Comparing the use of LSTM and BERT for Text Classification

Haemin Choi (201807000)

1. Research Hypothesis and Objectives

Text classification is a machine learning technique that assigns a set of predefined categories to open-ended text and is a classic problem in Natural Language Processing (NLP). In this research proposal, the task of text classification is implemented using two different models, Long-short term memory (**LSTM**) and Bidirectional Encoder Representations from Transformers (**BERT**). To train LSTM for text classification, each of the word embeddings, Word2vec and GloVe, is used to help the model reflect the similarity and relationships among the words. LSTM with word embeddings and BERT would be trained with different datasets for the text classification tasks, and their results would be analyzed in quantitive and qualitive perspectives.

With text being one of the most common types of unstructured data, it is usually estimated that around 80% of all information is unstructured. Because of the mess nature text data has, there has been a lot of research to analyze, understand, organize, and sort through text data so that people can use it to its full potential. This is where text classification with machine learning plays an important role. Text classification can automatically structure all manner of relevant text from various businesses, for example, from emails, legal documents, social media, chatbots, surveys and more. This enables saving time analyzing text data and automating business processes, so that many companies can make data-driven business decisions.

Particularly the use of machine learning text classification automatically analyzes millions of text data at a fraction of the cost with its scalability and applies consistent criteria while human annotators can make mistakes when processing and classifying textual data manually. Furthermore, once machine learning text classifier properly applied to a business environment, it can follow and identify critical information in real-time with the needs of users to take an action right away.

Substantial work has shown that pre-trained models on large corpus are beneficial for text classification and other NLP tasks, which can avoid training a new model from scratch. These pre-trained models include from word embeddings such as Word2vec and GloVe to sentence-level pretrained model such as OpenAI GPT and BERT. The word embeddings are often used as additional features for the main task, and the large language models utilize a large amount of unlabeled data. Although BERT has achieved amazing results and regarded as state-of-art pre-trained language model on several natural language processing tasks, there is little research that mainly aims to compare it to LSTM with word embedding. This research proposal could be critical and novel considering that it is not the always best to use large language model when designing text classifier and there are some cases when it is suitable to use relatively lighter model.

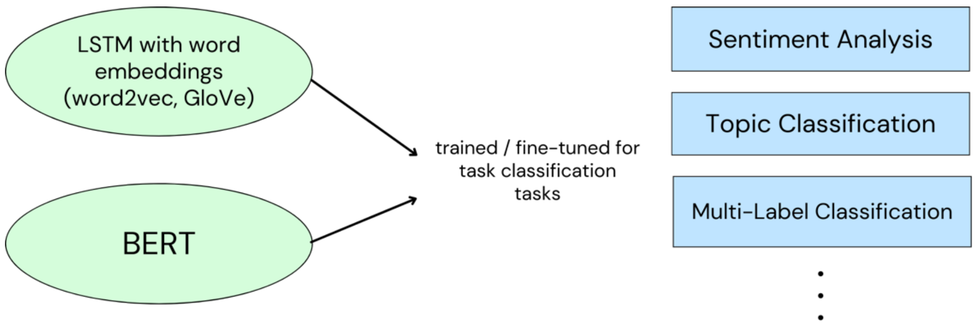

LSTM with word embeddings (word2vec, GloVe)
BERT
trained / fine-tuned for task classification tasks
Sentiment Analysis
Topic Classification
Multi-Label Classification

Figure 1: Simple research pipeline comparing two models on task classification tasks

The overall aim of the project is to develop useful text classifiers that can be applied on different environments by training two different neural network models, LSTM and BERT on text classification task. The project also aims to suggest how the experiment results of the two models vary when trained on different datasets with different classification tasks (e.g. sentiment analysis, topic classification, multi-label classification). By developing machine learning based text classifiers and analyzing their results, it is expected that the research can identify which model is suitable for which text classification task, suggesting the potential for being applied in real world.

2. Background

The history of text classification dates back to the early days of information retrieval and natural language processing. Text classification is the process of assigning predefined categories or labels to textual documents based on their content. Its evolution is closely tied to advancements in machine learning and computational linguistics.

In the 1990s, with the advent of the internet and the exponential growth of digital textual data, the need for automated text classification became more apparent. Various research began investigating machine learning approaches for text classification, including techniques such as naive Bayes, decision trees, and support vector machines (SVM). One of the seminal works in text classification is the "A Maximum Entropy Approach to Natural Language Processing" (Berger et al., 1996), which introduced the maximum entropy model for text classification. This approach laid the groundwork for probabilistic models that could effectively handle large-scale text classification tasks.

In the mid-2000s, there was the emergence of deep learning techniques, especially deep neural networks, as powerful tools for text classification. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) including LSTM networks began to outperform traditional machine learning algorithms on various text classification tasks. More recent advancements in text classification include the application of transfer learning techniques, such as pre-trained language models like BERT and GPT, which have significantly improved the performance of text classification models by leveraging large-scale pre-training on diverse text corpora.

This project focuses on the implementation of text classification with two machine learning models, LSTM and BERT. "Long Short-Term Memory Networks for Machine Reading" (Sutskever et al., 2014) introduced the application of LSTM networks for text classification and machine reading tasks. LSTMs are a type of recurrent neural network (RNN) architecture designed to capture long-term dependencies in sequential data. In this research, LSTM networks were applied to tasks such as sentiment analysis and document classification, demonstrating their ability to effectively model the contextual information in text data. The research showed that LSTM networks outperformed traditional machine learning algorithms and other RNN variants on various text classification benchmarks, opening the way for further investigation of LSTM-based models in NLP tasks.

