PERFORMANCE COMPARISON OF BERT-BASED METHODS ON SENTIMENT ANALYSIS

by

Enes Arda

A report submitted for EE492 senior design project class
in partial fulfillment of the requirements for the degree of
Bachelor of Science
(Department of Electrical and Electronics Engineering)
in Boğaziçi University

June 10th, 2022

Principal Investigator:
Prof. Dr. Levent Arslan

ACKNOWLEDGMENTS

I would like to show my great appreciation to Prof. Dr. Levent Arslan for giving me the opportunity to research and experiment on my ideas and providing me a chance to search for the answers in the area that I am interested in through my academic life.

ABSTRACT

Sentiment analysis is an NLP task that tries to extract a subjective information from text. Over the last decade businesses and industrial communities have increased their focus on sentiment analysis techniques as it is a powerful tool to understand customers, employees and demands. In this project a language model BERT is utilized to create models for three sentiment analysis tasks. (1) Basic Sentiment Analysis, (2) Fine-Grained Sentiment Analysis and (3) Aspect-Based Sentiment Analysis. I proposed different methods for three different tasks using RoBERTa and ALBERT. The results of different methods are compared.

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Sentiment Analysis

Today we live in a world where information transaction is at very large speeds in digital media and most of the information shared is in text. Manually finding the required information in this age and processing it is beyond the ability of a human being. On the other hand, computers can be utilized to help people tackle this problem. Natural language processing (NLP) is a branch of artificial intelligence concerned with making computers interpret a natural language and process the text data. One of the main tasks of NLP is sentiment analysis, whose aim is analyzing people's opinions in textual data (e.g., product reviews, movie reviews, or tweets), and extracting their polarity and viewpoint. Basic sentiment analysis classifies texts as positive or negative. Moreover, there are also other types of sentiment analysis:

1) Fine-grained sentiment analysis: This is a multi-category sentiment analysis and it is conducted across the following polarity categories: very positive, positive, neutral, negative or very negative. Hence the polarity is fine-grained.
2) Aspect-based sentiment analysis: It determines the particular aspect people are talking about. For example with aspect based analysis in the review "The camera lacks a good quality but the performance of the phone was good." It can be concluded that the reviewer commented positively on the performance of the phone and negatively on the camera.

Sentiment analysis is crucial for understanding user-generated tweets, news reports, product reviews or survey responses. It could help businesses understand how people feel about their brand or product at scale, help companies enhance their customer service and uncover the areas of improvement for products. Moreover, sentiment analysis can be used to identify emerging trends, analyze competitors and find new markets. It overall could be very beneficial for businesses, governments and individuals.

NLP is a diversified field with many distinct tasks and most task-specific datasets, such as sentiment analysis datasets, are insufficient for deep-learning-based NLP models to fully benefit from. In order to overcome this problem researchers have developed general purpose language representation models and trained them with huge amount of unannotated text on the web. This process is called pre-training and creates a model for a specific natural language. Later the pre-trained model can be fine-tuned on a specific NLP task like sentiment analysis. This fine-tuning of a pre-trained model has resulted in substantial accuracy improvements compared to training on small task-specific datasets from scratch [1].

1.2 BERT

A recent NLP pre-training technique published by Google is called Bidirectional Encoder Representations from Transformers (BERT). In [2] they show that in eleven NLP tasks BERT performs better than other approaches, presenting state-of-the-art results in all tasks of understanding a language. The old approaches suffer from two main problems:

1) They are slow.
2) They are context free or don't capture the context very well.

They are slow because the words are passed in sequentially and therefore it takes a long time for the model to learn. On the other hand, BERT uses transformer architecture where words can be processed simultaneously which takes the parallel computing power of GPUs and hence works faster. Moreover, some of the pre-trained representations are context-free, such as word2vec [3] or GloVe [4]. They generate a single word embedding representations for each word in vocabulary regardless of the context. For example, the word "bank" has the same representation in both "bank account" and "bank of the river" although they have totally different meanings. However, contextual models generate a representation for each word based on the other words in the sentence. For instance, in the sentence "I accessed my bank account” a left-to-right unidirectional contextual model would represent “bank” based on “I accessed my”. Bidirectional models, on the other hand, represent “bank” using both its previous and next context. Most of them do this by learning left-to-right and right-to-left contexts separately and concatenating them. This causes the true context to be slightly lost. BERT represents each word using its both left and right context starting from the very bottom of a deep neural network, making it deeply bidirectional.

As stated before BERT can be pre-trained on a large corpus to understand what a language is and then it can be fine-tuned to learn a specific task such as sentiment analysis. BERT learns language by training on two unsupervised tasks simultaneously [2]:

1) Masked language modeling
2) Next sentence prediction

For masked language modeling BERT takes in a sentence with random words masked and the goal is to output these masked tokens. This helps BERT understand the bidirectional context in a sentence. In case of next sentence prediction BERT takes in two sentences and it determines if the second sentence actually follows the first sentence. This helps BERT understand context across different sentences themselves. Using both these together BERT gets a good understanding of what a language is.

For the fine tuning phase all one has to do is to replace the fully connected output layers of the network with a new set of output layers that can give the output of the desired task. Then supervised training can be performed using a dataset. It doesn’t take long because only the output parameters are learnt from scratch and the rest of the model parameters are just slightly fine-tuned. Therefore the training time is fast. The pre-training and fine-tuning procedures of BERT are illustrated in Fig. 1.1.

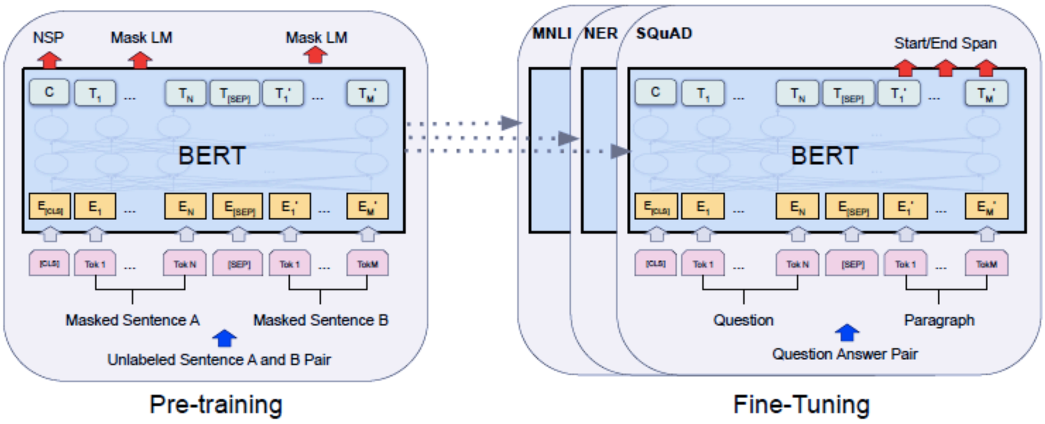


Fig. 1.1 Pre-training and fine-tuning procedures of BERT. For pre-training some of the tokens are masked and [CLS] token is used to determine if the next sentence comes after the first sentence.

1.3 Difference Between BERT, RoBERTa and ALBERT

In the original paper [2] BERT is optimized with Adam [16] using the parameters $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e - 6$ and weight decay 0.01. The learning rate is initially warmed up over the first 10000 steps with the maximum value 1e-4 and then linearly decayed. In RoBERTa [6] they tune the peak learning rate and the number of warm-up states separately for each setting instead of using constant values. Moreover, they conclude that setting $\beta_2 = 0.98$ and tuning Adam epsilon term improved stability while training with large batches. In pre-training, unlike the original BERT paper, RoBERTa doesn't train with a reduced sequence length for the first 90% of the updates but only with full-length sequences. Furthermore, [17] shows that increasing data size results in improved performance. Therefore, RoBERTa is trained on a total of 160GB uncompressed text data in comparison to the 16GB of uncompressed text data that original BERT is trained on. In the original BERT implementation the model is pre-trained on two unsupervised tasks simultaneously:

1) Masked language modeling
2) Next sentence prediction

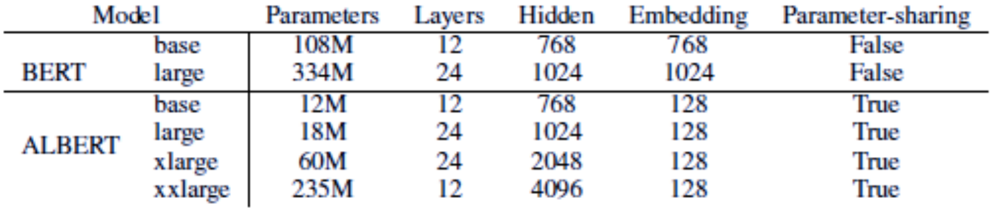
The masking is performed once during preprocessing, which results in a static mask. However, in RoBERTa to avoid using the same mask in each training instance they duplicated the data 10 times and each sentence is masked 10 different ways, which resulted in dynamic masking. In addition, in contrast to the original BERT, RoBERTa removes next sentence prediction loss and the inputs are packed with full sentences. In [6] they show that this improves downstream task performance. Lastly, the original BERT implementation utilizes a Byte-Pair Encoding of size 30K words. However, RoBERTa is trained with BPE vocabulary containing 50K words.

Simply increasing the model parameters when pre-training a language improves the performance on downstream tasks. However, at a point increasing the number of parameters becomes harder due to GPU/TPU memory limitations. A lite BERT (ALBERT) [7] tries to address this problem by using significantly fewer parameters than the traditional BERT architecture without hurting the performance. Moreover, so as to improve the performance, it introduces a self-supervised loss for sentence-order prediction instead of NSP of original BERT.

In BERT and RoBERTa the WordPiece embedding size is tied with the hidden layer size. However, unlike hidden layer embeddings, the objective of WordPiece embeddings is not to learn context-dependent representations. Moreover, the power of BERT comes from utilizing the context of the input. Therefore, they propose in [7] that untying WordPiece embedding size (E) and hidden layer size (H) by making hidden layer size much larger than the embedding size, make the usage of model parameters much more efficient. Nevertheless, increasing the hidden layer size could result in billions of parameters. Therefore in ALBERT, they use a factorization of embedding parameters by decomposing them to smaller matrices. Instead of projecting vectors directly in hidden space, they are first projected onto a lower dimensional embedding space and then to the hidden space. This factorization results in significant parameter reduction if $H \gg E$. Moreover, they suggest cross-layer parameter sharing to improve parameter efficiency. By sharing all parameters across layers they stabilize and reduce the number of parameters used.

