VILNIUS UNIVERSITY

FACULTY OF MATHEMATICS AND INFORMATICS

COMPUTER MODELING STUDY PROGRAMME

Master’s thesis

Comparative analysis of deep learning models when classifying text with a significant class imbalance

Giliojo mokymosi modelių lyginamoji analizė, sprendžiant teksto klasifikavimo uždavinį, esant dideliam klasių disbalansui

Karolis Kvedaravičius

Supervisor : Prof. dr. Olga Kurasova

Vilnius
2026

Acknowledgements

The author is thankful the Information Technology Research Center, Faculty of Mathematics and Informatics, Vilnius University, for providing the High-Performance Computing (HPC) resources for this research.

Summary

In this work, a comparative analysis of different deep learning models and methods that fight class imbalance is performed to identify the best performing models/methods in a variety of imbalanced text classification tasks. Various different papers, which describe and try to tackle the class imbalance problem in deep learning classification tasks with different methods are analyzed. An examination of existing overviews on the methods which tackle class imbalance in deep learning was performed. Several shortcomings are identified in that often it is not tested how the method performance transfer between different models and data sets. During this work 45 different experiments were performed when training deep learning models in a binary text classification task with different model/data set/method combinations. The results show the methods which perform with one model/data set combination often do not perform the same with the other. Several possible explanations are discussed down to model architecture and data set differences. The best method which fights class imbalance was found to be synthetic minority class sample generation using a LLM (Large Language Model). It provided the largest gains of classification metrics and improved those metrics in the largest number of model/data set combinations.

Keywords: Deep learning, text classification, class imbalance, text generation, BERT

Santrauka

Šiame magistro baigiamajame darbe buvo atlikta skirtingų giliojo mokymosi modelių ir metodų, siekiančių kovoti su klasių disbalansu, lyginamoji analizė siekiant nustatyti modelius/metodus, kurie pasiekia geriausius rezultatus įvairiuose teksto klasifikavimo uždaviniuose su aukštu klasių disbalansu. Buvo analizuoti skirtingi moksliniai straipsniai, kuriuose aprašyti bandymai kovoti su klasių disbalansu giliojo mokymosi klasifikavimo uždaviniuose pasitelkiant skirtingus metodus. Buvo aptartos egzistuojančios metodų skirtų kovoti su klasių disbalansu apžvalgos. Keli trūkumai buvai identifikuoti, dažnai šiose apžvalgose testuojamas mažas skaičius skirtingų duomenų aibių ir dažniausiai tik vienas giliojo mokymosi modelis. Neanalizuojama kaip ir kodėl metodai skirtingai veikia su skirtingomis duomenų aibėmis ir modeliais. Šiame darbe buvo atlikti 45 skirtingi eksperimentai mokant giliojo mokymosi modelius binariniame teksto klasifikavimo uždavinyje su skirtingomis modelių/duomenų aibių/metodų kombinacijomis. Iš rezultatų galima matyti kad jei vienas metodas pasiekia gerus rezultatus su vienu modeliu/duomenų aibe nereiškia kad pasieks gerus rezultatus su kitu modeliu/duomenų aibe. Yra aptartos kelios galimos to priežastys: nuo modelių architektūrinių skirtumų iki skirtingų duomenų aibių charakteristikų. Geriausi klasifikavimo rezultatai buvo pasiekti pasitelkiant mažumos klasės sintetinį generavimą naudojant LLM (angl. Large Language Model). Šis metodas turėjo didžiausią teigiamą poveikį klasifikavimo rezultatams ir pagerino klasifikavimo metrikas didžiausiam skaičiui modelių/duomenų aibių kombinacijų.

Raktiniai žodžiai: Gilusis mokymasis, teksto klasifikacija, klasių disbalansas, teksto generavimas, BERT

List of Figures

List of Tables

Contents

List of symbols

- w denotes a weight in the loss function.
- $\hat{y}$ denotes the label predicted by a classification model.
- y denotes the actual label of an object.

List of abbreviations

CNN	Convolutional Neural Network
SVM	Support Vector Machine
LLM	Large language models
BERT	Bidirectional Encoder Representations from Transformer
NLP	Natural Language Processing
SMOTE	Synthetic Minority Over-sampling Technique

Introduction

Deep learning models can achieve high performance in text classification tasks; however, they face difficulties when there is a significant class imbalance, which in some cases leads to a considerable drop in classification metrics for the minority class. This study aims to examine the performance of different deep learning models in text classification tasks under conditions of high class imbalance in the training data set, conduct a comparative analysis of the models, and explore solutions to the class imbalance problem.

In certain classification tasks a class imbalance is an unavoidable issue (for example fraudulent transactions detection). It is important to evaluate different deep learning models and the results they achieve so that a more informed decision can be made when choosing a model and approach for a specific classification task.

Existing research mostly focuses on older deep learning models, and are often performed just on a single data set and with a single model. The main goal of this thesis is to perform a comparative analysis of different deep learning models and methods that tackle class imbalance to identify the best performing models/methods in a variety of text classification tasks. The main objectives are:

- Train different deep learning with various imbalanced data sets. Identify the best performing model for the different data sets based on the classification results. Provide hypotheses on why model performance differs in the classification tasks.
- For each model/data set combination apply several different methods which fight class imbalance in classification tasks. Compare how classification results vary when applying these methods to different data sets/models. Provide possible reasons on why the methods exhibit different results with different models/data sets.
- Identify the methods which improve classification performance in the largest number of model/data set combinations and which method provides the highest average increase of the f1 score for the minority class.

1 Literature review

The class imbalance issue is not a recent phenomenon. Many real life applications of deep learning (spam detection, fraud identification etc.) naturally are predisposed to the class imbalance issue. This sections provides an overview of a variety of existing research on the imbalance issue, its effects on the results of deep learning models, the methods used to fight this issue and the recent advancements on this topic.

1.1 Effects of class imbalance on classification results

Class imbalance in the training data causes several different issues when trying to train a deep learning model for a particular task. It causes the trained model to over-fit and be biased towards the majority class. This is true for both binary and multi-label classification. This is explored in great detail in a paper which provides an overview of the class imbalance issue in CNNs (Convolutional neural networks) [3]. The CNN model is trained on multiple different data sets that differ by how many classes they contain and the level of imbalance between each class. A clear relation between the severity of class imbalance to the classification results measured by ROC (area under the receiver operating characteristic curve) is demonstrated. As more classes are introduced and when the imbalance ratio is higher the classification performance falls considerably for the CNN model. This demonstrates that class imbalance is a significant hurdle in achieving satisfactory results in certain classification tasks.

It is also beneficial to know how specifically various classification metrics are affected by class imbalance in the training set. In the paper [1] a number of different classification metrics are examined. It is noted that classification methods like accuracy can be deceptive. When trying to evaluate the models overall performance often the classification of the minority class is the point of interest. Accuracy can become ’boosted’ when calculating because of the majority class. In turn hiding the issues of classifying the minority class. Metrics like f1 score are often more useful when evaluating class performance, because it is more ’sensitive’ to classification results of the minority class.

1.2 Methods to fight class imbalance

When training deep learning models for a task with a high class imbalance a variety of approaches can be applied to potentially mitigate the issue. Most approaches that tackle this issue can be categorized into two types: data augmentation methods and algorithmic methods. Data augmentation methods try to modify the training data set to achieve higher results without modifying the method itself.

Conversely algorithmic approaches do not try to modify the data but apply changes to the model architecture or the training process itself. Algorithmic methods range from simple solutions like an extra class weight layer to more complex adjustments like adding Multiple Kernel Learning (MKL) to Support Vector Machines (SVM) which modifies how the weights are calculated [1]. In this section a variety of works on these methods and their application are examined.

1.2.1 Under-sampling and over-sampling

One of the most common methods which seek to remedy the class imbalance issue is under-sampling and over-sampling. These methods are very simple they modify the existing training data set by duplicating or discarding certain objects in the data set based on their class. Their relative simplicity, ease of use and applicability to generally all data sets makes them a popular choice. But depending on the type of model being trained results may wary. For example, while using over sampling for classical models overfitting may occur, but with CNNs over sampling often out performed under sampling on multiple data sets [3].