"Hierarchical Attention Networks for Document Classification" (Yang et al., 2016) proposed a hierarchical attention network architecture for document classification tasks, leveraging LSTM units to capture the hierarchical structure of documents. By incorporating LSTM units and attention mechanisms, the proposed model achieved state-of-the-art performance on several text classification datasets, demonstrating the effectiveness of LSTM-based architectures in handling document-level information and capturing fine-grained semantic cues.

"BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" (Devlin et al., 2018) introduced BERT, a pre-trained language representation model based on the transformer architecture. Being pre-trained on diverse textual data, BERT learns contextualized word embeddings that capture abundant semantic information, enabling it to achieve state-of-the-art performance on text classification tasks. Fine-tuning BERT on text classification datasets requires some task-specific architecture modifications, making it effective for various text classification tasks with limited labeled data.

“How to Fine-Tune BERT for Text Classification?” (C Sun et al., 2020) provided practical insights for effectively fine-tuning BERT models for text classification tasks. Focusing on addressing key considerations such as model architecture, hyperparameter tuning, and evaluation metrics, it compares the performance of different fine-tuning strategies and hyperparameters, highlighting their impact on classification accuracy and computational efficiency.

This research mainly investigates the implementation of various text classification tasks using LSTM and BERT and focuses on comparing the performance of the two models. Based on related research which explore each of them in detailed ways, this project aims to further figure out what is the key difference of the two models when used as text classifiers and how they can be used in real world applications in practical and effective ways.

3. Importance and Contribution to Knowledge

In the of modern society full of textual data generated every day, investigation about the development of text classification technology plays a pivotal role for economic success and addressing key societal challenges.

Application of text classification enables businesses to analyze customer feedback, inquiries, and sentiments at scale, leading to improved customer service and experience. By automatically categorizing and prioritizing incoming messages, companies can respond more efficiently to customer needs and resolve issues promptly. For example, Zendesk, a customer service software company, employs text classification algorithms to automatically categorize incoming support tickets based on their content.

Text classification algorithms also can be used to power recommendation systems that deliver personalized content to users based on their preferences and behavior. By analyzing text data from user interactions, the systems can suggest relevant articles, products, or services, thereby enhancing user engagement and driving revenue growth. A streaming service Netflix is known to utilize machine learning based text classification to analyze user reviews, viewing history, and preferences to recommend personalized content to its subscribers. According to Netflix, their recommendation system contributes to over 80% of content watched on the platform, highlighting its effectiveness in driving user engagement and retention.

Moreover, text classification plays a crucial role in identifying and mitigating misinformation and fake news spread in online platforms and social media. By analyzing textual content and identifying unreliable sources or misleading information, the integrity of information can be improved, and we can protect public discourse. For instance, fact-checking organizations like Snopes and PolitiFact leverage text classification algorithms to assess the accuracy of news articles and social media posts.

Text classification system can assist healthcare providers in analyzing electronic health records (EHRs), medical literature, and patient feedback to enhance diagnostic accuracy, treatment planning, and patient care. By extracting insights from unstructured data using automated text classification, healthcare providers can do evidence-based decision making and improve healthcare outcomes. IBM Watson for Oncology uses text classification technology to analyze medical literature and patient records to assist oncologists in developing personalized treatment plans for cancer patients. According to Somashekha SP et al. (2018), Watson for Oncology provided treatment recommendations that aligned with expert oncologists' opinions in 96% of breast cancer cases.

Investigation and improvement of text classification can lead to facilitating access to information and education. It enables the organization and categorization of vast amounts of textual information, making it more accessible and navigable for users. Google's search engine employs text classification algorithms to understand user queries and retrieve relevant web pages from its index. According to Google, their search engine processes over 3.5 billion searches per day, providing users with access to a wealth of information across various domains.

Text classification also help detect signs of mental health issues, sentiment analysis, and identifying individuals at risk of self-harm or suicide. By analyzing textual data from social media, online forums, and mental health helplines, these algorithms can provide early intervention and support to vulnerable individuals. A non-profit organization Crisis Text Line utilizes text classification technology to analyze incoming messages from individuals in distress. Trained volunteers and mental health professionals use these insights to provide real-time support and resources to individuals experiencing mental health crises.

In conclusion, the continuous investigation and development of text classification system offers current and future economic success by enhancing customer service, driving user engagement, and enabling personalized experiences. Moreover, it addresses key societal challenges by combating misinformation, improving healthcare outcomes, facilitating access to information, and supporting mental well-bing. By leveraging the insights derived from textual data with text classification, organizations and policymakers can develop targeted interventions and initiatives to create positive social impact and build more resilient and inclusive communities.

4. Pilot Study

To demonstrate the feasibility of the project, an initial pilot study for binary text classification was implemented. The objective of the pilot study is to make two machine learning based useful text classifiers which detect sarcasm on news headlines, using LSTM and BERT. First, two types of LSTM model were trained, one of them using Word2vec embedding algorithm and the other using pre-trained GloVe word embedding. Secondly, pre-trained base version of BERT was fine-tuned to do the sarcasm detection task. Both text classification models were trained on News Deadlines Dataset for Sarcasm Detection, which consists of 28,619 headlines and 2 classes (1 or 0).

