0%	55.96%	0%	27.98%
Religion	11.28%	100.0%	100.0%	0.61%	20.28%	1.21%	10.75%
Social Commentaries	35.99%	100.0%	0%	0.0%	52.93%	0%	26.47%
Average	*27.76%*	*99.79%*	*43.82%*	*1.3%*	*41.4%*	*2.42%*	*21.91%*

Source: The Author

Table 5.12 – Focal Loss results with best thresholds

Topic	**Threshold**	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	0.6	60.87%	26.92%	83.86%	95.64%	37.33%	89.36%	63.35%
Arts & Sciences	0.6	54.24%	47.76%	82.45%	85.89%	50.79%	84.14%	67.46%
Living	0.6	59.93%	95.3%	67.89%	13.5%	73.59%	22.53%	48.06%
Love	0.6	63.21%	47.86%	86.48%	92.29%	54.47%	89.29%	71.88%
Mythology & Folklore	0.6	44.83%	19.4%	95.72%	98.69%	27.08%	97.19%	62.13%
Nature	0.6	57.64%	78.1%	87.28%	72.36%	66.33%	79.12%	72.73%
Relationships	0.6	57.7%	73.11%	79.42%	65.95%	64.5%	72.06%	68.28%
Religion	0.6	38.92%	49.66%	93.41%	90.15%	43.64%	91.75%	67.69%
Social Commentaries	0.6	62.59%	71.61%	82.63%	75.94%	66.8%	79.14%	72.97%
Average		*55.55%*	*56.63%*	*84.35%*	*76.71%*	*53.84%*	*78.29%*	*66.06%*

Source: The Author

After seeing these questions, we evaluated the classifiers of this experiment with much more granularity, out of curiosity to see if the improvements that could be found in smaller slices were even more impressive. So we scanned every threshold from 0.001 to 0.999 with increments of 0.001. Table 5.13 shows the results we found. We do not take these results to a serious comparison with the other experiments, as it would be unfair with the other experiments that were evaluated only in the standard range we proposed, however we still find a couple of noteworthy things. Firstly, that there is indeed some improvement from the normal scan although it is nothing groundbreaking. A great improvement can be seen for Living, though, which increased in more than 15% its F1 score. It is worthy remembering that Living is the most balanced topic we have in the dataset. And secondly, that the average F1 score achieved with this approach (69.10%) would actually surpass the average F1 score of the baseline models (67.42%).

Looking back at the table for the results with the standard threshold, we can further notice that almost all of the topics have 100% recall for the positive class, with 0% for the negative one. This indicates that we are interpreting all of the responses of the classifier

Table 5.13 – Focal Loss results with more granular scan

Topic	**Threshold**	**Precision(P)**	**Recall(P)**	**Precision(N)**	**Recall(N)**	**F1(P)**	**F1(N)**	**F1**
Activities	0.59	45.49%	42.69%	85.78%	87.11%	44.05%	86.44%	65.24%
Arts & Sciences	0.595	51.2%	57.31%	84.41%	80.88%	54.08%	82.6%	68.34%
Living	0.619	68.46%	70.03%	58.0%	56.2%	69.24%	57.09%	63.16%
Love	0.591	57.04%	57.86%	88.29%	87.94%	57.45%	88.12%	72.78%
Mythology & Folklore	0.594	44.44%	23.88%	95.94%	98.37%	31.07%	97.14%	64.1%
Nature	0.607	61.9%	73.1%	85.8%	78.33%	67.03%	81.89%	74.46%
Relationships	0.604	61.34%	67.33%	77.87%	73.04%	64.2%	75.38%	69.79%
Religion	0.625	55.56%	37.93%	92.46%	96.16%	45.08%	94.27%	69.68%
Social Commentaries	0.613	72.16%	60.22%	79.54%	86.94%	65.65%	83.07%	74.36%
Average		*57.51%*	*54.48%*	*83.12%*	*82.77%*	*55.32%*	*82.89%*	*69.1%*

Source: The Author

as predicting the positive class. By picking one of them, namely the classifier for the Activities topic, and examining the values it outputs for all instances of our test set when running predictions over them, we noticed that the lowest value output was 0.5082, with the highest value output being 0.6494, and the average across all outputs falling at 0.5716, all rounded to 4 decimal digits. This confirms that, by setting the threshold at 0.5, for this particular classifier all samples in the test dataset would be considered positive, simply because the range that the classifier considered fell completely within 0.50 and 0.65.

Table 5.14 shows the minimum, average and maximum values that were output by the classifier when running predictions on the test set. While it shows that it is not always the case that values are restricted to 0.5 to 0.65, they do tend to be clumped together around this range, only leaving it by small margins. The smallest value output overall was 0.4289, and the highest was 0.7560, which is a significant departure from the range of 0.0 to 1.0 that classifiers are expected to use.