When utilizing these methods it is also important to consider the ratios between classes. In the paper [16] random under-sampling - RUS and random over-sampling - ROS methods are used with a random forest classifier to classify a highly imbalanced data set from a High School Longitudinal Study of 2009. The authors of the paper compare how different class ratios in the training data set effect the classification results and different methods differ based on the skewness of the data. The authors find that RUS method performs worse for moderately imbalanced data compared to the baseline classification. For extremely imbalanced data ROS showed signs of overfitting while a hybrid approach which combines both under-sampling and over-sampling techniques was the best while reaching mean precision of 0.911 compared to the baseline mean precision of 0.665.

Another consideration for both over-sampling and under-sampling is to what ratio is over-sampling/under-sampling is performed. In the previously discussed study on CNN performance when training data is imbalanced [3] multiple experiments where performed. From having all the classes be under-sampled/over-sampled to an equal number of representative for each class to a class imbalance ratio of 1000 between the classes. Their results show that their CNN model responds best to a minimal class imbalance after under-sampling/over-sampling.

These methods can be utilized effectively, but also have several draw backs. A certain subset of the majority class needs in the training data to be discarded to make under-sampling work. This is not feasible for all types of classification tasks if the training data set is too small.

1.2.2 Data generation

This is where data generation methods come in, which try to improve model classification performance in regards to the minority class by introducing synthetically generated minority class instances into the training data set. One of the earliest methods of this is SMOTE (Synthetic Minority Over-sampling Technique)[4]. Methods like SMOTE can be applied to data to create new synthetic objects for the minority class. It works by calculating the nearest neighbors by Euclidean distance of each instance of the minority class. Then using the distance between the instance and its neighbors new synthetic objects are interpolated. This allows us to avoid discarding already existing data.

For many years SMOTE has been applied on various deep learning task and a variety of variations of this method have been developed. For example Borderline-SMOTE which generates new synthetic samples on the borderline between new classes.

SMOTE is being utilized to this day. In this article [8], a CNN model and a DNN model (Deep Neural Network) are trained multiple times with 24 different imbalanced binary (two class) data sets and a variety of methods which fight class imbalance were utilized and their results compared. The results of the study show that SMOTE out performed other methods and greatly improved the accuracy of the classification. The long lasting use of SMOTE shows the power of synthetic data generation methods.

1.3 Advancements on the class imbalance issue

In recent years more sophisticated models like transformers have reached best in class results in a variety of text classification tasks [14]. Since they were introduced in 2017 they continue to be utilized to great effect and there is value in exploring how these models respond to the class imbalance issue. Furthermore, the rise of LLMs (Large Language Models) which are based on the transformer architecture have allowed a new approach for synthetic data generation which may provide new avenues for training models with synthetic generated data.

In 2017 a new deep learning model architecture was proposed called transformers [15]. These models differ greatly from CNN based models in that they discard recurrence and convolutions. Instead they utilize a self-attention mechanism. Self-attention allows the model to track the relations between all tokens in a sequence. This allows transformer models to reach better results in NLP (Natural Language Processing) tasks then CNN models. In tasks like machine translation transformers at the time outperformed all other models. Another advantage of transformers is that they can be more easily parallelized, which allows them to be trained faster than CNN models.

These transformer based models have been applied to tasks with class imbalance in a variety of fields. In the paper [11] a process of creating a deep learning model based on the transformer architecture is described. The specific model used is BERT (Bidirectional Encoder Representations from Transformer) and for this task it performs Short-Term Voltage Stability Assessment (STVSA). In this task class imbalance is often an issue. The authors of the paper propose the use of Wasserstein Generative Adversarial Network With Gradient Penalty (WGAN-GP) to generate synthetic data. This approach creates a more optimal data set for training the transformer model. They also perform semi-supervised clustering learning. This approach leads to better results than the other models in this field and maintains high results even when the class imbalance is 100:1.

Another highly imbalanced classification task that BERT can also be applied for is spam email detection [2]. A combined data set was collected from 7 other data sets and more classical deep learning models like random forest and the transformer based models distilBERT and roBERTa where trained on it. The transformer architecture models greatly outperformed the classical models and by utilizing methods like class weighted loss and random oversampling mitigated the class imbalance problem and reached the overall accuracy of 99.62%.

With the rise of LLMs a new approach of synthetic data generation has been developed. By fine tuning an LLM with a data set it is possible to generate synthetic data that may represent entries in that data set. By utilizing LLMs oversampling can be performed by generating new members of the minority class. This method has been successfully applied in combination with transformers in various NLP tasks. In the paper [6] it is demonstrated that BERT results can be improved by utilizing LLM generation when classifying multi-label data sets. Often achieving the best results compared to other methods. Although it is noted that for binary classification the results are not so impressive achieving just marginal improvements.

Increased performance of the BERT models and new ways to generate data have greatly improved the results that are achieved in NLP tasks. By using all these new models and methods we have additional approaches when trying to tackle the class imbalance issue.

1.4 Existing overviews of the class imbalance problem

It is also worth to explore some of the existing overviews on this issue. So we can identify what work on this issue has already been done

In this review article [12], different models and methods are analyzed to see their results when classifying an imbalanced data set. It categorizes the methods used to tackle class imbalance into 3 different groups: preprocessing, algorithms and hybrid methods. The test was performed on 23 different data sets. The most effective data set preprocessing methods were over-sampling and under-sampling, in these methods the distribution of the classes in the training data set is modified to reach better results.

Unfortunately, when comparing the different model results the article mostly focuses on Support Vector Models and does not perform tests on more recent models (for example: transformer based models). Nevertheless, the results show the IFTSVM (Intuitionistic Fuzzy State Vector Machine) achieved the best results when evaluating the selected data sets.

An overview on the class imbalance issue was performed in 2025, mainly focusing on various methods to fight imbalance [1]. They analyze data level, algorithmic and hybrid methods. Their analysis show that many methods have trouble with large class imbalances and that there is a need for more general methods that can be applied effectively on a diverse set of use cases. However, this research mostly focuses on application of these methods on older models likes CNNs and SVMs and analysis on more modern models is also absent.

1.5 Conclusions

When analyzing the literature several conclusions were reached. Class imbalance is a long standing issue that has plagued deep learning for decades. Several methods have been developed to try to tackle this issues. Utilizing methods like under/over sampling and synthetic data generation classification can be improved in a variety of tasks. Modern deep learning models based on the transformer architecture (like BERT), which benefit greatly from an increased size of the the training data set and can reach high results with synthetic data generation methods.

More complex data generation approaches like LLM based synthetic data generation allow us to generate synthetic data for highly imbalanced NLP tasks which can show improvements compared to synthetic data generation using long standing methods like SMOTE.

Existing overviews on this issue often cover older deep learning models and research regarding this specific issue on more modern BERT models are slim. In these overviews often the focus is on a limited number of models or a small number of data sets. How different methods to fight class imbalance perform on different model/data set combinations are rarely explored. Because of these reasons a wider exploration of application of these methods on a variety of data sets and models seems worthwhile.

2 Methodology

In this section the data sets and deep learning models used for this work are described. Also the used tools are listed and an overview of methods used in the following experiments which tackle class imbalance problem in deep learning is provided.

2.1 Tools used

In this thesis Python was used for the expansive existing deep learning and data set manipulation libraries and tools.

The specific tools, libraries and resources used for the following experiments are:

- Python: was chosen for the mature environment of libraries for deep learning and deep learning operations.
- pyTorch version 2.7.1: was chosen for it's extensive utility in NLP tasks.
- numpy version 1.26: for operations with arrays and matrices.
- transformers library from Hugging face version 4.36.2: was used for the pretrained BERT and distilBERT models. Also for using google/flan-t5-xl LLM to generate synthetic data.
- pandas version 2.1.4: used for data set manipulation, various python scripts to manipulate and clean up the custom data set used in this work.
- sklearn version 1.7.2: provides a variety of methods that allow to evaluate the classification metrics of the trained model (accuracy, precision, recall, f1 score).
- nlpaug version 1.1.11: this python library was used for a variety of methods that help with NLP tasks. Specifically a method for synthetic data generation was used.
- nltk version 3.9.1: language processing library used together with nlaug for generating synthetic text.
- VU hpc resources to perform training/fine tuning of deep-learning models.
- personal computing resources were also utilized, training/fine tuning was performed with GPU acceleration using an Nvidia RTX 4070 GPU.

2.2 Data sets

In this thesis the deep learning models were trained on 3 different highly imbalanced data sets. In this section the data sets are described, compared and the reasons for picking these specific data sets are discussed.

