


12-2022

## Movie Reviews Sentiment Analysis Using BERT

Gibson Nkhata  
*University of Arkansas, Fayetteville*

Follow this and additional works at: <https://scholarworks.uark.edu/etd>

 Part of the [Artificial Intelligence and Robotics Commons](#), [Graphics and Human Computer Interfaces Commons](#), and the [Human Ecology Commons](#)

---

### Citation

Nkhata, G. (2022). Movie Reviews Sentiment Analysis Using BERT. *Graduate Theses and Dissertations*  
Retrieved from <https://scholarworks.uark.edu/etd/4768>

This Thesis is brought to you for free and open access by ScholarWorks@UARK. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of ScholarWorks@UARK. For more information, please contact [scholar@uark.edu](mailto:scholar@uark.edu), [uarepos@uark.edu](mailto:uarepos@uark.edu).

# Movie Reviews Sentiment Analysis Using BERT

A thesis submitted in partial fulfillment  
of the requirements for the degree of  
Master of Science in Computer Science

by

Gibson Nkhata  
Mzuzu University  
Bachelor of Science in Information and Communication Technology, 2018

December 2022  
University of Arkansas

This thesis is approved for recommendation to the Graduate Council.

---

Susan Gauch, Ph.D.  
Thesis Director

---

Justin Zhan, Ph.D.  
Committee member

---

Ukash Nakarmi, Ph.D.  
Committee member

---

YanJun Pan, Ph.D.  
Committee member

## ABSTRACT

Sentiment analysis (SA) or opinion mining is analysis of emotions and opinions from texts. It is one of the active research areas in Natural Language Processing (NLP). Various approaches have been deployed in the literature to address the problem. These techniques devise complex and sophisticated frameworks in order to attain optimal accuracy with their focus on polarity classification or binary classification. In this paper, we aim to fine-tune BERT in a simple but robust approach for movie reviews sentiment analysis to provide better accuracy than state-of-the-art (SOTA) methods. We start by conducting sentiment classification for every review, followed by computing overall sentiment polarity for all the reviews. Both polarity classification and fine-grained classification or multi-scale sentiment distribution are implemented and tested on benchmark datasets in our work. To optimally adapt BERT for sentiment classification, we concatenate it with a Bidirectional LSTM (BiLSTM) layer. We also implemented and evaluated some accuracy improvement techniques including **S**ynthetic **M**inority **O**ver-sampling **T**Echnique (SMOTE) and NLP Augmenter (NLPAUG) to improve the model for prediction of multi-scale sentiment distribution. We found that including NLPAUG improved accuracy, however SMOTE did not work well. Lastly, a heuristic algorithm is applied to compute overall polarity of predicted reviews from the model output vector. We call our model BERT+BiLSTM-SA, where SA stands for Sentiment Analysis. Our best-performing approach comprises BERT and BiLSTM on binary, three-class, and four-class sentiment classifications, and SMOTE augmentation, in addition to BERT and BiLSTM, on five-class sentiment classification. Our approach performs at par with SOTA techniques on both classifications. For example, on binary classification, we obtain 97.67% accuracy, while the best performing SOTA model, NB-weighted-BON+dv-

cosine, has 97.40% accuracy on the popular IMDB dataset. The baseline, Entailment as Few-Shot Learners (EFL), is outperformed on this task by 1.30%. On the other hand, for five-class classification on SST-5, the best SOTA model, RoBERTa+large+Self-explaining, has 55.5% accuracy, while we obtain 59.48% accuracy. We outperform the baseline on this task, BERT-large, by 3.6%.

## DEDICATION

This thesis is dedicated to my late mother, Lincy Pyera Nyavizala Mphande. May her soul  
continue resting in peace.

## ACKNOWLEDGEMENTS

Firstly, I am grateful to my advisor Dr. S. Gauch for her valuable help towards the completion of this work. I am also thankful to my former advisor, Dr J. Zhan, and Dr. usman Anjum for their initial support towards this thesis.

I am so grateful to the Institute of International Education/Agricultural Transformation Initiative (IIE/ATI) scholarship programme for making it possible for me to study at the University of Arkansas. I also thank the Data Analytics that are Robust and Trusted (DART) project for supporting this work through the University of Arkansas CSCE department.

Last but not least, I thank all who in one way or another contributed in the completion of this thesis, your efforts are not taken for granted.

## TABLE OF CONTENTS

1	Introduction . . . . .	1
2	Related Work . . . . .	4
2.1	Sentiment Analysis . . . . .	4
2.2	Deep learning . . . . .	5
2.2.1	Deep Learning on Sentiment Analysis . . . . .	5
2.2.2	Deep Learning on Movie Reviews Sentiment Analysis . . . . .	6
2.3	BERT . . . . .	8
2.3.1	BERT and Sentiment Analysis . . . . .	8
2.3.2	BERT and Movie Reviews Sentiment Analysis . . . . .	9
3	Our Approach . . . . .	11
3.1	Methodology . . . . .	11
3.1.1	Sentiment Analysis . . . . .	11
3.1.2	BERT . . . . .	11
3.1.3	Fine-tuning BERT with BiLSTM . . . . .	13
3.1.4	Classification . . . . .	15
3.1.5	Accuracy Improvement Approaches . . . . .	17
3.1.6	Overall polarity . . . . .	19
3.1.7	Overview of Our Work . . . . .	23
4	Experiments . . . . .	25
4.1	Datasets . . . . .	25
4.1.1	IMDb movie reviews . . . . .	25
4.1.2	SST . . . . .	26
4.1.3	MR Movie Reviews . . . . .	27
4.1.4	Amazon Product Data dataset . . . . .	27
4.2	Data preprocessing . . . . .	28
4.3	Experimental settings . . . . .	29
4.4	Evaluation Metrics . . . . .	29
4.5	Results . . . . .	30
4.5.1	Evaluation of Goal 1 . . . . .	30

4.5.2	Evaluation of Goal 2 . . . . .	32
5	Conclusion . . . . .	34
5.1	Conclusion . . . . .	34
5.2	Future Work . . . . .	35
	Bibliography . . . . .	36
A	All Publications Submitted . . . . .	41



## LIST OF FIGURES

Figure 3.1:	Simplified diagram of BERT . . . . .	13
Figure 3.2:	Fine-tuning part of BERT with BiLSTM . . . . .	15
Figure 3.3:	Binary Tree Splitting . . . . .	18
Figure 3.4:	Overview of our work . . . . .	24

## LIST OF TABLES

Table 4.1: Accuracy (%) Comparisons of Models on Benchmark Datasets for Binary Classification . . . . .	31
Table 4.2: Accuracy (%) Comparisons for Three and Four Class Classification on IMDD . . . . .	31
Table 4.3: Accuracy (%) Comparisons of Models on Benchmark Datasets for Five Class Classification . . . . .	31
Table 4.4: Accuracy (%) of Our Model with Accuracy Improvement Techniques on SST-5 . . . . .	32
Table 4.5: Overall Polarity Computation on All the Datasets . . . . .	33

## 1 Introduction

Sentiment Analysis aims to determine the polarity of emotions like happiness, sorrow, grief, hatred, anger, and affection and opinions from text, reviews, and posts, which are available in many media platforms [1]. Sentiment analysis helps in tracking people’s viewpoints. For example, it is a powerful marketing tool that enables product managers to understand customer emotions in their various marketing campaigns. It is an important factor when it comes to social media monitoring, product and brand recognition, customer satisfaction, customer loyalty, advertising and promotion’s success, and product acceptance. It is among the most popular and valuable tasks in the field of NLP [2]. Sentiment analysis can be conducted as polarity classification or binary classification and fine-grained classification or multi-scale sentiment distribution.

Movie reviews is an important approach to assess the performance of a particular movie. Whereas providing a numerical or star rating to a movie quantitatively tells us about the success or failure of a movie, a collection of movie reviews is what gives us a deeper qualitative insight on different aspects of the movie. A textual movie review tells us about the strengths and weaknesses of the movie and deeper analysis of a movie review tells if the movie generally satisfies the reviewer. We work on Movie Reviews Sentiment Analysis in this study because movie reviews have standard benchmark datasets, where salient and qualitative works have been published on, in [3], for example.

