An Influence-guided data filtering method for data augmentation on sentiment analysis

Master Thesis

HUANG XIAOWEI
M225766

Hiroshima University
Graduate School of Advanced Science and Engineering
Informatics and Data Science Program
Social Computing Laboratory

Supervisor:
Professor Yasuhiko Morimoto

Sub Supervisor:
Associate Professor Sayaka Kamei
Professor Yusuke Hayashi

February 2024

Abstract

We propose a new approach to data filtering called Influence-guided data filtering applied to data augmentation. Data augmentation methods are commonly employed to mitigate overfitting and improve the generalization of deep neural network models. Current approaches involve examining additional replacement words and immediately substituting them. However, our objective is to improve the model by filtering out the most important sentences, which we expect will lead to better training results because more relevant sentences are obtained in the context of a smaller corpus. In the rapidly evolving digital media landscape, where emotions are extensively expressed online, our experiments on sentiment analysis tasks demonstrate that augmenting importance-filtered expectations yields superior improvements.

Acknowledgements

I extend my sincere appreciation to Professor Yasuhiko Morimoto and Associate Professor Sayaka Kamei for their invaluable guidance throughout my master's course and for their thorough review of this thesis as my supervisor. Thanks to Professor Hayashi for providing me with a good career plan and valuable and insightful advice on revising my thesis. Furthermore, I would like to express my heartfelt thanks to Hiroshima University for providing a conducive learning environment. The academic resources and facilities offered by the university have contributed significantly to the successful completion of my research. I am also grateful to all members of the Social Computing Laboratory for their unwavering support and engaging discussions on related academic matters, enriching my research journey. Last but not least, I would like to thank my family for providing me with tons of support so that I can study without any distractions! Their support has been instrumental in shaping my academic endeavors.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The demand for sentiment analysis has been increasing in recent years because of the development of technology and the increase in social media. Sentiment analysis, commonly known as opinion analysis, involves computational methods to investigate people's feelings, attitudes, evaluations, and appraisals regarding diverse aspects of products, services, entities, individuals, topics, events, and issues. Sentiment analysis is a common task in the natural language process(NLP) involving categorizing texts. The rapid advancement of neural networks and deep learning has driven this field's development and widespread adoption. This task achieved relatively perfect results due to the advent of transformers. Subsequently, a model named Bidirectional Encoder Representations from Transformers (BERT) based on transformers has shown impressive performance in the NLP, which was introduced in [1]. This development has provided a reasonable basis for various downstream tasks in the text domain, including sentiment analysis. Both Transformer and BERT are based on deep learning to achieve better performance.

The amount of data required for deep learning is generally large, although it is more accurate than traditional machine learning. Data augmentation becomes very important because too little data can lead to over-fitting in deep learning. The main idea of data augmentation is generating or synthesizing data by various means to expand the data when there is only a small amount of data. Data augmentation was first studied and explored in computer vision tasks. At the image level, it is also easier to perform data augmentation through various image transformations without destroying the meaning of the original image, so the data augmentation technique will also be more mature in image vision. Data augmentation techniques are also more mature in the image field, and we can easily see this observation in work [2], [3].

Data augmentation is also still needed in data processing for NLP, But at first, the idea could have worked out better. It may be due to the difficulties associated with the discrete nature of language, which prevents text from being augmented as simply as images. Despite the challenges, many large pre-trained models have emerged with the rapid development of NLP, and naturally, more

relevant downstream tasks can be explored. In this background, many new domains or tasks are low-resource; in other words, the datasets will generally be small and lack training data; thus, data augmentation is significant in this condition. After a large number of researchers in this area of research has made tremendous progress and also confirmed that the data enhancement use in the model training process through a particular method of generating new data to improve the generalization ability of the model in the actual scenario, and in today's trendy field of deep learning can also get excellent results.

However, current research on data augmentation predominantly focuses on substituting specific words within a sentence, a practice that may only partially capture the complexity of sentence-level augmentation. To address this limitation, we propose a novel approach inspired by real-life learning scenarios, where emphasis on critical content yields better results. Our method identifies essential data during model training using the influence function[4] from the Explainable Artificial Intelligence (XAI) field, and there is a detailed summary and introduction to this concept in work [5]. XAI seeks to enhance the transparency and understandability of machine learning models, contributing to improved decision-making processes. By applying the influence function to data augmentation, we shift the focus from merely replacing words in isolated sentences to augmenting entire sentences.

This work explicitly concentrates on identifying and augmenting crucial data to enhance performance. Leveraging a small dataset, we fine-tuned BERT for sentiment analysis. Subsequently, utilizing the influence function, we compute influence scores to identify significant data portions. Augmenting this essential data results in a more comprehensive dataset, contributing to improved model validation and overall performance.

The main contributions of this thesis are the following:

- A new data filtering method is proposed, which can improve the performance of data augmentation methods and achieve better performance on a data-sparse NLP task.
- Combining the influence function in XAI with BERT, one of the most famous representatives of big models today.
- Explored a popular and successful data augmentation named Easy Data Augmentation with excellent performance gains in our proposed framework and found the optimal augmentation parameters.

The subsequent sections of the paper will adhere to the following structure. Chapter 2 will provide a comprehensive review of pertinent methods and related work essential for conducting the experiments. In Chapter 3, the model structure and the overarching methodology employed in this thesis will be elucidated. Chapter 4 will discuss the experiments' specifics and present the corresponding results. Finally, Chapter 5 will encapsulate the theory with a concise summary.

Chapter 2

Related Works

2.1 Data Augmentation

In NLP, various approaches to data augmentation techniques are also commonly employed. Among these, lexical replacement stands out as a widely utilized method. This approach involves the substitution of a specific part of the original text. Notably, the challenge lies in ensuring the replacement keeps the sentence's overall meaning identical. One mature method entails randomly selecting a word from a sentence and substituting it with a synonym. Often leveraging resources like the WordNet database, this technique has been implemented in works such as [6], [7]. The latter utilized a similar method for their sentence similarity model, generating an additional 10,000 pieces of data.

Another prominent method involves word embedding replacement. In this straightforward approach, pre-trained word vectors, such as Word2Vec[8], GloVe[9], and FastText[10], are employed to identify the closest words in vector space, replacing words in the original sentence. [11] used this technique to enhance the language model's ability to mitigate overfitting in downstream tasks.

Moreover, various methodologies, such as back-translation, have been explored in NLP. Back-translation involves machine translation to translate text backward, altering the sentence composition without changing its meaning. In work [12], it demonstrated the successful application of back-translation to extend an unlabeled model, achieving commendable results.

In addition to back-translation, an important focus within the data augmentation domain is the Easy Data Augmentation(EDA) concept[13]. EDA encompasses a systematic approach to augmenting training data by applying synonym replacement, random insertion, and paraphrasing techniques. These methods aim to introduce controlled variations into the original text, enhancing the model's performance. While some researchers have explored data augmentation by injecting random noise into the training data, our emphasis in this related work lies on the principles and applications of EDA. Adding noise, which involves random insertion, swapping, and deletion, has shown noteworthy re-

sults in diverse NLP tasks.

However, traditional data augmentation methods, particularly those persistently modifying and replacing words, have shown limitations and encountered bottlenecks. At this juncture, our proposed method comes to the forefront. Our approach signifies a paradigm shift from a word-centric focus to a more holistic sentence-level perspective.

Within the scope of our study, EDA assumes a central and pivotal role. We leverage the fundamental principles of EDA to amplify the performance of our specific model architecture. Importantly, our research introduces a novel perspective by incorporating the influence function to enhance the efficacy of EDA further. While EDA inherently contributes to substantial improvements in the performance of deep learning models, our method goes a step further by harnessing influence function. This synergy enables us to achieve heightened performance gains, effectively demonstrating the superior capabilities of our approach. Our work not only builds upon the documented successes of EDA in existing literature but also establishes the unique superiority of our method. By employing the influence function in conjunction with EDA, we showcase the adaptability and effectiveness of EDA and the additional performance boost afforded by our innovative integration.

2.2 Influence Function

XAI is pivotal in understanding and interpreting the intricate workings of machine learning models. The advent and deployment of large models have fueled interest in interpretable learning, prompting extensive research in methods that shed light on model behaviors observed in previous works[14], [15]. Previous efforts primarily aimed to quantify the importance of input data using mathematical techniques or intermediate model quantities, thereby explicating model behavior through such essential measures. Various methods, including attention mechanisms mentioned in [16], [17], perturbation analysis[18], and approximation techniques[19], have been employed to illustrate model predictions.

The influence function, a cornerstone in XAI, analyzes a model's sensitivity to individual sample predictions. This analytical approach allows researchers to pinpoint samples or features significantly impacting the model's predictions, thereby enhancing model interpretability. Initially applied in the early stages of machine learning for identifying crucial training samples [4], the influence function found its way into deep learning, particularly in computer vision. In [20], it was used to identify detrimental data and address issues related to data distribution instability. In the realm of NLP, [21] demonstrated the application of the influence function in uncovering biases in word embeddings, showcasing its versatility.

Crucially, in our research, the influence function plays a transformative role. Leveraging its capacity to discern the impact of individual samples on model predictions, we integrate the influence function into the model training process. By identifying and prioritizing crucial training samples, we harness the influence

function's superior capabilities to enhance model performance substantially. Incorporating the influence function in our experimental design provides a unique avenue for augmenting universal data augmentation techniques. Through the targeted selection of necessary training samples during model training, we witness a notable boost in performance, addressing bottlenecks encountered in conventional augmentation methods. The influence function, acting as a guiding force, not only enhances our understanding of the model's decision-making but also serves as a practical tool for achieving substantial performance gains.

In summary, the influence function, an essential element of XAI, enriches our understanding of model predictions and proves transformative in augmenting model training. By identifying and leveraging crucial training samples, its superior capabilities are harnessed to achieve significant performance improvements, thereby elevating the overall effectiveness of the model.