The main criteria for the chosen data sets was:

- data set is made up of two highly imbalanced classes so binary classification could be performed by the deep learning models. The target amount of imbalance was 95% to 5%.
- data set has at least 50,000 entries. This is needed to test certain methods like under-sampling and to provide enough training data for transformer models, which need a substantial amount of data to effectively fine tune them.
- The text needs to be in English because most accessible LLM models which were used for minority sample text generation perform best with and were trained on English text.

2.2.1 YouTube advertisement data set

The first chosen data set [10] was created by myself and is made up of transcribed lines of text form videos on the YouTube platform (referred to as ”YT” when displaying classification results in tables). This custom data set has two classes: lines of text that are part of advertisements (class 1) and lines of text that are not advertisements (class 0). The data set has 740,000 unique lines of text, 95% of the lines are attributed to class 0 and 5% are attributed to class 1. Examples of both of the class are provided in Table 1. For training and validation a smaller subset of randomly sampled data was used, around 100000 lines of text. This was because of the length of time it would take to train a transformer architecture based model on the full data set. This data set has a high class imbalance, which is useful to evaluate the performance of different deep learning models for this particular paper.

Each line of text is of variable length, from a single word to whole sentences. This is because of how the YouTube API returns transcribed text. The data-set contains lines only in the English language. Also all punctuation, emojis and other non text symbols were removed from the data set in hopes that the model would have an easier time classifying the text. The average length of each line of text is around 40 symbols.

The transcribed YouTube video text was collected using an undocumented, but an openly accessible part of YouTube API. The API returns the line of text, the lines start time and duration in seconds. The collected lines of text are identified as advertisements and non advertisements using the open source database from the SponsorBlock browser extension which tracks advertisements in YouTube videos based on community reporting. Each video has marked times were users have reported that adds appear. This data is combined to create this advertisements data set.

Another reason for this specific data set is that not a lot of research has been done on utilizing deep learning models to detect advertisements in video content. Research on this specific data set could lead to use in a variety of applications, for example: undisclosed advertisement detection to enforce platform guidelines, advertisement detection not based on manual human input etc.

Table 1. *YouTube text data set examples*

text	class
and they would replace it for like 49 US	0
which is less than it used to be	0
for free for two months by being one of	1
the first 500 people to sign up using	1

2.2.2 YouTube advertisement block data set

As we can see from the examples of the YouTube advertisement data set the individual instances of text are quite short. And depending on how the large ad block is split up, the lines could lack the necessary context for the models to classify them correctly.

Because of this reason a new data set was created from this YouTube text data set. The new data set (referred to as ”YT block” when displaying classification results in tables) is made up of combined lines of the same class. In the original data set the text is continuous meaning that the lines of text are from the same video and are sequential. This allows for combining the lines into larger blocks based on class. There is a limit to the maximum block size, because the non-ad blocks would be much longer compared to the ad blocks, because of the lower number of ad lines. The max block length is 4 lines and this length was chosen to roughly equalize the length of both class blocks, so the deep learning models do not become biased based on the length of the input. The resulting data set has around 50,000 blocks of text, with ad blocks making up around 6% of the data set. The average length of non-ad blocks is 28 words, while ad blocks are around 25 words.

Of course by modifying the text to be in blocks instead of the original lines the task being performed by the models is fundamentally different from the original task. The models trained one one or the other data set can not be effectively used for the other task. Nevertheless, this allows us to test how the models perform with the same source data but in a form that gives the models increased context.

2.2.3 Toxic Wikipedia comments data set

Another data set used was the Jigsaw toxic comment data set [5]. This is a data set (referred to as ”TOXIC” when displaying classification results in tables) of Wikipedia comments that where classified into toxic and non-toxic comments. These comments where also labeled on the severity and type of toxicity. These extra labels where discarded because the goal was to perform binary classification and not multi-label classification. The original data set had around 160,000 different comments with no duplicates. The minority class (toxic comments) make up around 9.5% of all objects in the data set. Examples for both types of comments can be seen in Table 2.

This data set is not as heavily imbalance as the YouTube data set. As a result the models would achieve greater results with it and to test the affect of various class imbalance mitigating methods the data set was artificially resampled to a 95/5 imbalance ratio. The new more heavily imbalance data set has around 120,000 lines of text. Another reason for the resampling was that it would match the imbalance level of the YouTube text data set to allow for better comparison.

This data set was chosen because allows us to compare how different models and methods react to a highly imbalanced task in a completely different domain. Also the text itself has different characteristics compared to the YouTube text data set. A single instance of the comment contains the whole comment, unlike the YouTube text data set in which a single add could be spread across multiple instances. Because of this the instances of the toxic comments data set are on average longer taking up around 70 words compared to the 7 average words of the first data set.

Also advertisements in text often try to blend in with the general content of the video and in turn can be hard to spot. While on the other hand toxic comments are by their nature provocative and they stand out from the rest of the comments. Furthermore, the lexicon of toxic comments greatly differs from the non toxic comments, with vulgar language and curse words appearing often in toxic comments and rarely in non toxic comments. Because of these reasons it seemed useful to test how models would handle these two very different data sets. The assumption was that a data set like this would be easier to classify for the models.

Table 2. *Toxic comments data set examples*

text	class
The Mitsurugi point made no sense - why not argue to include Hindi on Ryo Sakazaki's page to include more information?	0
Would you both shut up, you don't run wikipedia, especially a stupid kid.	1

2.3 Deep learning models

In this section the 3 models that were trained/fine-tuned for the experiments in this thesis are described. The models are compared and the reasoning of choosing them is provided.

2.3.1 BERT

BERT is a language presentation model based on the transformer architecture [7]. The model looks at text sequences from left to right and right to left to build better understanding of the texts meaning. BERT has great performance in various text processing tasks. Because the model is pre-trained it can be fine-tuned relatively easily and with modest resources for a variety NLP tasks. Also because the BERT model is pretrained, for fine-tuning we can use less data than when training from scratch because it already has language understanding ”built in”.

For this work a BERT model with additional layers for classification was used from the transformers library ”BertForSequenceClassification”. This model is specifically designed for text classification tasks It also has an additional dropout layer with a dropout ratio of 0.1.

Because of BERTs excellent results in various NLP tasks it was chosen for these experiments, to test how transformer models deal with the class imbalance issue.

2.3.2 distilBERT

Although BERT can be fine-tuned faster than when training a transformer based architecture model from scratch, the computational resources and time required are still quite significant. Because of this a variety of more light-weight models based on the BERT architecture were introduced. As one alternative distilBERT was proposed.

The main difference between the BERT and distilBERT models is the size and the complexity of the model. The distilBERT model is around 40% smaller and is around 60% faster when performing inference while retaining most of BERT models language understanding abilities [13]. This allows the distilBERT model to be fine-tuned much faster than the full-size BERT model. As with BERT a custom model designed specifically for classification tasks is used from the the transformer library: ”DistilBertForSequenceClassification”. When displaying classification results in tables, it is referred to as ”DBERT” throughout this work.

These specific BERT models were chosen because of their proven utility and results in a large number of NLP tasks. Another reason is because the models are pretrained we can fine-tune them for our specific needs relatively quickly and with a modest data set, which allows for faster iteration. And finally transformer based models do not demand the same amount of computational resources as LLM models which utilize a similar architecture. A LLM model (like Chatgpt) was not chosen because of the opaque black box nature of fine tuning a model like it.

Another reason to choose these models is to test how model complexity affects the results in classification. An assumption was made that a more complex model with more features like BERT would outperform a simpler model with less features like distilBERT in classification tasks with a large class imbalance. Even though they share a similar architecture.

2.3.3 CNN

To compare these BERT architecture based models with more classical models a CNN model was also trained. One reason to compare BERT based models with a CNN is to see if the more advanced architecture of these more modern models improves performance in tasks with a large class imbalance. Another reason is to test how various different methods to to fight class imbalance translate between vastly different deep learning model architectures. Which is not possible if only BERT models are compared.

The CNN model in this work is based on the CNN model proposed by Yoon Kim [9], its basic idea is to take in text as a embeddings and pass it trough multiple convolutional layers, after convolutions a max-over-time pooling layer is applied which allows to capture the most important features. The features are passed to a dropout layer and finally to a fully connected output layer. For this thesis the specific implemented CNN model has 128 input channels and 3 convolutional layers with kernel sizes of 3, 4 and 5, each producing 100 feature maps.