BERT is a popular pre-trained language representation model and has proven to perform well on many NLP tasks like named entity recognition, question answering and

text classification [4]. It has been used in information retrieval in [5] to build an efficient ranking model for industry use cases. The pre-trained language model was also successfully utilised in [6] for extractive summarization of text and used for question answering with satisfactory results in [7]. Yang et al. [8] efficiently applied the model in data augmentation yielding optimal results. BERT has been primarily used in [9] for sentiment analysis, but the accuracy is not satisfactory.

In this paper, we fine-tune BERT for sentiment analysis on movie reviews, comparing both binary and fine-grained classifications, and achieve, with our best method, accuracy that surpasses state-of-the art (SOTA) models. Our fine-tuning couples BERT with Bidirectional LSTM (BiLSTM) and use the resulting model for binary and fine-grained sentiment classification tasks. To deal with class imbalance problem for fine-grained classification, we also implement oversampling and data augmentation techniques.

Fine-tuning is a common technique for transfer learning. The target model copies all model designs with their parameters from the source model except the output layer and fine-tunes these parameters based on the target dataset. The main benefit of fine-tuning is no need of training the entire model from scratch. Hence, we are fine-tuning BERT by adding BiLSTM and train the model on movie reviews sentiment analysis benchmark datasets. BERT processes input features bidirectionally [4], so does BiLSTM [10]. The primary idea behind bidirectional processing is to present each training sequence forwards and backwards to two separate recurrent nets, both of which are connected to the same output layer [10]. That is, both BERT and BiLSTM do not process inputs in temporal order, their outputs tend to be mostly based on both previous and next contexts.

Following that, we compute an overall polarity on the output vector from BERT+BiLSTM-

SA using a heuristic algorithm adopted from [11]. The algorithm in their paper is applied on the output vector from three-class classification on Twitter dataset by LSTM. In this work, we apply the algorithm differently depending on whether the output vector is from binary, three-class , four-class, or five-class classification. Hence, there are four different versions of the algorithm corresponding to four different classifications being carried out in our work. We present all the algorithms in **Section 3.1**.

Therefore, we divide our work into 2 main goals which aim to provide answers to the following questions:

1. How to effectively fine-tune BERT to improve accuracy measure on Movie Reviews Sentiment Analysis.
2. How to compute an overall polarity of a collection of movies reviews sentiments predicted by BERT+BiLSTM-SA.

In order to achieve goal 1, we will try to fine-tune BERT with BiLSTM, and goal 2 will be achieved by variably applying a heuristic algorithm on BERT output vector from BERT+BiLSTM-SA.

## 2 Related Work

### 2.1 Sentiment Analysis

Determining sentiment orientation of a text is an active research domain in NLP. This can be conducted in the context of polarity classification, fine-grained analysis, aspect-based analysis, and multi-lingual analysis. We start by presenting work done on the former two. A step-by-step lexicon-based sentiment analysis using the R open-source software is presented in [12]. The study conducted polarity classification by assessing the predictive accuracy of built-in lexicons using 1,000 movie review. The approach was applied on IMDb dataset and an 81.30% accuracy was obtained. Conversely, in [1], a different study implemented and compared traditional machine learning techniques like Naive Bayes (NB), K-Nearest Neighbours (KNN), and Random Forests (RF) for sentiment analysis on movie reviews. Their results on IMDb benchmark showed that NB was the best classifier on the task with 81.45% accuracy. Mesnil et al. [2] used a slightly distinct approach by deploying an ensemble generative technique for various machine learning approaches on movie reviews sentiment analysis and obtained 90.57% accuracy, whilst Dael et al. [13], in another study with a different dataset called Cornell movie review dataset, used only KNN with the help of information gain technique and obtained 90.8% accuracy. These results indicate that KNN has been a powerful traditional machine learning technique for movie reviews sentiment analysis. In [14], the authors proposed training document embeddings using cosine similarity and feature combination with NB weighted bag of n-grams. Preliminary, they compared

training document embeddings with cosine similarity against dot product, and cosine similarity gave best results. Their experiments on IMDB dataset achieved higher accuracy of 91.42%. In addition, Singh et al. [15] applied mixed objective function for binary classification on sentiment analysis on IMDB benchmark. Their approach reported error rate of 9.95%. The aforementioned models targeted polarity classification only. Nevertheless, both binary classification and fine-grained classifications on sentiment analysis were implemented in the following two studies. Semwal et al. [16] has used transfer learning, while Wang et al. [17], has utilised entailment and few-shot learning. Both studies used IMDB, SST-2, and MR benchmarks for binary classification, and Yelp and SST-5 for fine-grained classification. Average accuracy of 87.57% is reported in [16] on binary datasets, while [17] has reported 88.16%. For fine-grained classification, they reported average accuracy of 52.65% and 54.65%, respectively.

In our work, we adopt both polarity and fine-grained classifications from [17] but use deep learning techniques and BERT pre-trained language model. We also adopt transfer learning from [16].

## **2.2 Deep learning**

### **2.2.1 Deep Learning on Sentiment Analysis**

Deep Learning (DL) is a SOTA technique for most NLP tasks, sentiment analysis is not exceptional. Zhang et al. [18] explored Character-level Convolution Neural Networks (CCNNs) for text classification on Yelp and Amazon benchmarks and compared them against bag of words, n-grams and their TF-IDF variants, word-based CNNs and Recurrent Neural

Networks (RNNs). Best error rates of 7.82% on Yelp and 6.93% on Amazon were reported. On yelp, their result was outperformed by n-grams, which had 6.36% error rate. Overall results from the paper suggest that performance of the model relied on many factors, such as dataset size, whether texts were curated, and choosing the alphabet by distinguishing between uppercase and lowercase letter. In a different study, Shen, et al. [19] devised a different approach with CNNs. They applied meta network to learn context-sensitive convolutional filters for text processing in order to abstract the contextual information of a sentence or document into a set of input-aware filters. The approach was applied on Yelp and produced 4.89% error rate, which is better than the former approach. However, as these networks go deeper, the associated computational complexity increases, which poses serious challenges in practical applications. Additionally, DL pipelines are data hungry. Hence, shallow word-based Deep Pyramid CNNs (DPCNN) for text categorization were proposed in [20] to mitigate these problems. The authors studied deepening of word-level CNNs to capture global representations of text, and they obtained best accuracy by increasing the network depth without largely increasing computational cost. Their technique was evaluated on Yelp and Amazon datasets, and obtained error rates of 7.88% and 7.92%, respectively.

### **2.2.2 Deep Learning on Movie Reviews Sentiment Analysis**

To start with, RNNs and CNNs architectures performances were explored for semantic analysis of movie reviews in [21]. Predefined 300-dimensional vectors from word2vec were used instead of training the word vectors along other parameters using samples. RNNs were outperformed by CNNs, which gave best accuracy of 46.4% on SST dataset. It was concluded that basic RNNs were not an efficient model to represent structural and contextual



properties of the sentence. On the same note, basic RNNs suffer from the problem of vanishing or exploding gradients when the network goes very deep, thereby leading to model underfitting and overfitting. As a result, Coupled Oscillatory RNN (CoRNN), which is a time-discretization of a system of second-order ordinary differential equations, was proposed in [22] to mitigate the exploding and vanishing gradients problem. The time-discretization feature enabled them to model networks of controlled nonlinear oscillators, which proved precise bounds on the gradients of the hidden states, thereby mitigating the exploding and vanishing gradient problem for basic RNNs. They achieved 87.4% accuracy on IMDb. LSTMs also help in mitigating the problem in question. Hence, Bodapati et al. [23] used LSTM on movie reviews sentiment analysis by investigating the impact of different hyper parameters like dropout, number of layers, and activation functions. LSTMs are good in modeling very long sequences of data. Applying their experiments on IMDb, the network configuration comprised embedding, LSTM layer, dense layer, 0.5 dropout, and 100 LSTM units provided maximal accuracy of 88.46%. LSTMs adopt usage of additional gates for the purpose of memorizing longer input data sequences. Now, whether the gates incorporated in the LSTM architecture already offers a good generalization or additional training of data would be necessary to further improve the prediction is not clear [24]. As a result, a BiLSTM network for the task of text classification has also been applied via mixed objective function in [15] using both supervised and unsupervised approaches. Their results show that a simple BiLSTM model using maximum likelihood training can result in a competitive performance on polarity classification. In fact, a 6.07% error rate is reported from the study.