2.3 BERT

Pre-trained language models have received much attention in NLP since 2018, achieving excellent performance in several tasks. One of the best known is the BERT model, based on the transformer structure mentioned in [22], which captures rich language representations in the unsupervised pre-training phase and then achieves significant performance gains on specific tasks through fine-tuning. Using the Transformer encoder structure, the BERT model breaks the traditional limitation that text can only be modeled from left to right. It introduces bi-directional contextual information, i.e., the model uses both the left and right contexts of the text during the pre-training process. After the pre-training phase, the BERT model can be fine-tuned to perform well in various downstream tasks. In the fine-tuning phase, task-specific labeled data is linked to the pre-trained model, and then the model's parameters are fine-tuned through supervised training. The proposal of BERT started a revolution in the field of NLP, and its pre-training-fine-tuning framework not only set new performance records on several benchmark tasks but also inspired several subsequent pre-training models, such as the GPT family, Roberta, and XLNet mentioned in [23], among others. These models are all based, to varying degrees, on BERT and continue to drive the development of NLP technology.

2.4 Sentiment Analysis

Through work [24], we can get the definition of sentiment analysis. Sentiment analysis, which aims to recognize the sentiments or emotional states expressed in text, is essential in NLP. This task helps to gain insight into people's attitudes and emotional tendencies towards products, events, topics, etc. It has many applications in social media analysis, consumer reviews, and opinion monitoring. Generally, this task can be categorized into the following methods.

Traditional methods: Early research focused mainly on rule-based and dictionary-based methods in sentiment analysis. Researchers construct sentiment dictionaries and rules to identify sentiment tendencies in texts. However, these methods are often limited by vocabulary coverage and semantic understanding, making it difficult to deal with complex linguistic expressions and polysemous words.

Machine Learning Approaches: With the rise of machine learning techniques, sentiment analysis has gradually shifted from traditional methods to data-driven approaches. At this stage, researchers began to explore the use of machine learning algorithms such as Support Vector Machines (SVMs)[25], Plain Bayesian Classifiers and Random Forests to learn sentiment analysis models from labeled data automatically. Feature engineering is also a key aspect at this stage, where researchers try to extract meaningful features from text to use as inputs to the model.

Deep learning methods have achieved remarkable advancements in sentiment analysis tasks in recent years, as evidenced by notable breakthroughs in the field [26], [27]. The introduction of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) has enabled models to better capture contextual information between sentences and text. Models such as Long Short-Term Memory Networks (LSTMs)[28] and Gated Recurrent Units (GRUs)[29] have been widely applied to sentiment analysis of sequential data, effectively solving the problem of long-range dependency on text.

In the current landscape, developing large language models (LLMs) has become a standard in sentiment analysis. The experimentations in this study will align with this trend, opting for using the classic BERT model, showcasing the influence of cutting-edge large-scale models in advancing sentiment analysis capabilities.

Chapter 3

Methodology

To accomplish the sentiment analysis task, we choose the publicly available sentiment analysis dataset named The Stanford Sentiment Treebank(SST)[30], which is an expanded version of the MR (Movie Review) dataset. There are two versions of this SST dataset: SST -1, which contains five labels, and the other is the dataset we use, SST-2. The structure of our proposed method is illustrated in Figure3.1. Initially, we select the first 500 data instances from the dataset, leveraging them to train a model through fine-tuning with the pre-trained BERT model. Subsequently, some of the parameters of the fine-tuned model, the training data, and the data from the test set are fed into the influence function to compute the importance scores of all the training data. Following this, the training set's data is sorted according to the importance scores obtained, dividing it into the crucial and general data. In the final phase, data labeled as important is augmented by more multiples, like 24 times, when the data augmentation method is applied. In contrast, general data retains a standard augmentation multiplier, like 16 times. To validate the effectiveness of our proposed method, the new dataset generated by our proposed method is examined on the pre-trained BERT model.

The forthcoming sections will intricately describe each module within the method and the underlying mathematical frameworks. The initial module involves the training of sentiment analysis predictions utilizing BERT. The second module encompasses the application of the influence function, and the concluding module addresses the data augmentation methods pertinent to this experiment.

This structural framework will be presented in the subsequent section, demonstrating how our proposed method provides a discernible performance boost to the data augmentation approach. This comprehensive description outlines the sequence of operations, from initial data selection to the final validation. It explains the methodology's inner workings and potential impact on augmenting and enhancing the dataset.

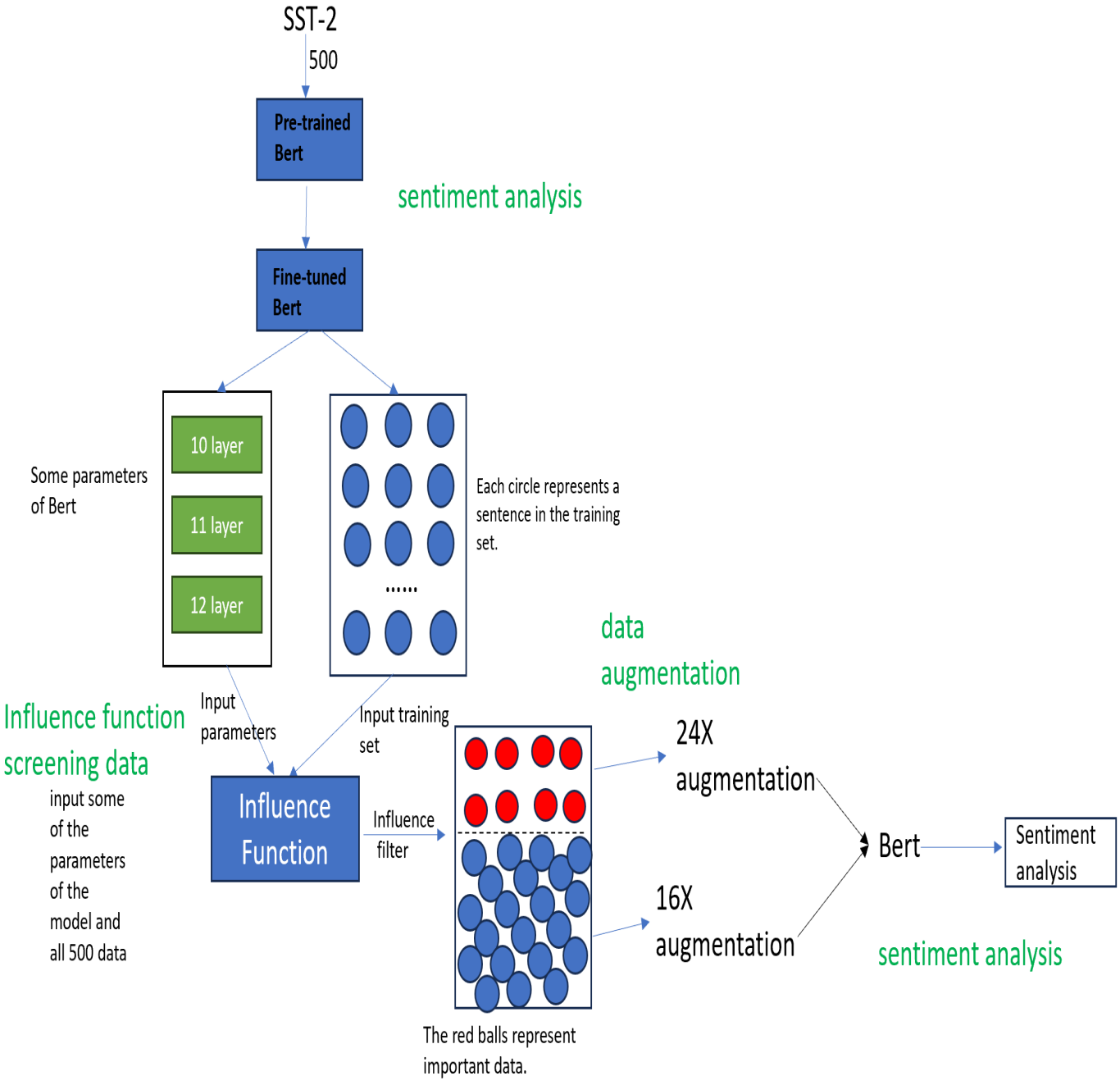

Figure 3.1: The structure of our proposed method.

3.1 Dataset

There are two variations of the SST(The Stanford Sentiment Treebank) dataset available: SST-1, including five labels, while the version we use, SST-2, is identical to the one we refer to. [31] trained the model with this dataset. This task is given the sentiment of a sentence as shown in Figure 3.2. From the figure, we can see that the first column is the sentence, and the second column is the label corresponding to the sentence. This dataset is a dataset with two categories of data. By the labels, we can see that all the sentences are categorized into two categories. 0 represents the sentences with negative emotions, and 1 illustrates those with positive emotions. Since we will perform a sentiment analysis task, this dataset fits the scenario we need. This dataset contains 67,350 training data and has 1821 test data to provide validation operations. Our experiments will be carried out based on this dataset to simulate scenarios with very little data to verify that our proposed method is effective in data augmentation methods. Therefore, we randomly selected 500 entries from the training dataset as training data for our experiments. Then, 100 more data were randomly selected from the training set as validation data. Finally, 100 randomly selected data from the test dataset are used as the test set.

```
do n't work in concert  0
the direction has a fluid , no-nonsense authority , and the performances by harris , phifer and cam ` ron seal the deal .   1
would have liked it more if it had just gone that one step further  1
it 's too harsh to work as a piece of storytelling ,    0
hawaiian shirt  1
```

Figure 3.2: Example of dataset

3.2 Model Structure

3.2.1 Sentiment Analysis Module

The sentiment analysis module can be achieved through various methods and models. The recent emergence of large language models has led to unparalleled advancements in this task. Additional effective methods and models include ELMO [32] and GPT [33]. However, they can not perform as well as BERT in sentiment analysis. ELMO is a bi-directional LSTM-based model that generates context-sensitive word vectors. However, unlike BERT, ELMO omits the Transformer architecture during training, which may impact its effectiveness in capturing long-distance dependencies. GPT operates as a unidirectional generative model, producing text through unidirectional autoregression. In sentiment analysis, a bidirectional encoding model holds an advantage as sentiment may be impacted by both preceding and subsequent text. Therefore, BERT is deemed the most appropriate for sentiment analysis; this experiment utilizes the BERT

model. BERT is a pre-trained model founded on the Transformer architecture, which has gained significant attention in the NLP field. The fundamental concept behind BERT is to comprehend contextual data of the text using the bi-directional encoder. Such capability facilitates the model's understanding of each word's context within a sentence. The training of BERT comprises two phases, As shown in Figure 3.4 mentioned in [1], namely, pre-training and fine-tuning. As mentioned, our proposed approach suggests using the pre-trained BERT model to fine-tune it on a randomly selected 500-unit training set from SST-2. This process will lead to the development of a fine-tuned model, which can be applied to the sentiment analysis dataset. Additional validation of the fine-tuned model's accuracy is then conducted to test the validity of the chosen validation method.