Although CNN have existed for a long time this CNN architecture can achieve respectable results in NLP tasks. Of course this model is not pretrained like the BERT models and needs to be trained from zero. Nevertheless because of its relatively simple architecture this model can be trained more quickly and with less computational resources than even the more light weight distilBERT model.

2.4 Methods to tackle class imbalance

In this section the methods that where used for the following experiments are listed and a general overview on how they work is given.

2.4.1 Under-sampling

One of the most simple methods to tackle class imbalance in deep learning is under-sampling (Figure 1). Its goal is to remove a certain amount of members form the majority class, to create a more balanced data set to train the model on while keeping the validation data set unchanged. There are many different under sampling methods, for example the objects of the majority class can be removed form the training data set at complete random or more sophisticated methods can be utilized like ENN (Edited Nearest Neighbors) which utilizes distances between objects to remove the outlier members of the majority class.

These methods can be used to create a balanced data set of course, but the training data set can be modified to create any ratio between the classes. In this thesis, random under-sampling was utilized to create two different data sets of ratios 80-20 and 50-50 from the original 95-5 to compare how the classification metrics change. The down side of this method is that we do discard a certain amount of data that can not be used when training the model. Therefore a large enough data set is required to effectively utilize this method.

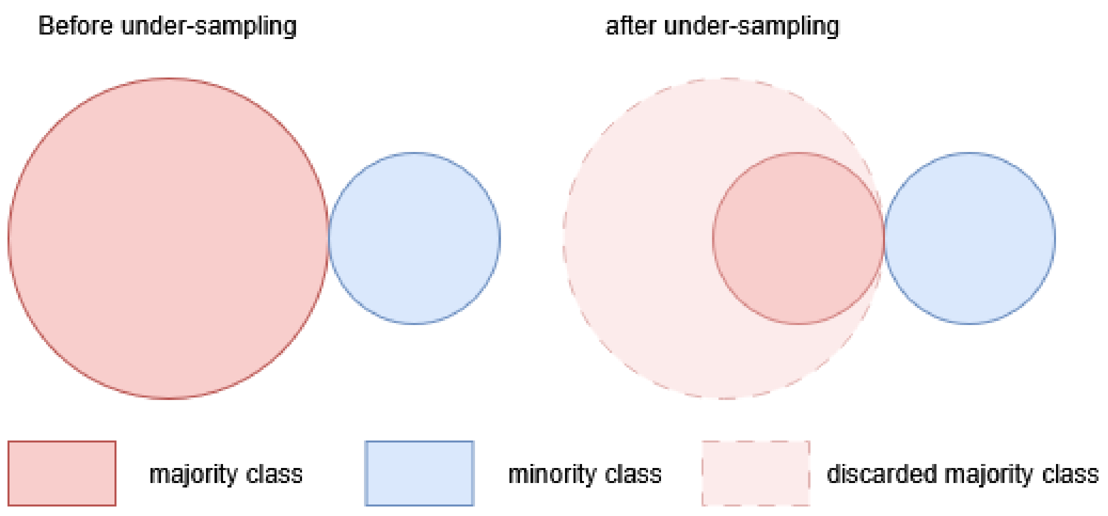


Figure 1. *Under-sampling example*

2.4.2 Class weights

Class weights are a different approach to the class imbalance problem. Instead of discarding a part of the data set when performing under-sampling, weights are added to the loss function to artificially add bias to the model during training. The weights can be added to multiple different loss functions like cross entropy and focal loss. However, no significant difference was found when using either cross entropy loss of focal loss in classification results, the following experiments where performed only using cross entropy loss. Each class can have a different weights and higher weights lead to the model being more biased to that class. Class weights are simple to implement, but unfortunately the results may vary for data sets with a high level class imbalance.

For the data sets used the ”optimal” weights were calculated using the sklearn library for Python. For the YouTube text data set the calculated weights were: w_0 - 0.5297, w_1 - 8.9104. For the YouTube text blocks data set: w_0 - 0.5335, w_1 - 7.9734. For the toxic comment data set: w_0 - 0.5264, w_1 - 9.9750.

$$\mathcal{L} = -\left(w_1\, y \log(\hat{y}) + w_0\, (1-y) \log(1-\hat{y})\right) \tag{1}$$

2.4.3 Synthetic data generation using synonym replacement

Another approach is to utilized synthetic data generation to over-sample the minority class. Methods like these are useful when loosing data by under-sampling is not desired. For example, in image classification tasks, images can be cropped, rotated, inverted etc. to artificially increase the number of objects in the training data and provide more samples to improve classification accuracy and reduce overfitting.

SMOTE works great for creating new synthetic data for numerical data. But in this thesis the deep learning model training is performed on text, even if text is interpreted as numerical data the resulting synthetic text fragments may not be coherent as text. Because of this another method is utilized in this paper.

For the minority class of the data sets a method which generates new sentences by replacing words of the original line with synonyms was used. This creates new data while keeping the meaning of the text mostly the same. For this process nlpaug library was used which provides methods and tools to replace words with their synonyms.

Table 3. *Toxic comments generated with synonym replacement*

text	Is generated
It was already deleted so there is no need to discuss now!! Go To Hell	No
It was already edit thus there is no need to discuss now! ! Go To Hell	Yes
It was already deleted so there is no need to discourse now! ! Go To Hell	Yes

Synonym replacement allows us to create unique sentences without changing the overall meaning of the sentence. Utilizing this multiple new data sets were created with different oversampling amounts. From generating enough instances of the minority class to match the majority task to a less aggressive tripling of the minority classes for both data sets. But it is possible to see some limitations of this text generation method. In one of the examples ”deleted” is replaced with ”edit” which changes the overall meaning of the text unintentionally (Table 3).

2.4.4 Synthetic data generation using LLMs

Another method of text generation was utilized: LLMs. A local instance of the LLM ”google/flan-t5-xl” was hosted and used to generate new instances of the minority class by prompting. The assumption was that the context that LLMs keep would allow for more sophisticated generation of text (Table 4). With text generation using synonym replacement there is a risk that the meaning of the sentence can dramatically change meaning or even not make contextual sense after augmentation.

The greater awareness of the given tasks context provided by the improved language understanding of LLMs could potentially allow for more sophisticated text generation that creates more convincing minority samples. For each data set the text was generated using the training data sets so that the LLM would not generate new minority instances that closely match the samples in the validation set. The generated samples were filtered out if they were too long or too short (based on the length of the original samples) and too similar to the original sample. By adding the remaining generated outputs to the training data sets, the amount of minority samples were roughly doubled compared to the original training data sets.

By generating text using LLM prompting we can change certain parameters to tune the text output:

- top probability: how ”sure” the model needs to be when evaluating did the generated output match the prompt. For generating text in this work top probability of 0.92 was used.
- temperature: controls how much variation to introduce to the model outputs. A value of 0.95 was used in this work.
- repetition penalty: the model is penalized for tokens that already appeared in the output. A value of 1.15 was used in this work.
- Maximum new tokens: caps the length of the output. Value was 80 for toxic comment data set and YouTube text block data set, for the original YouTube text data set 48 was used.

Table 4. *Toxic comments generated with LLM*

text	Is generated
Atlant is a spectactular idiot, superseded only by yourself.	No
Atlant is a spectatular idiot, only a few seconds behind you.	Yes
Atlant is a spectactular moron, superseded only by yourself.	Yes

The following prompts where used to generate text with the google/flan-t5-x model:

- Prompt for the toxic comment data set ($original$ is the original minority sample that is added to the prompt): "Write a DIFFERENT toxic or aggressive online comment with the SAME intent and emotional intensity. Keep it hostile, confrontational, or threatening. Do NOT make it polite. Do NOT add explanations or advice. Use informal internet style (caps, repeated punctuation, short sentences are OK). Keep it between 3 and 60 words. Return ONLY the comment. Example: $original$".
- Prompt for both of the YouTube text data sets: "Rewrite the following YouTube sponsorship disclosure in a natural, casual YouTuber style. Keep the meaning and keep it clearly an ad (mention sponsor or a call to action like link/promo code). Keep length between 7 and 30 words. Return ONLY the rewritten line. Example: $original$.

2.5 Classification metrics

Relying purely on accuracy for evaluating the model results when classifying a imbalanced data set can be misleading. For this reason we take note of multiple different classification metrics for each class in the data set. This helps us better understand how performance differs when classifying the majority and the minority classes.