BiLSTM gave better results compared with other deep learning methods. However, with a 6.07% error rate, there is still room to improve performance on this task. Therefore,

in our work, we adopt BiLSTM here.

## 2.3 BERT

### 2.3.1 BERT and Sentiment Analysis

BERT is a popular SOTA pre-trained language model, which has provided salient results on NLP tasks. In sentiment analysis, for example, Xu et al. [25] analyzed the attentions and pre-trained hidden representations learned from a corpus on BERT for tasks in Aspect-Based Sentiment Analysis (ABSA). Since this work focused on ABSA, it is concluded in the paper that BERT uses very few self-attention heads to encode context words, e.g. prepositions or pronouns that indicate an aspect, and opinion words for an aspect. Conversely, Li et al. [26] investigated the modeling power of contextualized embeddings from BERT on an end-to-end ABSA task. The focus was on exploring to couple the BERT embedding component with various neural models. Their results suggest that BERT is also impressive when coupled with other models, and it worked well with Gated Recurrent Units (GRU). Gao et al. [27] also applied the pre-trained language model on target-dependent sentiment classification. The emphasis was to verify if its context-aware representation can achieve similar performance improvement in ABSA. Their findings suggest that coupling BERT with complex neural networks that used to work well with embedding representations does not show much value on ABSA.

Other studies have used BERT by applying transfer learning. In [28], they fine-tuned a pre-trained BERT on ABSA by constructing an auxiliary sentence from the aspect and convert ABSA to a sentence-pair classification task, e.g. Question Answering (QA) and

Natural Language Inference (NLI). Then, BERT was fine-tuned on the transformed task and an accuracy of 92.8% was attained on SentiHood dataset. In a different study, BERT has also been explored for fine-tuning on a novel task of Review Reading Comprehension (RRC) and ABSA in [29] by first building an RRC dataset called ReviewRC based on a popular benchmark for ABSA, then they explored a novel post-training fine-tuning approach on BERT. The study achieved 90.47% accuracy. Results from these studies suggest that their fine-tuning techniques are more effective.

It can be observed that BERT has been widely used on ABSA task. In this work, we adopt the coupling technique as in Li et al. [26], but we couple BERT with BiLSTM on the general movie reviews sentiment analysis task not ABSA.

### **2.3.2 BERT and Movie Reviews Sentiment Analysis**

On Movie Reviews, Maltoudoglou et al. [30] used BERT for turning words into contextualized word embeddings with parameters fine-tuned on IMDb movie reviews corpus by applying Inductive Conformal Prediction (ICP). Using BERT with ICP, they have reported 92.28% accuracy. Alaparthi and Mishra [31] have used a different approach. They have compared BERT against SentiWordNet, logistic regression, and LSTM for Movie Reviews sentiment analysis on IMDb dataset. The study aimed at finding relative efficacy of the four sentiment analysis algorithms and undisputed superiority of the pre-trained advanced supervised BERT in sentiment classification from text. BERT outperformed other models with 92.31% accuracy. Both studies focused on binary classification. On the contrary, Munikar et al. [9] used BERT for both binary and fine-grained classifications on SST-2 and SST-5 datasets, respectively. The model outperformed other deep learning based models, like CNN

and RNN, on both tasks attaining 93.7% accuracy on SST-2 and and 55.5% accuracy on SST-5.

It can be concluded that deep learning techniques have been the most accurate approaches for sentiment analysis. In general, transfer learning by fine-tuning BERT has provided best results. However, considering the results reported and the capabilities of BERT, there is still more room to improve their algorithms. Additionally, most of the studies focused on either polarity classification or fine-grained classification, and most researchers have not evaluated their approach on a wide variety of available SOTA benchmark datasets for sentiment analysis. That is, for example, results are reported on either IMDB only or SST variants only. As a result, in this work, we aim to fine-tune BERT by coupling with BiLSTM [26] for both polarity classification and fine-grained sentiment classification, since these techniques have given optimal results from the literature, and we adopt transfer learning from [16]. We also extend the previous works by computing overall polarity of sentiments as done in [11]. While this study has computed overall polarity regarding a single output vector from three-class classification by LSTM, we will compute from output vector of BERT coupled with BiLSTM depending on a classification task. We will experiment our model on IMDB, SST, MR, and Amazon benchmarks.

## 3 Our Approach

### 3.1 Methodology

We present different techniques used in our work starting with description of sentiment analysis and BERT. Afterwards, we explain how BERT is fine-tuned with BiLSTM, elucidate how classification is applied in our tasks, explain accuracy improvement techniques, describe overall polarity computation, and finish the chapter by talking about overview of our work.

#### 3.1.1 Sentiment Analysis

Sentiment analysis is a sub-domain of opinion mining, which aims at the extraction of emotions and opinions of people towards a particular topic from a structured, semi-structured, or unstructured textual data [32]. It can be conducted as polarity classification or fine-grained classification. Machine learning models following polarity classification classify a text as either carrying a positive or negative sentiment. On the other hand, fine-grained classification models use more than two classes in order to compute multi-scale sentiment distribution. Both classifications are implemented in our context.

#### 3.1.2 BERT

BERT [4] stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers and was introduced by researchers from Google. BERT focuses on pre-training deep bidirectional representations from unlabeled text by jointly conditioning on both left and right contexts in all layers of the model. As a result, BERT can be fine-tuned with just one additional layer

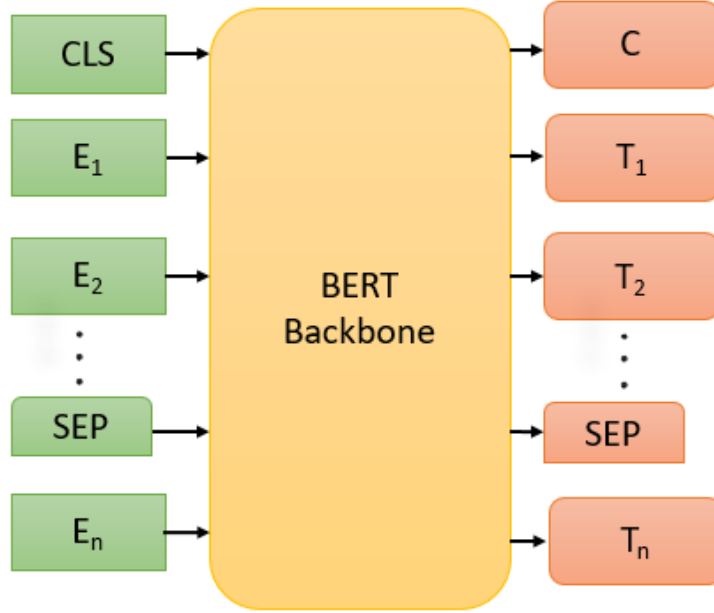
for a downstream task like sentiment analysis or question answering. BERT was pre-trained using the following two unsupervised tasks.

**Masked Language Modeling:** In this task, 15% of the tokens in the input sequence are randomly masked. Then, the entire input sequence is fed to a deep bidirectional transformer encoder, and the output softmax layer tries to predict the masked words[4, 9].

**Next Sentence Prediction:** BERT learns a relationship between two input sentences,  $A$  and  $B$ , by predicting if the sentences follow each other in a particular monolingual corpus. 50% of the inputs are a pair in which the second sentence is the subsequent sentence in the source document, while in the other 50% a random sentence from the corpus is chosen as the second sentence during training [4, 9].

Because of multiple attention heads, BERT processes a sequence of input tokens in parallel. There are two models of BERT, namely, BERT<sub>BASE</sub> and BERT<sub>LARGE</sub>. We are using BERT<sub>BASE</sub>, which has 12 layers, 768 hidden states, 12 attention heads, and 110M parameters, whereas BERT<sub>LARGE</sub> has almost 2 times of each of these specifications. Specifically, we use the uncased version of BERT<sub>BASE</sub> known as *bert-base-uncased*, which accepts input tokens as lowercase.

BERT has its own format for the input tokens. The first token of every sequence is denoted as [CLS]. This token corresponds to the last hidden layer, aggregates all the information in the input sequence and is used for classification tasks. Sentences are packed into a single input sequence and differentiated in two ways: using a special token [SEP] to separate them and adding a learned embedding to every token identifying a sentence where



**Figure 3.1:** Simplified diagram of BERT

it belongs to.

