Handling Class Imbalance when Detecting Dataset Mentions with Pre-trained Language Models

Yousef Younes, Brigitte Mathiak
GESIS – Leibniz-Institute for the Social Sciences
Cologne, Germany
{yousef.younes,brigitte.mathiak}@gesis.org

Abstract

Understanding the links between datasets and publications is a hard task. Yet it is very important to improve dataset discovery and FAIRness of research data. However, only a few datasets with such high-quality links exist. Human annotations are the gold standard for producing such links, but it is a time-consuming manual task, much of which is spent reading text that is not connected to dataset mentions at all. In this paper, we propose a filter to pre-screen scientific publications' sections, so that we can find candidates for dataset mentions reliably. For this, we test both BERT and RoBERTa on sections content as well as on sections title. The main challenge is the inherent imbalance in the data, which we tackle using different imbalance handling techniques, such as re-sampling and variations of the loss function. The best result was obtained when using RoBERTa on section contents by combining re-sampling, balanced focal loss, and a recall-biased validation metric to get a fairly high recall and acceptable precision. The source code and the best obtained model are available here [1].

1 Introduction

Manual textual annotation is a very cumbersome and expensive process, yet it acts as the gold standards that is used for training computational learning models (Pustejovsky and Stubbs, 2012). Unfortunately, this importance turned to be a bottleneck for NLP because the annotation is a time-consuming, laborious, yet error-prone process. To alleviate these difficulties, a lot of tools have been introduced to pre-process the data in order to reduce the amount of human effort needed and to make that effort more focused on the essential task that can not be automated (Pan and Yang, 2009). Detecting research dataset mentions in text (Fan et al., 2022) is an example of an NLP task that needs manually annotated data to train a model.

Research datasets play a crucial role in science. They act as a unifying point that allows comparison of different research ideas and also facilitate reproducibility of results (Wilkinson et al., 2016). For these reasons, the problem of finding datasets mention in research papers has gotten more attention recently (Zhao et al., 2018). To tackle this problem, researchers like in (Färber et al., 2021)(Mesbah et al., 2018) treat it as a domain-specific, supervised NER task that needs annotated dataset to operate on. The Coleridge dataset [2] used in Kaggle's "Show US the Data" competition is an example of such dataset in which research papers are annotated with the datasets mentioned in them. One of the challenges that annotators face when doing such annotation is the sparsity of these mentions in scientific text. By sparsity, we mean that most of the texts do not have dataset mentions and only a small amount contains such mentions. This sparsity has negative consequences not only on the annotation process but also on the annotation result. During annotation, the annotators spend a lot of time reading texts that do not contain dataset mentions. Naturally, the resulting annotated dataset is imbalanced towards the samples without mentions.

One way to help annotators is to perform blocking (Azzalini et al., 2021) in order to pick the texts that contain dataset mentions so they can focus on them. To achieve this, we define a binary classification task in which we classify text as belonging to the negative class (N) when it does not have dataset mention and to the positive class (P) when it has. For example, the following excerpt "...classification problem on a subset of *ADNI database* consisting

[1] https://github.com/YousefYounes15/dataset_mention_detection

[2] https://www.kaggle.com/c/coleridgeinitiative-show-us-the-data/data

of 531 subjects with sMRI and DTI scans..." makes the paper's section that contains it belongs to the P class since ANDI is an abbreviation for a dataset. From here onward, N and P will be used to refer to the negative and positive classes respectively.

Our goal is to get a binary classifier with the highest possible P-recall and high precision as well. Such classifier will help reduce the time and effort needed for the manual annotation. To achieve this goal we chose to work with language models assuming that the context plays a vital role for the task at hand. Since our data is imbalanced, we used the chance to compare the performance of two popular models: BERT (Kenton and Toutanova, 2019) and RoBERTa (Liu et al., 2019) on such data. To the best of our knowledge, there is no such comparison in the literature. We have experimented on different parts of sections content and also on the sections title to find out which settings work better. We also have used different imbalance remedy techniques like re-sampling and cost-sensitive learning i.e., balanced focal loss.