When using BERT for sentiment analysis tasks, the whole process of working with the model can be divided into the following steps: preprocessing, pre-training, and fine-tuning. Each step is described in detail below:

- Preprocessing: In the preprocessing stage, it is necessary to transform the raw text data into a suitable input format for the BERT model. BERT accepts tokens (usually words or subwords), each paired with a corresponding number.
- Pre-training: In the pre-training phase, the model learns an ability to make sense of the contextual representation of words by absorbing a large amount of the corpus. We use the pre-trained model in [22] for experiments
- Fine-tuning: During this phase, we employ the pre-trained BERT model to fine-tune it for the sentiment analysis task. The weights of the pre-training phase for BERT are initially loaded to ensure fast training. Subsequently, a classification layer must be added on top of BERT. In contrast, the last hidden layer of [CLS] serves as the terminal representation of the input sentence, as shown in Figure 3.3. Using the softmax activation function, we may add a fully connected layer at the end to map the CLS to sentiment categories. Our task involves binary categorization (positive, negative). We should also note that there are 12 encoder layers in the figure 3.3, and this parameter will be used later on.

This module ends with a sentiment classification for each sentence based on the scores obtained after a softmax function on the [CLS] labels in BERT. As shown in Figure 3.5, Each sentence will have a score X ($0<X<1$) and then be differentiated into positive and negative emotions based on the score. For example, If $X<0.5$, the sentence sentiment is negative. If $X>0.5$, the sentence sentiment is positive. Within the structural framework of the proposed method, as depicted in Figure 3.1, a sentiment analysis task is conducted using the fine-tuned model both at the commencement and conclusion of the method. It is crucial to note that the former represents the training stage, while the latter assesses accuracy using the fine-tuned model.

This sentiment analysis task serves as a critical benchmark for evaluating the superiority of our proposed method. The results will be elaborated upon in

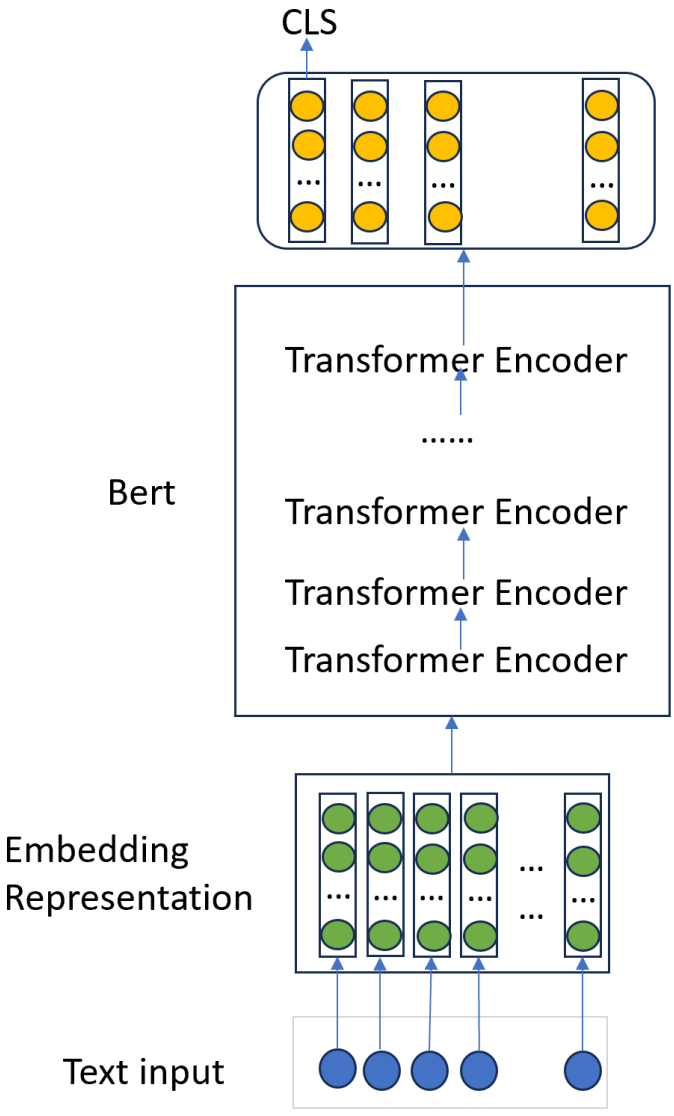

Figure 3.3: Overall fine-tuning procedures for BERT.

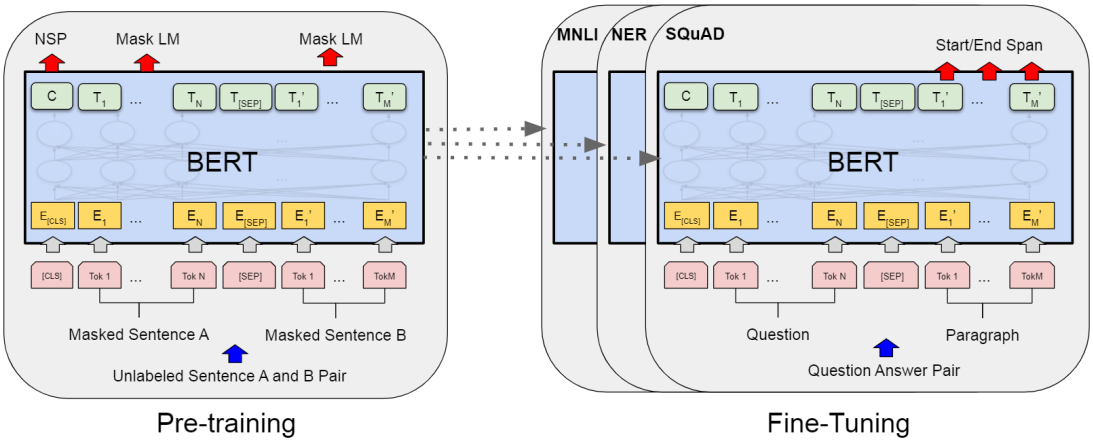

Figure 3.4: Overall procedures for BERT[1]

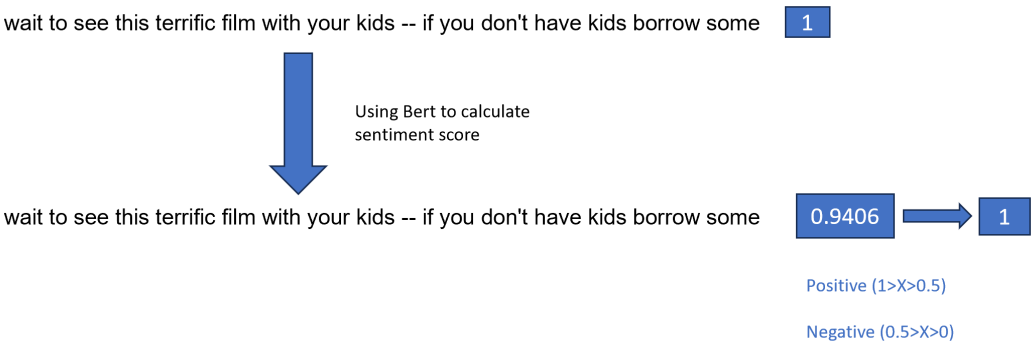

Figure 3.5: Example of sentiment analysis

the next section, showing how effectively our method captures and categorizes sentiment within sentences. This validation underscores the efficacy and distinct advantages of our proposed approach.

Moreover, an integral aspect is that after fine-tuning the BERT model on the sentiment analysis task dataset, specific parameters of the fine-tuned model are input into the subsequent module (influence function screening data) for computing importance scores. as shown in the beginning sentiment analysis module in Figure 3.1, This dual employment of the sentiment analysis module serves a dual purpose. The initial iteration is dedicated to obtaining the fine-tuned model, allowing the transfer of parameters to the influence function to calculate the essential importance scores our method requires. The final iteration of the sentiment analysis module is then utilized to validate the effectiveness of our proposed method. This iterative approach ensures the seamless integration of sentiment analysis, fine-tuning, and influence function calculations, thereby substantiating the efficacy of our methodology.

3.2.2 Influence Function module

The influence function gauges a model's sensitivity to each sample within the training data. It works by locally perturbing a piece of data in the training set and then measuring the importance of this perturbed data in the training set by observing how much the model's performance on the test set is affected after being trained on this perturbed training set. The influence function serves not only to identify anomalies in the training data but also to aid comprehension of the reliability of model predictions and how the removal of a sample affects such predictions. Implementing the influence function will also involve the choice of parameters of the model; for example, the model used in our experiments is the BERT model, which has 12 layers of encoder and the last layer of fully connected layers. How many parameters we chose, will we choose in the next section of the experiment? Next, let's discuss the basics of the influence function and mathematically demonstrate why the computation can be done without retraining the model.