4.1. Case Study

Scola et al. (2021) explored several deep learning models such as Bidirectional Long Short-Term Memory (BiLSTM) and Bidirectional Encoder Representations from Transformers (BERT) to address the task of sarcasm detection. While most research has been implemented using textual data form social media, the researchers evaluated their models using a new headlines dataset. This is the first study to apply BERT for sarcasm detection in texts that do not come from social media, and the experiment results showed that the BERT-based approach overcomes the performance of BiLSTM and the state-of-the-art on this type of dataset, the hybrid neural architecture based on BiLSTM.

4.2. Acquiring Sample Data

In most of the past studies about sarcasm detection, twitter datasets collected using hashtag - based supervision were used. However, such datasets are noisy in terms of labels and language and require the availability of contextual tweets since many tweets are replies to other tweets.

News Headlines Dataset for Sarcasm Detection which is collected from two news website overcomes these limitations of Twitter datasets related to noise. The Onion aims at producing sarcastic versions of current events and all the headlines from News in Brief and News in Photos categories (which are sarcastic) were collected. Non-sarcastic news headlines were collected from Huffpost. Each record of the dataset consists of three attributes: *is_sarcastic*, *headline*, and *article_link*. *is_sarcastic* is 1 if the record is sarcastic, and otherwise it is 0. *headline* represents the headline of the news article, and *article_link* is the link to the original news article, which is useful in collecting supplementary data.

Ststistic / Dataset	Headlines	SemEval
Total Records	28,619	3,000
Sarcastic records	13,635	2,396
Non-sarcastic records	14,984	604

Table 1: The general statistics of the dataset along with SemEval-2018 Twitter dataset

According to the general statistics of the dataset, there are 28,619 records in total and 13,635 sarcastic records (labeled 1) and 14,984 non-sarcastic records (labeled 0). Compared to a high-quality Twitter dataset provided SemEval-2018 challenge, there are much more records in terms of quantity. Moreover, there are no spelling mistakes and informal usage since news headlines are written by professionals in a formal manner, which reduces the sparsity and increases the chance of finding pre-trained embeddings such as GloVe or Word2vec.

4.3. Sarcasm Detection with Bidirectional LSTM and BERT

For the implementation of pilot study, two types of LSTM model were trained, one of them using Word2vec embedding algorithm and the other using pre-trained GloVe word embedding. The news headline dataset was preprocessed before being used for training by getting rid of 151 stopwords in it and punctuation marks such as ‘,’, ‘.’, ‘-’, ‘/’. Preprocessed data was tokenized and padded to be ready to be used in the training process.

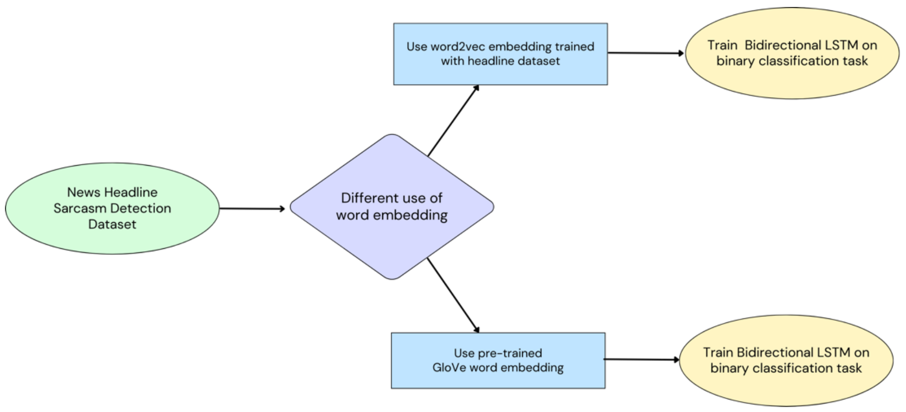

News Headline Sarcasm Detection Dataset
Different use of word embedding
Use word2vec embedding trained with headline dataset
Train Bidirectional LSTM on binary classification task
Use pre-trained GloVe word embedding
Train Bidirectional LSTM on binary classification task

Figure 2: Flowchart of training Bidirectional LSTM using different word embedding

First, a LSTM model was trained with word2vec embedding layer on it which is created by using words from the news headlines dataset. Word2vec genism model was imported to make a word embedding, and the vector size of it was set to be 50. The 'trainable' parameter of embedding layer was set to be 'True', and dropout was used on dense layers of the model to prevent overfitting.

Secondly, another LSTM model was trained with pre-trained GloVe word embedding on it which is created and pre-trained by using 12,00,000 words from twitter data. The dimension of embedding vector was set to be 25, which is the smallest size of the pre-trained embedding. The 'trainable' parameter of embedding layer was set to be 'False' since the pre-trained embedding layer does not need to be trained by the model.

	Accuracy	**Precision**	**Recall**	**F1 Score**
Word2Vec	0.8034	0.8119	0.7066	0.7556
GloVe	0.7126	0.6904	0.6012	0.6428

Table 2: Performance of LSTM with different word embeddings on sarcasm detection task

In terms of all performance score, LSTM trained with Word2vec word embedding outperformed LSTM with pre-trained GloVe embedding. LSTM with Word2vec embedding showed almost 80% accuracy in classifying sarcastic new headlines and the F1 score was 0.7556, which is moderately better than another one.