The contributions of this paper can be outlined in the following points:

- Re-sampling has more effect than cost-sensitive learning on language model-based classifiers.
- We can not depend on sections title to perform this classification using language models, albeit dataset mentions are more present under particular section titles.
- RoBERTa outperforms BERT even when the data is imbalanced.
- The best results with a P-recall of 86% were achieved with RoBERTa when combining re-sampling, cost-sensitive learning, and a recall-oriented validation metric.

The rest of this paper is organized as follows. Section 2 reviews how text classification was improved by word embeddings before it summarizes bias handling techniques. Section 3 describes the data that we used in the paper before it gives details on the different loss functions used in the experiments. Section 4 describes the experimental settings then it reports the results of the different experiments. After that, the results are discussed in section 5. Finally, the paper concludes in section 6.

2 Related Work

2.1 Text Classification with Deep Learning

Text classification is a supervised machine learning task in which a given piece of text is predicted to belong to a class of a set of predefined classes (Vijayan et al., 2017). To perform this task, many traditional algorithms such as Naive Bayes, Support Vector Machines (SVM), Tree-based models (Kowsari et al., 2019); and modern neural-based algorithms such as RNN, CNN, DBN have been used (Minaee et al., 2021). All of these models share a common problem to which each has devised a different solution which is text representation.

Transfer learning is a learning framework that revolutionized the field of machine learning by breaking the isolation of both tasks and models (Pan and Yang, 2009). It enables a model which is trained on one task to transfer the knowledge it gained to another related task via fine-tuning (Torrey and Shavlik, 2010). As a sub-field of machine learning, NLP witnessed a huge leap due to transfer learning thanks to the introduction of pre-trained word embeddings.

Word embeddings are one of the most successful examples of transfer learning because they are learned in one task and used to solve other tasks. Earlier word embeddings such as word2vec (Mikolov et al., 2013a)(Mikolov et al., 2013b) and GloVe (Pennington et al., 2014) represent a word by a high-dimensional vector. The similarities between words are reflected as similarities between their vector representations. These embeddings are usually used to initialize new models but the whole model needs to be re-trained from scratch on the new data. That is why they are known as shallow transfer learning. Contextual embeddings are an improvement over word embeddings in which the context of the word is involved in its representation (Liu et al., 2020). Pretrained language models such as ELMo (Peters et al., 2018), GPT-n (Radford et al., 2019)(Brown et al., 2020), BERT (Kenton and Toutanova, 2019), RoBERTa (Liu et al., 2019) are used to obtain this type of representation. Unlike word embeddings, these models are usually used as part of new models which are fine-tuned on the data for a particular task. This significantly reduces the need for huge training data thus reducing the computational cost.

The great success achieved by language models clearly indicates the vital role played by context in natural language processing. Dataset mentions in the scientific text are not an exception. Although there is no standard method to mention a dataset in a research paper, we assume that there is some kind of unwritten rules that govern the way researchers mention a dataset. These rules have something to do with the context in which the dataset title or label is mentioned. For this reason, we make use of language models to solve our problem. More particularly, the focus is on BERT and RoBERTa since they provide a contextual embedding that proved to be successful for many NLP tasks. In addition, we take the chance to compare the performance of these two models on imbalanced data.

The classifiers used in the experiments are constructed by adding one linear layer on top of the base version of BERT and RoBERTa. So in the rest of this paper, we will be using BERT and RoBERTa as shortcuts to indicate the classifiers based on them. While this is a very straightforward way to turn these into classifiers, due to the inherent limitation of the language model, they can only handle a limited input of up to 512 tokens at a time. One way to address this issue, which was introduced in (Sun et al., 2019), is to select a representative set of tokens from the document that results in the best quality for the classifier. One of our intentions is to use and extend this work for imbalanced data. Another approach is to get a proper document representation by dividing the text into equally-sized chunks then using word-level encoder and pooling followed by chunk-level encoder and pooling as illustrated in (Su et al., 2021).

2.2 Handling Class Imbalance

Class imbalance is a feature that characterizes many labeled datasets. This imbalance has a direct effect on the classifier so many techniques have been proposed to deal with it. These techniques fall under two categories: re-sampling and cost-sensitive learning (Iikura et al., 2020).