**Figure 3.1** shows a simplified diagram of BERT.  $E_n$  is an input representation of a single token constructed by summing the corresponding token, segment, and position embeddings; *BERT Backbone* represents main processing performed by BERT;  $T_n$  is a hidden state corresponding to token  $E_n$ ; and  $C$  is a hidden state corresponding to aggregate token [CLS]. So, we use  $C$  as the input to the fine-tuning component for sentiment classification.

### 3.1.3 Fine-tuning BERT with BiLSTM

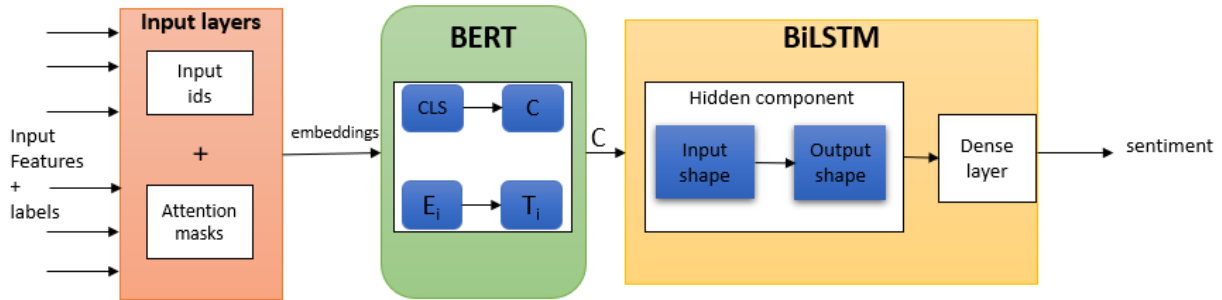
Since BERT is pretrained [4], there is no need of training the entire model from scratch. Hence, we just need to transfer information from BERT to the fine-tuning layer that is added and train this layer for sentiment analysis. This saves training time of the resulting model.

Our fine-tuning works as follows. After data preprocessing, we build two input layers

to BERT, where names of the layers need to match the input values. These input values are attention masks and input ids. In other words, attention masks and input ids are input embeddings to the model.

The input embeddings are propagated through BERT afterwards. Dimensionality of the embeddings depends on the input sequence length, batch-size, and number of units in the hidden state. BiLSTM is then concatenated at the very end of BERT, and it includes the dense layer. Therefore, BiLSTM receives information from BERT and feeds it into its dense layer, which then predicts respective classes for the input features. BERT and BiLSTM shared same hyperparameters, and we specify all hyperparameters in section 4 under experimental settings. We illustrate the fine-tuning part of our model in **Figure 3.2**. In the figure, *input features* are tokens in a review, *input ids* represent an input sequence, and *attention masks* are binary tensors indicating the position of the padded indices to a particular sequence so that the model does not attend to them. For the *attention mask*, we use 1 to indicate a value that should be attended to, while 0 is used to indicate a padded value. Padding helps in making sequences have same length when sentences have variable lengths, which is common in NLP. Therefore, padded information is not part of the input and should rarely be used in model generalisation. The output from BERT should have same dimension as input to BiLSTM, which is set to 768 and is represented by  $C$ . On the same note, we only feed  $C$  to BiLSTM.  $E_i$  and  $T_i$  mean similar items as  $E_n$  and  $T_n$ , respectively, in **Figure 3.1**. BiLSTM has one hidden component. The hidden state have input dimension of 128 by 256 by 768. 128 represents batch-size, 256 is time steps, and 768 represents number of units. Finally, there is a fully connected layer (dense layer) at the end, which has output dimension of batch-seize by 1, since we are working on binary classification. The dense layer





**Figure 3.2:** Fine-tuning part of BERT with BiLSTM

is used to predict a sentiment polarity of the input sequence.

Multi-class fine-tuning for BERT requires specifying number of classes to be used in classification. Adding BiLSTM layer also made it easy specifying number of classes and transitioning among all the classification tasks.

We also freeze weights of BERT first layers so that our focus dwells on the last layers close to the fine-tuning component. These layers contain trainable weights which are updated to minimize the loss during training of the model on our downstream task of sentiment analysis.

### 3.1.4 Classification

We are fine-tuning BERT on both polarity classification and fine-grained classification. Fine-grained classification is further divided into three modes of classification, which are three-class or 3-point scale, four-class or 4-point scale and five-class or 5-point scale classifications.

**Polarity classification:** Polarity classification or binary classification in this context is defined as follows. Given a movie review  $R$ , classify it as carrying either a positive sentiment

or a negative sentiment. This is a primary classification task that is usually carried out for sentiment analysis, since positive and negative polarities are the main sentiments that a text can portray [2].

**Three-class classification:** This extends the binary classification by introducing a neutral class [33]. The idea is that reviewers may not primarily and absolutely assign a positive or negative sentiment to a movie, because they might not have a right choice at hand [11]. Additionally, there might be a tie between positive and negative words used in the review. Therefore, these observations trigger inclusion of a neutral polarity. Three-class classification is defined as follows. Given a movie review  $R$ , classify it as whether carrying negative, neutral, or positive sentiment.

Alternatively, the output vector from binary classification can also be directly converted into three-class classification instead of manipulating labels in the training data and restarting the training process for the three class task. Since a sigmoid activation function predicts a class for a given review by assigning varying confidence levels to negative and positive classes, a neutral class can be included by using a delta value,  $\delta$ , whereby, the actual output label can be overwritten by a neutral label if the difference between the probabilities of the original two classes is less than  $\delta$ . However, this approach requires careful definition of  $\delta$  in order to have sensible results.

**Four-class classification:** We hierarchically extend binary version of IMDb to four classes by applying binary tree splitting. Binary tree splitting was applied in [34] using the concept of binary segmentation to find homogeneous nodes in a tree. Binary splitting is applied to IMDb

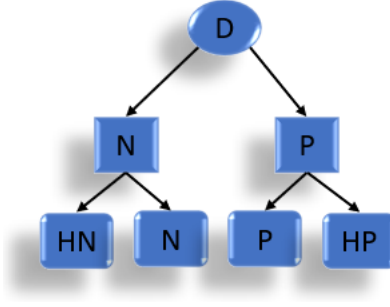
movie reviews dataset only in our work to split negative reviews into highly negative and negative, and split positive reviews into positive and highly positive. **Figure 3.3** illustrates this idea, where  $D$  represents Dataset, and  $N$ ,  $P$ ,  $HN$ , and  $HP$  represent, Negative, Positive, Highly Negative, and Highly Positive reviews, respectively. An extended explanation of how binary tree splitting is applied is provided in **Chapter 4** under IMDB dataset. Hence, four-class classification is defined as follows. Given a movie review  $R$ , classify it as whether carrying a highly negative, negative, positive, or highly positive sentiment.

**Five-class classification:** Five classes are used here as in [9] and [33]. While [33] used the output vector obtained from this classification to estimate the distribution of data examples across the five classes, we use the vector to find the overall polarity of all the predictions. Five-class classification is defined as follows. Given a movie review  $R$ , classify it as whether carrying a highly negative, negative, neutral, positive, or highly positive sentiment.

We use different classification point scales aiming at exploring how the polarity of a given review and overall polarity of a collection of reviews varies as the classification becomes more fine-grained, e.g. if a review is predicted as negative in a dataset that is used for binary classification, how will its polarity be maintained if we extend labels in the binary dataset to highly negative, negative, neutral, positive, and highly positive? The same applies for overall polarity.

### 3.1.5 Accuracy Improvement Approaches

We separately resorted to some data oversampling and augmentation techniques in order to enhance our model accuracy on fine-grained classification.



**Figure 3.3:** Binary Tree Splitting

**Oversampling:** We used SMOTE oversampling technique. SMOTE (Synthetic Minority Over-sampling TEchnique) first appeared in [35] to improve model performance on imbalanced datasets. For SMOTE only works with numerical data, we first converted reviews in minority classes into numerical features and fed the output into SMOTE, which then oversampled the features.