Table 5.14 – Focal Loss min, average and max output values on test set

Topic	**Min**	**Average**	**Max**
Activities	0.5082	0.5716	0.6494
Arts & Sciences	0.4607	0.5831	0.6817
Living	0.5651	0.6230	0.6715
Love	0.4289	0.5651	0.6932
Mythology & Folklore	0.4622	0.5322	0.6572
Nature	0.5173	0.6006	0.7106
Relationships	0.5106	0.5994	0.6790
Religion	0.4712	0.5682	0.7560
Social Commentaries	0.5130	0.5974	0.7175

Source: The Author

Overall, we can see that for the setup we used in this work, while performing threshold moving was helpful yet not essential for all the other classifiers, for those using focal loss it was an absolute necessity to get meaningful results. Without it, by just assuming the naive 0.5 threshold as the only option, this strategy would appear to be innocuous

for this problem.

Table 5.15 shows the comparison between the F1 scores obtained by the models trained with the baseline configuration and those trained using the Focal Loss as the loss function. Here we are using the values presented in table 5.12 for the Focal Loss results, as they were obtained following the same methodology for threshold moving as we used for the other experiments. We can see that the baseline outperforms the results achieved with Focal Loss in almost all cases, except for Mythology & Folklore and Relationships. Its average performance is also better.

Our best interpretation of this underperformance and unstable behavior by the Focal Loss is that it would require a deeper search for values to its hyper-parameters. For this work, as described in the methodology section, we used the values proposed by the original paper, but several things might make these sub-optimal for our case. For instance, the fact that the original paper was developed in the context of computer vision, while we are working with NLP, and the fact that the value proposed for its most impactful hyperparameter, γ, is absolute, instead of being data-dependent.

A comprehensive search for good values to the hyperparameters is beyond the scope of this work. We still leave it as a suggestion in the section describing possibilities of future work, however, as we expect that a carefully-chosen set of values could greatly improve the results achieved in this work.

Table 5.15 – Comparison Baseline vs Focal Loss F1 scores

Topic	**Baseline**	**Focal Loss**
Activities	**68.89%**	63.35%
Arts & Sciences	**69.14%**	67.46%
Living	**64.63%**	48.06%
Love	**74.79%**	71.88%
Mythology & Folklore	45.47%	**62.13%**
Nature	**74.49%**	72.73%
Relationships	67.62%	**68.27%**
Religion	**68.05%**	67.69%
Social Commentaries	**73.69%**	72.97%
Average	***67.42%***	*66.06%*

Source: The Author

5.5 Comparing all experiments

Table 5.16 shows the comparison between the F1 scores obtained by the models for each experiment we performed. From it, we can see that undersampling and the weighted loss function show the best results overall, with the former having a slight advantage. From the 9 topics we used, undersampling performed better for 6, with weighted loss performing better on 3 - also, when we average the scores over all topics, the undersampling method is the only one that reaches a score above 70%, standing at 71.07%, 1.35% higher than when we use class weights. The baseline and the Focal Loss methods were not the best method for any topic.

It is also noteworthy that the topic with the biggest imbalance in the dataset, Mythology & Folklore, shows improvement under all of the tested methods in comparison with the baseline. The Living topic, which has the lowest imbalance, has a stable performance across all methods (including the baseline), except for the Focal Loss. This highlights our perception that the adoption of the Focal Loss requires a more cautious fine-tuning of the value used for its hyperparameters, as it seems to struggle with scenarios that stray even a little from its specific use case.

It is somewhat surprising that undersampling can achieve the best result overall, since in trying to solve one of the issues of this dataset (imbalance), it takes another (small amount of data) and worsen it. We believe that this points not to the fact that undersampling is a superior method, but instead that there is space to improve the performance by tinkering further with the loss function. Not only with the hyperparameters for the Focal Loss function, which we already mentioned, but also for the class weights method. The strategy we used to compute weights has as positive points the fact that a) it is the default solution implemented to this problem on a library that is an industry-standard for machine learning; and b) that it makes intuitive sense. However, these by themselves do not guarantee it is the optimal solution for this particular dataset. Another thing that might have given undersampling an unfair advantage over the other methods is that in our methodology we experimented with 5 different levels of undersampling, while for the other ones we tried only one set of parameters. This gives one extra degree of freedom to this method, although it must be mentioned that it is not a completely free advantage, as taking any level other than the highest imbalance would impact negatively on the amount of data available to learn with, which should reduce the performance. Both these possibilities can be mitigated at the same time by running new experiments exploring further

the adjustments to the loss functions, and we again suggest this as a possibility for future works.