The metrics are calculated by first counting the true positives -TP, false positives FP, true negatives - TN, false negatives - FN.

- Precision - measures how often is the model correct when predicting a class.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

- recall - measures the models ability to identify the class instances.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

- f1 score - geometric mean of precision and recall, useful for evaluating overall performance for a class.

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

The main goal of these experiments is to try to improve classification performance for the minority class, because of this in the following experiments the models are evaluated and compared based on their f1 score of the minority class.

3 Analysis

In this section experiments combining the data sets and models are described. Their results are displayed and discussed. The goal of all of these experiments is to find out how different models handle different highly imbalanced data sets for a classification task, identify the best performing model based on the minority f1 metrics.

Another goal of these experiment is to test how different methods translate between different data sets and different model architectures and the different level of complexity.

3.1 Model training

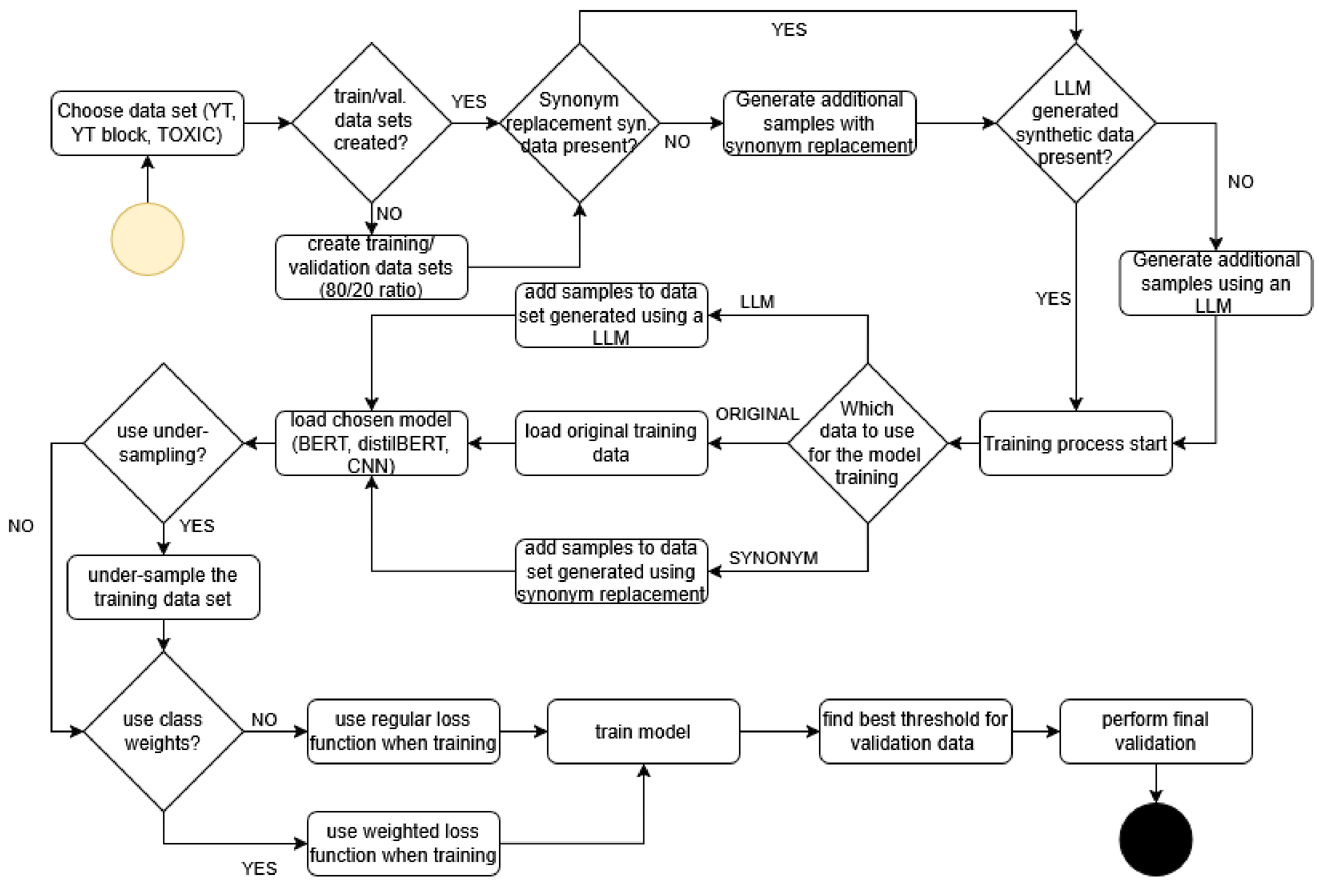
In this thesis 3 different models with 3 of the described data sets were trained/fine-tuned and then trained/fine-tuned by applying 4 different methods to fight class imbalance. In total that makes up 45 different training/fine-tuning permutations.

Additional experiments were performed with different model hyperparameters, different dropout layers, different synthetic data generation methods, but because of the amount of data the focus is only on the 45 main experiments.

All the experiments were performed by creating a training/fine-tuning loop in python that would run all 45 experiments sequentially, changing the methods/models/data sets after each iteration. The general flow for a single experiment can be seen here in Figure 2 . The CNN model was trained for maximum of 20 epochs and the BERT models where fine-tuned to a maximum of 6 epochs. The CNN model was trained for longer because it is trained from scratch and the BERT models are already pretrained. Also an early stopping mechanism was implemented if no improvements based on the f1 score were gained after a specified number of epochs: 3 - for BERT based models, and 6 for CNNs.

After training the the best threshold in regards to the f1 score of the minority would be identified and for the final validation this threshold would be used. During validation accuracy, precision, recall and f1 score would be saved for each class in a file for later review. When evaluating results metrics are taken form the epoch with the best results and not the final one.

The full training/fine-tuning loop would take around 16 hours on the RTX 4070 GPU.



Choose data set (YT, YT block, TOXIC)
train/val. data sets created?
YES
NO
create training/ validation data sets (80/20 ratio)
Synonym replacement syn. data present?
YES
NO
Generate additional samples with synonym replacement
LLM generated synthetic data present?
NO
YES
Generate additional samples using an LLM
Training process start
Which data to use for the model training
LLM
ORIGINAL
SYNONYM
add samples to data set generated using a LLM
load original training data
add samples to data set generated using synonym replacement
load chosen model (BERT, distilBERT, CNN)
use under-sampling?
YES
NO
under-sample the training data set
use class weights?
NO
YES
use regular loss function when training
use weighted loss function when training
train model
find best threshold for validation data
perform final validation

Figure 2. *Model training process*

3.2 Model hyperparameters

When trying to find the best hyperparameters for each of the models multiple experiments where done with different combinations of hyperparameters. The best performing hyperparameters where used for all further experiments (Table 5).

Table 5. *Model hyperparameters*

	Epochs	batch size	learning rate	max length
BERT	6	8	$2.5e^{-5}$	128
distilBERT	6	8	$2.5e^{-5}$	128
CNN	20	16	$1e^{-3}$	128

3.3 Baseline model performance

At first glance all the models reach great results with all the data sets. Accuracy for all model/data set combinations is over 90%. However, when looking at the classification metrics for the minority class it can be seen that they are significantly worse than the classification metrics for the majority class.

3.3.1 Baseline model performance with the YouTube text data set

When evaluating the different model results (Table 6) for the YouTube text data set several things stand out. A clear divide between the BERT based models and the CNN model. the BERT and distilBERT models are neck to neck in their results achieving overall accuracy of 94% and their f1 score for the minority differs only by 1% with a slight lead to the more complex BERT model. Meanwhile the CNN falls back slightly when looking at overall accuracy by 2%, but the minority class f1 score is worse by 7% when compared to the BERT model. In particular the precision of the CNN model is worse than the BERT models.

When looking at the results of all models it can be clearly seen that they all struggle greatly when trying to identify ads in this data set. All the classification metrics are worse by more than half compared to the majority class. This indicates that the class imbalance creates significant issues for all deep leaning models tested.