**Augmentation:** Data augmentation aims at reducing the degree of class imbalance and maximizing the amount of information that can be extracted from limited resources available [36]. We used NLP Augmenter (NLPAUG), which applies operations to textual input based on abstractive summarization and utilises synonym replacement based on proximity of a word embedding vector. This technique was successful in our experiments, and we briefly illustrate it in (3.1).

$$V_{AUG} = F(V_{IN}) \quad (3.1)$$

Where  $V_{AUG}$  is an output matrix of augmented sentences,  $F$  is the substractive summarisation augmentation function, and  $V_{IN}$  is the input matrix with raw text data. Both the input and output matrices contain  $n$  vectors depending on the number of misrepresented or minority classes in a dataset. Additionally, data examples in input vectors are randomly sam-

pled from a particular misrepresented class, and there is a one-to-one mapping of sentences between an input vector and a corresponding output vector.

After implementing these techniques, we first concatenated the output of SMOTE with the original input data and trained the model on SST-5. Last, we separately did the same with  $V_{AUG}$ .

### 3.1.6 Overall polarity

We define overall polarity as the following. Given an output vector from BERT+BiLSTM-SA containing sentiment labels of  $N$  reviews, compute the dominating polarity or sentiment label in the vector. To compute the overall polarity of reviews, we feed the output vector of BERT+BiLSTM-SA into a heuristic algorithm adopted from [11]. That is, BERT+BiLSTM-SA first predicts a sentiment category of each review and the results are collected in an output vector; labels of each class are then counted in the output vector; the output is then given to the heuristic algorithm to compute a dominating polarity for the reviews altogether.

In our work, we extend the algorithm to also compute overall polarity from output vector of binary, four-class, and five-class classifications. The reason being that the algorithm is dependent on number of classes in the output vector to compute the overall polarity, so we derived three variants of the algorithm accordingly.

**Algorithm 1** shows how overall polarity is computed from output vector of three-class classification. Consider the input as Twitter replies not reviews. The overall polarity is first considered to be neutral if the proportion of neutral Twitter replies to a given tweet is at least higher than a threshold, which is set to be 85%. The reason being that most replies to a tweet are generally expected to be neutral, for not every reply can be expected to carry

a positive or a negative polarity. Then, negative and positive sentiments are considered in the output vector. Again, there is usually no exclusively positive or negative reply, that is why a positive overall sentiment is assigned if there is at least 1.5 times as many negative replies as positive replies, and vice versa. Meaning that the proportion of a dominating class must be at least higher for all the replies to carry its sentiment polarity. Lastly, a neutral sentiment polarity is given again when the total numbers of positive and negative replies are close to each other, implying nonexistence of dominance between the two sentiments in the replies.

In our formulations, 1.2 was used as variable coefficient instead of 1.5 to determine majority gap for decision making. That is, the numbers, 1.2 and 1.5, are just weights or coefficients used for comparisons in the algorithms. Additionally, we are using movie reviews not twitter replies. In the source paper, their results worked well with 1.5 as shown in **Algorithm 1**, whilst 1.2 gave satisfactory results in our case.

To compute overall polarity from binary classification, we derived **Algorithm 2** from **Algorithm 1** by removing lines 1 to 3 in **Algorithm 1**, since binary classification does not contain a neutral polarity. Following that, everything else is reminiscent of **Algorithm 1**. Although there is not a neutral sentiment in the binary output vector of BERT+BiLSTM-SA, we introduce it for the overall polarity computation if the output of the first two conditions in **Algorithm 2** is false, implying a tie between quantities of positive and negative reviews.

**Algorithm 2** is extended to **Algorithm 3** for computation of overall polarity from four-class output vector of BERT+BiLSTM-SA. The algorithm is applied hierarchically considering binary tree splitting in **Figure 3.3**. That is, we start comparing base classes followed by comparing sub-classes of a base class that has majority samples. Therefore, we start by

---

**Algorithm 1:** Overall polarity computation from three-class classification output vector.

---

**Result:** Dominating sentiment polarity for all reviews.

```
1 if #total neutral reviews > 85% of the total reviews then
2   | overall polarity ← neutral;
3 else
4   | if #total positive reviews > 1.5 × # of total negative reviews then
5     | overall polarity ← positive;
6   | else if #total negative reviews > 1.5 × # of total positive reviews then
7     | overall polarity ← negative;
8   | else
9     | overall polarity ← neutral;
```

---

---

**Algorithm 2:** Overall polarity computation from binary classification output vector.

---

**Result:** Dominating sentiment polarity for all reviews.

```
1 if #total positive reviews > 1.2 × # of total negative reviews then
2   | overall polarity ← positive
3 else if #total negative reviews > 1.2 × # of total positive reviews then
4   | overall polarity ← negative
5 else
6   | overall polarity ← neutral
```

---

summing the number of samples of all fine-grained classes under each super class, like negative super class total is computed from highly negative and negative sub-classes totals. Then, three cases are considered. First case, if there is at least 1.2 times as many total negative reviews as total positive reviews for the super classes, a highly negative overall polarity is assigned if a highly negative subclass total is at least 1.5 times the number of negative subclass reviews. Else, an overall negative polarity is assigned. Second case, just vice-versa by interchanging positive and negative in the first case. Last case, a neutral overall polarity is assigned if there is no dominance between the super classes: total number of negative reviews and total number of positive reviews in the base classes. For the base classes a majority of times 1.2 is optimal for the dominating class, while for the sub-classes (mostly

---

**Algorithm 3:** Overall polarity computation from four-class classification output vector.

---

**Result:** Dominating sentiment polarity for all reviews.

```
1 if # (highly negative reviews + negative reviews) > 1.2 × # of (positive reviews
  + highly positive reviews) then
2   | if #highly negative reviews > 1.5 × # of negative reviews then
3   |   | overall polarity ← highly negative
4   | else
5   |   | overall polarity ← negative
6 else if # (positive reviews + highly positive reviews) > 1.2 × # of (highly
  negative reviews + negative reviews) then
7   | if #highly positive reviews > 1.5 × # of positive reviews then
8   |   | overall polarity ← highly positive
9   | else
10  |   | overall polarity ← positive
11 else
12  | overall polarity ← neutral
```

---

highly positive and highly negative) a majority of times 1.5 is better, since sub-classes are more fine-grained so the sample size of the dominating subclass has to be higher to assign its label as the overall polarity.

For five-class classification, we derive **Algorithm 4** from **Algorithm 3** by just initially adding one step. That is, the overall polarity is firstly considered to be neutral if the proportion of neutral reviews is at least higher than a threshold, which is similarly set to be 85%. Then, everything is just the same as steps in **Algorithm 3**.

A naive approach to computing the overall polarity would be just counting the number of labels for each class in BERT+BiLSTM-SA output vector and assign the overall polarity depending on majority class. However, the overall polarity computed from this approach cannot represent a good dominating majority class. Consider binary classification, for example, if the output vector has 49 positive reviews and 51 negative reviews, the overall polarity will be negative. However, a difference of 2 is not optimal in this case to decide the



---

**Algorithm 4:** Overall polarity computation from five-class classification output vector.

---

**Result:** Dominating sentiment polarity for all reviews.

```
1 if #total neutral reviews > 85% of the total reviews then
2   | overall polarity  $\leftarrow$  neutral
3 else
4   | if (#highly negative reviews + #negative reviews) > 1.2  $\times$  # of (positive
      | reviews + highly positive reviews) then
5     |   if #highly negative reviews > 1.5  $\times$  negative reviews then
6       |     | overall polarity  $\leftarrow$  highly negative
7       |   else
8       |     | overall polarity  $\leftarrow$  negative
9   | else if #(positive reviews + highly positive reviews) > 1.2  $\times$  # of (highly
      | negative reviews + negative reviews) then
10  |   | if #high positive reviews > 1.5  $\times$  # of positive reviews then
11  |     | | overall polarity  $\leftarrow$  highly positive
12  |   | else
13  |     | | overall polarity  $\leftarrow$  positive
14  | else
15  | | overall polarity  $\leftarrow$  neutral
```

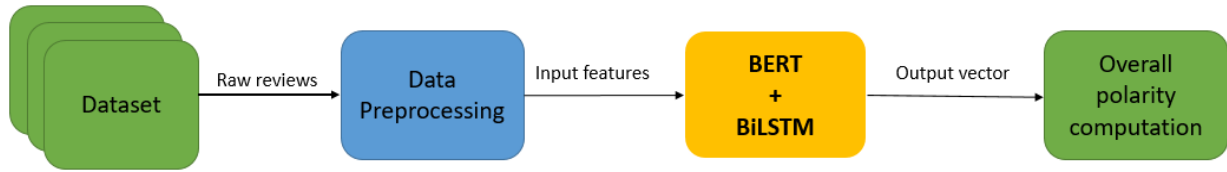
---

dominating polarity of all the reviews. Additionally, the level of positiveness or negativity is different for every review in the datasets. As a result, we use the formulations in the respective algorithms so that either class in the output vector must contain a higher majority to assign its label, otherwise the overall polarity becomes neutral.