As stated before, NSP loss is used in original BERT pre-training phase. However, studies [18] show that NSP's impact is ineffective and unreliable. In [7] they propose that this is because NSP mixes topic prediction (which overlaps with masked language modeling) and coherence prediction. In order to address this problem ALBERT uses a sentence-order prediction (SOP) loss, which ignores topic prediction and focuses primarily on coherence. This helps the model to learn finer distinctions regarding coherence properties. As a result, ALBERT improves downstream task performance for multi-sentence encoding tasks.

Due to the design choices, ALBERT models have much smaller parameter size compared to BERT models. For instance BERT-base mode has layer size 12 and hidden size 768 with 108M parameters, whereas ALBERT-base has the same layer size and hidden size with 12M parameters. A significant reduction in parameter size can be observed. The original comparison from the paper [7] is in Table 1.1.



Model		Parameters	Layers	Hidden	Embedding	Parameter-sharing
BERT	base	108M	12	768	768	False
	large	334M	24	1024	1024	False
ALBERT	base	12M	12	768	128	True
	large	18M	24	1024	128	True
	xlarge	60M	24	2048	128	True
	xxlarge	235M	12	4096	128	True

Table 1.1: The configurations of the main BERT and ALBERT models

1.4 Objective

The objective of this project is to compare BERT based methods such as RoBERTa [6], ALBERT [7] on an NLP text classification task called sentiment analysis. First of all, the methods will be tested on the basic sentiment analysis where the text is classified as positive or negative. Later on different methods can also be compared on other sentiment analysis techniques to see whether some methods perform better on specific tasks or not

Text data for sentiment analysis can come from different sources, including web data, emails, chats, social media, tickets, insurance claims, user reviews, and questions and answers from customer services, which I think also has an influence on the performance of different models. Therefore different datasets will also be compared in the project. For example twitter datasets such as Stanford Twitter Sentiment Dataset [10], Sentiment Strength Twitter Dataset [11], Sanders Twitter Dataset or movie review datasets such as Stanford Sentiment Treebank [12], IMDB Movie Reviews Dataset [13] or product review datasets such as OpinRank Review Dataset [14], Amazon Review Data [15].

Moreover, the methods will also be compared on other sentiment analysis tasks such as fine grained sentiment analysis and aspect based sentiment analysis to conclude whether some methods perform better on specific sentiment analysis tasks or not. The results will give us the best sentiment analysis method for a specific task and specific dataset which is a very important result as the sentiment analysis could be very important for businesses, governments, companies and individuals.

CHAPTER 2
BASIC SENTIMENT ANALYSIS

2.1 Dataset

Initially, my objective was to use the IMDB Movie Review Dataset [13] to compare the performance of pre-trained RoBERTa and ALBERT models on basic sentiment analysis. The goal in basic sentiment analysis is to classify a text as positive or negative sentiment. The IMDB dataset contains 50000 long reviews labeled as positive or negative sentiment. Both RoBERTa and ALBERT take the input of a sequence no more than 512 tokens. Therefore, the reviews that contain more words than 512 are truncated. Moreover, due to time constrictions only 10000 reviews are used. The 70% of the dataset is used in training 30% of dataset is used in testing and validation.

2.2 Machine Learning Approach

Both RoBERTa and ALBERT use the hidden size of 768 and the first token of the input sequence is always [CLS] which contains the special classification embedding. First, I wanted to use the last layer hidden-states of the first token as features of size 768 and then train machine learning models to classify IMDB reviews as positive or negative sentiment. The weighted average F1 Scores of the 5 different ML models are displayed in Table 2.1.

	Support Vector Classifier	Random Forest Classifier	KNN Classifier	Decision Tree classifier	Gradient Boosting Classifier
RoBERTa	**0.89**	0.84	0.76	0.76	0.76
ALBERT	**0.80**	0.77	0.65	0.65	0.65

Table 2.1: Weighted average F1 scores of 5 ML models using the last layer hidden-states of the first tokens as inputs.

One can see from Table 2 that Support Vector Classifier gave the best results and pre-trained RoBERTa performed better than ALBERT. The confusion matrices of SVC for both methods are shown in Fig2.1 and Fig2.2.

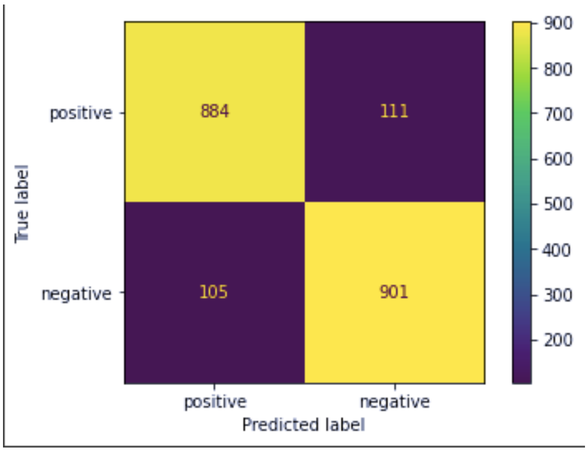


Fig. 2.1 Confusion matrix of SVC using output of RoBERTa

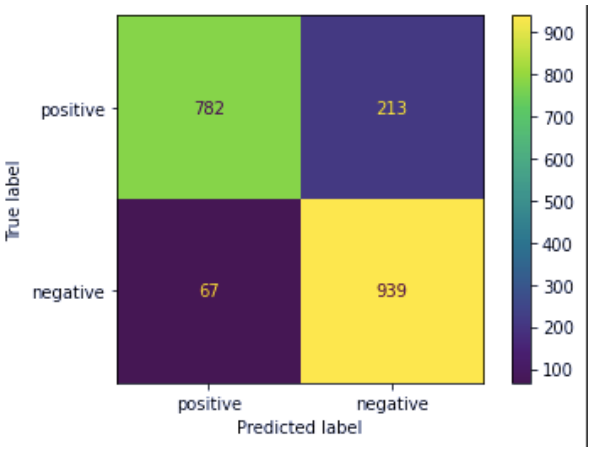


Fig. 2.2 Confusion matrix of SVC using output of ALBERT

2.3 Fine Tuning Approach

Then I wanted to fine tune the BERT-based models and compare the results with the ML approach. BERT-based models are fine-tuned on Tesla K80 that Google Colab provides and batch size is set to 32 to ensure that GPU memory is fully utilized. Since our task is to classify a text as positive or negative sentiment the output of the model should be of size 2 that represents the probability of 2 sentiments. In order to get output of size 2, two dense layers that perform linear transformations are defined. The first layer reduces the dimension from 768 to 512 and the other layer is the output layer that reduces the dimension to two. Between these two layers a rectified linear unit function is applied as it helps to reduce the likelihood of the gradient to vanish and introduces sparsity. The dropout layer is used with dropout probability 0.1 to prevent overfitting and regularize. Finally, a softmax function is used to convert the outputs of the output layer to probabilities and use these probabilities to classify the text. AdamW optimizer is used with $\beta_1 = 0.9$, $\beta_2 = 0.999$, base learning rate 2e-5 and $\epsilon = 1\text{e-}8$. No warm up state is used and the learning rate is decreased linearly to 0. The negative log likelihood loss is used as the loss function and the label weights are considered in calculating the loss. The epoch is set to 4 due to time considerations and the fine-tuning is done. The learning curve of RoBERTa can be seen in Fig 2.3.

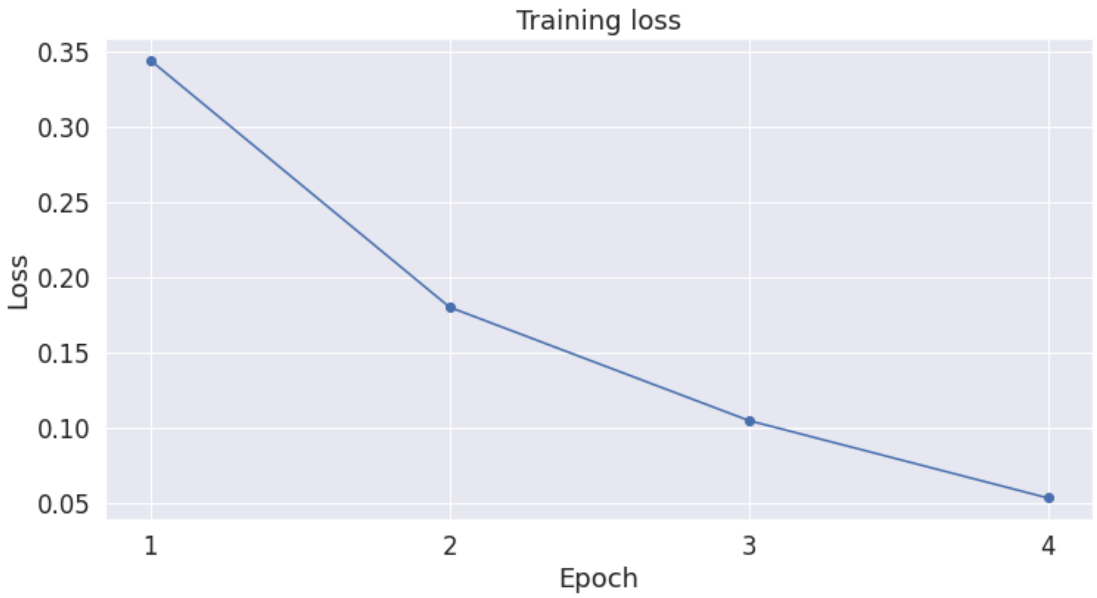
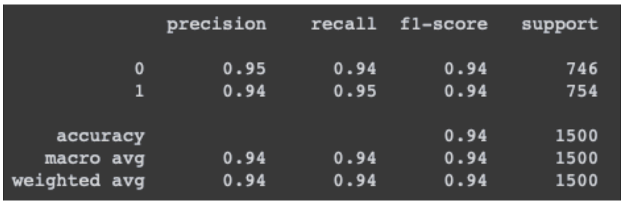


Fig. 2.3. RoBERTa learning curve for Basic Sentiment Analysis

Fine-tuning resulted in a model of size 477.1MB. The classification report and the confusion matrix on the test set can be seen in Fig 2.4 and Fig 2.5.



```
              precision    recall  f1-score   support

           0       0.95      0.94      0.94       746
           1       0.94      0.95      0.94       754

    accuracy                           0.94      1500
   macro avg       0.94      0.94      0.94      1500
weighted avg       0.94      0.94      0.94      1500
```