Table 5.16 – Comparison of F1 scores between all experiments

Topic	**Baseline**	**Undersampling**	**Weighted Loss**	**Focal Loss**
Activities	68.89%	**73.67%**	67.94%	63.35%
Arts & Sciences	69.14%	**73.24%**	69.25%	67.46%
Living	64.63%	**65.70%**	64.61%	48.06%
Love	74.79%	74.85%	**75.38%**	71.88%
Mythology & Folklore	45.47%	64.50%	**65.20%**	62.13%
Nature	74.49%	**75.15%**	73.60%	72.73%
Relationships	67.62%	**69.22%**	67.95%	68.27%
Religion	68.05%	**69.28%**	69.03%	67.69%
Social Commentaries	73.69%	73.99%	**74.49%**	72.97%
Average	*67.42%*	***71.07%***	*69.72%*	*66.06%*

Source: The Author

6 FINAL REMARKS

In this work, we applied the BERT pre-trained language model to create text classifiers for a dataset of poems from the Poetry Foundation. We built this dataset by starting from a previously available dataset with poems and their ids, and then proceeding to scrape the organization's website and fetching from it all the metadata that could be interesting for such a work. We then proceeded to analyse the available metadata in order to define what were the most useful lines of approach and, based on the one we found the best, namely to identify the topics based on the textual content of the poems, we selected the samples that would be used.

We identified some of the main issues that the dataset could create to the language model and focused on the issue of imbalance of the dataset. We experimented with three possible strategies to mitigate it: undersampling of the majority class, the use of different weights for each class on the loss function, and the use of the Focal Loss function. We presented the results and compared each of the strategies with a baseline implementation, as well as presenting a general comparison considering all of the strategies implemented.

We found that Focal Loss function was not able to consistently outperform the baseline results within the constraints of our methodology, while at the same time presenting some peculiar behaviors that suggest it might perform better with a more careful choice for the values of its hyperparameters.

We found that both undersampling of the majority class and the use of different weights for each class when computing the loss function were able to outperform the baseline results, strengthening the idea that the class imbalance was a significant source of problems for the classifiers. Counter-intuitively, undersampling was able to perform even better than the weighted loss function. Given the fact that it uses less data, and therefore has less content for the model to learn with and exploring patterns, we would not expect that to be the case. We believe that the reason is the fact that the strategy used to generate the weights for each class, although being a standard for this purpose, was not optimal, suggesting that there is a latent potential for this approach to yield even better results, through deeper experimentation with different ways to generate the weights.

Along the way, we identified some possibilities that fell outside the scope of the current work, but that we can suggest for future ones. The first of them being what was just mentioned, tinkering further with the parameters of the loss functions used. And although we restricted our scope to work with only a couple of them, there are also other

loss functions proposed to deal with imbalance issues that could be explored in future works, one example of which being Dice Loss (LI et al., 2020).

One simple extension of the current work would be to mix the two approaches that outperformed the baseline, by using the best level of undersampling and on top of that scaling the loss function by using class-dependent weights. Another natural extension of this work would be to attempt using ROC and PR curves to find the best thresholds, since we only performed threshold moving, as well as employing other metrics to study the performances of the classifiers.

A decision we made early on for this work was to use binary classifiers for each topic. However, it is also possible to treat this same context with a multi-label approach.

A future work could also be performed by comparing the results obtained with language models and those obtained with more traditional machine learning algorithms. Lou, Inkpen and Tanasescu (2015), for instance, found that SVMs performed well for a similar task, so putting them side by side with a language model might yield interesting insights.

While we explored undersampling of the majority class, it is also possible to attempt the oversampling of the minority class as an alternative which has the benefit to allow for more data to be used. Both approaches were tested together with BERT by Jayaraman et al. (2023) on a dataset composed of websites news articles, as we mentioned in section 3. It remains to be seen if the most common techniques to achieve oversampling, such as simply repeating samples or reproducing samples of the minority class with small changes will work as well for poetry, given the fact that it has a more rigid formal structure than daily language and that the texts are created for aesthetic purposes more than just to communicate. An interesting technique that might fit well with this nature of poetry is LAMBADA, proposed by Anaby-Tavor et al. (2020), in which a generative language model is first trained on the samples of the minority class and then used to generate more samples that are similar to those in order to augment the data available.

Finally, more sophisticated language models can be tested with this dataset. We restricted our scope to BERT Base Uncased as a standard, however BERT itself has more powerful variants, such as BERT Large. The use of the Cased variants also might bring about different results and is an interesting comparison to be made. Moving away from the BERT family, there are bigger models such as RoBERTa (LIU et al., 2019) that have the potential to better handle complex texts such as poems. Another interesting idea is to work with ensembles of these models.