Table 6. *Model results with the YouTube text data set*

(a) *BERT model results*

class	precision	recall	f1 score	support
0	0.97	0.97	0.97	24950
1	0.46	0.44	0.45	1497
accuracy	0.94			

(b) *DistilBERT model results*

class	precision	recall	f1 score	support
0	0.97	0.97	0.97	24950
1	0.47	0.42	0.44	1497
accuracy	0.94			

(c) *CNN model results*

class	precision	recall	f1 score	support
0	0.97	0.95	0.96	24950
1	0.35	0.43	0.38	1497
accuracy	0.92			

3.3.2 Baseline model performance with the YouTube text block data set

When looking at the model results when trying to classify ads in YouTube video text that has been combined into longer blocks (Table 7) it can clearly be seen that the models perform significantly better when classifying the minority class when compared to the original YouTube text dat set. The transformer based models pull ahead again, achieving overall accuracy of 96% while the CNN model reaches accuracy of 94%.

When looking at the f1 score of the minority class the BERT model slightly outperforms the leaner distilBERT model, which in turn outperforms the CNN models by 7%. Nevertheless all models f1 score for the minority class achieves a respectable boost just under 20%. This may indicate that the increased text length of the objects in this data set provides increased context about the class of the object. Which in turn improves the performance of all models.

Table 7. *Model results with the YouTube text (blocks) data set*

(a) *BERT model results*

class	precision	recall	f1 score	support
0	0.97	0.98	0.98	9411
1	0.69	0.60	0.64	624
accuracy	0.96			

(b) *DistilBERT model results*

class	precision	recall	f1 score	support
0	0.97	0.98	0.98	9411
1	0.66	0.59	0.62	624
accuracy	0.96			

(c) *CNN model results*

class	precision	recall	f1 score	support
0	0.97	0.97	0.97	9411
1	0.56	0.54	0.55	624
accuracy	0.94			

3.3.3 Baseline model performance with the toxic comment data set

With the Wikipedia toxic comment data set all the models achieve the highest results yet (Table 8). All the models reach overall accuracy within 1% of each other with a slight disadvantage for the CNN model. The CNN model again has a worse f1 score for the minority class compared to the transformer models. This shows a pattern of the CNN models struggling more in classifying the minority class compared to the BERT models in all the data sets. Even more intriguing is that when looking at the f1 score of the minority class, the less complex distilBERT model outperforms the larger BERT model although it is only by 1%, bucking the trend of BERT outperforming the other two models in every data set.

In this data set the delta between the classification metrics of the majority class and the minority class is the lowest of all data sets. This shows that the models can more easily detect toxic comments than advertisements in video content. This maybe because toxic comments are very distinct from regular comments, with heavy use of upper case letters, excessive punctuation and expletives.

Table 8. *Model results with the toxic comment data set*

(a) *BERT model results*

class	precision	recall	f1 score	support
0	0.99	0.99	0.99	28871
1	0.77	0.74	0.76	1503
accuracy	0.98			

(b) *DistilBERT model results*

class	precision	recall	f1 score	support
0	0.99	0.99	0.99	28871
1	0.81	0.73	0.77	1503
accuracy	0.98			

(c) *CNN model results*

class	precision	recall	f1 score	support
0	0.98	0.99	0.99	28871
1	0.82	0.64	0.72	1503
accuracy	0.97			

3.3.4 Conclusions of the baseline experiment results

From these baseline experiments where no methods that fight class imbalance were applied, several patterns can be seen. The BERT model achieves the best results when looking at accuracy and f1 score of the minority class, being outperformed by distilBERT only once and only by 1%. Both of the models based on the transformer architecture are relatively close, the larger size of the BERT model only provides small advantages. This may indicate that added model complexity will not necessarily improve performance in highly imbalanced classification tasks.

A more significant difference can be seen when comparing the BERT models to the CNN model. The CNN model is outperformed in all data sets. Another interesting note is that the delta between the CNN model and the BERT model increases when the minority class is harder to classify. When classifying the toxic comment data set (in which all the models perform the best when classifying the minority class) the delta between the different architecture models is only 5%. But in the YouTube text data set (which the models struggled the most with) the delta increases to 7%. It may indicate that the more complex architecture of the BERT models provides them with an advantage in more complex tasks with a high class imbalance.

What all models have in common is that they struggle when classifying the minority class to varying degrees depending on the data set, which can be clearly be seen when looking at f1 score of the minority class (Table 9). It is not possible to brute force better performance by only utilizing more modern and advanced deep learning models. This shows the need to explore how the models perform when applying methods which specifically try to increase the classification performance for the minority class.

Table 9. *Model performance when class weights are applied*

Data set	BERT	DBERT	CNN
YT	0.45	0.44	0.38
YT blocks	0.64	0.62	0.55
TOXIC	0.76	0.77	0.72

3.4 Methods for the class imbalance problem

Various methods to improve classification accuracy for the minority class were applied when training the models. For each data set and model combination 4 different methods were applied during training. This creates 36 different experiments and the results of each experiment are represented, discussed and compared.

The tested methods are:

- Class weights.
- Random under-sampling.
- Synthetic data generation using synonym replacement.
- Synthetic data generation using LLM generation.

3.4.1 Class weights

Class weights is an algorithmic method that introduces bias to the model by applying it to the loss function. Because of its ease of implementation in to the training process, this method was a prime candidate for testing. For these experiments the class weights were calculated individually for each data set and cross entropy loss was used for all models.

When looking at the results of the models when class weights are applied in training (Table 10) we see quite varied outcomes. The most disappointing aspects it the when applying class weights it fails to improve performance in all models for the original YouTube text and toxic comments data sets. Not only does it not improve results it often hurts the minority class f1 score.

The only exception was YouTube text block data set which saw a considerable improvement for all models. The CNN models sees the biggest improvement, by utilizing class weights the f1 score rises by 4%. The transformer based models also see an improvement of 2%, the larger increase of the CNN model may indicate that the CNN architecture benefits more from a weighted loss function. Also for the data sets that fail to see improvement the f1 score for the CNN model stays the same while for the distilBERT model it falls for both data sets and BERT performance falls in the original YouTube text data set. This may also indicate that the CNN architecture has some greater affinity for this method. Nevertheless the BERT model leads results with all data sets.

When examining the mediocre results for the YouTube text and Toxic comment data set several possibilities are possible. An incorrect of suboptimal implementation of class weights is ruled out because the same implementation improves results for the YouTube text block data set. The derived YouTube data set may perform better than the original YouTube text data set because of the increased information provided to the models by the extended text length. Even when adding bias to the models if the line of text has too much ambiguity it does not help the model in identifying the minority class.

The classification of the toxic comment data set already reaches higher results in regards to the minority class compared to classification of the other two data sets. This may indicate that the models are already reaching a limit in their performance for this particular data set.

Table 10. *Model performance measured by minority class f1 score when class weights are applied*

Data set	BERT	DBERT	CNN
YT	0.44	0.42	0.37
YT blocks	0.66	0.64	0.59
TOXIC	0.76	0.75	0.72

3.4.2 Under-sampling

Another method that was applied was random under-sampling. Data in from the majority class is randomly discarded to balance the data set. The distilBERT model was trained with YouTube text block data set under-sampled to 3 different ratios:

- Class 0 - 2517, class 1 - 2517.

- Class 0 - 1678, class 1 - 2517.
- Class 0 - 10068, class 1 - 2517.

Unfortunately, aggressive under-sampling does not provide good results for this particular data set as can be seen from Table 11. With equal ratios between classes or when enough majority instances are discarded that the minority becomes the dominant class recall increases over the baseline but the precision is decreased leading to an overall lower f1 score compared to a less aggressive under-sampling ratio. Because of these results it was decided to use a 80/20 ratio for all experiments, otherwise the amount of needed experiments would have grown outside the scope of this work.

Table 11. *Model results with different ratios of under-sampling*

(a) *Equal class imbalance*

class	precision	recall	f1 score	support
0	0.97	0.97	0.97	9411
1	0.55	0.60	0.57	624
accuracy	0.94			

(b) *Class imbalance 8/2*

class	precision	recall	f1 score	support
0	0.97	0.97	0.97	9411
1	0.60	0.62	0.61	624
accuracy	0.95			

(c) *Class imbalance 2/3*

class	precision	recall	f1 score	support
0	0.98	0.95	0.96	9411
1	0.46	0.64	0.54	624
accuracy	0.93			

When applying under sampling we can see that the models under perform significantly compared to the base line results (Table 12). Out of the 9 experiments only the CNN model when trained on the YouTube video text block data set sees a respectable improvement of 3% to the minority f1 score compared to the base line results. And even then, it fails to see any improvement for the other data sets. With all these experiments the minority class recall increases, but precision falls.