Fig. 2.4. Fine-Tuned RoBERTa classification report for Basic Sentiment Analysis

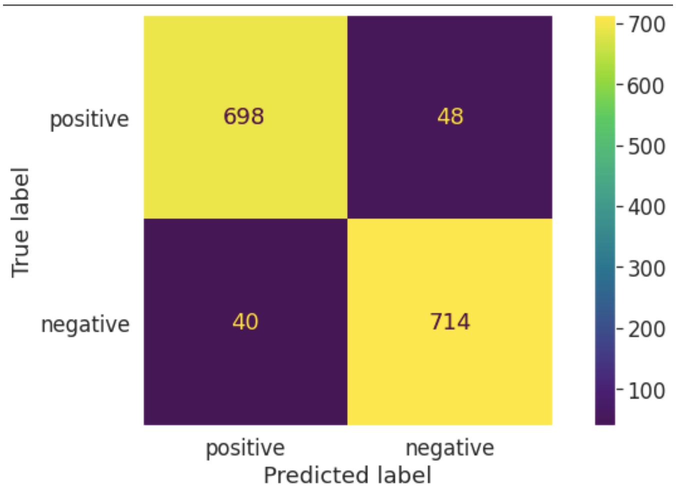


Fig. 2.5. Fine-Tuned RoBERTa confusion matrix for Basic Sentiment Analysis

ALBERT is fine-tuned utilizing the same functions and parameters. Fine-tuning resulted in a model 46.1MB, which is much smaller than the RoBERTa model as it uses significantly fewer parameters. The learning curve of ALBERT can be seen in Fig2.6.

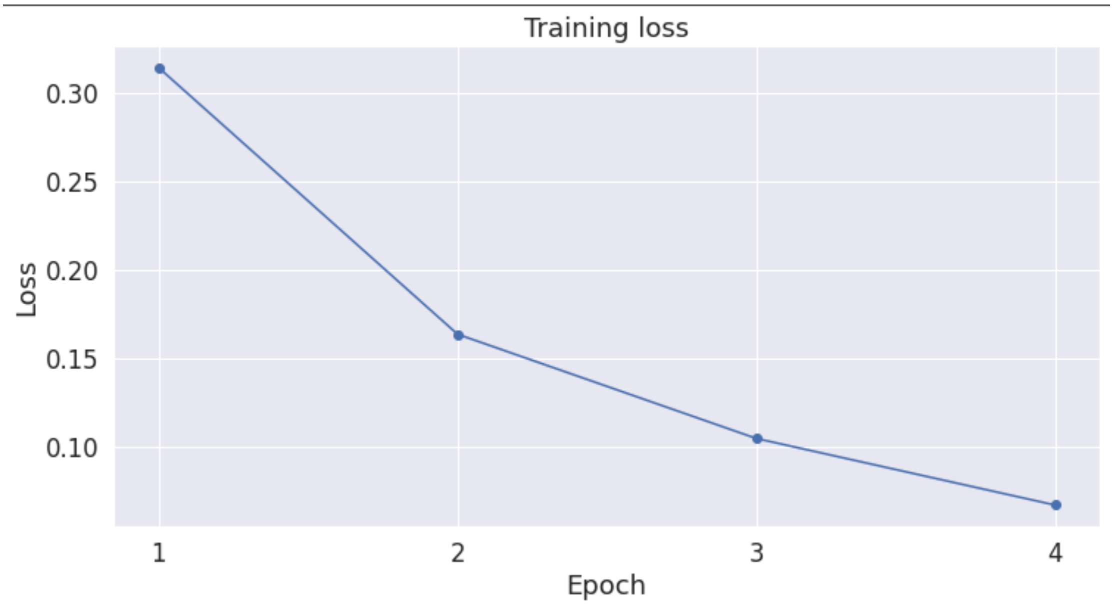
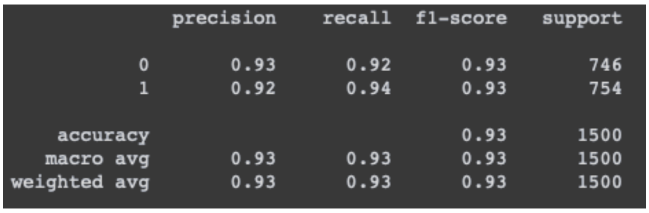


Fig. 2.6. ALBERT learning curve for Basic Sentiment Analysis

The end loss of the ALBERT seems to be smaller than the end loss of RoBERTa. The classification report and the confusion matrix on the test set can be seen in Fig2.7 and Fig2.8.



```
              precision    recall  f1-score   support

           0       0.93      0.92      0.93       746
           1       0.92      0.94      0.93       754

    accuracy                           0.93      1500
   macro avg       0.93      0.93      0.93      1500
weighted avg       0.93      0.93      0.93      1500
```

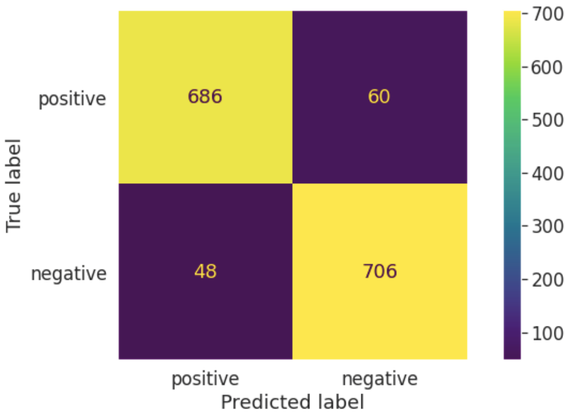


Fig. 2.7. Fine-Tuned ALBERT classification report for Basic Sentiment Analysis

Fig. 2.8. Fine-Tuned ALBERT confusion matrix for Basic Sentiment Analysis

One can conclude that fine-tuning models resulted in much better results than machine learning approaches. Moreover, similar to the ML approach RoBERTa gave better results than ALBERT but in fine-tuning case the difference is smaller. The comparison of RoBERTa and ALBERT can be made in Table 2.2.

	ML F1 Score	Fine-Tuning F1 Score	Model Size
RoBERTa	0.89	0.94	477.1MB
ALBERT	0.80	0.93	46.1MB

Table 2.2: Comparison of performance of RoBERTa and ALBERT on Basic Sentiment Analysis

2.4 Testing Model on a Different Domain

In order to see the effect of the domain on the performance of basic sentiment analysis models, models that are trained on IMDB Movie Review dataset are tested on Amazon Food Review dataset. The results are summarized in Table 2.3.

	Train on Movie dataset Test on Movie dataset F1 Score	Train on Movie dataset Test on Food dataset F1 Score	Train on Combined dataset Test on Food dataset F1 Score
RoBERTa	0.94	0.91	0.93
ALBERT	0.93	0.88	0.91

Table 2.3: Comparison of performance of RoBERTa and ALBERT Basic Sentiment Analysis models tested on different datasets.

One can see from Table 2.3 that domain knowledge affects the performance of the model. A basic sentiment analysis model that is trained in a specific domain performs better in that domain. The performance drop of RoBERTa model is 3% whereas the performance drop of ALBERT model is 5%. This is probably because the reduction of parameters prevents the model to generalize to other domains and makes the model more domain-specific. In addition, both datasets are combined and split into to training and test sets. As seen in Table 2.3 this approach gave the mediocre scores.

These results show the importance of the domain knowledge in sentiment analysis. A BERT model that is trained in one domain performs better in that domain.

CHAPTER 3
FINE-GRAINED SENTIMENT ANALYSIS

3.1 Dataset

The aim in fine-grained sentiment analysis is not only to find if a sentiment is positive or negative but also to capture the text's sentiment degree. For example to find how positive or how negative a text's sentiment is. Amazon Food Review dataset [15] is used for this purpose. This dataset contains 568,454 food reviews that are rated from 1 to 5. The reviews are not very long so the maximum sequence length is selected as 256 to save memory. Moreover, due to time constrictions only 12,495 reviews are used. The dataset is not very balanced and the number of reviews for different scores are not the same. Hence, weighting should be utilized in loss function to handle the imbalance. The 70% of the dataset is used in training 15% of dataset is used in testing and 15% of dataset is used in validation.

Due to the low performance observed in machine learning approach in the previous chapter, only fine-tuning is used in this chapter.

3.2 Classification

The difference between the basic sentiment analysis classification and the fine-grained sentiment analysis classification is that fine-grained has more outputs. In this case fine-grained should have 5 outputs to represent the probability of each sentiment level. As before, the last layer hidden-states of the first [CLS] token of size 768 is used as the input. And this input is put into the same architecture as in Chapter 2 only with the difference in output layer reducing the dimension to 5 instead of 2. The other part of the architecture is the same as the basic sentiment analysis. Because the dataset is imbalanced label weights should be considered in calculating the loss. BERT-based models are fine-tuned on Tesla P100 that Google Colab provides and batch size is set to 32 to ensure that GPU memory is fully utilized. The number of epochs is set to 8 and the resulting learning curve for RoBERTa, its classification report and the confusion matrix can be seen in Fig3.1, Fig3.2 and Fig3.3.