Re-sampling methods try to address the problem by duplicating samples of minority classes, removing samples of majority classes or generating new samples of minority classes. These methods have been used successfully for text classification e.g., (Estabrooks et al., 2004)(Tepper et al., 2020). But the removal and duplication of samples change the data distribution and can cause the models to overfit the minority classes. For this reason, approaches like in (Chawla et al., 2002)(Guo and Viktor, 2004) have been introduced to produce synthetic data. In this work, we will use sample removal and duplication and leave sample generation for future work because text generation is out of the scope of this paper.

Cost-sensitive learning methods approach the problem by injecting data bias in the cost function also known as the loss function. The loss function condenses the whole learning system into a single number called the loss whose value must be minimized in order to improve the performance of the model. Cost-sensitive learning tailors the loss in favor of a particular class(es) by multiplying it by a weight value. One way to choose that weight is to use the inverse class frequency like in Balanced Cross Entropy (BCE) (Phan and Yamamoto, 2020). Another way is to use the difficulty of the sample as a weight like in Focal Loss (FL) (Lin et al., 2017).

To the best of our knowledge, a comparison of the effect of imbalance handling techniques on BERT- and RoBERTa-based classifiers was not performed. In this work, we investigate empirically the effect of using re-sampling and cost-sensitive learning on such classifiers.

3 Methodology

This section describes the data and loss functions that are used in this study. It starts by explaining how the data is prepared. Then an overview of the loss functions is presented.

3.1 Data

The experiments in this paper use the Coleridge dataset. This dataset consists of a collection of research papers ($\sim$19.6 K) out of which 14.3K papers are unique and 5.3K papers are duplicated due to the fact that they have multiple dataset mentions. Each of these papers is stored in a JSON-file that wraps each section's title and content in a JSON object. In addition to the JSON files, there is a CSV-file that contains basic metadata (file name, paper title, dataset title, dataset label, cleaned dataset label) about each paper. To prepare the data for our experiments, it was processed as follows: scan through the sections of each paper to extract the title and text. Then a binary label is generated for every section. This label contains (1) if a dataset associated with the paper is mentioned in the section either using its title or its label, otherwise it contains (0). The final dataset that we got contains approximately 233K samples. Each sample consists of the following pieces of information:

(file name, paper name, section title, section text, dataset mention, label). The sections are considered documents whose contents and titles are used in the experiments. Using sections as our basic text units is a compromise between using all of the paper, which would be trivial, and sentence level, which often does not have enough information to make the decision and introduce even more bias. Sections can be easily identified, even when they are not part of the metadata of the publication (Mathiak et al., 2009).

Before we run experiments on this dataset, we divided it randomly into two parts: train and test set. The test set accounts for 20% of the dataset ($\sim$ 47K samples) and is used to test the classifier after training is complete. This testing set is the same for all experiments. The remaining 80% ($\sim$ 187K samples) is kept for training and validation sets. During training, 80% of this data is selected randomly to be used for training and the remaining are used for validation. Since the splitting is done randomly, different sections from the same paper could appear in train, val and test sets. In all operations involving randomization, a seed is used to ensure reproducibility.

Inspecting the data, we have found that among the existing 233297 samples only 27856 samples contain dataset mentions i.e., they belong to the P class. We also have found that the majority of the sections ($\sim$200K) contain long text. In summary, only 12% of the samples in the data belong to the P class. Besides that 85% of the samples contain long text that will represent a challenge to the models used in this work.

3.2 Loss Functions

Focal Loss (FL) was introduced in (Lin et al., 2017) to handle the bias of the object detection problem in computer vision. In this study, we apply it to a binary text classification along with three other functions, which are also variations of cross entropy (Murphy, 2012). They can be generalized in one formula, Eq.1, from which different loss functions can be derived by assigning different values to the parameters.

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

Equation 1 can be split into three components. The main component is the binary cross entropy $-log(p_t)$ where p_t holds either the model's estimated probability ($p \in [0, 1]$) for class P or its complement $(1 - p)$ for class N. Another part is the balancing factor α_t which, when it is activated, takes a different positive non-zero value for each class to reflect its weight. But α_t could be deactivated by giving it the same value for all classes. In the context of this paper, we will write $\alpha_t = (x, y)$ to indicate that x is the weight for class N and y is the weight for class P. Since we have only two classes, we will set $\alpha_t = (1, 1)$ when the class's weight is not considered. Otherwise, α_t will be assigned values that can be chosen by the user or set by some criteria like the inverse class frequency (Phan and Yamamoto, 2020). The third part is the modulating factor $(1 - p_t)^\gamma$ where $\gamma >= 0$ is the focusing parameter. This modulating factor adjusts the contribution of the sample in the loss based on its difficulty. For difficult samples about which the classifier is not certain, p_t will have low values which will increase the modulating factor resulting in more contribution of these samples in the loss. While for easy samples, p_t will have high values and the modulating factor approaches zero thus reducing the contribution of these samples in the loss. Now let us derive the loss functions.

