|  |
| --- |
| IE4483 |
| Sentiment Analysis |
|  |

|  |
| --- |
| KOH JIAHUI, SHAO SIYING, PYAE SONE KHIN (U2021752A)  10-29-2023 |

## Data Preprocessing Procedures

For this project, we have three models which have their own feature format. However, we have general data cleaning procedures before proceeding to training our models. The general data cleaning procedures are as follows.

Lowercasing:

## All text data is converted to lowercase to maintain consistency and prevent case-related variations.

Removing Punctuation:

Special characters, punctuation marks, and symbols are removed from the text to focus on meaningful words. As removing them simplifies the data and reduce noise.

Removing Numbers:

Numerical digits or symbols are excluded from the text data, as the focus is often on the emotional tone rather than numerical values. This might help streamline the text data for better model performance.

Removing Stopwords:

Common, non-discriminative words (stopwords) are eliminated from the text, such as "the," "and," "is," etc. With this, it can helps in focusing on words that carry more sentiment.

Removing Duplicated Rows:

Duplicate entries, if present, are removed to ensure data quality and to avoid bias that may arise from duplication. With this, we can maintain a clean and unbiased dataset.

## BERT

### Feature Format

For sentiment analysis using a BERT model, the chosen feature format is tokenized and encoded text data. This format is a standard practice in natural language processing (NLP) and ensures that raw textual data can be effectively utilized by the model for sentiment classification.

### Data Preprocessing Procedures

1. Text Cleaning:

* Lowercasing: All text data is converted to lowercase to maintain consistency and prevent case-related variations.
* Removing Punctuation: Special characters, punctuation marks, and symbols are removed from the text to focus on meaningful words.
* Removing Numbers: Numerical digits or symbols are excluded from the text data, as they may not be relevant for sentiment analysis.
* Removing Stopwords: Common, non-discriminative words (stopwords) are eliminated from the text, such as "the," "and," "is," etc.
* Removing Duplicated Rows: Duplicate entries, if present, are removed to ensure data quality.

1. Tokenization:

Breaking down text into words or subword units is an important step in NLP. When using BERT, words are represented as subword pieces through subword tokenization. This process relies on a trained tokenizer which can be found in the Hugging Face Transformers library.

1. Padding and Truncation:

BERT models require input sequences with a fixed length. To achieve this, padding is used to extend shorter sequences and truncation is used to shorten longer sequences. This ensures that all input sequences have the same length.

1. Encoding:

Text data is converted into numerical values that can be understood by the BERT model. The encoding process involves;

* input\_ids: A series of integer IDs that represent the tokens.
* attention\_mask: A binary mask that distinguishes between actual tokens and padding tokens.
* token\_type\_ids: Utilized to differentiate between two distinct sentence inputs. It is not used for BERT but included for compatibility.

1. Dataset Creation:

The structured dataset contains encoded text as well as target labels representing sentiments, where 0 indicates a negative sentiment and 1 represents a positive sentiment.

1. Data Splitting:

The dataset is divided into two sets, 80% for training and the other 20% for validation purpose.

1. Batching:

During the training and prediction phase, the data is organized into batches for efficient processing. To manage these batches effectively, data loaders like PyTorch's DataLoader are utilized.

By following these procedures, the unprocessed text data undergoes a transformation to a suitable format that can be fed into a BERT model. This format allows the model to comprehend and generate predictions based on the encoded text information.

### Parameter and Setting Selection

Maximum Sequence Length (max\_length)

After weighing the trade-offs between computational efficiency and model complexity, we chose to set max\_length at 128. We can maintain reasonable computing requirements while capturing context by using 128 as the max\_length. This choice is appropriate for sentiment analysis jobs and is consistent with NLP community practices. While using a lesser length could mean losing important contextual information, using a greater length would need more computational resources.

Number of Training Epochs (num\_train\_epochs)

Initially, we started with three training epochs. By allowing the model to go through training iterations, this value helps the model identify the sentiment patterns that are underlying the data. Hyperparameter tuning allows us to further optimize this, though, if needed. We can identify the ideal balance between training time and model convergence by experimenting with the number of training epochs. As we are able to attain 92% accuracy this first decision appears to be both practical and efficient.

Number of Labels (num\_labels)

For binary sentiment analysis, it is appropriate to set num\_labels to 2. This selection illustrates the task's binary nature, in which the objective is to categorize material as having a positive or negative sentiment. The model architecture is optimized for this binary classification by setting num\_labels to 2, which satisfies the demands of the sentiment analysis task.

## Model Selection and Result