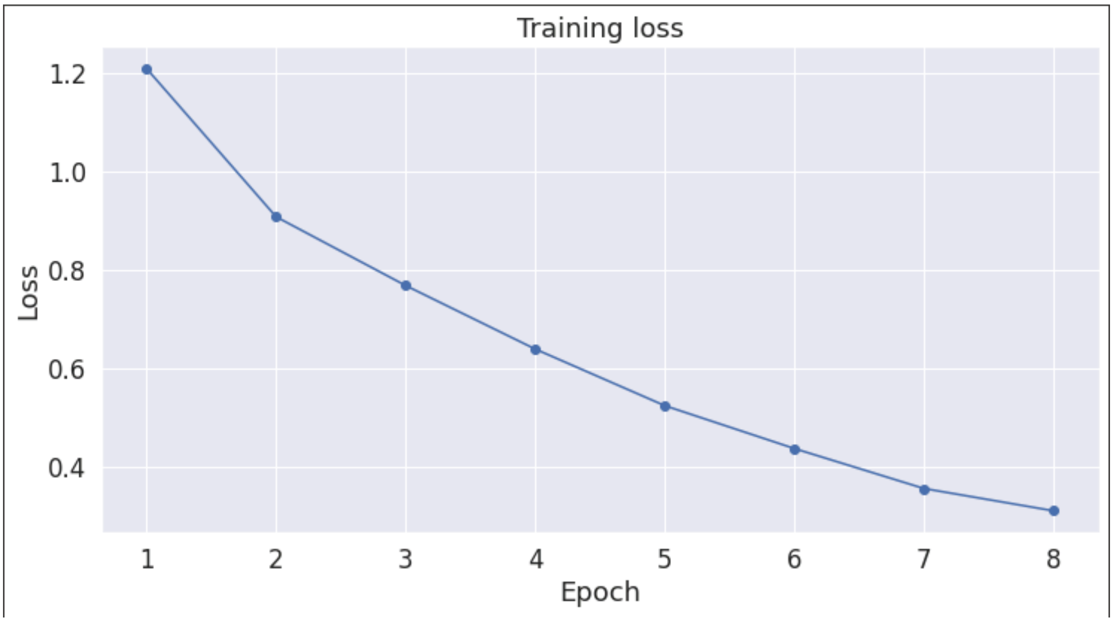
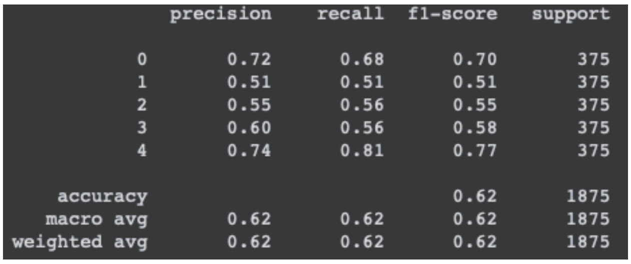


Fig. 3.1. Training loss for Fine-Grained Sentiment Analysis classification model with RoBERTa



```
              precision    recall  f1-score   support

           0       0.72      0.68      0.70       375
           1       0.51      0.51      0.51       375
           2       0.55      0.56      0.55       375
           3       0.60      0.56      0.58       375
           4       0.74      0.81      0.77       375

    accuracy                           0.62      1875
   macro avg       0.62      0.62      0.62      1875
weighted avg       0.62      0.62      0.62      1875
```

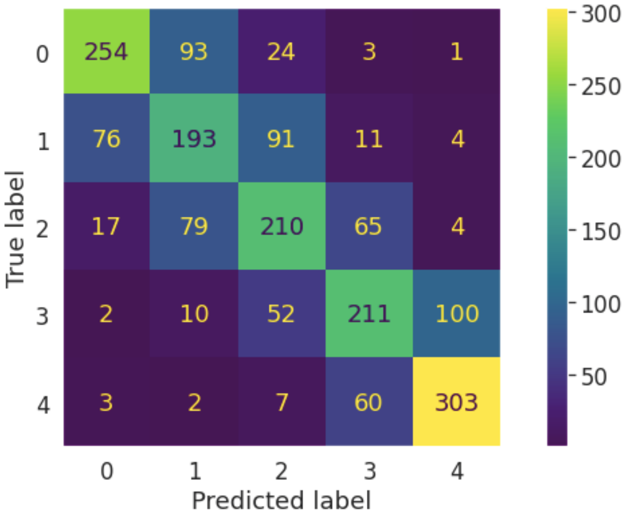
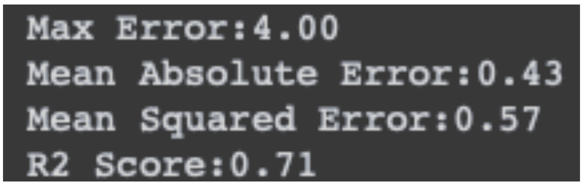


Fig. 3.2. RoBERTa classification report for Fine-Grained Sentiment Analysis

Fig. 3.3. RoBERTa confusion matrix for Fine-Grained Sentiment Analysis

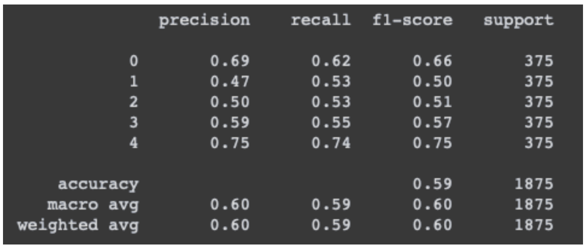
Incorrectly classified classes are mostly classified to ones that are next to their actual scores. Because the classified variable in this task is quantitative other metrics can also be used to evaluate the model. These regression scores can be seen in Fig 3.4.



```
Max Error:4.00
Mean Absolute Error:0.43
Mean Squared Error:0.57
R2 Score:0.71
```

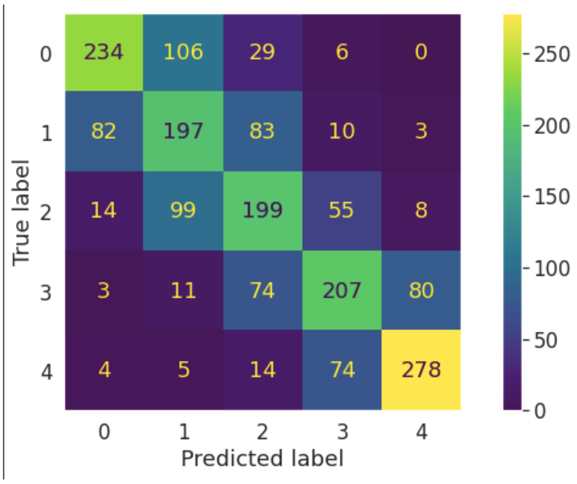
Fig. 3.4. RoBERTa regression scores for Fine-Grained Sentiment Analysis classification model

The same architecture is used for training the ALBERT model. Its classification report, the confusion matrix and the regression scores can be seen in Fig3.5, Fig3.6 and Fig3.7.



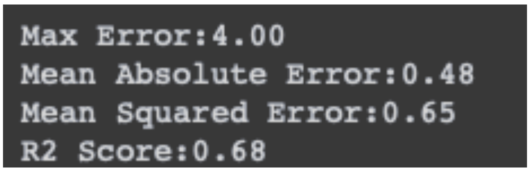
```
              precision    recall  f1-score   support

           0       0.69      0.62      0.66       375
           1       0.47      0.53      0.50       375
           2       0.50      0.53      0.51       375
           3       0.59      0.55      0.57       375
           4       0.75      0.74      0.75       375

    accuracy                           0.59      1875
   macro avg       0.60      0.59      0.60      1875
weighted avg       0.60      0.59      0.60      1875
```

Fig. 3.5. ALBERT classification report for Fine-Grained Sentiment Analysis classification model



True label \ Predicted label	0	1	2	3	4
0	234	106	29	6	0
1	82	197	83	10	3
2	14	99	199	55	8
3	3	11	74	207	80
4	4	5	14	74	278

Fig. 3.6. ALBERT confusion matrix for Fine-Grained Sentiment Analysis classification model



```
Max Error:4.00
Mean Absolute Error:0.48
Mean Squared Error:0.65
R2 Score:0.68
```

Fig. 3.7. ALBERT regression scores for Fine-Grained Sentiment Analysis classification model.

Comparison of two models are made in Table 3.1.

	F1	MSE	R2
RoBERTa	0.62	0.57	0.71
ALBERT	0.60	0.65	0.68

Table 3.1: Comparison of performance of RoBERTa and ALBERT Fine-Grained sentiment models on different metrics.

One can see from Table 3.1 that that RoBERTa scores outperform ALBERT scores. Moreover, this time the difference between two models is larger than that in Basic Sentiment Analysis.

3.3 Regression

Although the dataset consists of discrete sentiment scores from 1-5 we don't have to use a classifier to classify between these 5 discrete classes. Instead a regression model can be utilized to capture the level of sentiment more finely. In this way the sentiment score can be any value between 1-5 and the performance of the model can be evaluated better because a score 3.4 is a better estimation of 4 than a score 3. As before, the last layer hidden-states of the first [CLS] token of size 768 is used as the input. Therefore, the output of the model should be of size 1 that represents the degree of sentiment. In order to get output of size 1, two dense layers that perform linear transformations are defined. The first layer reduces the dimension from 768 to 512 and the other layer is the output layer that reduces the dimension to one. Between these two layers a rectified linear unit function is applied as it helps to reduce the likelihood of the gradient to vanish and introduces sparsity. The dropout layer is used with dropout probability 0.1 to prevent overfitting and regularize. AdamW optimizer is used with $\beta_1 = 0.9$, $\beta_2 = 0.999$, base learning rate 2e-5 and $\epsilon =$ 1e-8. No warm up state is used and the learning rate is decreased linearly to 0. The mean squared error loss is used as the loss function and the label weights are considered in calculating the loss. The epoch is set to 4 due to time considerations and to prevent overfitting. The learning curve of RoBERTa can be seen in Fig 3.8.

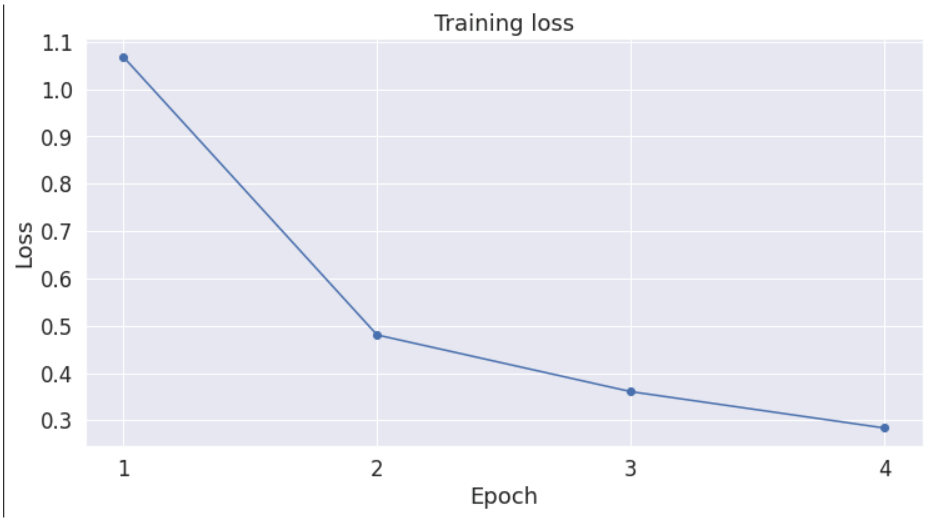
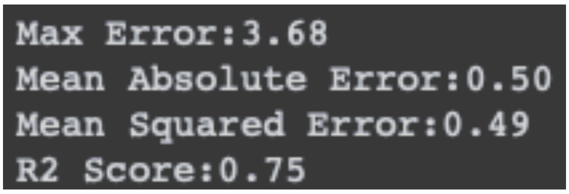


Fig. 3.8. Training loss for Fine-Grained Sentiment Analysis regression model with RoBERTa

The regression scores and the distribution of the predictions can be seen in Fig3.9 and Fig3.10.



```
Max Error:3.68
Mean Absolute Error:0.50
Mean Squared Error:0.49
R2 Score:0.75
```

Fig. 3.9. RoBERTa regression scores for Fine-Grained Sentiment Analysis regression model.