Through work [34], we can explain the theory by deriving it as follows: Suppose we have n existing training samples Z_1, Z_2... ,Z_N, where $Z_i(x_i,y_i)$, L(Z, θ) denotes the loss of z trained under the theta parameter, where x represents the textual content of the training samples, while y represents the actual label. With the above definition, then, the loss function or empirical risk function can be expressed as R(θ):

$$R(\theta) = \frac{1}{n}\sum_{i=1}^{n} L(Z_i, \theta) \tag{3.1}$$

When we minimize the empirical risk loss, we can get the parameter at this point expressed by the following equation:

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} L(Z_i, \theta) \tag{3.2}$$

The influence function is used to observe the effect of a slight change in a specific data point (x_i) in the training dataset for a given model parameter theta. To utilize the influence function, we alter or remove a particular sample (Z_i) and observe the resultant change in theta. The purpose of this experiment is to demonstrate the effectiveness of the influence function by making a slight increase in x_i for Z_i. At this time, the parameter a is expressed as follows:

$$\hat{\theta}_{\epsilon,Z} = \arg\min_{\theta} \left(\frac{1}{n} \sum_{i=1}^{n} L(Z_i, \theta) + (Z, \theta) \right) \tag{3.3}$$

Looking at equations 3.1 and 3.3, we can see that we can substitute equation 3.1 into equation 3.3 to obtain:

$$\hat{\theta}_{\epsilon,Z} = \arg\min_{\theta} \left[R(\theta) + \epsilon L(Z, \theta) \right] \tag{3.4}$$

Define $\Delta\epsilon = \hat{\theta}_{\epsilon,Z} - \hat{\theta}$ to measure the amount of change in the parameter θ, again because θ is a consequence of $\arg\min R(\theta)$, independent of ϵ, So the following equation holds:

$$\frac{\mathrm{d}\hat{\theta}_{\epsilon,Z}}{\mathrm{d}\epsilon} = \frac{\mathrm{d}\Delta\epsilon}{\mathrm{d}\epsilon} \tag{3.5}$$

Since $\hat{\theta}_{\epsilon,Z}$ is an extreme value, we can obtain it by taking the derivative of $\arg\min_{\theta} \left[R(\theta) + \epsilon L(Z, \theta) \right]$ and making the derivative equal to zero: $\nabla R(\hat{\theta}_{\epsilon,Z}) + \epsilon \nabla L(Z, \hat{\theta}_{\epsilon,Z}) = 0$, when $\hat{\theta}_{\epsilon,Z} \longrightarrow \hat{\theta}$ expanding equation 3.5 can be done as a first-order Taylor's formula; that is, it is obtained by expanding $\hat{\theta}_{\epsilon,Z}$ in the neighborhood of $\hat{\theta}$:

$$\Delta\epsilon \approx \frac{1}{\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})} \cdot \left[\nabla R(\hat{\theta}_{\epsilon,Z}) + \epsilon \nabla L(z, \hat{\theta}) \right] \tag{3.6}$$

The following expression can be obtained by simplifying from equation 3.2 and reasoning:

$$\Delta\epsilon \approx -\Delta^2 R(\hat{\theta})^{-1} \cdot \Delta L(z, \hat{\theta})\epsilon \tag{3.7}$$

And $\Delta^2 R(\hat{\theta})$ is the Hessian matrix $H(\hat{\theta})$mentioned in [4]

By combining equations 3.5 and 3.7, the expression for the final influence function can be expressed as:

$$I_up, params(Z) = \frac{\mathrm{d}\hat{\theta}_{\epsilon,Z}}{\mathrm{d}\epsilon} = -H_{\hat{\theta}}^{-1} \nabla L(Z, \hat{\theta}) \tag{3.8}$$

We can observe from Equation 3.8 that the impact score of every training sample Z solely depends on $\hat{\theta}$. Therefore, there is no requirement to retrain the model,

and we can precisely calculate the score, signifying the importance of every training sample.

Through the intricate mathematical procedures detailed above, we acquire crucial importance scores for the training data during the training process. This accomplishment is a critical outcome and forms the core of our research endeavor. The series of mathematical operations provides profound insights into the significance of each data point and establishes a robust foundation for our study.

Following the acquisition of these importance scores, we gain a nuanced understanding of the relative importance of each data point. This signifies a shift from perceiving data as homogeneous to identifying critical factors. Subsequently, we systematically rank all training data based on their importance levels. This hierarchical ranking system enables us to establish rules, specifying which data points are paramount—whether within the top one hundred, two hundred, or three hundred. This ranking system offers a layered understanding of the data, empowering us to utilize its latent value purposefully.

Integrating this with the visual representation in Figure 3.1's influence function filtering module, a dynamic portrayal of our method's operational process emerges. The output of this module divides the original dataset into two distinct subsets. As evident from the colors in the figure, the initially trained data is represented in all blue. After the influence function's calculation and filtering, a portion of the data is identified as crucial (depicted in red). In contrast, the remaining data is considered general (retaining the original blue color). This division vividly illustrates our method's process, showcasing the identification of essential data through influence function analysis and the subsequent categorization into distinct subsets.

This pivotal step is an indispensable and fundamental component within our proposed methodology and serves as the starting point for detailed data analysis. Our research goes beyond the performance of the overall dataset; it delves into uncovering highlights and crucial data points, providing robust support for our methodology. This in-depth analysis informs subsequent experiments and model refinements, allowing us to tailor data processing strategies precisely and ultimately enhance the performance and practicality of our proposed approach.

3.2.3 Data Augmentation module

Nowadays, data augmentation methods are also developing rapidly in the field of NLP, in which substituting a certain proportion of words in a sentence for words with similar meanings by various methods is the primary method. Of course, there is also the back translation method. The classic EDA method is mentioned in surveys [35], [36], and so on, and they all praise the efficiency of this method. The idea is simple but can achieve excellent results. Therefore, I will use EDA in my experiments to validate my proposed method and explore the parameters in EDA that work best in our proposed method.

We implemented EDA to generate more diversified training samples by applying four primary operations on raw text data. As figure3.6[13] shows, the

four primary operations we employed are as follows:

- SR: Replacing synonyms is one of the earliest EDA procedures. To perform this task, a random word is chosen from the original text and switched out with a synonym extracted using NLP libraries such as NLTK and WordNet. Not only does this technique improve the diversity of the text, but it also leads to an enhanced comprehension of the surrounding context and meaning by the model.

- RI: Random insertion is another crucial operation in EDA. In this process, we arbitrarily choose a word and insert a randomly selected word into a random location in the original text. This inserted word can be selected from a thesaurus or randomly chosen in the text. This helps to extend the sentence's length and structure, thus enhancing the model's ability to adjust to various sizes and syntactic structures.

- RS: The random swap operation aims to increase the diversity of the sentence structure of the text. In this operation, we randomly select two words in the original text and swap their positions. This process simulates the rearrangement of words within the sentence and helps the model better understand the relationship between different words.

- RD: The random omission process emulates the lack of information in a text. With some probability, a word is randomly selected and then omitted from the original text. This process compels the model to deduce insufficient information, thereby enhancing model stability.

The operations conducted within our Data Augmentation Module serve as more than just mechanisms to introduce controlled variations into the original text—they form the cornerstone of our innovative augmentation approach. Specifically, our method, based on the insights gained from the influence function, identifies and seamlessly integrates influential data points into the EDA process.

In this crucial step, we employ the influence function to compute importance scores for each data point. Those identified as highly influential are then strategically chosen to undergo intensified augmentation within the EDA framework. This nuanced approach enables us to concentrate the augmentation efforts on data points deemed most critical by the influence function, enhancing the model's understanding of significant patterns and contexts. It involves meticulously exploring various parameters within the EDA framework to optimize its effectiveness with influence function. By creating a symbiotic relationship between EDA and influence function, our method aspires to achieve superior performance, showcasing its distinctive contributions to the field.

Adding a visual layer to this description, referring to Figure 3.1's data augmentation module, vividly illustrates the process. Following the previous module's data partitioning into crucial and general subsets, we observe distinct levels of augmentation applied to each subgroup. Notably, the numerical indicators provide a clear distinction—24X signifies a 24 times augmentation for crucial

Operation	Sentence
None	A sad, superior human comedy played out on the back roads of life
SR	A **lamentable**, superior human comedy played out on the **backward** road of life.
RI	A sad, superior human comedy played out on **funniness** the back roads of life..
RS	A sad, superior human comedy played out on **roads** back **the** of life.
RD	A sad, superior human out on the roads of life.

Figure 3.6: EDA's operations

data, while 16X indicates a 16 times augmentation for general data. This aligns seamlessly with the original EDA framework, further enhancing the richness of our augmentation process and emphasizing the differential treatment based on the identified importance of each subset.

In summary, the Data Augmentation Module in our study intricately combines the conventional power of EDA with sophisticated insights from the influence function. By intensifying augmentation on strategically identified influential data points, our methodology aims to diversify training samples and elevate the model's performance through a targeted focus on critical elements, thereby contributing to advancing our research objectives.

Chapter 4

Experiment

To illustrate the efficacy of our proposed approach, we carried out experiments in this chapter. This chapter details the experimental setup and steps, presents and analyzes the results, and explores variations in the results with different parameter configurations.

4.1 Experiment settings

4.1.1 Dataset

This experiment employs the SST-2 publicly available dataset to simulate data enhancement in limited data. Therefore, we will randomly select 500 and 100 data samples from the dataset as a training and validation set and 100 as a test set. The random sampling is assigned to compose a balanced training set, so we will randomly select 250 positive and 250 negative samples to ensure this dataset is reasonable. If any of the categories are too many or too few, it will significantly affect the overall results of the experiment.

4.1.2 Baseline

Because we propose filtering to improve the effectiveness of data augmentation methods, the Baseline of our experiment will be the result obtained with the same dataset under the optimal parameter configuration of EDA's data augmentation. The parameters of the Baseline method will be set precisely according to the optimal parameter settings as mentioned in the [13], and the new dataset will be formed by augmenting the dataset by 16 times to be trained and tested in BERT's model.

4.1.3 Experiment setup

We designed the experiment to include three modules: sentiment analysis, influence function filtering, and data augmentation. I'll detail the parameters of

each module separately below:

- Sentiment analysis module: In our experiments, we used the BERT-base model, which consists of a 12-layer Transformer Encoder structure. We tuned some key parameters, including setting the hidden layer dimension of each layer to 768, the number of self-attentive heads to 12, and the fully connected feed-forward network dimension of each layer to 3072. The size of the vocabulary table was 30,000. In addition, we conducted multiple rounds of training for our experiments, using a batch size of 128 for each round. The maximum sequence length was 50 for both the training and validation sets to ensure that the input sequences were short when processed. We used the Adam optimizer, put the learning rate to 1e-5, and set the random seed to 42 to ensure the repeatability of the results. Twenty training epochs were performed for better convergence.