Cross entropy is used as the loss function in the BERT- and Roberta-based classifiers. It takes neither the class's weight nor the samples' difficulty into account. It can be derived from Eq.1 by setting $\alpha_t = (1, 1)$ and γ= 0. Balanced cross entropy considers the class imbalance but not the difficulty of the samples. To derive it from Eq.1, we set $\gamma = 0$ and α_t could take different values. We will use the classes' percentages in the data $\alpha_t = (0.12, 0.88)$ beside other values in the experiments (cf. section 4.4). Focal loss pays attention to the difficulty of the sample ($\gamma > 0$) but not on the weight of the classes ($\alpha_t = (1, 1)$). Finally Balanced Focal Loss (BFL) brings everything together so $\alpha_i > 0$ for $i = 0, 1$ and $\gamma > 0$.

4 Results

In this section, we start by describing the experimental settings. After that, the results of different experiments are presented.

Our goal is to find the best model for selecting texts with dataset mentions. The major challenges in this are the length of the texts and the inherent imbalance in the data (cf. section 3.1). We start by experimenting with the section contents to find the text features that give the best results. Then we use these features to select the best sampling rates and loss function settings. After that, we combine all of these results together. Additionally, we experiment with the section titles using different configurations. Finally, we try to improve the results by changing the validation metric.

4.1 Experimental settings

All experiments were implemented in Pytorch and ran on a 4-GPU machine with a GeForce RTX 2080 Ti with 11 GB RAM each. The classifiers were trained for 4 epochs using the AdamW optimizer (Loshchilov and Hutter, 2019) with a learning rate of 2x10-5 to overcome the catastrophic forgetting problem following the recommendation of (Sun et al., 2019). Each epoch consists of one full scan of the training set to compute the loss followed by one pass through the validation set then the accuracy is used as the validation metric to measure the classifier progress. Different loss functions described in section 3.2 will be used. To make full use of GPU memory, the batch size was chosen to be 32 for experiments conducted on section contents and 256 for the ones on section titles (cf. section 4.6). We have made use of the same seed for all operations that involve randomization to make the experiments repeatable. In some experiments, we do diverge from these settings by changing the validation metric which will be clearly stated in the corresponding section.

To make sure that our results are reliable, we adopt the following strategy. We start by running a set of initial experiments to help us find settings that produce the best results. If these settings were in line with results obtained from previous research we use them; otherwise, we run experiments using 5-fold cross-validation to verify our results and use settings that produce the best average result on the test set. In all experiments, we use the test data to report the values of precision, recall and F1-score for both the P and N classes. We also report the accuracy (ACC) and Matthew Correlation Coefficient (MCC) metrics. Since the model is intended to filter samples of the P class, we are mainly interested in improving the P-recall. So we will be comparing models based on their P-recall but in case of a tie, other metrics will be used.

4.2 Feature Selection

In this subsection, we investigate which of the text's features serve our goal the best. Particularly, we are interested in token selection, as we try to keep within the 510 token limit imposed by BERT.

The sections content are used as input for the classifier. In this context, every section is considered a document by itself (cf. section 3.1). BERT accepts a maximum input of 512 tokens (cf. section 2.1), yet most sections are longer than this (cf. section 3.1). We also know that the choice of input tokens for BERT has an impact on the quality of the classifier (Sun et al., 2019). As such, we want to find out which part of the text achieves the best classification results. Working on balanced data, it was found in (Sun et al., 2019) that picking three-quarters of BERT's input tokens from the beginning of the document and one-quarter from the end of the document results in the best classification outcome. Since our data is imbalanced, we want to examine whether that imbalance has an impact on the choice of the tokens. To do so, we run several experiments using different parts of the sections. The results reported in Table 1 show that the tokens (F 382+L 128) produced the best result (P-recall = 0.79 and MCC = 0.835) for this dataset. This complies with the result obtained in (Sun et al., 2019) which implies that the imbalance nature of the data has no impact on the choice of the input tokens for the task at hand. Based on that, we consider BERT with these input settings as our base model. The experiments to follow will use and build upon these settings.