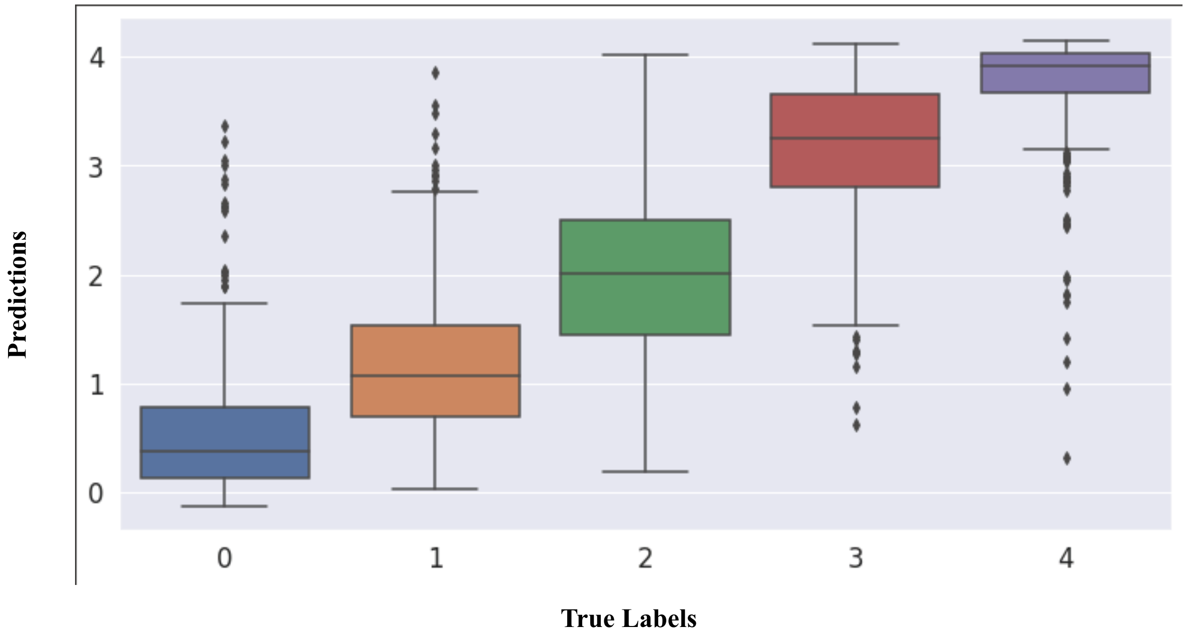
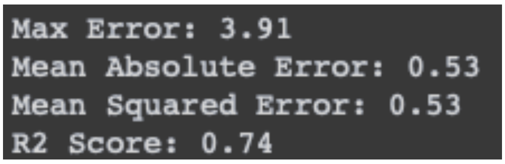


Fig. 3.10. RoBERTa prediction distributions for Fine-Grained Sentiment Analysis regression model.

The same architecture is used to train ALBERT. The resulting regression scores and the distribution of the predictions can be seen in Fig3.11 and Fig3.12.



```
Max Error: 3.91
Mean Absolute Error: 0.53
Mean Squared Error: 0.53
R2 Score: 0.74
```

Fig. 3.11. ALBERT regression scores for Fine-Grained Sentiment Analysis regression model.

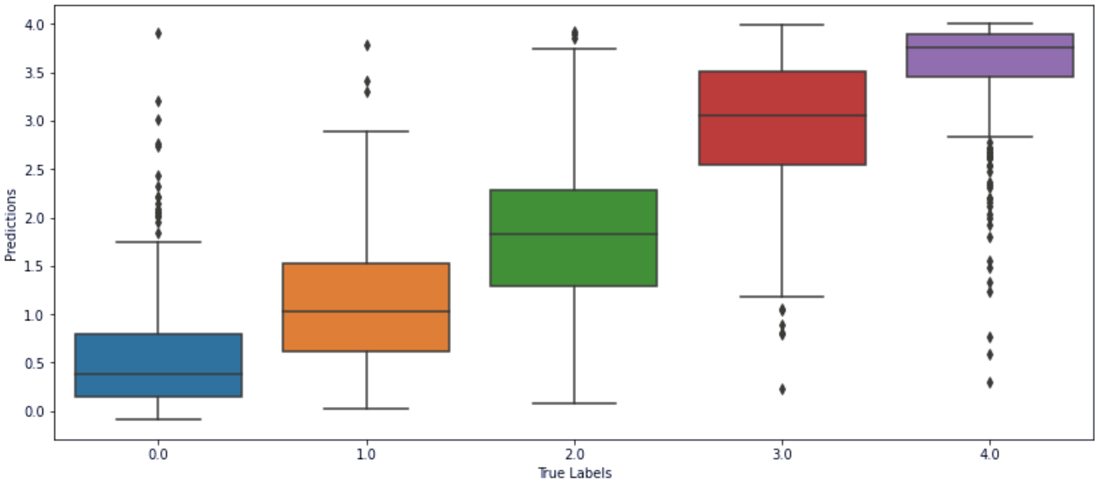


Fig. 3.12. ALBERT prediction distributions for Fine-Grained Sentiment Analysis regression model.

The comparison of regression and the classification models are made in Table 3.2.

	Classification F1	Classification MSE	Regression MSE
RoBERTa	0.62	0.57	0.50
ALBERT	0.60	0.65	0.53

Table 3.2: Comparison of performance of RoBERTa and ALBERT Fine-Grained sentiment models.

One can see that by using the regression model we can get better MSE scores for both the models. This is expected because in regression we use MSE as the loss function and try to minimize it. Furthermore, RoBERTa model again outperforms the ALBERT model on all metrics. This time the difference between two models are greater due to the fact that the fine-grained sentiment analysis is a harder task than basic-sentiment analysis and extra parameters used in RoBERTa improves the score.

CHAPTER 4
ASPECT-BASED SENTIMENT ANALYSIS

4.1 Dataset

The aim in aspect-based sentiment analysis is to categorize the data by aspect and find the sentiment attributed to each aspect. SemEval-2014 Restaurant Review dataset [19] is used for this purpose. This dataset consists of 3044 restaurant review texts with aspect terms and their corresponding sentiment labeled. For example, the review “The price is reasonable although the service is poor.” includes 2 aspect terms: “price” and “service”. For “price” the corresponding sentiment is “positive”, whereas for “service” the corresponding sentiment is “negative”.

Hence, Aspect-Based Sentiment Analysis consists of 2 parts:

1. Aspect term detection: Determine the term in sentence that has a corresponding sentiment.
2. Sentiment analysis: Determine the polarity of the sentiment associated with the aspect term.

The dataset includes 4 types of sentiment polarities: “positive”, “negative”, “neutral” and “conflict”. “Conflict” is when both a positive and a negative review is made on an aspect term. For instance the review “The service varies from day to day-sometimes they're very nice, and sometimes not.” includes both a positive and a negative opinion about aspect term “service”; hence, the sentiment polarity is “conflict”. The dataset is not balanced and the number of reviews for “neutral” and “conflict” is way less than the number of reviews for “positive” and “negative”. Therefore, weighting should be utilized in loss function to handle the imbalance.

The longest review in the dataset consists of 91 words, so the maximum sequence length is chosen as 128 to save memory. The reviews shorter than 128 tokens after encoding are padded to the maximum sequence length. The 70% of the dataset is used in training 15% of dataset is used in testing and 15% of dataset is used in validation.

4.2 Aspect Term Detection

In order to detect the aspect terms the last hidden state of each token is utilized to determine whether the token belongs to an aspect term. Given the input token sequence $\mathbf{x} = \{x_1, \dots, x_T\}$ of length T, BERT is employed with 12 transform layers and the last hidden state $H^{12} = \{h_1^{12}, \dots, h_T^{12}\} \in \mathbb{R}^{T\times768}$ is fed to the aspect detection layer to predict the tag sequence $\mathbf{y} = \{y_1, \dots, y_T\}$, where $y_t \in \{B, I, O\}$. B represents the beginning of the aspect term I represents the inside of the aspect term and O represents the outside of the aspect term. Overall architecture of the model can be seen in Fig4.1.