This lackluster performance of random under-sampling may indicate that the models are relying of the majority class imbalance to build a decision boundary between when performing classification. The improvement of the CNN model may indicate that in some cases it becomes biased because of the increased number of majority samples.

Table 12. *Model performance when under-sampling is performed*

Data set	BERT	DBERT	CNN
YT	0.44	0.43	0.36
YT block	0.64	0.62	0.58
TOXIC	0.75	0.72	0.70

3.4.3 Synthetic data generation using synonym replacement

If removing majority samples hurts classification performance, creating new minority samples may lead to a different result. Often random over-sampling is performed so that the models encounter more minority samples during training. For these experiments a different approach was taken. New minority samples were generated by utilizing text generation using synonym replacement of the existing minority samples. This would introduce new unique samples to the model when training, compared to random over-sampling that would repeat existing data. The total number minority samples were tripled by the generated text for each data set.

The results are quite varied as can be seen in Table 13, the synthetic data generation method fails to increase classification results in regards to the minority class for most of the model/data set combination. It does however, improve the results for the CNN model with the YouTube text block data-set by a quite significant 5%.

For the original YouTube text data set the length of text may be the limiting factor when generating text. The failure to improve the classification of toxic comments data set may be down to the fact the synonym replacement performed by the nlpaug data set creates ”softened” results. The nlpaug libraries synonym map replaces curse words with often non-fitting alternatives in this specific context. A more specialized synonym replacement approach may be needed for text generation in this domain.

Table 13. *Model performance when data generation using synonym replacement is applied*

Data set	BERT	DBERT	CNN
YT	0.41	0.43	0.36
YT blocks	0.64	0.62	0.60
TOXIC	0.74	0.74	0.70

Synthetic data generation using LLMs

The final tested method that tackles the class imbalance problem was synthetic data generation using LLMs. The assumption was that the increased language understanding of LLMs would allow for more convincing minority class generation compared to the fairly rudimentary synonym replacement. However, there are some disadvantages compared to the synonym replacement method. Firstly it is way more resource intensive requiring more computational power and time. Also the LLMs can hallucinate and generate non fitting or nonsensical text that needs to filtered out. Because of these reasons less minority samples were generated for each data set. Just doubling the amount of minority sample instead of tripling when performing synonym replacement.

LLM generated data application provides interesting results (Table 14). Compared to synthetic data generation using synonym replacement LLM generation is able to improve performance in 3 cases, furthermore it improves f1 score by a greater margin. ALL of the models are improved when classifying the YouTube text block data set. The distilBERT and BERT results are improved by 2%, while the results for the CNN model are increased by 6% which is the largest increase in all of the performed experiments.

Only the YouTube text block data set sees any improvement. This may indicate that the model fails to create convincing minority samples when the minority samples are very short as in the original YouTube text data set. It also may fail to generate convincing toxic comment instances, because of the content it was trained on. These LLMs can be fine tuned and perhaps by training them with toxic comments they would be able to generate more convincing examples.

Table 14. *Model performance when LLM generated minority samples are used*

Data set	BERT	DBERT	CNN
YT	0.43	0.43	0.38
YT blocks	0.66	0.66	0.61
Toxic comments	0.75	0.74	0.71

3.4.4 Comparison of synthetic data results with real data

For performing these experiments as mentioned before the toxic comment data set was re-sampled to have a larger class imbalance. Because of this the previously discarded minority samples can be introduced back into the data set and test how the model performs after training compared to expanding the data set with generated samples of the minority class.

Only the distilBERT model was trained on these ”new” real minority samples, because of the similar relative performance between the models. As the results show, by introducing actual minority samples back into the data set the classification results for the model improve (Table 15). This shows that distilBERT has potential for further improvements and that they have not reached the Bayes limit. The limiting factor when training the models with generated data may be the quality of the samples generated.

The data is generated based on existing samples and the model may not get new useful info to better recognize samples it hasn’t encountered yet. Maybe a more drastic generation method that would alter the text more dramatically would have achieved better results. Another possible approach is too use more complex and computationally expensive LLMs to generate more varied and convincing data for training. Fine tuning the LLM could also be a viable approach.

Table 15. *DistilBERT model with additional real minority samples*

class	precision	recall	f1 score	support
0	0.986	0.98	0.98	28859
1	0.80	0.84	0.82	3056
accuracy	0.65			

3.4.5 Other applications of methods that tackle class imbalance

As the results show for these particular models and data sets the effectiveness of the methods tested is quite varied. Nevertheless, even if the methods fail improving the classification results for the minority class sometimes there are benefits that may not be apparent at first.

For example when training the CNN model for 7 epochs the baseline results are better than results when applying class weights. However, if the model is trained in a time constrained environment just for a single epoch, by utilizing class weights we reach better results for the minority class when training with the YouTube text data set (Table 16).

The precision for the majority and minority classes falls, but the f1 score for the minority class improves by 14%. This could be useful when there is a need to train a model more quickly and with better classification performance for the minority class. This show that the tested methods could have further utility outside of improving the final classification metrics.

Table 16. *Model results with the YouTube text data set*

(a) *CNN model baseline results after 1 epoch* ***(b)*** *CNN model with weights results after 1 epoch*

class	precision	recall	f1 score	support	class	precision	recall	f1 score	support
0	0.97	1.00	0.97	24950	0	0.96	0.97	0.97	24950
1	0.67	0.12	0.21	1497	1	0.40	0.32	0.35	1497
accuracy	0.94				accuracy	0.95			

Experiment analysis conclusions

Overall 45 different experiments were performed when trying to measure the performance of methods that try to mitigate the class imbalance problem. With additional experiments performed to find the most optimal hyperparameters and under-sampling ratios.

Table 17. *All experiment comparison*

	BERT TOXIC	DBERT TOXIC	CNN TOXIC	BERT YT	DBERT YT	CNN YT	BERT YT block	DBERT YT block	CNN YT block
baseline	0.76	0.77	0.72	0.45	0.44	0.38	0.64	0.62	0.55
weights	0.76	0.75	0.72	0.44	0.42	0.37	0.66	0.64	0.59
undrsmpl.	0.75	0.72	0.70	0.44	0.43	0.36	0.64	0.62	0.58
Syn. gen.	0.74	0.74	0.70	0.41	0.43	0.36	0.64	0.62	0.60
LLM gen.	0.75	0.74	0.71	0.43	0.43	0.38	0.66	0.66	0.61

The methods that were utilized for these experiments show varied results (Table 17). Only in 8 out of the 36 experiments do we see improvements to the f1 score when classifying the minority class. This shows that these explored methods can not be generalized for these data sets and models.

When looking at the differences between the data sets and models several interesting patterns can be identified. The only data set that was able to see improvement was the YouTube text data block data set. There are several possible reasons when compared to the other data sets.

Both of the data sets sourced from YouTube video text are quite similar but have key differences. One reason maybe that the models already have a better baseline performance when classifying the block data set so they have more potential to see improvement when applying class imbalance mitigation methods.

Both the higher baseline results and the increased effectiveness of the methods may also be down to the fact that the objects in the block data set have significantly longer text. This longer text contains the whole advertisement and it may be easier to identified compared to the short text lines in the original data set.

When looking at the toxic comment data set the answer is not so clear. The baseline results with the toxic comment data set are the best out of all data sets tested. The models may already be reaching the limit of their performance and models can not easily improve classification of the minority class. Although by introducing minority samples back in to the training data set that were earlier discarded to create a larger imbalance the models are able to reach higher results. This shows that there is room to grow.

Another reason for the lackluster method performance with this data set is the differences in the domain of the task and the characteristics of the text. The YouTube data sets have all punctuation removed and the toxic comment data set often has excessive punctuation which may increase the difficult in text classification tasks.

Also text generation methods may create more convincing examples for the YouTube data sets. When looking at specific instances of text generated by synonym replacement this method fails to substitute curse words with convincing alternatives introducing noisy samples into training.

Examining the results between the 3 different models a few interesting points stand out. Firstly the CNN model results see improvement with every method when classifying the YouTube text block data set. The methods improve the CNN model results in the range from 3% to 6% of the f1 minority score. With the synthetic data generation using LLMS seeing the highest increase out of all methods. This may indicate that the CNN models is a lot more receptive to the methods than the BERT based models who see improvement with only half of the methods.