In addition, we fine-tuned the coefficient to 1.2 and 1.5 in the algorithms based on different observations on the behavior of our model in different scenarios. That is, these numbers empirically gave accurate overall polarity computation reflecting the original overall polarity of input features.

### 3.1.7 Overview of Our Work

**Figure 3.4** provides an overview of our work. In a nutshell, we start by preprocessing raw text data into features that can be input to BERT and then we feed those features into



**Figure 3.4:** Overview of our work

BERT+BiLSTM through the fine-tuning layer, which specifies the hyperparameters that BERT+BiLSTM should use. Lastly, an output vector from BERT+BiLSTM predictions is used to compute overall polarity.

## 4 Experiments

In this chapter, we start with a description of datasets that are used in the experiments followed by data preprocessing. Afterwards, we describe experimental settings, explain evaluation metrics used in experiments, and lastly discuss experimental results in relation to defined goals.

### 4.1 Datasets

Datasets used in our experiments consist of movie reviews annotated for sentiment analysis on a 2-point, 3-point, 4-point and 5-point scales. Following is the description of the datasets.

#### 4.1.1 IMDb movie reviews

IMDb movie reviews dataset [37] is a popular binary sentiment analysis dataset consisting of 50,000 reviews from the Internet Movie Database (IMDb). It comprises equal number of negative and positive reviews. In our work, the dataset is used for binary classification and extended to three-class and four-class classifications. The dataset consists of three columns, namely, reviews, sentiment score, and label. Sentiment scores are integers from 1 to 10. Reviews with score from 1 to 5 have a negative label and 6 to 10 positive.

To apply four-class classification to the dataset, we used the idea of binary tree splitting in **Figure 3.3**. There are eight unique sentiment scores for every review in the dataset because there is no single review with scores 5 and 6. Then, we assign highly

negative label to reviews with scores 1 and 2 and negative label to reviews with scores 3 and 4. On the other hand, a positive label is given to reviews with scores 7 and 8, and a highly positive label is given to reviews with scores 9 and 10. For three class, scores 1, 2, and 3 represented negative sentiment; scores 4 and 7 represented neutral sentiment; and scores 8, 9, and 10 represented positive sentiment. We use IMDB-2, IMDB-3, and IMDB-4 to mean binary, three-class, and four-class versions of the dataset, respectively.

#### **4.1.2 SST**

The SST (Stanford Sentiment Treebank) is a corpus with fully labeled parse trees that allows for a complete analysis of the compositional effects of sentiment in language. The corpus is based on the dataset introduced in [38], and it consists of 11,855 single sentences extracted from movie reviews. It was parsed with the Stanford parser and includes a total of 215,154 unique phrases from those parse trees, each annotated by three human judges. Each phrase in the dataset is labelled as either negative, somewhat negative, neutral, somewhat positive or positive, corresponding to highly negative, negative, neutral, positive, and highly positive labels respectively in our annotations. The dataset has two versions, SST-5 or SST fine-grained and SST-2 or SST binary. SST-5 uses the five labels, whilst SST-2 uses two labels, which are negative and positive. Negative label is taken from negative or somewhat negative reviews, whereas positive label is taken from somewhat positive or positive reviews. Neutral reviews are discarded in SST-2. We use SST-2 and SST-5 for binary and five-class classifications, respectively.

### 4.1.3 MR Movie Reviews

MR Movie Reviews dataset comprises collections of movie reviews documents labeled with respect to their overall sentiment polarity, positive or negative, or subjective rating, for example two and a half stars, and sentences labeled with respect to their subjectivity status, subjective or objective, or polarity. In this paper, we use the version introduced in [39], which consists of 5331 positive and 5331 negative processed reviews. MR Movie Reviews dataset is solely used for binary classification task in our experiments.

### 4.1.4 Amazon Product Data dataset

This dataset contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May, 1996 through July, 2014. The dataset includes reviews, product metadata, and links. It was introduced in [40] for sentiment analysis using product review data, and in [41] to build a recommender system in collaborative filtering setting on amazon products. We only focus on video reviews in our work. The dataset originally contained labels with scores from 1 to 5 corresponding to polarity strength variation from highly negative to highly positive. We prepared the dataset for binary classification by replacing 1 and 2 scores with a negative label and 4 and 5 scores with a positive label, and score 3, which represents a neutral class, was discarded as in SST-2 [38]. In our experiments, we denote amazon-2 referring to the binary version of the dataset, and amazon-5 represents five-class version.

## 4.2 Data preprocessing

Our data preprocessing steps aim at transforming the raw input data into a format that BERT can understand. This involves carrying out two primary steps.

First, creating input examples using the constructor provided by BERT. The constructor accepts three main parameters, which are *text\_a*, *text\_b*, and *label*. *text\_a* is the text that we want the model to classify, which in this case, is the collection of movie reviews without their associated labels. *text\_b* is used if we are training a model to understand the relationship between sentences, for example sentence translation and question answering. This scenario hardly applies in our work, so we just leave *text\_b* blank. *label* has labels of input features. In our case, *label* implies sentiment polarity of every movie review. Refer to BERT original paper [4] for more details about this step.

Then, we conduct the following preprocessing steps.

- Lowercase our text, since we are using the lowercase version of BERT<sub>BASE</sub>.
- Tokenize all sentences in the reviews. For example, "this is a very fantastic movie" to "this", "is", "a", "very", "fantastic", "movie".
- Break words into word pieces. That is "interesting" to "interest" and "###ing".
- Map our words to indexes using a vocab file that is provided by BERT.
- Adding special tokens: [CLS] and [SEP], which are used for aggregating information of the entire review through the model and separating sentences respectively.
- Append index and segment tokens to each input to track a sentence that a specific token belongs to.

The output of the tokenizer after these steps is *input ids* and *attention masks*. These are then taken as inputs to our model in addition to the reviews labels.

### 4.3 Experimental settings

We fine-tune *bert-base-uncased*, which is a version of BERT that accepts lower case tokens. Some layers of BERT were not initialized from the model checkpoint, hence they are newly initialized.

We carried out many simulations on the datasets to find optimal hyperparameters for the model. As a result, optimal results from our experiments were obtained by the following hyperparameters. For binary classification, the model worked well with adam optimizer, 32 batch size, 3e-5 learning rate, 1e-08 epsilon, 128 maximum sequence length, and binary cross entropy loss. We trained the model for 5 epochs and repeated steps for each batch. On the other hand, all multi-class classifications used 64 batch-size, 1e-4 learning rate, 256 maximum sequence length and 1e-5 decay, in addition to adam optimizer and sparse categorical cross entropy loss. This version was trained for 7 epochs and also followed the repetition of steps for batches. We noticed overfitting when increasing the number of epochs for the respective models.

### 4.4 Evaluation Metrics

Similar to evaluation metrics used in [9], we use the accuracy performance measure to evaluate the performance of our model and compare it with other models. Accuracy is simply defined as follows:

$$accuracy = \frac{\text{number of correct predictions}}{\text{total number of predictions}} \times 100 \quad (4.1)$$

## 4.5 Results

### 4.5.1 Evaluation of Goal 1

We use experimental settings in **Section 4.3**. Then, we present accuracy comparisons between our model and other models on all datasets for binary classification in **Table 4.1**, followed by three-class and four-class classifications on IMDB dataset only in **Table 4.2**, and lastly, five class classification on SST-5 and amazon-5 in **Table 4.3**. Our model outperforms all other models on all classification tasks on all the datasets, thereby achieving new SOTA accuracy on the benchmark datasets.