Lastly, BERT-base model was fine-tuned with the news headline dataset to do sarcasm detection. Data preprocessing was not conducted since BERT was designed to understand the overall context of sentences, without need to remove stopwords or punctuation marks.

	Accuracy	**Precision**	**Recall**	**F1 Score**
BERT-base	0.8735	0.8719	0.8703	0.8711

Table 3: Performance of BERT-base on sarcasm detection task

In terms of all performance score, BERT turned out to highly outperform two types of LSTM classifiers especially with about 87% accuracy.

4.4. Evaluation

Overall, different three text classifiers were experimented in the pilot study: two LSTM and fine-tuned BERT. Even though LSTM with Word2Vec outperforms another one with GloVe, it turned out that BERT, which is regarded as transformer based Large Language Model, has better performance technically. However, using BERT required more computing power than LSTM since it was inevitable to use GPU to fine-tune BERT, whereas LSTM required less time and memory.

5. Programme and Methodology

The research work programme makes use of a methodology for Artificial Intelligence projects, Cross-Industry Standard Process for Data Mining (CRISP-DM). It consists of 5 processes: business and data understanding, data preparation, modeling, evaluation, and deployment.

5.1. Business and Data Understanding

From business perspective, this project recognizes that a useful text classifier that can be applied to various industry domains is needed to automate different business processes that operates on textual data and information such as emails, legal documents, social media, chatbots, surveys and more. Moreover, it is considered that use of automated text classification technology in different business environments can solve business problems and lead to data-based decision making at the fraction of cost and time. Therefore, the research aims to develop two different text classifiers using bidirectional LSTM and BERT and figure out which model is appropriate to which task by comparing their performance on various text classification tasks, such as sentiment analysis, topic classification, multi-label classification.

Collection of initial data would be conducted by considering that datasets should be suitable for experimenting text classification tasks. To acquire suitable data, social media communities which share related research and data such as *Papers with Code*, *GitHub*, or *Kaggle* would be explored. The sources of data, the way it is collected, record and label format, number of records, or field identities would be examined. Exploratory Data Analysis (EDA) would be conducted by visualizing, identifying relationships among the data, and verifying the quality of data.

5.2. Data Preparation

IMDb movie review, Yahoo! Answers, and GoEmotions datasets would be used for each of the text classification tasks: sentiment analysis, topic classification, and multi-label classification. All datasets would be cleaned by correcting, imputing, or removing erroneous values. This preprocessing could include integrating data by combining data from multiple sources or re-formatting data in necessary.

5.3. Modeling

The modeling part of this project focuses on the use of neural network modeling techniques. Bidirectional LSTM would be built as a traditional neural network model, and pre-trained BERT would be loaded and fine-tuned for classification task as a way of using large language model. Two LSTM models would be built by using different word embeddings, Word2vec and pre-trained Glove to assess which kind of word embedding is useful when performing each of text classification tasks. Two LSTM models and fine-tuned BERT would be assessed on four classification criteria, accuracy, precision, recall, and f1 score. During all the modeling process python and its machine learning toolkit tensorflow would be used as a modeling tool.

5.4. Evaluation

Whereas technical model assessment would be implemented in the modeling process, which model meets the business objective would be assessed in the evaluation phase. Qualitive results of the models would be evaluated by considering if the models meet the business criteria or which one(s) should be approved for the business. Whether to proceed to deployment or iterate further would be determined based on the evaluation process.

5.5. Deployment

A plan for deploying text classification models in real world business would be documented. A thorough monitoring and maintenance plan would be developed to avoid issues during the operational phase or post-project phase of a model. Not only deployment plan but also a summary of the project would be documented by the project team so that it could include a final presentation of data mining results. Finally, a project retrospective about what went well, what could have been better, and how it can be improved in the future would be conducted by researchers to share any feedback about the research.