- Influence function filtering module: Among the general parameters in this module, the batch size is set to 32, and the maximum sentence length of each data is set to 128. To ensure that the experiment can be repeated each time, we set the random seed to 42. There is also the critical hyperparameter that needs to be fixed, which is the parameter that we will study in our experiments, and it is the number of parameters that will be fed in. Since we are analyzing which data the BERT model values more during training, we need to decide which of the total 12 layers of transformer encoder in BERT to input. As mentioned in [37], the influence function can only become accurate if few parameters are input. Therefore, we will experiment with the last layer of BERT's encoder, the final two layers of BERT's encoder, and the final three layers of BERT's encoder to demonstrate our proposed method.

- Data augmentation module: The EDA's data augmentation method is used, the base method uses the default optimal parameters for that method, and the augmentation is done 16 times. The previous introduction to the EDA mentioned that the process has four operations: random substitution, deletion, random swap, and random insertion. All four parameters are set to 0.1, meaning, i.e., each word has a 10% probability of performing these operations.

  However, for the object, we want to experiment with, there are two differences with the Base method; one is that some of the data are augmented more, exactly which part of the data, in our experiment, we set a range called essential data, and will be sorted to find out the top 100, 200, and 300 data as the critical data for the experiment respectively. Then the second difference is that instead of being 16 times, the essential data is enhanced 20, 24, 28, and 32 times.

4.1.4 Evaluation metrics

In our experiments, since the experiments are conducted based on the sentiment analysis task. So, we chose four key evaluation metrics to comprehensively assess the model's performance and thus evaluate the effectiveness of our proposed method. In our experiments, we validate our proposed method's effectiveness through a sentiment analysis task and use publicly available sentiment analysis datasets. This means that our result is a categorization task. So, we used the traditional evaluation criteria for categorization tasks. These metrics are Accuracy, Recall, Precision, and F1 value. The result will be between 0 and 1, with closer to 1 representing better performance. The following is a brief description of each metric:

- Accuracy: Accuracy is the number of samples for which the model's predictions are correct for the entire data set as a percentage of the total data. This is an important metric, and the higher this metric is, the better the model performs in terms of overall performance.
- Recall: It measures the proportion of all actual positive cases the model successfully predicts. A high recall rate means the model has good coverage of positive examples.
- Precision: Precision measures the proportion of positive cases that the model predicts out of the total number of positive cases. A high precision means the model is more reliable in predicting positive cases.
- F1: The F1 value is the harmonic mean of recall and precision. A higher value means the model performs better when considering both recall and precision.

  While we evaluated the four key metrics of accuracy, recall, precision, and F1 value, we believe that accuracy and F1 value have a more integrated and holistic significance in evaluating model performance. Accuracy intuitively measures the model's prediction accuracy on the overall data, while the F1 value provides a more global performance evaluation after considering the balance of Recall and Precision. So, we will focus on accuracy and the F1 value in interpreting our experimental results better to understand the model's performance in sentiment analysis tasks and better explain the validity of our proposed approach.

4.2 Sentiment analysis

Figures 4.1 and 4.2 shows the training process of our proposed and Base methods. The x-axis indicates the number of epochs trained, while the y-axis indicates the accuracy and loss during training. We can see that the losses at the end of the training have been shallow, and the fluctuation of the model accuracy is reasonable.

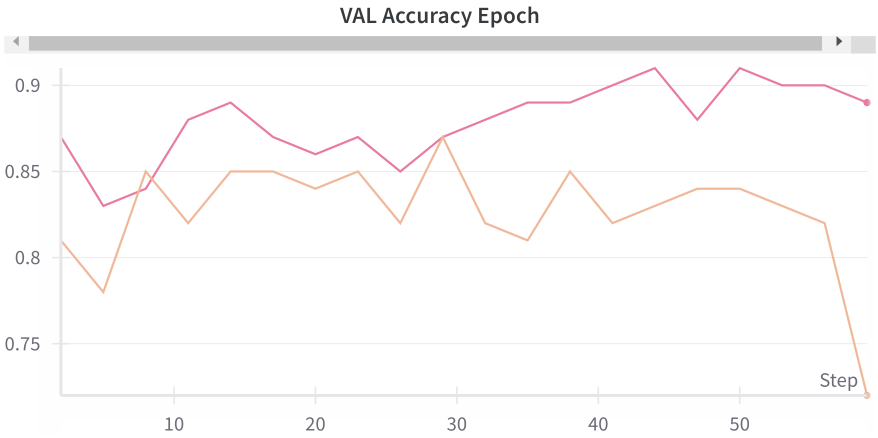

Figure 4.1: Validation accuracy of Sentiment Analysis Module

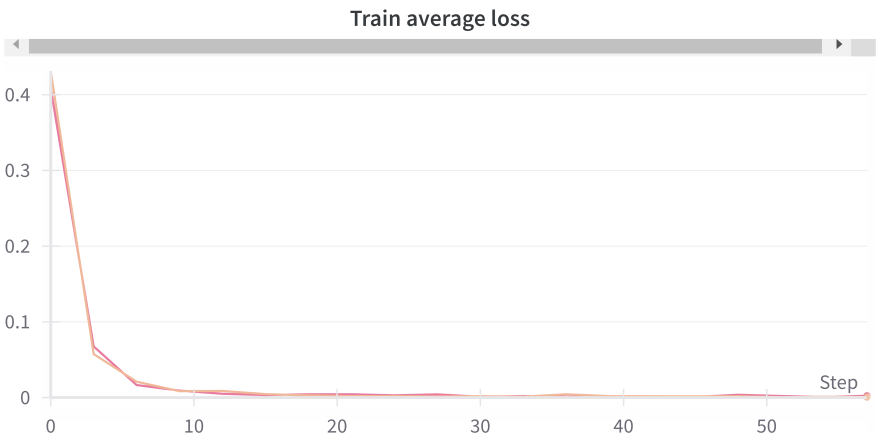

Figure 4.2: Train loss of Sentiment Analysis Module

The pink $line(2layer\_200)$ in the figure represents that I input the parameters of the last two layers of the encoder of the BERT model into the influence function. After calculating the importance scores of each data and then sorting, I set the first 200 of the 500 training data as the essential data to be augmented 24 times by EDA to form the training data to be involved in the training process. The other line represents the Base method, which uses the default optimal parameters of EDA to augment all the data 16 times and participate in the training process.

4.3 Important score of sentences

One of the many experiments is shown here as an example. The importance scores for each data calculated after we input the training data and some of the parameters of the fine-tuned BERT model into the influence function are shown.

```
seems to want to be a character study ,  0
5.579573036157697

stretched over the nearly 80-minute running time  0
4.273906245388003

the gloomy film noir veil  0
4.109010334764418
```

Figure 4.3: Examples of importance scores on negative samples

```
be incomprehensible to moviegoers not already clad in basic black  1
-1.258519442154423

feature to hit theaters since beauty and the beast 11 years ago  1
-1.228093632528826

the footage of the rappers at play and the prison interview with suge knight  1
-1.2015562102462358
```

Figure 4.4: Examples of importance scores on positive samples

An excellent absolute value of the importance score means a more critical role in the training process. Figure 4.4 shows the three most important pieces of data in the positive sample labeled 1. In comparison, Figure 4.3 shows the three most important pieces of data that the negative sample labeled 0 focused on during model training. We can see that each piece of data will have an importance score, as shown in the figure in general, and we can sort the positive

and negative samples in order of importance, respectively, and then filter out a certain number of 'the most important data.' Finally, because the scores are

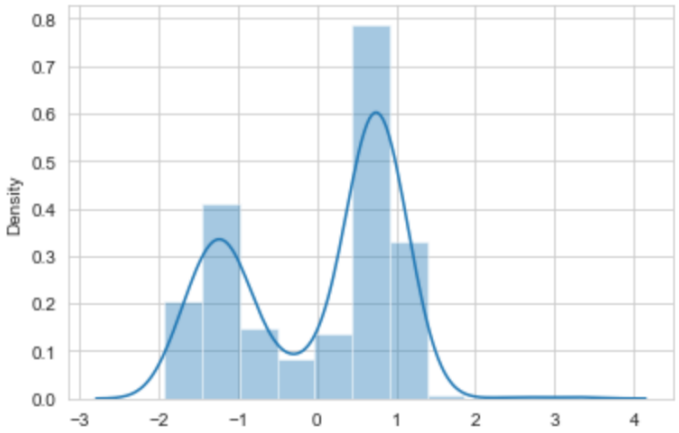

Figure 4.5: Distribution of importance scores for all 500 training data

discrete, we normalized the scores for the 500 pieces of data and showed the distribution of the importance scores for all the data in Figure 4.5. We can see that the positive and negative samples seem to conform to a normal distribution, meaning the critical data are still in the minority.

4.4 Result

In this subsection, we will show the entirety of our experiment. Our proposed method will be applied to EDA, and a comparison will be made with the technique using only EDA to verify the correctness of our proposed method. In the above section on the experimental setup, I have mentioned that there are three critical variables in our experiment, which are the number of parameters input into the influence function of the fine-tuned BERT, the range of the vital data, and the diversity of augmentation of the essential data.