Results indicate that our model performs better on binary classification tasks, and the model accuracy decreases as the classification scale becomes more fine-grained on IMDB dataset as shown in **Table 4.2**. The general reason being that adding classes in a dataset makes distinction between classes harder for the model. In addition, number of samples in each class are taken from the original dataset size despite increasing number of classes. Hence, less number of examples per class to successfully learn the model for effective prediction as number of classes are increasing in a dataset.

We also report experimental results on oversampling using SMOTE and augmentation using NPLPAUG in **Table 4.4** on SST-5 dataset. BERT+BiLSTM+SMOTE-SA uses SMOTE in addition to BERT and BiLSTM for Sentiment Analysis, while BERT+BiLSTM+NPLPAUG-SA uses NPLPAUG. Including SMOTE in our model does not have any impact on accuracy be-



**Table 4.1:** Accuracy (%) Comparisons of Models on Benchmark Datasets for Binary Classification

Model name	Dataset			
	<i>IMDb-2</i>	<i>MR</i>	<i>SST-2</i>	<i>amazon-2</i>
RNN-Capsule [42]	84.12	83.80	82.77	82.68
coRNN [16]	87.4	87.11	88.97	89.32
TL-CNN [16]	87.70	81.5	87.70	88.12
Modified LMU [43]	93.20	93.15	93.10	93.67
DualCL [44]	-	94.31	94.91	94.98
L Mixed [45]	95.68	95.72	-	95.81
EFL [17]	96.10	96.90	96.90	96.91
NB-weighted-BON+dv-cosine [14]	97.40	-	96.55	97.55
SMART-RoBERTa Large [46]	96.34	97.5	96.61	-
<b>Ours</b>	<b>97.67</b>	<b>97.88</b>	<b>97.62</b>	<b>98.76</b>

**Table 4.2:** Accuracy (%) Comparisons for Three and Four Class Classification on IMDD

Model name	Dataset	
	<i>IMDb-3</i>	<i>IMDb-4</i>
CNN-RNF-LSTM [47]	73.71	63.78
DPCNN [20]	76.24	66.17
BERT-large [9]	77.21	66.87
<b>Ours</b>	<b>81.87</b>	<b>70.75</b>

**Table 4.3:** Accuracy (%) Comparisons of Models on Benchmark Datasets for Five Class Classification

Model name	Dataset	
	<i>SST-5</i>	<i>amazon-5</i>
CNN+word2vec [21]	46.4	48.85
TL-CNN [16]	47.2	58.1
DRNN [48]	-	64.43
BERT-large [9]	55.5	65.83
BCN+Suffix+BiLSTM-Tied+Cove [49]	56.2	65.92
RoBERTa+large+Self-explaining [50]	59.10	-
<b>Ours</b>	<b>60.48</b>	<b>69.68</b>

cause mere BERT+BiLSTM-SA, which uses BERT and BiLSTM only without any accuracy improvement techniques, has even higher accuracy compared with BERT+BiLSTM+SMOTE-SA, thereby implying that the model does not catch well the semantics from data produced

**Table 4.4:** Accuracy (%) of Our Model with Accuracy Improvement Techniques on SST-5

Classification task	Accuracy
BERT+BiLSTM-SA	58.44
BERT+BiLSTM+SMOTE-SA	58.36
BERT+BiLSTM+NLPAUG-SA	60.48

by the oversampling technique. Conversely, BERT+BiLSTM+NLPAUG-SA improves performance of the model from 58.44% to 59.48%. The explanation to the two observations is that SMOTE takes text, which has been transformed into BERT features, as input. Hence, the transformed features tamper with the effectiveness of SMOTE, since some semantic information is lost during transformation. On the contrary, NLPAUG directly works on raw text data, making learning of semantic information easier from the data.

#### 4.5.2 Evaluation of Goal 2

We finish the discussion of results by talking about the overall polarity computation on all datasets by our model. The overall polarity computation is presented in **Table 4.5**, where OP stands for Overall Polarity. *Original OP* is known before input embeddings are fed into BERT+BiLSTM-SA, while *Computed OP* is computed after the model has made predictions on reviews. The table shows that the *Computed OP* is the same as the *Original OP* for all the datasets. The *Original OP* was calculated by counting the number of samples of each label in the input features and use the result in a particular heuristic algorithm depending on classification scale. The output was then used to verify the *Computed OP*. It can also be observed from the table that different versions of the same dataset have a consistent *Computed OP*. Therefore, without loss of generality, we are confident that the heuristic algorithm computes the expected and accurate overall polarity on the output vector

**Table 4.5:** Overall Polarity Computation on All the Datasets

<b>Dataset</b>	<b>Original OP</b>	<b>Computed OP<sup>b</sup></b>
IMDb-2	Neutral	Neutral
IMDb-3	Neutral	Neutral
IMDb-4	Neutral	Neutral
MR reviews	Neutral	Neutral
SST-2	Neutral	Neutral
SST-5	Neutral	Neutral
amazon-2	Positive	Positive
amazon-5	Positive	Positive

<sup>b</sup>OP stands for Overall Polarity

**Table 4.5** shows overall polarity computation for all the datasets. The Original OP is the same as the Computed OP for all the datasets.

from the model.

## 5 Conclusion

In this chapter, we summarise our work, contributions to the domain knowledge, and propose possible future work depending on our observations.

### 5.1 Conclusion

Sentiment analysis is an active research domain in NLP. In this work, we extend the existing domain knowledge of sentiment analysis by giving solutions to the following two questions, which also acted as main goals of our research. (1) How to effectively fine-tune BERT to improve accuracy measure on Movie Reviews Sentiment Analysis. (2) How to compute an overall polarity of a collection of movies reviews sentiments predicted by a model, BERT+BiLSTM-SA, for example. In order to answer question 1, we employed the technique of transfer learning by coupling BERT with BiLSTM. We froze first layers of BERT, which generated word embeddings that were inputs to BiLSTM. That is, BiLSTM acted as a classifier on BERT generated features. We used our model for both polarity classification and fine-grained classification. We further divided fine-grained classification into three-class, four-class, and five-class classifications. For binary classification, we experimented our model on IMDb, MR, SST-2, and Amazon-2 datasets. We also extended the binary IMDb dataset to three classes, and four classes by using binary tree splitting up to two levels. Amazon-5 and SST-5 were used on five-class classification task. We have shown that our model outperforms other works on all classification tasks and on all datasets. However, in order to further improve the model accuracy on five-class classification, we explored the effects of

using SMOTE and NLPAUG on SST-5, which is a most difficult fine-grained classification benchmark. We found that employing SMOTE actually decreased accuracy from 58.44% to 58.36%, while NLPAUG did improve accuracy to 60.48%. To give solution to question 2, we variably applied a heuristic algorithm to BERT-BiLSTM+SA output vector depending on the classification task being conducted. For all the datasets, we have demonstrated that the original overall polarity is the same as the computed overall polarity, and different versions of the same dataset give consistent computed overall polarity. To the best of our knowledge, this is the first work to couple BERT with BiLSTM and apply the resulting model on different sentiment classification tasks and different benchmark datasets and use the model output vector to compute overall sentiment polarity. We also contribute to the domain knowledge by exploring and showing how the polarity of a given review and overall polarity of a collection of reviews varies as the classification becomes more fine-grained.

## 5.2 Future Work

We propose future work to dwell on how to effectively apply accuracy improvement techniques to transformed BERT features despite loss of semantic information in them and focus on how different components of a sentence contribute to its sentiment prediction since this is information that is not generally exploited by current approaches.