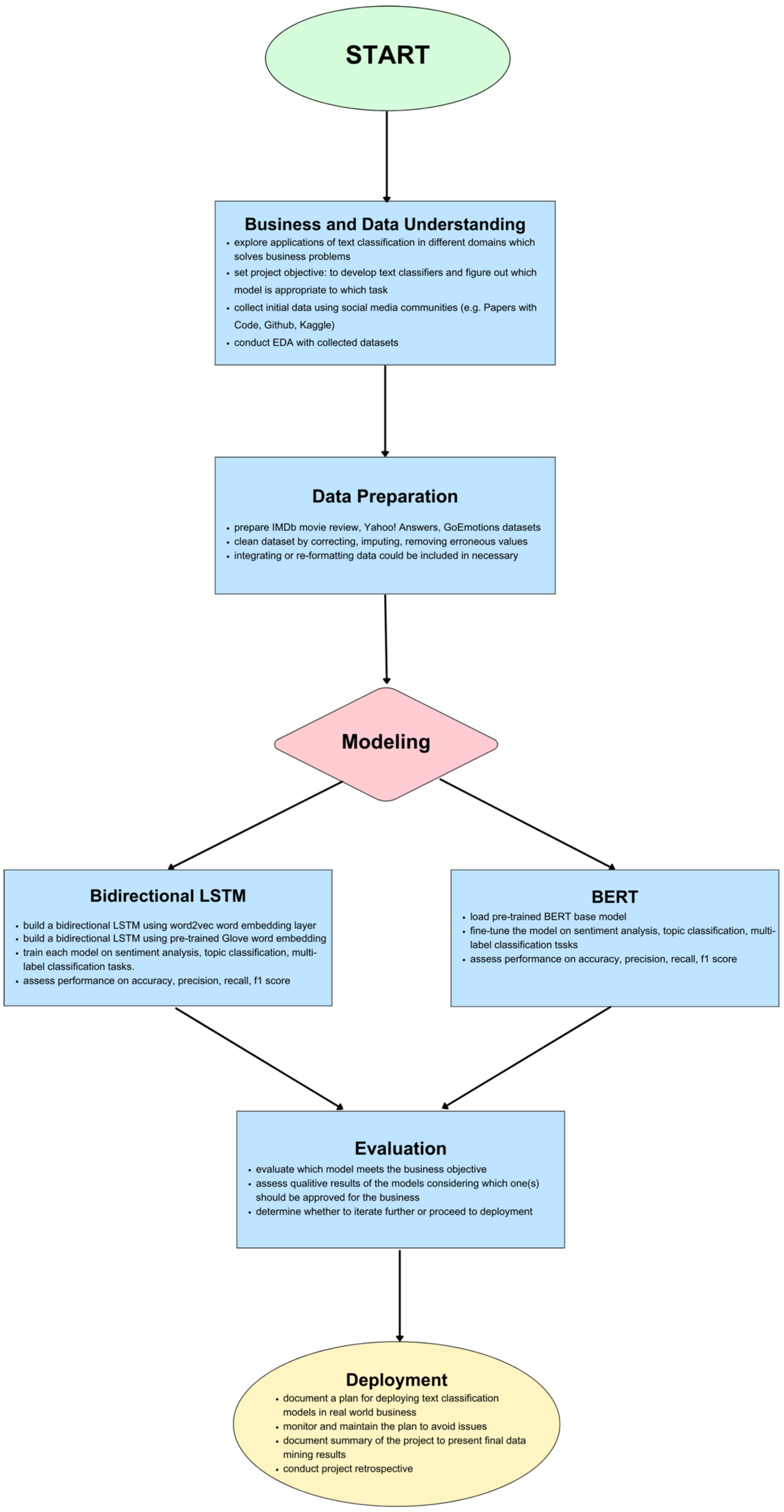

START
Business and Data Understanding
explore applications of text classification in different domains which solves business problems
set project objective: to develop text classifiers and figure out which model is appropriate to which task
collect initial data using social media communities (e.g. Papers with Code, Github, Kaggle)
conduct EDA with collected datasets
Data Preparation
prepare IMDb movie review, Yahoo! Answers, GoEmotions datasets
clean dataset by correcting, imputing, removing erroneous values
integrating or re-formatting data could be included in necessary
Modeling
Bidirectional LSTM
build a bidirectional LSTM using word2vec word embedding layer
build a bidirectional LSTM using pre-trained Glove word embedding
train each model on sentiment analysis, topic classification, multi-label classification tasks.
assess performance on accuracy, precision, recall, f1 score
BERT
load pre-trained BERT base model
fine-tune the model on sentiment analysis, topic classification, multi-label classification tssks
assess performance on accuracy, precision, recall, f1 score
Evaluation
evaluate which model meets the business objective
assess qualitive results of the models considering which one(s) should be approved for the business
determine whether to iterate further or proceed to deployment
Deployment
document a plan for deploying text classification models in real world business
monitor and maintain the plan to avoid issues
document summary of the project to present final data mining results
conduct project retrospective

Figure 3: Workplan diagram of the project

Reference

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing, *Computational Linguistics*, 22(1): 39–71.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Banard, J. (2024, January 23). *What are Word Embeddings?* IBM. https://www.ibm.com/topics/word-embeddings

Cynthia Van Hee, Els Lefever, and Véronique Hoste. 2018. SemEval-2018 Task 3: Irony Detection in English Tweets. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 39–50, New Orleans, Louisiana. Association for Computational Linguistics.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. 2020. GoEmotions: A Dataset of Fine-Grained Emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4040–4054, Online. Association for Computational Linguistics.

Gibert, O., Perez, N., García-Pablos, A., and Cuadros, M. 2018. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long Short-Term Memory-Networks for Machine Reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas. Association for Computational Linguistics.

Mathur, M. (2020). *Sarcasm Detection with GloVe/Word2Vec(83%Accuracy).* Kaggle. https://www.kaggle.com/code/madz2000/sarcasm-detection-with-glove-word2vec-83-accuracy/notebook

Meng, J., Zhu, Y., Sun, S., and Zhao, D. 2024. Sarcasm detection based on BERT and attention mechanism, *Multimed Tools Appl*, 83: 29159–29178.

Misra, R. and Arora, P. 2023. Sarcasm Detection using News Headlines Dataset, *AI Open*, 4: 13-18.

MonkeyLearn. (2018, October 4). *Text Classification: What it is and Why it Matters*. MonkeyLearn. https://monkeylearn.com/text-classification/

SA. (2023, October 7). *Case Study: How Netflix Uses AI to Personalize Content Recommendations and Improve Digital Marketing*. Medium. https://medium.com/@shizk/case-study-how-netflix-uses-ai-to-personalize-content-recommendations-and-improve-digital-b253d08352fd

Scola, E., Segura-Bedmar, I. 2021. Sarcasm Detection with BERT, *Procesamiento del Lenguaje Natural*, 67:13-25.