This is because several experiments were conducted to verify that our results were not accidental. Then, the calculated mean and deviation values were written as results to be presented in a table. As shown in Table 4.1, we can see that the vertical axis results from combining the Base method according to the parameters mentioned earlier. The horizontal axis is the four metrics mentioned earlier in the section on evaluation methods. I want to make a detailed explanation of the different parameter combinations in the figure, as an example, where 1layer with 100 means that we get the importance score of all the data after inputting the last layer of encoder parameters in the fine-tuned Bert model into the influence function to participate in the calculation. Then, all the data are ranked according to the importance score, and the top 100 data

	Accuracy	Precision	Recall	F1
EDA(Base)	84 ± 2.51	82.78 ± 4.40	87.89 ± 1.83	84.89 ± 3.13
1layer with 100	83.66 ± 2.51	81.18 ± 3.55	89.52 ± 6.41	85.04 ± 3.45
1layer with 200	84.27 ± 2.00	85.05 ± 4.08	85.13 ± 8.64	84.96 ± 3.06
1layer with 300	81.33 ± 3.05	80.68 ± 4.12	84.46 ± 6.08	82.45 ± 4.08
2layer with 100	$\mathbf{86 \pm 2.64}$	84.36 ± 4.47	88.37 ± 4.56	$\mathbf{86.75 \pm 3.31}$
2layer with 200	$\mathbf{86.67 \pm 3.78}$	85.32 ± 4.43	89.61 ± 4.62	$\mathbf{87.49 \pm 4.31}$
2layer with 300	84.67 ± 2.31	83.34 ± 3.31	88.37 ± 4.1	85.73 ± 2.92
3layer with 100	85 ± 3	84.07 ± 5.55	87.89 ± 1.83	85.94 ± 3.39
3layer with 200	$\mathbf{86 \pm 1}$	84.98 ± 0.89	88.94 ± 5.1	$\mathbf{86.81 \pm 2.32}$
3layer with 300	85 ± 2.64	83.26 ± 2.36	88.94 ± 5.48	85.98 ± 3.69

Table 4.1: Experiments on each set of parameters

are selected as important data for more augmentation. In the case of 2layer, the last two layers of encoder parameters are input into the influence function, while 3layer means that the final three layers of encoder parameters are input. Here, we have set to augment all the critical data 24 times while increasing the remaining data by 16 times, as in the Base method. Then, because this experiment has three variables, we will augment the critical data in this table 24 times. Then, we will keep one variable fixed to compare the change of the other variable. The experiment here is to test the validity of our proposed method and explore under what parameter our proposed filtering structure can improve the EDA. Why did we only experiment within three layers of the encoder? That is because we have seen a study about the influence function in [37], which shows that approximation is needed in the mathematical calculation process because of the influence function. Therefore, the more parameters we input, the more unstable the result will be. The paper shows that the performance is relatively better and more stable when the parameters are input within three layers.

4.4.1 The effectiveness of our approach

From Table 4.1, we can evaluate and analyze our method through the results. According to our previous section's description of the experimental evaluation criteria, we will prioritize comparing these two metrics to judge our method. Accuracy and F1 can better indicate the model's performance. Therefore, we also introduce Figures 4.6 and 4.7 to show only these two critical metrics separately to provide a more effective and intuitive presentation of our results. We focus on the first part of Table 4.1, and we show Figures 4.6 and 4.7. The horizontal dotted line shows the accuracy obtained by the Baseline method.

First, we focus on the effect of the number of input layers; in the case of using only one layer, i.e., only the last layer of the encoder parameter is entered into the influence function, the result is not very good; selecting the first 200 data as the critical data in the performance of The performance improvement

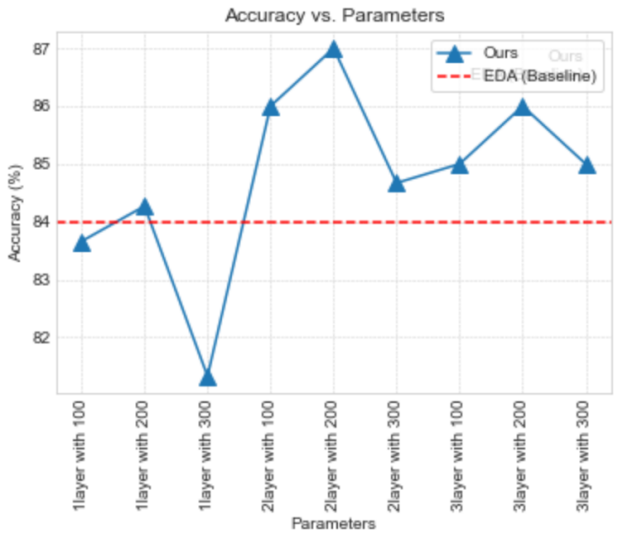

Figure 4.6: accuracy of different parameter combinations

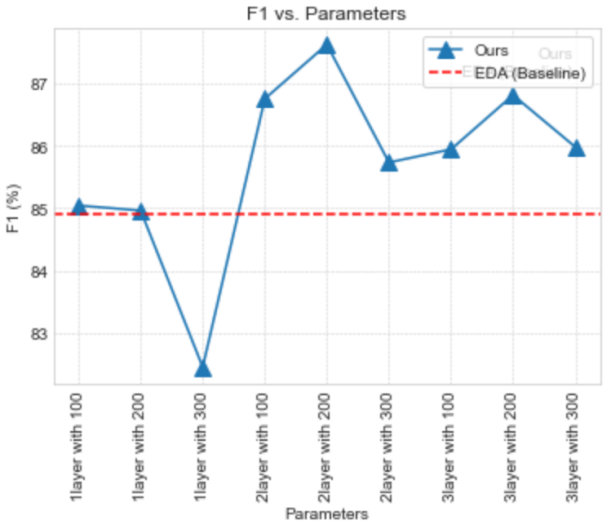

Figure 4.7: F1 score of different parameter combinations

of choosing the first 200 data as essential data is minimal compared to Baseline. In contrast, the performance of selecting the first 100 as crucial data and the first 300 as critical data is reduced. Subsequently, we can see in the remainder of the table that a significant improvement is obtained when using 2layer, i.e., inputting the parameters of the last two sides of the encoder into the influence function and setting the first 200 data as the critical data, Baseline's accuracy and F1 score are 84 and 84.89, respectively, but after using our method, they can reach 86.67 and 87.49, which are improved by 2.67 and 2.6 respectively. This is a nice boost and the best parameter combinations we have experimented with.

We not only focus on the results but also take the Deviation of the results into account, as shown in Figures 4.8 and 4.9,

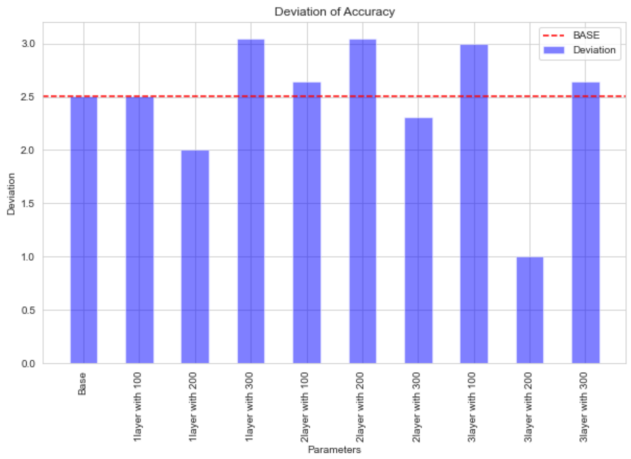

Figure 4.8: Deviation of the accuracy results for each parameter combination

we calculated the Deviation of the experimental results obtained for each set of parameters. Deviation implies the stability of the values and is as small as possible. We can only see from the data that even the Deviation of the Baseline method is not low; it reaches 2.51 and 3.13 for the accuracy and F1 scores. I think this is probably because each training data set is randomly filtered from the SST-2 dataset, and the data distribution will differ in each set. The length of the sentences may also be different. The length of the sentences may also be different, thus bringing bias. Combining the table and the figure, we can see that the Deviation of the results obtained by combining layer 2 with 200 parameters is slightly higher than the Baseline. In comparison, the results obtained by combining three layers with 200 parameters are more stable than the Baseline, which is also surprising.

In this subsection, we have analyzed the effectiveness of our method concerning the experimental results, explaining and justifying that our method is indeed effective. However, the analysis in this subsection is based on the experiments conducted by fixing the multiplicity of augmentation at 24 times in our three critical parameters, and in the following subsection, we will also an-

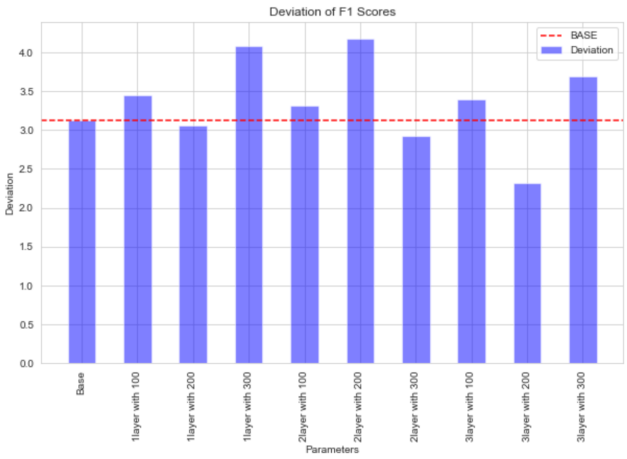

Figure 4.9: Deviation of the F1 score results for each parameter combination

alyze the effects of different multiplicities of augmentation. This way, we can find our proposed method's most suitable parameters for EDA, a vital data augmentation method.

4.4.2 Comparison of different augmentation multiples

Previously, we mentioned three critical parameters of our proposed method, two of which we studied in the previous subsection: the number of input parameters and defining the range of essential data. In this subsection, we will experiment to find the most suitable parameters for EDA using our method. This is because the multiplicity of augmentation is a critical parameter for the data augmentation method. Too much data augmentation may lead to overfitting of the model on the training set, especially when the generated augmented samples are highly similar to the original samples. The model may be overfitted to the noise in the training set, and the generalization ability may decrease. Moreover, large-scale data augmentation increases the computational and storage overhead during training. This may lead to longer training time while requiring more hardware resources. However, too little data augmentation won't work either; if the augmentation times are too small, the model may not be able to learn the diversity of the data adequately, resulting in poor generalization performance when faced with unseen samples. With a small dataset, more data augmentation may prevent the model from prematurely overfitting the training data, limiting its effectiveness in real-world scenarios. The context of our study is the situation when faced with only small datasets.