## Bibliography

- [1] P. Baid, A. Gupta, and N. Chaplot, “Sentiment analysis of movie reviews using machine learning techniques,” *International Journal of Computer Applications*, vol. 179, no. 7, pp. 45–49, 2017.
- [2] G. Mesnil, T. Mikolov, M. Ranzato, and Y. Bengio, “Ensemble of generative and discriminative techniques for sentiment analysis of movie reviews,” *arXiv preprint arXiv:1412.5335*, 2014.
- [3] Z. Bingyu and N. Arefyev, “The document vectors using cosine similarity revisited,” *arXiv preprint arXiv:2205.13357*, 2022.
- [4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [5] W. Guo, X. Liu, S. Wang, H. Gao, A. Sankar, Z. Yang, Q. Guo, L. Zhang, B. Long, B.-C. Chen *et al.*, “Detext: A deep text ranking framework with bert,” in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 2509–2516.
- [6] Y. Liu, “Fine-tune bert for extractive summarization,” *arXiv preprint arXiv:1903.10318*, 2019.
- [7] Y. He, Z. Zhu, Y. Zhang, Q. Chen, and J. Caverlee, “Infusing disease knowledge into bert for health question answering, medical inference and disease name recognition,” *arXiv preprint arXiv:2010.03746*, 2020.
- [8] W. Yang, Y. Xie, L. Tan, K. Xiong, M. Li, and J. Lin, “Data augmentation for bert fine-tuning in open-domain question answering,” *arXiv preprint arXiv:1904.06652*, 2019.
- [9] M. Munikar, S. Shakya, and A. Shrestha, “Fine-grained sentiment classification using bert,” in *2019 Artificial Intelligence for Transforming Business and Society (AITB)*, vol. 1. IEEE, 2019, pp. 1–5.

- [10] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional lstm and other neural network architectures,” *Neural networks*, vol. 18, no. 5-6, pp. 602–610, 2005.
- [11] S. T. Arasteh, M. Monajem, V. Christlein, P. Heinrich, A. Nicolaou, H. N. Boldaji, M. Lotfinia, and S. Evert, “How will your tweet be received? predicting the sentiment polarity of tweet replies,” in *2021 IEEE 15th International Conference on Semantic Computing (ICSC)*. IEEE, 2021, pp. 370–373.
- [12] M. Anandarajan, C. Hill, and T. Nolan, “Sentiment analysis of movie reviews using r,” in *Practical Text Analytics*. Springer, 2019, pp. 193–220.
- [13] N. O. F. Daeli and A. Adiwijaya, “Sentiment analysis on movie reviews using information gain and k-nearest neighbor,” *Journal of Data Science and Its Applications*, vol. 3, no. 1, pp. 1–7, 2020.
- [14] T. Thongtan and T. Phienthrakul, “Sentiment classification using document embeddings trained with cosine similarity,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, 2019, pp. 407–414.
- [15] D. Singh Sachan, M. Zaheer, and R. Salakhutdinov, “Revisiting lstm networks for semi-supervised text classification via mixed objective function,” *arXiv e-prints*, pp. arXiv–2009, 2020.
- [16] T. Semwal, P. Yenigalla, G. Mathur, and S. B. Nair, “A practitioners’ guide to transfer learning for text classification using convolutional neural networks,” in *Proceedings of the 2018 SIAM international conference on data mining*. SIAM, 2018, pp. 513–521.
- [17] S. Wang, H. Fang, M. Khabsa, H. Mao, and H. Ma, “Entailment as few-shot learner,” *arXiv preprint arXiv:2104.14690*, 2021.
- [18] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” *Advances in neural information processing systems*, vol. 28, 2015.
- [19] D. Shen, M. R. Min, Y. Li, and L. Carin, “Learning context-sensitive convolutional filters for text processing,” *arXiv preprint arXiv:1709.08294*, 2017.

- [20] R. Johnson and T. Zhang, “Deep pyramid convolutional neural networks for text categorization,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 562–570.
- [21] H. Shirani-Mehr, “Applications of deep learning to sentiment analysis of movie reviews,” in *Technical report*. Stanford University, 2014.
- [22] T. K. Rusch and S. Mishra, “Coupled oscillatory recurrent neural network (cornn): An accurate and (gradient) stable architecture for learning long time dependencies,” *arXiv preprint arXiv:2010.00951*, 2020.
- [23] J. D. Bodapati, N. Veeranjanyulu, and S. Shaik, “Sentiment analysis from movie reviews using lstms.” *Ingenierie des Systemes d’Information*, vol. 24, no. 1, 2019.
- [24] S. Siami-Namini, N. Tavakoli, and A. S. Namin, “The performance of lstm and bilstm in forecasting time series,” in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 3285–3292.
- [25] H. Xu, L. Shu, P. S. Yu, and B. Liu, “Understanding pre-trained bert for aspect-based sentiment analysis,” *arXiv preprint arXiv:2011.00169*, 2020.
- [26] X. Li, L. Bing, W. Zhang, and W. Lam, “Exploiting bert for end-to-end aspect-based sentiment analysis,” *arXiv preprint arXiv:1910.00883*, 2019.
- [27] Z. Gao, A. Feng, X. Song, and X. Wu, “Target-dependent sentiment classification with bert,” *Ieee Access*, vol. 7, pp. 154 290–154 299, 2019.
- [28] C. Sun, L. Huang, and X. Qiu, “Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence,” *arXiv preprint arXiv:1903.09588*, 2019.
- [29] H. Xu, B. Liu, L. Shu, and P. S. Yu, “Bert post-training for review reading comprehension and aspect-based sentiment analysis,” *arXiv preprint arXiv:1904.02232*, 2019.
- [30] L. Maltoudoglou, A. Paisios, and H. Papadopoulos, “Bert-based conformal predictor for sentiment analysis,” in *Conformal and Probabilistic Prediction and Applications*. PMLR, 2020, pp. 269–284.
- [31] S. Alaparthi and M. Mishra, “Bert: A sentiment analysis odyssey,” *Journal of Marketing Analytics*, vol. 9, no. 2, pp. 118–126, 2021.



- [32] T. Gadekallu, A. Soni, D. Sarkar, and L. Kuruva, “Application of sentiment analysis in movie reviews,” in *Sentiment Analysis and Knowledge Discovery in Contemporary Business*. IGI global, 2019, pp. 77–90.
- [33] S. Rosenthal, N. Farra, and P. Nakov, “Semeval-2017 task 4: Sentiment analysis in twitter,” in *Proceedings of the 11th international workshop on semantic evaluation (SemEval-2017)*, 2017, pp. 502–518.
- [34] F. Mola and R. Siciliano, “A two-stage predictive splitting algorithm in binary segmentation,” in *Computational statistics*. Springer, 1992, pp. 179–184.
- [35] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: synthetic minority over-sampling technique,” *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.
- [36] G. Rizos, K. Hemker, and B. Schuller, “Augment to prevent: short-text data augmentation in deep learning for hate-speech classification,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 991–1000.
- [37] A. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, “Learning word vectors for sentiment analysis,” in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 2011, pp. 142–150.
- [38] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.
- [39] B. Pang and L. Lee, “Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales,” *arXiv preprint cs/0506075*, 2005.
- [40] X. Fang and J. Zhan, “Sentiment analysis using product review data,” *Journal of Big Data*, vol. 2, no. 1, pp. 1–14, 2015.

- [41] R. He and J. McAuley, “Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering,” in *proceedings of the 25th international conference on world wide web*, 2016, pp. 507–517.
- [42] Y. Wang, A. Sun, J. Han, Y. Liu, and X. Zhu, “Sentiment analysis by capsules,” in *Proceedings of the 2018 world wide web conference*, 2018, pp. 1165–1174.
- [43] N. R. Chilkuri and C. Eliasmith, “Parallelizing legendre memory unit training,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 1898–1907.
- [44] Q. Chen, R. Zhang, Y. Zheng, and Y. Mao, “Dual contrastive learning: Text classification via label-aware data augmentation,” *arXiv preprint arXiv:2201.08702*, 2022.
- [45] D. S. Sachan, M. Zaheer, and R. Salakhutdinov, “Revisiting lstm networks for semi-supervised text classification via mixed objective function,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 6940–6948.
- [46] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, “Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization,” *arXiv preprint arXiv:1911.03437*, 2019.
- [47] Y. Yang, “Convolutional neural networks with recurrent neural filters,” in *Proceedings of Empirical Methods in Natural Language Processing*, 2018.
- [48] B. Wang, “Disconnected recurrent neural networks for text categorization,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 2311–2320.
- [49] S. Brahma, “Improved sentence modeling using suffix bidirectional lstm,” *arXiv preprint arXiv:1805.07340*, 2018.
- [50] Z. Sun, C. Fan, Q. Han, X. Sun, Y. Meng, F. Wu, and J. Li, “Self-explaining structures improve nlp models,” *arXiv preprint arXiv:2012.01786*, 2020.

## **A All Publications Submitted**

”Movie Reviews Sentiment Analysis Using BERT”, Gibson Nkhata, Usman Anjum,  
and Justin Zhan. **Submitted for Reviewal**