Sigl, S. (2020, January 20). *Text Classification Demystified: An Introduction to Word Embeddings*. FreeCodeCamp. https://www.freecodecamp.org/news/demystify-state-of-the-art-text-classification-word-embeddings/

Somashekhar, S. P., Sepúlveda, M. J., Puglielli, S., Norden, A. D., Shortliffe, E. H., Rohit Kumar, C., Rauthan, A., Arun Kumar, N., Patil, P., Rhee, K., & Ramya, Y. (2018). Watson for Oncology and breast cancer treatment recommendations: agreement with an expert multidisciplinary tumor board. *Annals of oncology : official journal of the European Society for Medical Oncology*, 29(2): 418–423.

Sun, C., Qiu, X., Xu, Y., Huang, X. 2019. How to Fine-Tune BERT for Text Classification?. In: Sun, M., Huang, X., Ji, H., Liu, Z., Liu, Y. (eds) Chinese Computational Linguistics. CCL 2019. Lecture Notes in Computer Science, vol 11856. Springer, Cham.

Appendix

Appendix is organized into two sections: additional implementation details for sarcasm detection with bidirectional LSTM in Appendix A; and additional implementation details for sarcasm detection with fine-tuned BERT in Appendix B.

A. Additional Details for Sarcasm Detection with Bidirectional LSTM

Entire code is available at here.

A.1 Using Pre-trained GloVe Word Embedding

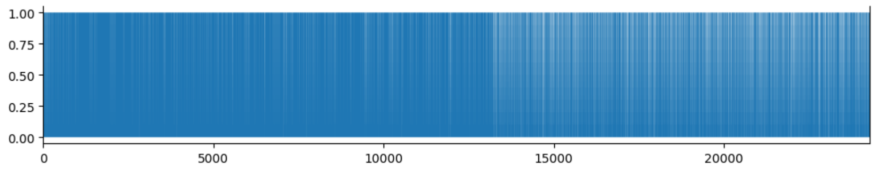
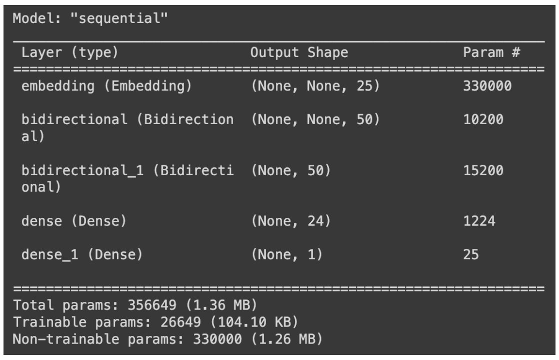


Figure 4: Words in our dataset which corresponds with the ones in GloVe embedding. 1 if the word is in the embedding else 0. As we can see, about 13,200 words corresponds with the ones in the embedding.

Pre-trained GloVe word embedding with vector size of 25 was loaded. To decide the size of vocabulary, it was needed to check how many words from our dataset corresponds with the ones in GloVe embedding. After checking out that about 13,200 words corresponds with the embedding, vocabulary size was set to be 13,200.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, None, 25)          330000

 bidirectional (Bidirection  (None, None, 50)          10200
 al)

 bidirectional_1 (Bidirecti  (None, 50)                15200
 onal)

 dense (Dense)               (None, 24)                1224

 dense_1 (Dense)             (None, 1)                 25

=================================================================
Total params: 356649 (1.36 MB)
Trainable params: 26649 (104.10 KB)
Non-trainable params: 330000 (1.26 MB)
```

Figure 5: Summary of LSTM model with pre-trained GloVe embedding

Using pre-trained GloVe word vectors, embedding matrix with size of (13,200, 25) was created. The embedding matrix was used as weights when building a neural network in the embedding layer. In the neural network, two bidirectional LSTM layer and two fully connected dense layers were stacked. On the last layer, sigmoid was used as activation function and Adam optimizer was used with learning rate of 0.00001. To calculate loss, binary cross entropy loss function was used. The model was fitted throughout 30 epochs.

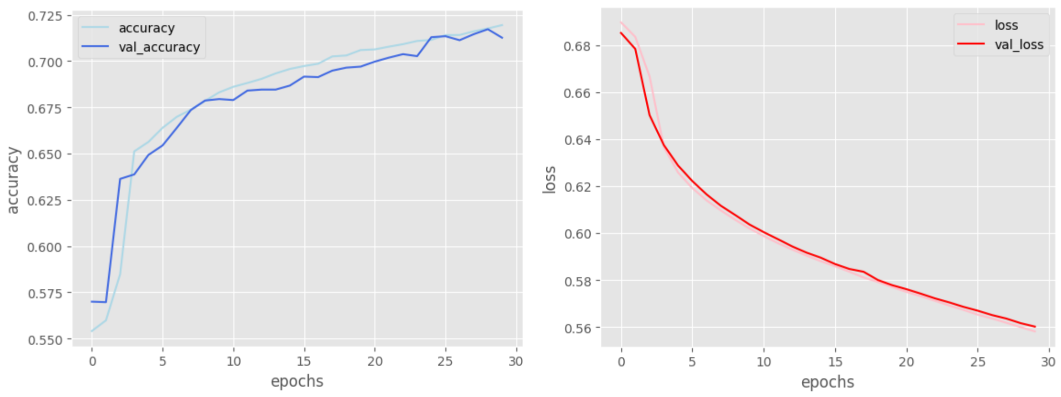
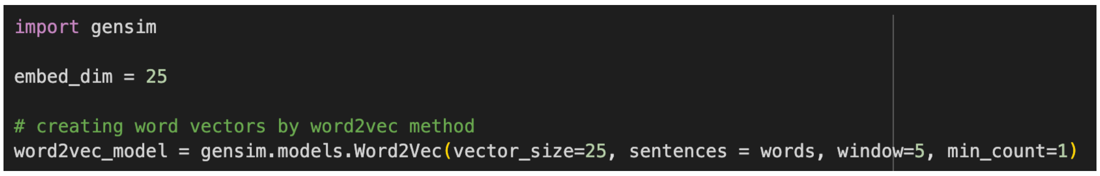


Figure 6: Accuracy and loss throughout 30 epochs. Even though validation accuracy is slightly lower than training accuracy, validation loss goes down very closely to the training loss. Overfitting seems to be prevented enough.

A.2 Using Word2Vec Word Embedding



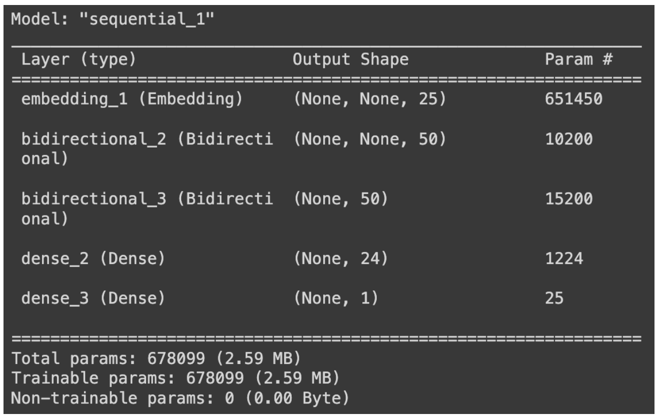
```python
import gensim