4.3 Handling Imbalance by Under- and Oversampling

As mentioned in the data description, most of the text does not contain dataset mentions and only 12% of the sections belong to the P class. One way to overcome this imbalance is to use re-sampling. Here we will over-sample the P samples, downsample the N ones and combine the two together to find the sampling ratio that gives the best result. Building on the results from the previous experiments, we have used F 382+ L 128 tokens of sections texts as input to BERT. We have run several experiments with different re-sampling settings to find that the best re-sampling factor for the dataset at hand is (4:0.55) which is to quadruple the P samples and take more than half of the N samples. The corresponding approximated ratio for this sampling factor obtained by dividing the number of positive samples by the number of negative samples is (1) indicating that the numbers of P and N samples are roughly the same. Putting it differently, the model works best when the dataset is balanced. To get a

Tokens	**Precision**		**Recall**		**F1-Score**		**ACC**	**MCC**
	N	P	N	P	N	P		
F 510	0.97	0.92	0.99	0.78	0.98	0.84	0.97	0.827
L 510	0.97	0.92	0.99	0.74	0.98	0.82	0.96	0.808
F 255 + L 255	0.97	0.92	0.99	0.79	0.98	0.85	0.97	0.834
F 128 + L 382	0.97	0.92	0.99	0.79	0.98	0.85	0.97	0.834
F 382 + L 128	0.98	0.93	0.99	**0.79**	0.98	0.85	0.97	**0.835**

Table 1: Feature Selection for Classifying Section Excerpts with BERT. In the "Tokens" column, F stands for First and L for Last. F 510 means the first 510 tokens. L 510 means the last 510 tokens.

reliable result, we ran an experiment using 5-fold cross-validation using a BERT-based classifier and reported the average testing result in Table 2. Although the accuracy is 95%, the P-recall is only 83%. This is the highest P-recall value that could be obtained using re-sampling as a technique to surpass the imbalance problem.

4.4 Imbalance Handling with Loss Functions

Loss-sensitive learning is another known technique to handle Imbalance. Experiments ran so far used cross entropy to compute the loss. Here we ran several experiments with different loss functions via changing the values of α_t, γ in Eq. 1 as explained in section 3.2. The goal is to find the best loss function settings for our task. To achieve this goal, we experiment not only with BERT but also with RoBERTa. Because, unlike text features and sampling factor which are data-related issues, the loss function is part of the model so we want to compare its effect on the two models.

From the initial experiments in which different values for α_t and γ were utilized, we found that when using the focal loss i.e., (α_t = (1,1) and $\gamma = 2$), RoBERTa outperformed BERT with respect to P-recall. Moreover, with balanced cross entropy ($\alpha_t = (.12, .88)$ and $\gamma = 0$) both models produce similar accuracy and P-recall. The best results for both models were obtained when BFL (α_t = (.12,.88), γ=4) is used, with BFL the P-recall value was 77% and 78% for BERT and RoBERTa respectively. Using higher values for γ i.e., $\gamma > 4$, the P-recall started to decline for both models. We also tried to improve RoBERTa results using different values for α_t but we were not able to get better results. To make sure that the best results that we obtained using BFL (α_t = (.12,.88), γ=4) are stable and not due to some randomness, We ran an experiment using 5-fold cross-validation and reported the average of the results obtained on the testing data in Table 3. These results show that RoBERTa is outperforming BERT with 1% increase in P-recall. The next step is to combine the best settings found so far in one experiment.

4.5 Combining Best Settings

In previous experiments, we have seen the effect of using re-sampling and BFL individually. In this subsection, we will combine all the results that we obtained so far. That is to use 382 tokens from the beginning of the section and 128 tokens from the end as the model's input. Furthermore use the sampling factor (4:0.55) that balanced the dataset i.e., ratio=1. In addition, use balanced focal loss with $\alpha_t = (.12, .88)$ and $\gamma = 4$. We have run an experiment using 5-fold cross-validation for both BERT and RoBERTa with these settings reported the average of all metrics in Table 4. The results for RoBERTa and BERT are competitive in general but RoBERTa achieved 3% increase in P-recall over BERT with these settings.