Therefore, we want to validate not only the effectiveness of our method but also how many times we should augment the part of the data that we recognize as 'important data' to be a good performer when using our method on EDA. In Table 4.1, we see that the most significant improvement is in the case of

the parameter combination 2layer with 200. Still, the Deviation of its accuracy and F1 scores are 3.78 and 4.31, respectively, while the Deviation of Baseline's method is 2.51 and 3.13 for these two metrics, respectively. In contrast, in the case of the parameter combination, In the case of 3layer with 200, we can see that there is still a good improvement. The Deviation of each index of this set of parameters is lower than that of the baseline method. The Deviation of its accuracy is only 1, which is much lower than that of the corresponding index of the baseline method. For Deviation, the lower the better; the more insufficient means, the more stable the result. However, the combination of 2layer with 200 gives the best results, so we will look for the best enhancement multiplier based on these two sets of parameters to find the best combination of these three important parameters.

	Accuracy	Precision	Recall	F1
20X	$\mathbf{86 \pm 3.61}$	85.04 ± 4.52	87.7 ± 4.78	$\mathbf{86.06 \pm 4.74}$
24X	$\mathbf{86.67 \pm 3.78}$	85.32 ± 4.43	89.61 ± 4.62	$\mathbf{87.49 \pm 4.31}$
28X	$\mathbf{85.33 \pm 2.52}$	83.38 ± 3.32	89.7 ± 4.97	$\mathbf{86.38 \pm 3.43}$
32X	$\mathbf{85.67 \pm 3.06}$	81.51 ± 0.83	90.46 ± 2.51	$\mathbf{86.8 \pm 3.47}$
Baseline(EDA)	84 ± 2.51	82.78 ± 4.40	87.89 ± 1.83	84.89 ± 3.13

Table 4.2: Different augmentation multiples of 2layer with 200.

Table 4.2 shows that inputting the parameters of the last two layers of the encoder into the influence function, the top two hundred importance scores in the 500 pieces of training data are classified as essential data. This part of the data is augmented 20, 24, 28, and 32 times, and different multiplications boost them to see the results of the final model test. The four indicators still evaluate the results.

By combining Figure 4.10 and Figure 4.11, we can very intuitively see that in the case of 24X augmentation of essential data, the four metrics accuracy, precision, recall, and F1 scores are the most obvious ones to be improved, the rest of the augmentation multiples are still improved, and the worst one, 28X, can still be improved in accuracy by 1.33. Therefore, this can illustrate the effectiveness of our validity of the proposed method. When exploring the best parameter combinations, a more precise answer can also be obtained through the combined presentation of tables and pictures. However, the Deviation of the results of this parameter combination could be more reasonable because accuracy and F1 scores are the evaluation metrics we are more concerned about. Through Figure 4.12 and Figure 4.13, we can see that all the results are worse than Baseline's method, except for the group of 28X, which is almost the same as Baseline's method. However, the instability here doesn't mean there's no performance improvement for a particular data set. By analyzing the results of each set of data in the experiments, we find that using this set of parameters (2layer with 200) can sometimes improve the performance by five percentage points. In comparison, sometimes it can only improve the performance by two

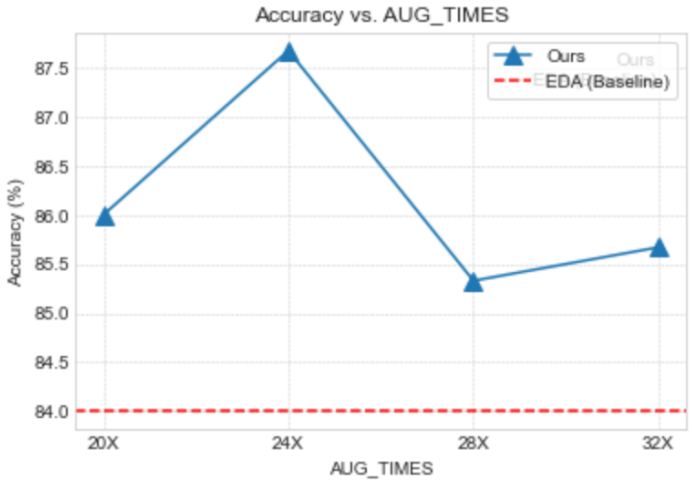

Figure 4.10: Accuracy of '2layer with 200' for different multiples

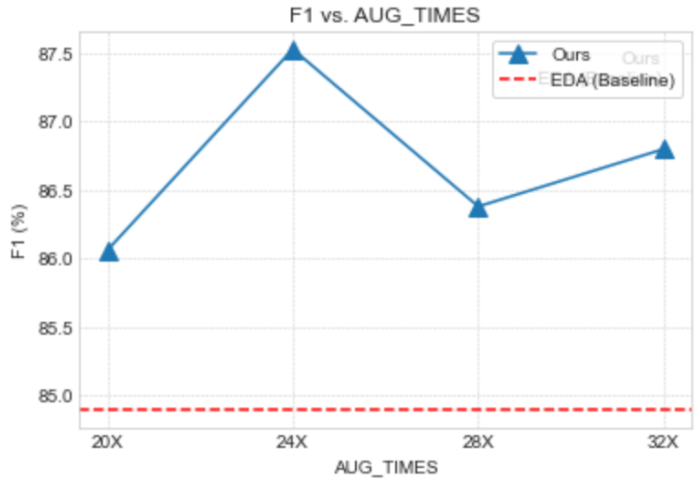

Figure 4.11: F1 score of '2layer with 200' for different multiples

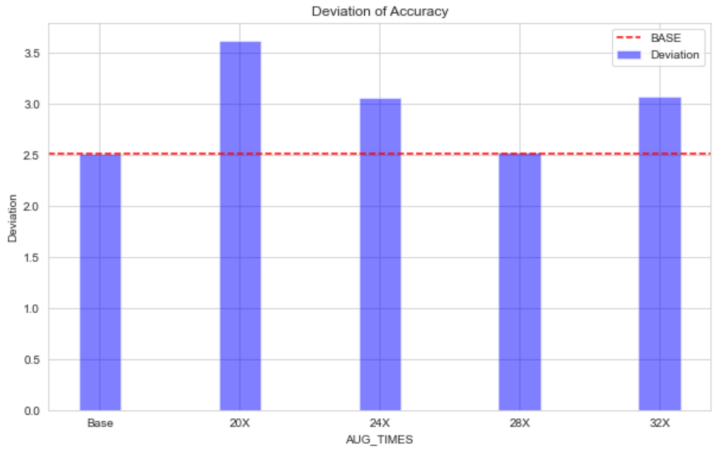

Figure 4.12: Deviation of the accuracy of '2layer with 200'

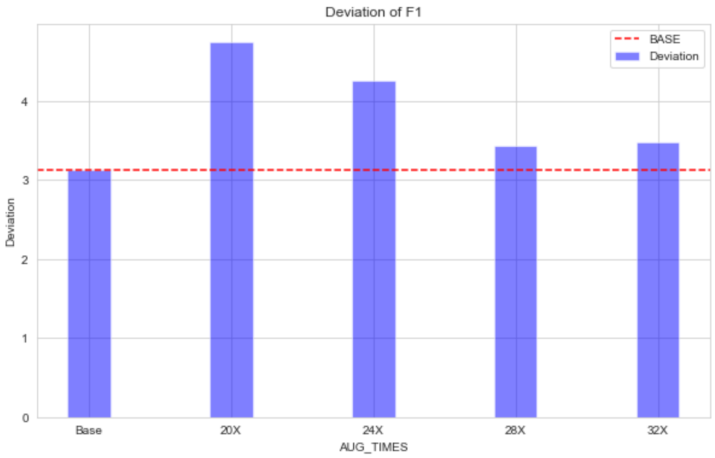

Figure 4.13: Deviation of the F1 score of '2layer with 200'

percentage points. The degree of improvement fluctuates, implying that this set of parameters can sometimes enhance a lot of performance when dealing with different data sets. This means that this set of parameters can sometimes be improved and sometimes only a little when dealing with different data sets. This means that this set of parameters can sometimes improve a lot and sometimes only a little when dealing with other data sets, but it will constantly improve the performance of the baseline method.

We did this experiment for 2layer with 200 and the parameter combination of 3layer with 200 because the experimental results for this set of parameters had a lower deviation. We also wanted to test whether this set of parameters would be optimal for EDA.

	Accuracy	Precision	Recall	F1
20X	$\mathbf{83.66 \pm 2.52}$	81.40 ± 4.44	88.13 ± 3.58	$\mathbf{84.54 \pm 2.28}$
24X	$\mathbf{86 \pm 1}$	84.98 ± 0.89	88.94 ± 5.1	$\mathbf{86.81 \pm 2.32}$
28X	$\mathbf{84 \pm 2}$	83.23 ± 2.89	86.05 ± 10.08	$\mathbf{84.29 \pm 3.46}$
32X	$\mathbf{85 \pm 1.73}$	83.23 ± 1.85	88.13 ± 2.19	$\mathbf{85.60 \pm 1.86}$
Baseline(EDA)	84 ± 2.51	82.78 ± 4.40	87.89 ± 1.83	84.89 ± 3.13

Table 4.3: Different augmentation multiples of 3layer with 200.