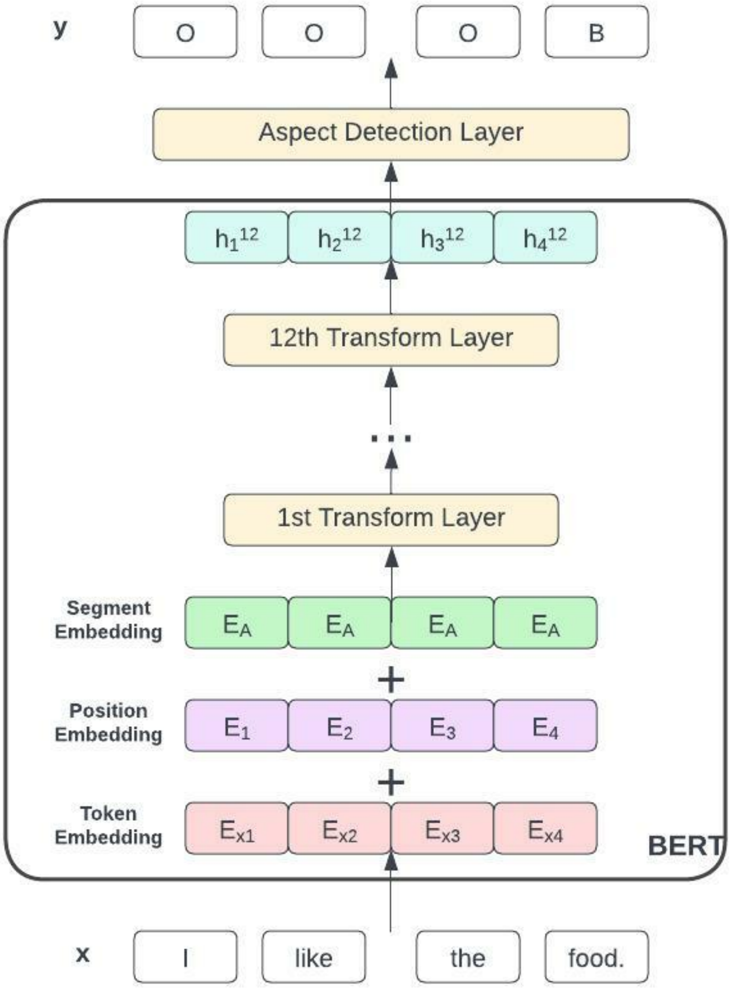
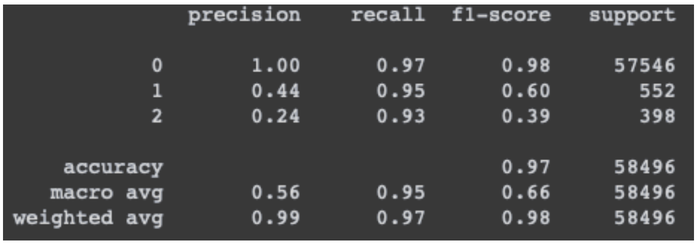


Fig. 4.1. Overall architecture of the Aspect Term Detection Model

Because the last hidden state is used as the input and the maximum sequence length is chosen as 128 the input to the aspect detection layer is 128x768. As before, two dense linear layers of size 768x512 and 512x3 are used to get the output matrix of size 128x3, where each column represents the probability of the token belonging to the tag B, I or O. Between these two layers a rectified linear unit function is applied as it helps to reduce the likelihood of the gradient to vanish and introduces sparsity. The dropout layer is used with dropout probability 0.1 to prevent overfitting and regularize. AdamW optimizer is used with $\beta_1 = 0.9$, $\beta_2 = 0.999$, base learning rate 2e-5 and ϵ = 1e-8. No warm up state is used and the learning rate is decreased linearly to 0. Only few of the tokens are aspect terms and therefore most of the tokens are tagged O. This creates a huge imbalance in dataset. Hence class weights should be considered in calculating the loss. Negative log likelihood function is used to calculate the loss. The number of epochs is set to 4 due to time considerations and to prevent overfitting. The classification report and the confusion matrix of the resulting model can be seen in Fig4.2 and Fig4.3. 0 is Outside, 1 is Begin and 2 is Inside.



```
              precision    recall  f1-score   support

           0       1.00      0.97      0.98     57546
           1       0.44      0.95      0.60       552
           2       0.24      0.93      0.39       398

    accuracy                           0.97     58496
   macro avg       0.56      0.95      0.66     58496
weighted avg       0.99      0.97      0.98     58496
```

Fig. 4.2. RoBERTa classification report for Aspect Term Detection model

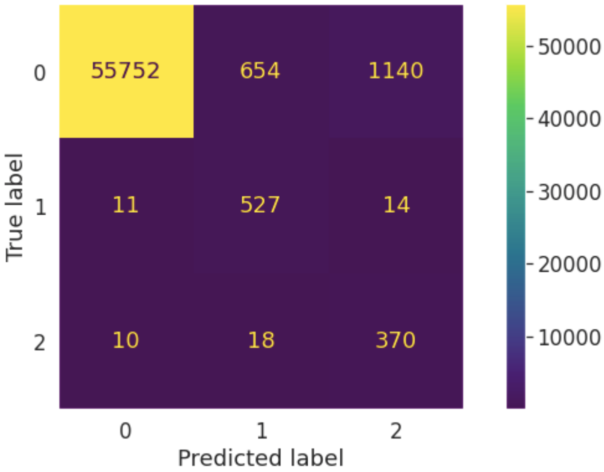
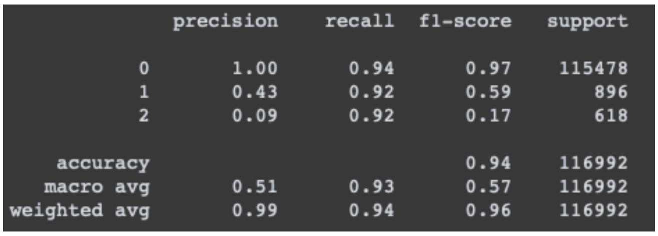


Fig. 4.3. RoBERTa confusion matrix for Aspect Term Detection model

One can see from Fig4.2 and Fig4.3 that although the recall values are good the precision for 1(B) and 2(I) are low and there are many false alarms. This means that our model tags some terms as aspect terms although they are not.

Deep learning models perform better as the number of samples increases. The Restaurant Review dataset only contains 3044 and it is very low to train a deep learning model. Therefore Laptop Reviews dataset from SemEval-2014 [19] is also added to increase the number of samples. Laptop Reviews dataset has 3048 reviews and in total they have 6092 reviews. This combined dataset is used to train the Aspect Term Detection model and the results can be seen in Fig4.4 and Fig4.5.



```
              precision    recall  f1-score   support

           0       1.00      0.94      0.97    115478
           1       0.43      0.92      0.59       896
           2       0.09      0.92      0.17       618

    accuracy                           0.94    116992
   macro avg       0.51      0.93      0.57    116992
weighted avg       0.99      0.94      0.96    116992
```

Fig. 4.4. RoBERTa classification report for Aspect Term Detection model using restaurant and laptop dataset.

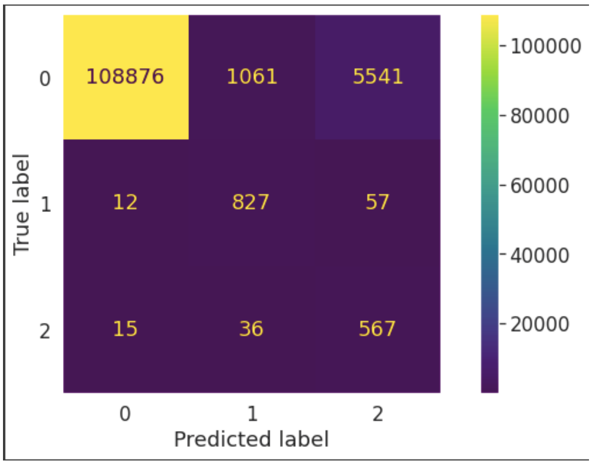
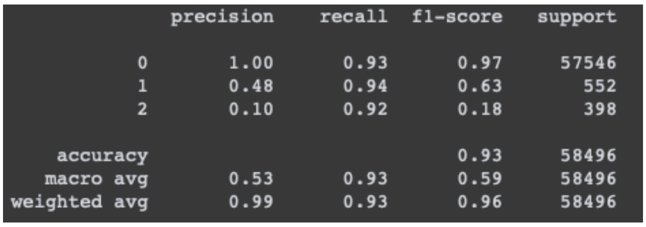


Fig. 4.5. RoBERTa confusion matrix for Aspect Term Detection model using restaurant and laptop dataset.

From Fig4.4 it can be seen that combining two datasets actually decreased the F1 scores. This is again due to the importance of the domain knowledge. Specifically in aspect term detection the domain knowledge becomes very important to detect the terms that are mainly used in that domain. Therefore, only the restaurant dataset is used next.

In order to improve the aspect term detection model another architecture is tried. In [20] it is suggested that using other intermediate layers instead of just the last layer enhances the performance of fine tuning of BERT. Therefore, this time instead of using the last hidden state as the input to the aspect detection layer the concatenation of last 4 hidden states are used. The other parts of the architecture are kept the same and the model is trained. The classification report and the confusion matrix of the resulting model can be seen in Fig4.6 and Fig4.7.



```
              precision    recall  f1-score   support

           0       1.00      0.93      0.97     57546
           1       0.48      0.94      0.63       552
           2       0.10      0.92      0.18       398

    accuracy                           0.93     58496
   macro avg       0.53      0.93      0.59     58496
weighted avg       0.99      0.93      0.96     58496
```

Fig. 4.6. RoBERTa classification report for Aspect Term Detection model using last 4 hidden states

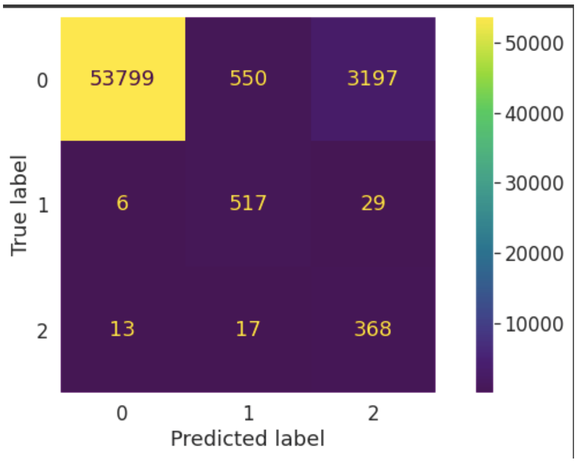


Fig. 4.7. RoBERTa confusion matrix for Aspect Term Detection model using last 4 hidden states

One can see that no improvement is seen in the model after utilizing the last 4 hidden states. Therefore, this method is abandoned.