4.6 Classifying Using Section Titles

Working on the training data, we have noticed that there is a tendency among researchers to mention datasets under sections with specific titles such as Introduction, Methodology, Discussion, Data, Datasets, Results, and Experiments among others. Based on that, we had the hypothesis that we can depend on the section titles to decide whether a section contains dataset mentions or not. To test this hypothesis, we experimented using four configurations (data as is, data with sampling, BFL, BFL with sampling) on both models. The best P-recall (68%) was obtained by RoBERTa when using BFL with sampling. But lower values of other measures such as P-precision = 0.22 and MCC=0.244 indicate that using the titles with these configurations to do the classification task at hand is not promising.

Precision		**Recall**		**F1-Score**		**ACC**	**MCC**
N	P	N	P	N	P		
0.98	0.80	0.97	**0.83**	0.97	0.81	0.95	0.789

Table 2: Average Testing Results using BERT when Data is Balanced using Re-sampling

Model	**Precision**		**Recall**		**F1-Score**		**ACC**	**MCC**
	N	P	N	P	N	P		
BERT	0.97	0.93	0.99	0.76	0.98	0.84	0.97	0.821
RoBERTa	0.97	0.93	0.99	**0.77**	0.98	0.85	0.97	0.830

Table 3: Average Testing Results Using Balanced Focal Loss with $\alpha_t = (.12, .88)$ and $\gamma = 4$

4.7 Experimenting With Validation Metrics

So far the best-known imbalance handling techniques were used individually and combined but we were not able to achieve a P-recall above 83%. Here we will add a new idea: that is to select the validation metric used on the validation set to be in line with our goal. Since our goal is to have a classifier that is able to find the overwhelming majority of the P samples, recall represents an exact match for this goal so we will use it as the validation metric. We ran an experiment using 5-fold cross-validation for each model on both section contents and titles. The average results of the experiments are documented in Table 5. Again, RoBERTa outperformed BERT on sections content with a P-recall of 85%. In other words, using recall as a validation metric resulted in 2% improvement on RoBERTa's P-recall reported in Table 4. Similarly, for the experiment that used the titles, the best P-recall obtained by RoBERTa was 70% which means a 2% improvement compared to the result from section 4.6.

To further improve the results, we have weighted the recall three times more than precision by using F-Beta score as the validation metric. With this, we ran the experiments and reported the results in Table 6. This modification improved the P-recall for RoBERTa on sections content by 1% with 5% decrease on P-precision compared to Table 5. So RoBERTa-based classifier on the sections content with Fbeta is the final result of this paper.

5 Discussion

The re-sampling experiments in section 4.3 show that there are limits for oversampling the P-samples and under-sampling the N-samples after which the performance of the classifier starts to decline. These limits correspond to having the same number of P- and N-samples i.e., achieving perfect balance, which is the expected outcome. The average classifier performance obtained when re-sampling balances the dataset is 83% P-recall and 80% P-precision see Table 2. This is better than the original results (79% P-recall) see table 1, but also encourages us to find even better ways to address the issue.

The experiments with loss functions in section 4.4 show that loss functions have more effect on RoBERTa than on BERT. As a side effect, we find that RoBERTa is better than BERT when dealing with imbalanced data. Nevertheless, using re-sampling with BERT produced better P-recall compared to using BFL with both BERT and RoBERTa. So we can conclude that re-sampling is actually more effective than using BFL in this particular use case. However, as seen in the experiments in section 4.5 combining BFL and re-sampling together improves the overall performance. RoBERTa outperformed BERT with 3% increase on P-recall see Table 4. That means when dealing with balanced data, the impact of the used loss function on RoBERTa is more than on BERT.

Starting from section 4.6, the experiments involved section titles to do the classification task. In these experiments, RoBERTa produced better P-recall than BERT. But the best P-recall value obtained was 70% is 16% lower than P-recall obtained on section contents. This is not surprising due to the lower amount of linguistic knowledge contained in the section titles compared to the section contents. This is in contrast to the findings in (Galke et al., 2017), where they have very promising results with only the titles, but also a very different classification task not related to dataset mentions. In section 4.7, the experimental results show that changing the validation metric has a good impact on the results of both models. When weighing

Model	**Precision**		**Recall**		**F1-Score**		**ACC**	**MCC**
	N	P	N	P	N	P		
BERT	0.97	0.81	0.97	0.8	0.97	0.80	0.95	0.779
RoBERTa	0.98	0.75	0.96	**0.83**	0.97	0.78	0.95	0.757

Table 4: Average Testing results using a combination of re-sampling and Balanced Focal Loss.