Table 4.3 shows the results of the experiments, which were conducted with the last three layers of the encoder parameters entered into the influence function, setting how many times the important data was enhanced if the data ranked in the top 200 according to the importance scores out of the 500 training data was the important data. As in the previous experiments, we focus more

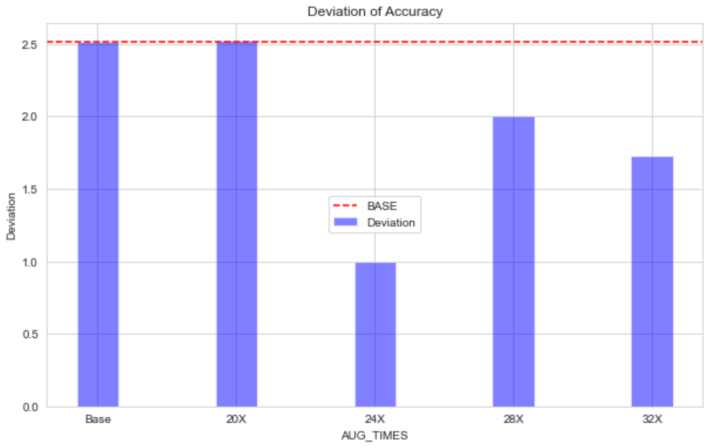

Figure 4.14: Deviation of the accuracy of '3layer with 200'

on the accuracy and F1 score among the four outcome metrics, and thus, we

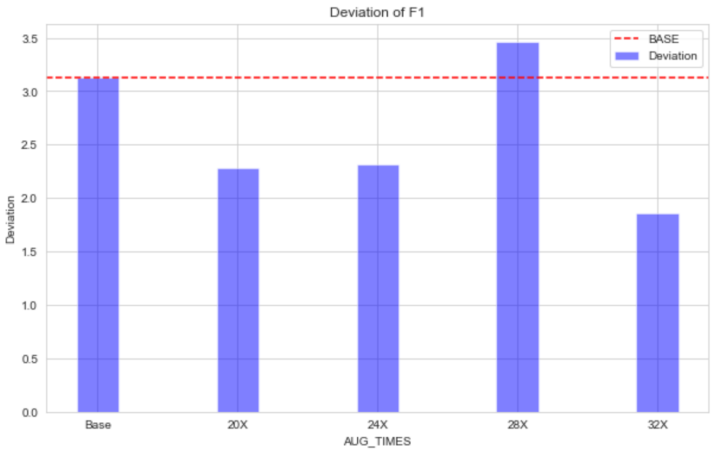

Figure 4.15: Deviation of the F1 score of '3layer with 200'

introduce Figures 4.14 and Figures 4.15. With these two figures and a table, we can see the advantage of this set of parameters, which is that, as we suspected, they both indeed have lower Deviation than the Baseline method, which means that They will have a more stable augmentation effect across different datasets. However, we can see the shortcomings of this set of parameters through Figures

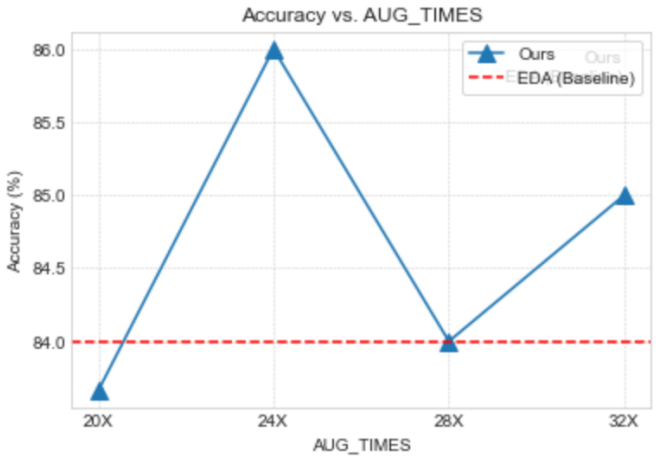

Figure 4.16: Accuracy of '3layer with 200' for different multiples

4.16 and Figures 4.17. Only the augmentation of 24X performs well; the rest of the augmentation seems to need to bring better performance, and even the performance of 20X augmentation is worse than that of the Baseline method. Moreover, compared to the result of using 2layer with 200, the performance of

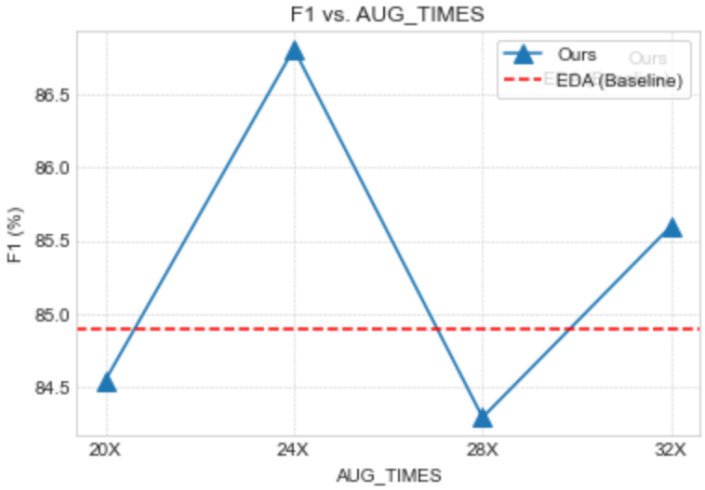

Figure 4.17: F1 score of '3layer with 200' for different multiples

3layer with 300 is worse under different augmentation multiples.

After conducting and analyzing our experiments, we have accomplished the two questions we posed in this chapter that we wanted to explore. The first is that our method can indeed improve the effectiveness of data enhancement because EDA is an outstanding representative of data enhancement methods and is also in line with the idea of the majority of data augmentation, which is the addition, deletion, exchange, and modification of words in a sentence. Therefore, if we can validate the success of our method on EDA, it will show its effectiveness. The second one is about exploring the optimal parameters for using our proposed method on EDA-based data augmentation methods. Our results described in the previous section show that the best results are achieved by enhancing the data labeled as important data 24 times and keeping the remaining data augmentation at the same augmentation multiple as in the Baseline method. However, there is a trade-off between deviation and performance results in the discussion about choosing the last two or three layers' parameters. Still, even though the Deviation of the results of 2layer with 200 is more significant, their different augmentation multiplicity can improve the original method. On the contrary, the results of the 3layer with 200 On the other hand, 3layer with 200 is not a very good combination of parameters, but it is stable. Therefore, we can think that for our proposed method, we choose to input the last two layers or three layers of encoder parameters of BERT into the influence function for the calculation of the importance score and then sort the training data according to the importance score of the first 40% of the data, in our experiment is the first 200 data of the 500 pieces of training data. This is also in line with the distribution of importance scores of all the data we showed in Section 4.3, where the critical data are in the minority. Therefore, the top forty percent

of the data are recognizable as essential. The best parameters described here are applied to EDA to improve performance in EDA as a data augmentation approach.

Chapter 5

Conclusion and Discussion

This thesis proposes an effective method to improve data augmentation effectiveness. In this paper, we introduce the two critical areas of XAI and data augmentation and point out areas that current data augmentation methods fail to focus on. Thanks to XAI's development, we have considered combining these two areas to make data enhancement methods more effective. Specifically, we wanted to explain the model's behavior through an influence function so that we could filter the data from it and then create an influence-guided filter to identify the critical data, which would make the model pay more attention to that part of the data. This behavior manifests by having the data augmentation method augment the critical data several times so the resulting dataset contains more meaningful data. The model then focuses more on the essential data in the dataset generated with our method because there are more meaningful data. Our experiments are conducted by randomly sampling 500 pieces of data in a large dataset as a training set to simulate the scarcity of data resources. The experiments are conducted in two parts. The first part tests the effectiveness of our method. The effectiveness of our proposed method is found by conducting experiments with different combinations of parameters. The second part is to find the best parameters for using our proposed EDA method. In conclusion, the presentation of the related work and principles of our method and the careful analysis of the whole experiment also show the validity and reliability of our proposed method.

This paper's contributions can be summarized as follows:

- Combining the fields of XAI and data augmentation gives LLMs better performance on sentiment analysis tasks.
- Proposes impact-guided filters that can effectively improve classical data augmentation methods such as EDA.
- Optimal parameters regarding the use of this method on EDA are investigated.

In future work, other large-scale pre-trained models besides BERT can be explored in our proposed method. Secondly, in our experiments, we have only computed the final three layers of encoder parameters of BERT's model; in future research work, it is possible to choose to input more parameters of BERT's or other big models into the influence function to explore whether it will give better results and thus improve the data augmentation method to a greater extent.

Bibliography

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[3] Jason Wang, Luis Perez, et al. The effectiveness of data augmentation in image classification using deep learning. *Convolutional Neural Networks Vis. Recognit*, 11(2017):1–8, 2017.

[4] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

[5] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

[6] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

[7] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[8] Kenneth Ward Church. Word2vec. *Natural Language Engineering*, 23(1):155–162, 2017.

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

[10] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[11] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*, 2019.

[12] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33:6256–6268, 2020.

[13] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*, 2019.

[14] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. *arXiv preprint arXiv:2005.04118*, 2020.

[15] Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031, 2021.

[16] Sarthak Jain and Byron C Wallace. Attention is not explanation. *arXiv preprint arXiv:1902.10186*, 2019.

[17] Ruiqi Zhong, Steven Shao, and Kathleen McKeown. Fine-grained sentiment analysis with faithful attention. *arXiv preprint arXiv:1908.06870*, 2019.

[18] Jiwei Li, Will Monroe, and Dan Jurafsky. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*, 2016.

[19] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[20] Zifeng Wang, Hong Zhu, Zhenhua Dong, Xiuqiang He, and Shao-Lun Huang. Less is better: Unweighted data subsampling via influence function. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[21] Marc-Etienne Brunet, Colleen Alkalay-Houlihan, Ashton Anderson, and Richard Zemel. Understanding the origins of bias in word embeddings. In *International conference on machine learning*, pages 803–811. PMLR, 2019.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[23] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[24] J Kaur, SS Sehra, and SK Sehra. A systematic literature review of sentiment analysis techniques. *International Journal of Computer Sciences and Engineering*, 5(4):22–28, 2017.

[25] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226, 2006.

[26] Cicero Dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers*, pages 69–78, 2014.

[27] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[28] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[29] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

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

[31] Maryam Heidari and James H Jones. Using bert to extract topic-independent sentiment features for social media bot detection. In *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0542–0547. IEEE, 2020.

[32] Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymanowska, Izabela Stefaniak, Michal Jarkiewicz, and Lukasz Okruszek. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research*, 304:114135, 2021.

[33] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[34] Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

[35] Connor Shorten, Taghi M Khoshgoftaar, and Borko Furht. Text data augmentation for deep learning. *Journal of big Data*, 8:1–34, 2021.

[36] Markus Bayer, Marc-André Kaufhold, and Christian Reuter. A survey on data augmentation for text classification. *ACM Computing Surveys*, 55(7):1–39, 2022.

[37] Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*, 2020.