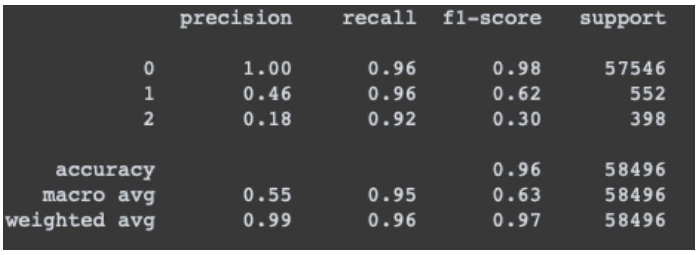
As stated before the domain information is thought to be very critical in determining the aspect terms. Knowing the domain and domain specific terms could be beneficial for finding the aspect terms and improve the performance of our model. Therefore, to exploit the domain information our model can be retrained for Masked Language Modeling (MLM) task. As stated before, this task is used to train the BERT and help it learn the basics of a natural language. In order to make the model a domain specific we can retrain it for MLM using a domain specific dataset. For this purpose, Yelp review dataset [21] is used. This dataset includes millions of restaurant reviews. However, because of computational time considerations only 20,000 reviews are used to retrain a RoBERTa model, which is not a large number for a retraining task. Before retraining, the RoBERTa model fills the mask in sentence "<mask> was very helpful." as follows:

He was very helpful.	probability: 0.3
It was very helpful.	probability: 0.2
That was very helpful.	probability: 0.17

The Yelp dataset consists of long reviews. Therefore maximum sequence length is chosen as 512. 15% of the input tokens are randomly masked and the model is trained to estimate the masked tokens. The batch size is set to 24 and the number of epochs is set to 4. After retraining the model is tested to fill the mask in the same sentence “<mask> was very helpful.” and the top three estimations were as follows:

Staff was very helpful.	probability: 0.61
Everyone was very helpful.	probability: 0.05
Service was very helpful.	probability: 0.03

One can see from the results that the domain information is learnt by the model. Now this retrained model can be fine-tuned for the aspect term detection task. The resulting classification report and the confusion matrix are in Fig4.8 and Fig4.9.



```
              precision    recall  f1-score   support

           0       1.00      0.96      0.98     57546
           1       0.46      0.96      0.62       552
           2       0.18      0.92      0.30       398

    accuracy                           0.96     58496
   macro avg       0.55      0.95      0.63     58496
weighted avg       0.99      0.96      0.97     58496
```

Fig. 4.8. RoBERTa classification report for Aspect Term Detection model using retrained model

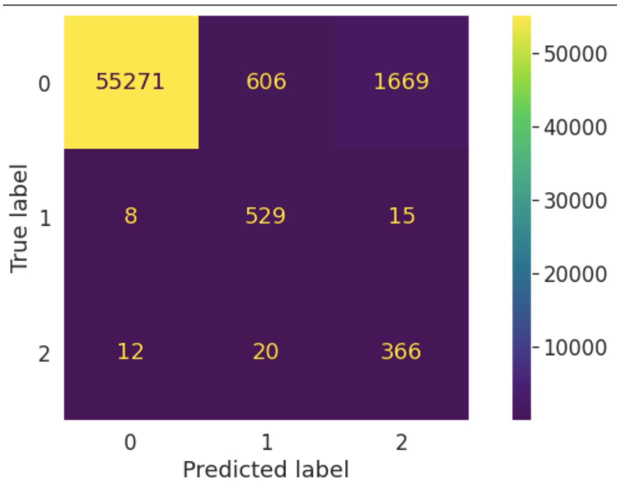


Fig. 4.9. RoBERTa confusion matrix for Aspect Term Detection model using retrained model

Unfortunately, exploiting the domain knowledge by retraining didn’t improve the performance of the aspect term detection model. Hence, this method is also abandoned.

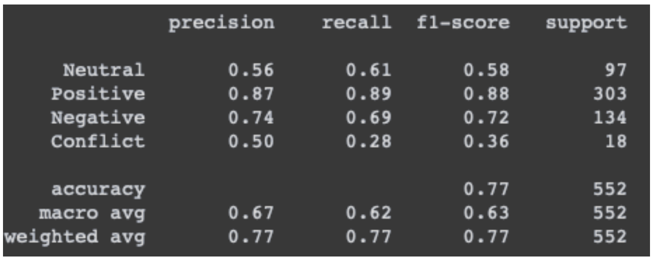
The same procedures are repeated for ALBERT using the same architectures and the results are summarized in Table 4.1.

	Last HS Model F1	Last 4 HS Model F1	Retrained Model F1
RoBERTa	0.66	0.59	0.63
ALBERT	0.59	0.58	0.58

Table 4.1: Comparison of performance of RoBERTa and ALBERT Aspect Term Detection models.

4.3 Aspect Based Sentiment Analysis

In order to detect the sentiment polarity associated with the aspect term the aspect term is fed together with the review into the model. The review and the aspect term are separated with the special [SEP] token and the last hidden state of the [CLS] token is used to classify the sentiment polarity associated with the aspect term as positive, negative, neutral or conflict. The same BERT architecture is used as before and the output is set to size 4 to represent the probability of each sentiment class. Batch size is set to 32 and the number of epochs is set to 8. The model is trained on the training set assuming that the aspect terms are known just to focus on the sentiment analysis part. The resulting classification report and the confusion matrix are in Fig4.10 and Fig4.11.



```
              precision    recall  f1-score   support

     Neutral       0.56      0.61      0.58        97
    Positive       0.87      0.89      0.88       303
    Negative       0.74      0.69      0.72       134
    Conflict       0.50      0.28      0.36        18

    accuracy                           0.77       552
   macro avg       0.67      0.62      0.63       552
weighted avg       0.77      0.77      0.77       552
```

Fig. 4.10. RoBERTa classification report for ABSA

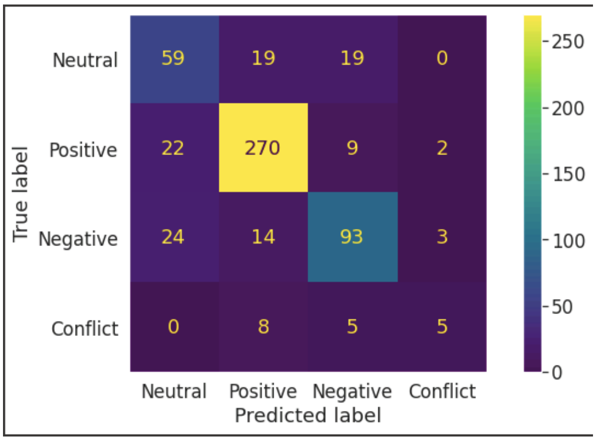
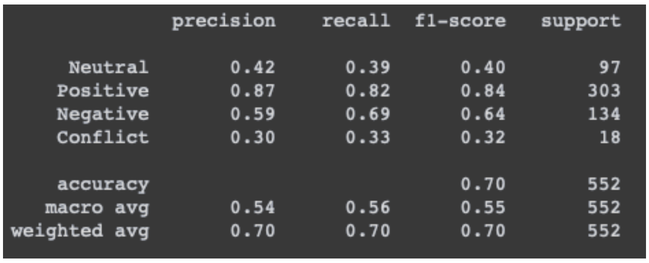


Fig. 4.11. RoBERTa confusion matrix for ABSA

One can see from Fig4.11 that classifying 'Conflict' labels is the hardest task. This is expected as it involves both a negative and a positive sentiment. A couple of observations about classifying the 'Conflict' reviews can be made as follows:

BERT is good at classifying the 'Conflict' labels when the grammar is simple and there is a pronoun that refers to the aspect term. For example in the review "There is usually a wait but it is well worth it." "wait" is said to be well worth it and referred to by using the pronoun "it", which makes the sentence grammatically simple and easy to find the conflict. On the other hand, both the reviews "The decor is nice, but more casual than fine dining" and "Even with a relatively inexpensive bottle of wine, if you can call $70.00 inexpensive, the cost is through the roof for better than average fare." are classified as "Positive". Both these examples have the explicit "Positive" sentiment but an implicit "negative" sentiment that is not grammatically expressed. Hence it is hard to classify them.

The same procedures are repeated for ALBERT using the same architectures and the results can be seen in Fig4.12 and Fig4.13.



```
              precision    recall  f1-score   support

     Neutral       0.42      0.39      0.40        97
    Positive       0.87      0.82      0.84       303
    Negative       0.59      0.69      0.64       134
    Conflict       0.30      0.33      0.32        18

    accuracy                           0.70       552
   macro avg       0.54      0.56      0.55       552
weighted avg       0.70      0.70      0.70       552
```

Fig. 4.12. ALBERT classification report for ABSA

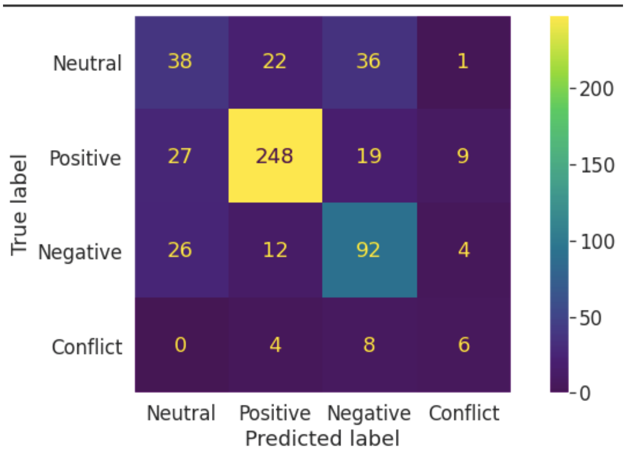


Fig. 4.13. ALBERT confusion matrix for ABSA

One can see that ALBERT performs worse than RoBERTa and the difference is even greater than fine-grained sentiment analysis.

4.4 Combination of Two Models

The aspect term detection model and the aspect based sentiment analysis model are combined and tested on SemEval-2014 test dataset [19]. Some observations can be made on the results.

review: I trust the people at Go Sushi, it never disappoints.
detected aspect terms: people (positive), Go Sushi (positive)
actual aspect terms: Go Sushi (positive)

In this review although it is not labeled as the aspect term the review includes a positive sentiment for people but maybe it is not included as it is not a domain specific term. This shows that the dataset has mislabels. This is one of the biggest constraints in sentiment analysis. Because labeling datasets is a subjective task it is prone to errors such as these. This makes it harder to train the model and misleads it.

review: Certainly not the best sushi in New York, however, it is always fresh, and the place is very clean, sterile.
detected aspect terms: sushi in New York (negative), place (positive)
actual aspect terms: sushi (negative)

Here although the actual aspect term is just “sushi” “sushi in New York” can also be considered the “aspect term”. Because the aspect term detection doesn’t have a single ground truth it is hard to evaluate the result and although the result isn’t wrong here it is considered a misclassification.

review: The portions of the food that came out were mediocre.
detected aspect terms: portions of the (negative), food (negative)
actual aspect terms: portions of the food (negative)

Here although they are parts of the same aspect term "portions" and the "food" are considered two separate aspect terms. This is probably because our model thought "mediocre" refers bot the portions and the food.