Setting	Model	**Precision**		**Recall**		**F1-Score**		**ACC**	**MCC**
		N	P	N	P	N	P		
BFL	BERT	0.97	0.90	0.99	0.77	0.98	0.83	0.96	0.815
	RoBERTa	0.97	0.91	0.99	0.78	0.98	0.84	0.97	0.820
BFL+S+C	BERT	0.97	0.75	0.96	0.82	0.97	0.78	0.94	0.749
	RoBERTa	0.98	**0.68**	0.94	**0.85**	0.96	0.75	0.93	0.721
BFL+S+T	BERT	0.94	0.22	0.70	0.68	0.80	0.34	0.70	0.252
	RoBERTa	0.94	0.21	0.66	0.70	0.78	0.32	0.66	0.232

Table 5: Average Testing Results Using Re-sampling and Balanced Focal Loss with Recall as Validation Metric. BFL stands for Balanced Focal Loss, C for Content, T for Titles, and S for Sampling.

Setting	**Model**	**Precision**		**Recall**		**F1-Score**		**ACC**	**MCC**
		N	P	N	P	N	P		
BFL+ S + C	BERT	0.98	0.73	0.96	0.81	0.97	0.77	0.94	0.741
	RoBERTa	0.98	**0.63**	0.93	**0.86**	0.95	0.72	0.92	0.692
BFL+ S + T	BERT	0.94	0.23	0.70	0.67	0.81	0.34	0.70	0.255
	RoBERTa	0.94	0.23	0.72	0.63	0.81	0.33	0.71	0.238

Table 6: Average Testing Results Using a Combination of Best Settings with F-beta as Validation Metric

the recall three times more than the precision using the F-beta score, we were able to obtain a classifier with 86% P-recall and 63% P-precision using RoBERTa. In other words, using such a filter will reduce the annotation time by more than 50% at the price of losing 14% of the positive samples.

We went further to investigate why the model was not able to pick the 14% P-samples by looking at some false negative cases which have dataset mentions but the model classified them as not having. We have found that the model has difficulty when the dataset is mentioned indirectly like in ”...Empirically, Schwellnus and Arnold (2008) uses data from OECD’s firms to show that increases in corporate taxes negatively impact firms...”. Here the source of the data is used but the actual name of the dataset is not used. In another case, the distance between the name of the dataset and the word that refers to it was quite long like in ”...*Studies* that have collected longitudinal data tend to be based on relatively small samples, whereas the *E CL S - B* provided a large, nationally representative sample a...”. Finally, we noticed that the dataset acronyms represent a challenge for the tokenizer.

6 Conclusion and Future Work

The re-sampling experiments in section 4.3 show that the models work best when the data is balanced. The experiments with loss functions in section 4.4 show that, with language models, using re-sampling to handle imbalance is more effective than using loss functions. In general, RoBERTa kept outperforming BERT when using imbalance handling techniques. The set-up with which RoBERTa classifier reached the highest P-recall (86%) was to use a balanced dataset of section contents with a proper loss function and a validation metric that is compatible with the task’s goal. Particularly, we used F-beta as the validation metric with recall weighted three times more than precision. We are planning to test this filtering algorithm with real-life annotators next. The idea is to provide them with candidates for extracts from full-text publications that allow them to efficiently find dataset mentions and annotate them. In the long run, we are interested in not only the dataset mentions themselves but also to enrich the mentions with additional metadata, such as direct links to the datasets.

References

Fabio Azzalini, Songle Jin, Marco Renzi, and Letizia Tanca. 2021. Blocking techniques for entity linkage: A semantics-based approach. *Data Science and Engineering*, 6(1):20–38.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and Amanda Askell. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. 2004. A Multiple Resampling Method for Learning from Imbalanced Data Sets. *Computational Intelligence*, 20(1):18–36.