The BERT models see improvements from two of the methods: class weights and synthetic data generation using LLMS. The distilBERT model sees the highest improvement of 4% when LMM data generation is applied, just like the CNN model. The BERT models which is the most complex out the 3 models tested sees a maximum improvement of 2% which is lower than the improvement that the other models see. This may indicate that with an increase of the complexity of the model the value of these methods diminishes.

When looking at the different methods themselves it is possible to see same differences. The class weights and LLM data generation outperform the other methods. Random under-sampling and synthetic data generation are only able to improve classification for the CNN model. This could mean that for transformer based models these methods do not provide much value in regards to improving the classification results for the minority class.

Although both LMM data generation and class weights improve performance for all models. The LLM data generation provides higher average improvement compared to class weights. While these LLM results are impressive it is important to remember that generation of synthetic data using LLM is significantly more computationally expensive and takes much longer than simply applying weights to the loss function. Depending what is more important: the highest classification results or slightly lower results but applied more easily and without additional work one or the method may be chosen for a given task.

Overall the tested methods provide mediocre results and can not be easily generalized between different methods and data sets. However, even though the methods fail to improve classification performance when looking at the f1 metric of the minority class in most cases, these methods have other applications. For example, if the CNN model is trained only for a single epoch, the model reaches much better results for the minority class when applying these methods then when not.

Results and conclusions

In this master's thesis a comparative analysis of multiple deep learning models and methods that tackle class imbalance was performed. A process that allows training models with different model/data set/method permutations sequentially was developed. Using it 45 different experiments were performed during which classification metrics were collected, which allows us to compare model/method performance in different highly imbalanced tasks. The results of the experiments are:

- Baseline model performance was compared by training each model without applying any methods that fight class imbalance. When looking at the classification results, we can see that the more modern transformer based models (BERT and distilBERT) outperform the CNN model in text classification tasks in all examined data sets. This may indicate that the pretrained models based on transformer architecture have an advantage over older CNN models because of their architecture and can be more effectively utilized for class imbalance. The BERT model achieves best results in 2 out of 3 data sets, while distilBERT achieves the best results in the toxic coment data set.
- Effectiveness of different methods was measured and compared by performing 36 different experiments with unique models/data sets. The generalization of the methods is quite poor, if one methods performs well in one data set or with one model that does not mean the same improvement will translate to another model/data set combination. The 4 applied methods were only able to improve the f1 score for the minority task only in 6 experiments out of 36. In addition, these methods improved classification only in one data set. This could mean that some data sets do not benefit from the tested methods do to the characteristics of the data. Nevertheless, all of the methods were able to improve classification with at least one data set/model combination showing that they all can be effective under certain circumstances.

  Furthermore, we can see that the CNN model sees improvements with all methods while the BERT models only see improvements with two methods. Also, the significantly more complex and larger BERT model can achieve only minimal improvements over the lighter distilBERT model. Indicating that model complexity helps only to a point when applying these methods in highly imbalanced tasks
- The performed comparison of the methods also show that best classification results are achieved by utilizing class weights and synthetic data generation using a LLM. They are both able to improve results with all models while under-sampling and synthetic data generation using synonym replacement only improves classification performance of the CNN model. Data generation using a LLM allows the BERT and distilBERT model to achieve the highest f1 score for the minority class, improving it over the baseline by 2%. The CNN model achieves a more substantial boost when training with LLM generated data increasing the results by 6%, the largest improvement of all tests.

- Although when looking at the overall results the methods that fight class imbalance improve the classification results in regards to the minority class in just one of the 3 examined data sets, they have other utility. In time constrained environments when training just for a single epoch, the application of these methods allows the models to reach better results for minority class then when not applying them in certain cases. This shows that these methods could have value when applying them in tasks were a faster training process is important.

Future work

In future research more models could be examined to see how the deal with class imbalance problem. Furthermore, attempts could be made at improving the results of the examined data sets by utilizing more methods (GANs, SMOTE) and performing additional experiments with already analyzed ones (with different hyperparameters). For example in this work only one set of class weights were tested, a variety of combinations could be tested.

Research could also be expended to include classification of non text data sets. For example in image classification a lot more methods can be used for over-sampling like image cropping, rotation filtering that could provide meaningfully different results form the data sets examined in this work.

References and sources

[1] M. Altalhan, A. Algarni, M. Turki-Hadj Alouane. “Imbalanced Data Problem in Machine Learning: A Review.” In: *IEEE Access* 13 (2025). Cited by: 44; All Open Access, Gold Open Access, pages 13686–13699. https://doi.org/10.1109/ACCESS.2025.3531662. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85216285363&doi=10.1109%2fACCESS.2025.3531662&partnerID=40&md5=6f9ef31b68623fcf88ebb3fa44837efc.

[2] H. Asliyuksek, O. Tonkal, R. Kocaoglu. “A Comparative Evaluation of a Multimodal Approach for Spam Email Classification Using DistilBERT and Structural Features.” In: *Electronics (Switzerland)* 14.19 (2025). Cited by: 0; All Open Access, Gold Open Access. https://doi.org/10.3390/electronics14193855. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-105019054531&doi=10.3390%2felectronics14193855&partnerID=40&md5=ace14519c710c4408e5c7bd1529bde46.

[3] M. Buda, A. Maki, M. A. Mazurowski. “A systematic study of the class imbalance problem in convolutional neural networks.” In: *Neural Networks* 106 (2018), pages 249–259. ISSN: 0893-6080. https://doi.org/https://doi.org/10.1016/j.neunet.2018.07.011. URL: https://www.sciencedirect.com/science/article/pii/S0893608018302107.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer. “SMOTE: Synthetic minority over-sampling technique.” In: *Journal of Artificial Intelligence Research* 16 (2002). Cited by: 25726; All Open Access, Gold Open Access, Green Open Access, pages 321–357. https://doi.org/10.1613/jair.953. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-0346586663&doi=10.1613%2fjair.953&partnerID=40&md5=41cb92c1b2bca5665ce846a089d3aae6.

[5] cjadams, J. Sorensen, J. Elliott, L. Dixon, M. McDonald, nithum, W. Cukierski. *Toxic Comment Classification Challenge*. https://kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge. Kaggle. 2017.

[6] N. A. Cloutier, N. Japkowicz. “Fine-tuned generative LLM oversampling can improve performance over traditional techniques on multiclass imbalanced text classification.” In: Cited by: 17. 2023, pages 5181–5186. https://doi.org/10.1109/BigData59044.2023.10386772. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85183823640&doi=10.1109%2fBigData59044.2023.10386772&partnerID=40&md5=3c475d17fb687f3c28441cf3370792fd.

[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. URL: https://arxiv.org/abs/1810.04805.

[8] J. Joloudari, A. Marefat, M. Nematollahi, S. Oyelere, S. Hussain. “Effective Class-Imbalance Learning Based on SMOTE and Convolutional Neural Network.” In: (2023).

[9] Y. Kim. *Convolutional Neural Networks for Sentence Classification*. 2014. URL: https://arxiv.org/abs/1408.5882.

[10] K. Kvedaravičius. *YouTube advertisements*. https : / / huggingface . co / datasets / KarolisKved/YoutubeAds. HuggingFace. 2025.

[11] Y. Li, J. Cao, Y. Xu, L. Zhu, Z. Dong. “Deep learning based on Transformer architecture for power system short-term voltage stability assessment with class imbalance.” In: *Renewable and Sustainable Energy Reviews* 189 (2024), page 113913. ISSN: 1364-0321. https://doi.org/10.1016/j.rser.2023.113913. URL: http://dx.doi.org/10.1016/j.rser.2023.113913.

[12] S. Rezvani, X. Wang. “A broad review on class imbalance learning techniques.” In: (2023).

[13] V. Sanh, L. Debut, J. Chaumond, T. Wolf. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.” In: *CoRR* abs/1910.01108 (2019). URL: http://arxiv.org/abs/1910.01108.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin. “Attention Is All You Need.” In: *CoRR* abs/1706.03762 (2017). URL: http://arxiv.org/abs/1706.03762.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin. “Attention Is All You Need.” In: *arXiv preprint arXiv:1706.03762* (2017).

[16] T. Wongvorachan, S. He, O. Bulut. “A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining.” In: *Information* 14.1 (2023). ISSN: 2078-2489. https://doi.org/10.3390/info14010054. URL: https://www.mdpi.com/2078-2489/14/1/54.