4.5 Another Approach to Aspect Based Sentiment Analysis

Instead of using two separate models for aspect term detection and sentiment analysis, one single model that carries out both tasks can also be used. In this model the aspect terms are tagged with their corresponding sentiments. For example in this case the available output tags are **y** = {O, B-POS, I-POS, B-NEG, I-NEG, B-NEU, I-NEU, B-CON, I-CON} for outside the aspect term, the beginning of the aspect term that has positive sentiment, inside of the aspect term that has positive sentiment, the beginning of the aspect term that has negative sentiment, inside of the aspect term that has negative sentiment, the beginning of the aspect term that has neutral sentiment, inside of the aspect term that has neutral sentiment, the beginning of the aspect term that has conflict sentiment and inside of the aspect term that has conflict sentiment respectively. The overview of the new model can be seen in Fig4.14.

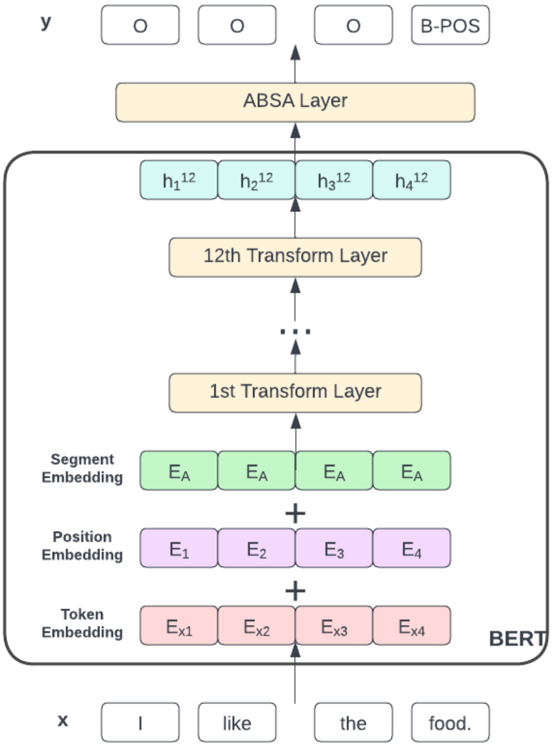
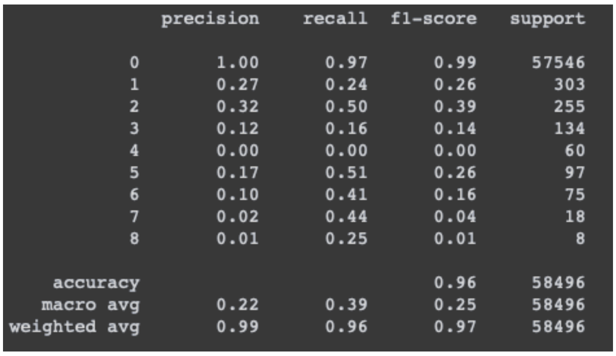


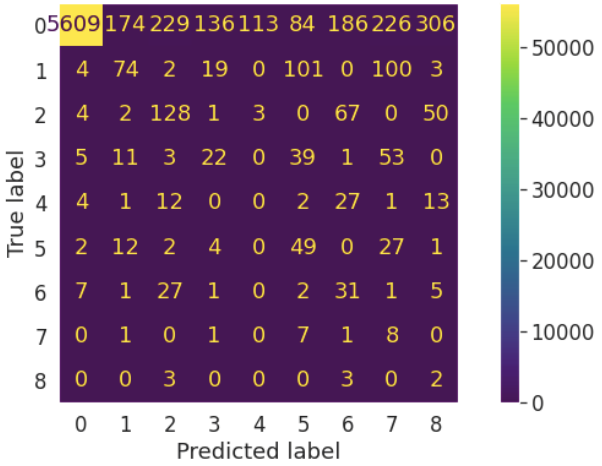
Fig. 4.14. Overall architecture of the Aspect Term Detection Model

This model is trained and tested on the same Restaurant Reviews dataset as before and the resulting classification report and the confusion matrix can be seen in Fig4.15 and Fig4.16.



```
              precision    recall  f1-score   support

           0       1.00      0.97      0.99     57546
           1       0.27      0.24      0.26       303
           2       0.32      0.50      0.39       255
           3       0.12      0.16      0.14       134
           4       0.00      0.00      0.00        60
           5       0.17      0.51      0.26        97
           6       0.10      0.41      0.16        75
           7       0.02      0.44      0.04        18
           8       0.01      0.25      0.01         8

    accuracy                           0.96     58496
   macro avg       0.22      0.39      0.25     58496
weighted avg       0.99      0.96      0.97     58496
```

Fig. 4.15. RoBERTa classification report for another ABSA approach



True label \ Predicted label	0	1	2	3	4	5	6	7	8
0	5609	174	229	136	113	84	186	226	306
1	4	74	2	19	0	101	0	100	3
2	4	2	128	1	3	0	67	0	50
3	5	11	3	22	0	39	1	53	0
4	4	1	12	0	0	2	27	1	13
5	2	12	2	4	0	49	0	27	1
6	7	1	27	1	0	2	31	1	5
7	0	1	0	1	0	7	1	8	0
8	0	0	3	0	0	0	3	0	2

Fig. 4.16. RoBERTa confusion matrix for another ABSA approach

One can see that this model gives much worse results than our previous combined model. Therefore this approach is abandoned.

CHAPTER 5
CONCLUSION

5.1 Results

In conclusion it can be said that BERT-based methods perform well for the NLP task called Sentiment Analysis. For Basic Sentiment Analysis we proposed a simple BERT architecture for classification. RoBERTa and ALBERT gives very similar results on Basic Sentiment Analysis with ALBERT giving a much small and compact model due to the smaller number of parameters. Therefore ALBERT model can be chosen for this case. For Fine-Grained Sentiment Analysis we proposed a regression BERT architecture. It is shown that the difference between ALBERT and RoBERTa grows in Fine-Grained Sentiment analysis. Hence it is better to choose RoBERTa in this case. For Aspect Term Detection task we proposed a model that tags each token by using the last hidden states. Finally, for the Aspect Term Sentiment Analysis we proposed a model to use the aspect term and the review together to find the sentiment related to the term and the review. For ABSA RoBERTa also outperforms ALBERT by a great amount and hence it should be preferred. The overall comparison between RoBERTa and ALBERT can be seen in Table 5.1.

	Basic Sentiment Analysis F1	Fine Grained Sentiment Analysis MSE	Aspect Term Detection F1	Aspect Based Sentiment Analysis F1
RoBERTa	0.94	0.50	0.66	0.63
ALBERT	0.59	0.53	0.59	0.55

Table 5.1: Comparison of performance of RoBERTa and ALBERT Aspect Term Detection models.

5.2 Realistic Constraints

In all NLP tasks one of the biggest constraints is the availability and the quality of data. It is a fact that the more data a deep learning structure utilizes the better its scores get. However, bigger data means more labeling and it is usually not an easy task to label an NLP dataset as it can be very subjective and sometimes sensitive to human perception. Moreover, since the data is usually collected from the web it is not very high quality or reliable. There are many abbreviations or slang words that are used, which makes the dataset harder to work with.

5.3 Social, Environmental and Economic Impact

Sentiment analysis can help businesses, governments and companies to understand their workers', people's and customers' needs better and take action accordingly. By utilizing sentiment analysis and understanding people's needs faster businesses can make more profit and this will create a positive impact to the economy. Moreover, understanding the people's needs helps the demands in our society to be supplied faster and therefore sentiment analysis also has a positive effect on social environment.

5.4 Cost Analysis

Fortunately the tools used in this project have very little cost and there are not going to be any mechanical parts that I will build. Therefore the economical costs will be minimal. I used Google Colab Pro which is monthly 138₺. Moreover, I spent approximately 80 hours on this project. An average engineer makes $35.35 an hour so this makes $2828 for labor costs. Hence in total the cost is $2836.

5.5 Standards

This project complies to Standard Personal Data Artificial Intelligence Agent (IEEE P7006), the General Data Protection Regulation by EU, Global Initiative for Ethical Considerations in Artificial Intelligence (IEEE P7000TM). Moreover, IEEE code of ethics is conformed during the making of the project. In addition, IEEE conference paper template is used in writing the project report.

BIBLIOGRAPHY

BIBLIOGRAPHY

1. Chen, T., Liu, S., Chang, S., Cheng, Y., Amini, L., & Wang, Z. (2020). Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 699-708).
2. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
3. Church, K. W. (2017). Word2Vec. *Natural Language Engineering*, *23*(1), 155-162.
4. Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532-1543).
5. Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.
6. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
7. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.
8. Basiri, M. E., Nemati, S., Abdar, M., Cambria, E., & Acharya, U. R. (2021). ABCDM: An attention-based bidirectional CNN-RNN deep model for sentiment analysis. *Future Generation Computer Systems*, *115*, 279-294.
9. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, *32*.
10. Go, A., Bhayani, R., & Huang, L. (2009). Twitter sentiment classification using distant supervision. *CS224N project report, Stanford*, *1*(12), 2009.
11. M. Thelwall, *Sentiment Strength Twitter Dataset*, University of Wolverhampton: 2012. Available: http://sentistrength.wlv.ac.uk/documentation/?C=D;O=A
12. Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher Manning, Andrew Ng and Christopher Potts, *Stanford Sentiment Treebank*, University of Stanford: 2013. Available: https://nlp.stanford.edu/sentiment/code.html
13. N. Lakshmipathi, IMDB Movie Review Dataset, 2019. Available: https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
14. Kavita Ganesan, Cheng Xiang Zhai, *OpinRank Review Dataset*, 2011. Available: https://archive.ics.uci.edu/ml/datasets/opinrank+review+dataset
15. J. McAuley, *Amazon Product Data*, 2018. Available: https://jmcauley.ucsd.edu/data/amazon/
16. Kingma, D. P., & Adam, J. B. (2015). A Method for Stochastic. *Optimization. In, ICLR*, *5*.
17. Baevski, A., Edunov, S., Liu, Y., Zettlemoyer, L., & Auli, M. (2019). Cloze-driven pretraining of self-attention networks. *arXiv preprint arXiv:1903.07785*.
18. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le QV, X. (2021). generalized autoregressive pretraining for language understanding; 2019. *Preprint at https://arxiv. org/abs/1906.08237 Accessed June*, *21*.
19. Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In SemEval, pages 27–35.
20. Song, Y., Wang, J., Liang, Z., Liu, Z., & Jiang, T. (2020). Utilizing BERT intermediate layers for aspect based sentiment analysis and natural language inference. *arXiv preprint arXiv:2002.04815*.
21. Asghar, N. (2016). Yelp dataset challenge: Review rating prediction. *arXiv preprint arXiv:1605.05362*.