embed_dim = 25

# creating word vectors by word2vec method
word2vec_model = gensim.models.Word2Vec(vector_size=25, sentences = words, window=5, min_count=1)
```

Figure 7: Creating word vectors by loading and using genism Word2Vec model.

To create word vectors with Word2Vec method, genism Word2Vec model with vector size of 25 was loaded and words from our dataset was used as sentences parameter. Using function to create weight matrix from Word2Vec genism model, embedding matrix was created in size of (26058, 25). As we can see form the size of embedding matrix, this time the size of vocabulary was set to be 26058, to use all the words in the vocabulary.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (None, None, 25)          651450

 bidirectional_2 (Bidirecti  (None, None, 50)          10200
 onal)

 bidirectional_3 (Bidirecti  (None, 50)                15200
 onal)

 dense_2 (Dense)             (None, 24)                1224

 dense_3 (Dense)             (None, 1)                 25

=================================================================
Total params: 678099 (2.59 MB)
Trainable params: 678099 (2.59 MB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 8: Summary of LSTM model with Word2Vec embedding

The embedding matrix was used as weights when building a neural network in the embedding layer. This time the *trainable* parameter of Embedding layer was set to be *True* since the embedding weights need to be trained. In the neural network, two bidirectional LSTM layer and two fully connected dense layers were stacked. To prevent overfitting, 0.2 dropout was applied on the LSTM layers. On the last layer, sigmoid was used as activation function and Adam optimizer was used as well with learning rate of 0.00001. Binary cross entropy was used to calculate loss throughtout training. This time, we can see that there are more trainable parameters and no non-trainable parameters because the weights of embedding layer was set to be trained unlike before. The model was fitted throughout 30 epochs.

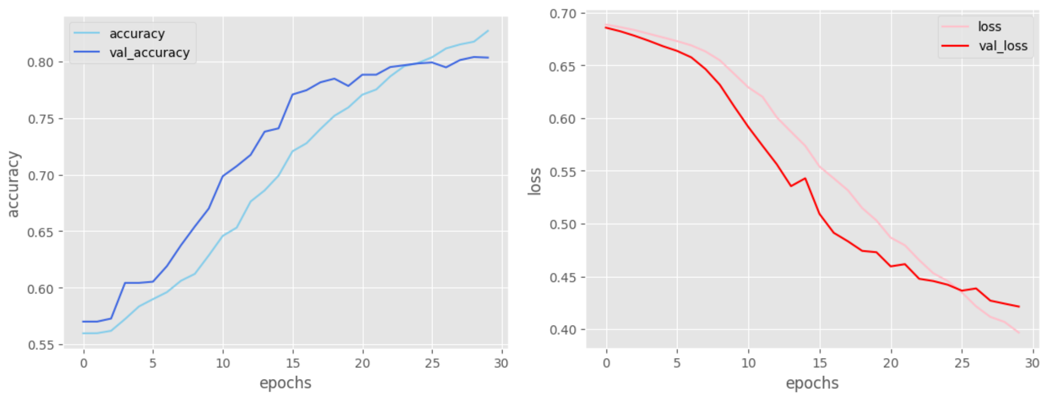


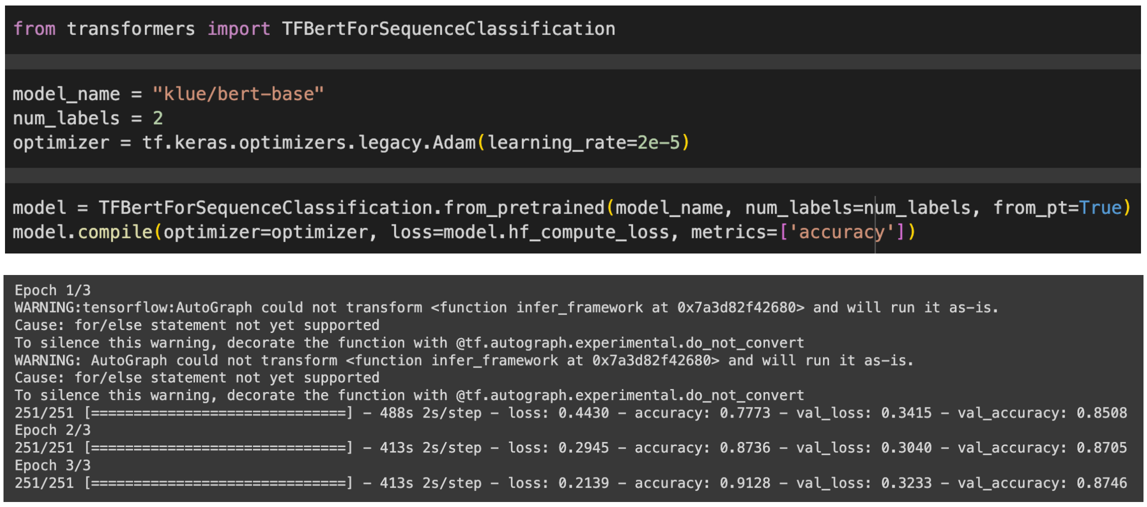
Figure 9: Accuracy and loss of second LSTM model throughout 30 epochs

Overall accuracy is higher, and the loss is lower than the experiment before. Validation accuracy is higher than training accuracy until about 24th epoch and validation loss is lower than training loss until 25th epoch. Overfitting seems to be prevented before 24th epoch. This time the validation accuracy reached 0.8034 and F1 score was 0.7556.

B. Additional Details for Sarcasm Detection with Fine-Tuned BERT

Entire code is available at here.

Before being tokenized, the dataset was split into train, validation, and test set in the proportion of 6:2:2. Training set length was 16,025 and validation and test set length was 5342 after split. BertTokenizerFast was loaded from transformers to tokenize training and validation set, and they were tokenized with padding and truncation and max length of 50.