Lizhou Fan, Sara Lafia, David Bleckley, Elizabeth Moss, Andrea Thomer, and Libby Hemphill. 2022. Librarian-in-the-loop: A natural language processing paradigm for detecting informal mentions of research data in academic literature. *CoRR*, abs/2203.05112.

Michael Färber, Alexander Albers, and Felix Schüber. 2021. Identifying used methods and datasets in scientific publications. In *Proceedings of the Workshop on Scientific Document Understanding: co-located with 35th AAAI Conference on Artificial Inteligence (AAAI 2021) ; Remote, February 9, 2021. Ed.: A. P. B. Veyseh*, volume 2831 of *CEUR Workshop Proceedings*. RWTH Aachen. ISSN: 1613-0073.

Lukas Galke, Florian Mai, Alan Schelten, Dennis Brunsch, and Ansgar Scherp. 2017. Using titles vs. full-text as source for automated semantic document annotation. In *Proceedings of the Knowledge Capture Conference*, pages 1–4.

Hongyu Guo and Herna L. Viktor. 2004. Learning from imbalanced data sets with boosting and data generation: the DataBoost-IM approach. *ACM SIGKDD Explorations Newsletter*, 6(1):30–39.

Riku Iikura, Makoto Okada, and Naoki Mori. 2020. Improving BERT with Focal Loss for Paragraph Segmentation of Novels. In *International Symposium on Distributed Computing and Artificial Intelligence*, pages 21–30. Springer.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. Text classification algorithms: A survey. *Information*, 10(4):150. Publisher: Multidisciplinary Digital Publishing Institute.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Qi Liu, Matt J. Kusner, and Phil Blunsom. 2020. A Survey on Contextual Embeddings. *arXiv:2003.07278 [cs]*. ArXiv: 2003.07278.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *arXiv:1711.05101 [cs, math]*. ArXiv: 1711.05101.

Brigitte Mathiak, Andreas Kupfer, and Silke Eckstein. 2009. Using Layout Data for the Analysis of Scientific Literature. In Djamel A. Zighed, Shusaku Tsumoto, Zbigniew W. Ras, and Hakim Hacid, editors, *Mining Complex Data*, volume 165, pages 3–22. Springer Berlin Heidelberg, Berlin, Heidelberg. ISSN: 1860-949X, 1860-9503 Series Title: Studies in Computational Intelligence.

Sepideh Mesbah, Christoph Lofi, Manuel Valle Torre, Alessandro Bozzon, and Geert-Jan Houben. 2018. Tse-ner: An iterative approach for long-tail entity extraction in scientific publications. In *International Semantic Web Conference*, pages 127–143. Springer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep learning–based text classification: a comprehensive review. *ACM Computing Surveys (CSUR)*, 54(3):1–40.

Kevin P. Murphy. 2012. *Machine learning: a probabilistic perspective*. MIT press.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359. Publisher: IEEE.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv:1802.05365 [cs]*. ArXiv: 1802.05365.

Trong Huy Phan and Kazuma Yamamoto. 2020. Resolving Class Imbalance in Object Detection with Weighted Cross Entropy Losses. *arXiv:2006.01413 [cs]*. ArXiv: 2006.01413.

James Pustejovsky and Amber Stubbs. 2012. *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*. " O'Reilly Media, Inc.".

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Xin Su, Timothy Miller, Xiyu Ding, Majid Afshar, and Dmitriy Dligach. 2021. Classifying Long Clinical Documents with Pre-trained Transformers. *arXiv:2105.06752 [cs]*. ArXiv: 2105.06752.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Naama Tepper, Esther Goldbraich, Naama Zwerdling, George Kour, Ateret Anaby Tavor, and Boaz Carmeli. 2020. Balancing via Generation for Multi-Class Text Classification Improvement. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1440–1452, Online. Association for Computational Linguistics.

Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI global.

Vikas K Vijayan, K. R. Bindu, and Latha Parameswaran. 2017. A comprehensive study of text classification algorithms. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1109–1113.

Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, and Philip E. Bourne. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9. Publisher: Nature Publishing Group.

Mengnan Zhao, Erjia Yan, and Kai Li. 2018. Data set mentions and citations: A content analysis of full-text publications. *Journal of the Association for Information Science and Technology*, 69(1):32–46.