REFERENCES

AHMAD, S. et al. Classification of poetry text into the emotional states using deep learning technique. **IEEE Access**, PP, p. 1–1, 04 2020.

ALI, H. et al. Imbalance class problems in data mining: A review. **Indonesian Journal of Electrical Engineering and Computer Science**, v. 14, 03 2019.

ANABY-TAVOR, A. et al. Do not have enough data? deep learning to the rescue! **Proceedings of the AAAI Conference on Artificial Intelligence**, v. 34, n. 05, p. 7383–7390, Apr. 2020. Available from Internet: <https://ojs.aaai.org/index.php/AAAI/article/view/6233>.

BARBADO, A.; GONZáLEZ, M. D.; CARRERA, D. **Lexico-semantic and affective modelling of Spanish poetry: A semi-supervised learning approach**. 2021. Available from Internet: <https://arxiv.org/abs/2109.04152>.

BOZINOVSKI, S. Reminder of the first paper on transfer learning in neural networks, 1976. **Informatica**, v. 44, 09 2020. Available from Internet: <https://www.informatica.si/index.php/informatica/article/view/2828>.

DEVLIN, J. et al. **BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding**. 2019. Available from Internet: <https://arxiv.org/abs/1810.04805>.

JAYARAMAN, A. K. et al. Imbalanced aspect categorization using bidirectional encoder representation from transformers. **Procedia Computer Science**, v. 218, p. 757–765, 2023. ISSN 1877-0509. International Conference on Machine Learning and Data Engineering. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S187705092300056X>.

KADHIM, A. I. Survey on supervised machine learning techniques for automatic text classification. **Artificial Intelligence Review**, v. 52, n. 1, p. 273–292, Jun 2019. ISSN 1573-7462. Available from Internet: <https://doi.org/10.1007/s10462-018-09677-1>.

LAKSHMI, T. J.; PRASAD, C. S. R. A study on classifying imbalanced datasets. In: **2014 First International Conference on Networks Soft Computing (ICNSC2014)**. [s.n.], 2014. p. 141–145. Available from Internet: <https://ieeexplore.ieee.org/abstract/document/6906652>.

LI, X. et al. **Dice Loss for Data-imbalanced NLP Tasks**. 2020. Available from Internet: <https://arxiv.org/abs/1911.02855>.

LIN, T.-Y. et al. **Focal Loss for Dense Object Detection**. 2018. Available from Internet: <https://arxiv.org/abs/1708.02002>.

LIU, Y. et al. **RoBERTa: A Robustly Optimized BERT Pretraining Approach**. 2019.

LOSHCHILOV, I.; HUTTER, F. **Decoupled Weight Decay Regularization**. 2019. Available from Internet: <https://arxiv.org/abs/1711.05101>.

LOU, A.; INKPEN, D.; TANASESCU, C. Multilabel subject-based classification of poetry. **Nature**, v. 2218, p. 30–7, 2015. Available from Internet: <https://www.site.uottawa.ca/~diana/publications/flairs_2015_paper.pdf>.

MADABUSHI, H. T.; KOCHKINA, E.; CASTELLE, M. **Cost-Sensitive BERT for Generalisable Sentence Classification with Imbalanced Data**. 2020. Available from Internet: <https://arxiv.org/abs/2003.11563>.

NGUYEN, N.; DUONG, A. Comparison of two main approaches for handling imbalanced data in churn prediction problem. **Journal of Advances in Information Technology**, v. 12, p. 29–35, 01 2021.

QIU, X. et al. Pre-trained models for natural language processing: A survey. **Science China Technological Sciences**, v. 63, n. 10, p. 1872–1897, Oct 2020. ISSN 1869-1900. Available from Internet: <https://doi.org/10.1007/s11431-020-1647-3>.

RUMA, J. F. et al. A deep learning classification model for persian hafez poetry based on the poet's era. **Decision Analytics Journal**, v. 4, p. 100111, 2022. ISSN 2772-6622. Available from Internet: <https://www.sciencedirect.com/science/article/pii/S2772662222000479>.

RUSSELL, S.; NORVIG, P. **Artificial Intelligence: A Modern Approach (4th Edition)**. Pearson, 2020. ISBN 9780134610993. Available from Internet: <http://aima.cs.berkeley.edu/>.

VASWANI, A. et al. **Attention Is All You Need**. 2017. Available from Internet: <https://arxiv.org/abs/1706.03762>.

ZHU, J.; WU, H.; YASEEN, A. Sensitivity analysis of a bert-based scholarly recommendation system. **The International FLAIRS Conference Proceedings**, v. 35, May 2022. Available from Internet: <https://journals.flvc.org/FLAIRS/article/view/130595>.