```python
from transformers import TFBertForSequenceClassification

model_name = "klue/bert-base"
num_labels = 2
optimizer = tf.keras.optimizers.legacy.Adam(learning_rate=2e-5)

model = TFBertForSequenceClassification.from_pretrained(model_name, num_labels=num_labels, from_pt=True)
model.compile(optimizer=optimizer, loss=model.hf_compute_loss, metrics=['accuracy'])
```

```
Epoch 1/3
WARNING:tensorflow:AutoGraph could not transform <function infer_framework at 0x7a3d82f42680> and will run it as-is.
Cause: for/else statement not yet supported
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert
WARNING: AutoGraph could not transform <function infer_framework at 0x7a3d82f42680> and will run it as-is.
Cause: for/else statement not yet supported
To silence this warning, decorate the function with @tf.autograph.experimental.do_not_convert
251/251 [==============================] - 488s 2s/step - loss: 0.4430 - accuracy: 0.7773 - val_loss: 0.3415 - val_accuracy: 0.8508
Epoch 2/3
251/251 [==============================] - 413s 2s/step - loss: 0.2945 - accuracy: 0.8736 - val_loss: 0.3040 - val_accuracy: 0.8705
Epoch 3/3
251/251 [==============================] - 413s 2s/step - loss: 0.2139 - accuracy: 0.9128 - val_loss: 0.3233 - val_accuracy: 0.8746
```

Figure 10: Loading and to fine-tuning BERT-base on sarcasm detection task.

BERT-base model for sequence classification was loaded to be fine-tuned. This time adam optimizer was used as well but with learning rate of 2e-5. The model was fitted for 3 epochs with batch size of 64 and T4 GPU on Colab Notebook was needed to train it. The accuracy of fine-tuned BERT-base was 0.8735, and F1 score was 0.8711, which outperformed both two LSTM models on sarcasm detection task.

sentence	label	pred	score
billy bush reportedly out at 'today' and negotiating exit from show	0	0	0.9951
flaws in how we evaluate leaders (from kahneman's thinking, fast and slow)	0	0	0.9988
charlottesville shows that states must amend their open-carry laws	0	0	0.9852
blm's alicia garza launches census project to mobilize black political power	0	0	0.9047
college graduate accepts position above parents' garage	1	1	0.7846
pastor going on little spiel about seeing how in love couple are despite not knowing them for very long	1	1	0.9979
how to read a bad book by a great author	0	0	0.9986
eu court issues landmark data ruling	0	1	0.8267
china's potemkin villages	0	0	0.8940
florida man killed after standing up for gay friends, witnesses say	0	0	0.9971

Table 4: Prediction and evaluation of fine-tuned BERT on test set. Sampled 10 sentences.

The fine-tuned model was evaluated on test set, which consists of 5342 records. Not only we can see the technical performance of the model with score, also we can see how the model classifies if a news headline is sarcastic or not with the prediction results. From the sampled 10 results above, the model predicts a headline 'how to read a bad book by a great author' as non-sarcastic with probability of 0.9986, and 'pastor going on little spiel about seeing how in love couple are despite not knowing them for very long' as sarcastic with probability of 0.9979.