GROUP 26

AML CHALLENGE-3

SENTIMENT ANALYSIS

INTRODUCTION

WHAT IS SENTIMENT ANALYSIS?

Sentiment analysis, also known as opinion mining, is the process of determining and categorizing the sentiment expressed in a piece of text. It involves analyzing the subjective information conveyed in the text to identify whether the sentiment is positive, negative, or neutral.

INTRODUCTION

OBJECTIVES OF THE CHALLENGE

- To explore and evaluate diverse natural language processing (NLP) techniques and models for sentiment analysis.
- To improve the accuracy and effectiveness of sentiment classification specifically in the context of social media.
- To address the crucial task of capturing and understanding sentiment in order to evaluate public opinion and reactions.
- To advance the state-of-the-art in sentiment analysis methods within the realm of social media platforms.

HOW TO SOLVE IT?

1. Data Exploration
2. Data Preparation
3. Model Selection
4. Model Performance

DATA EXPLORATION
TRAINING DATA

The training dataset comprises the text ID, the original tweet text, the selected text representing the words indicating sentiment, and the sentiment label (positive/negative/neutral).

	textID	text	selected_text	sentiment
0	28ac06f416	good luck with your auction	good luck with your auction	positive
1	92098cf9a7	Hmm..You can`t judge a book by looking at its ...	Hmm..You can`t judge a book by looking at its ...	neutral
2	7858ff28f2	Hello, yourself. Enjoy London. Watch out for ...	They`re mental.	negative
3	b0c9c67f32	We can`t even call you from belgium sucks	m suck	negative
4	7b36e9e7a5	not so good mood..	not so good mood..	negative
5	e158681396	Jumping im the shower after a long day of work...	ahmazing	positive

DATA EXPLORATION
TEST DATA

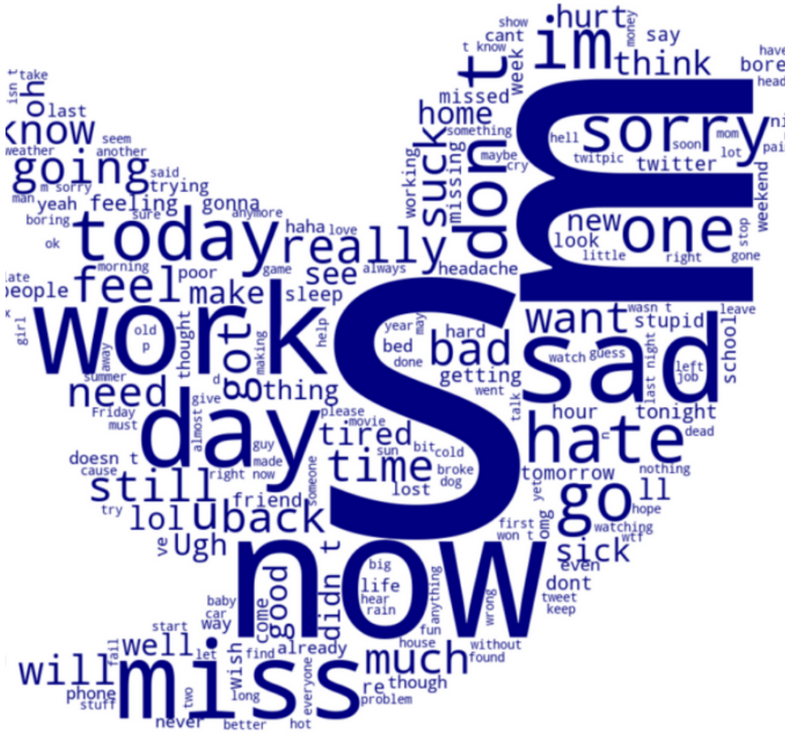
The test dataset includes the text ID, the original text of the tweet, and the corresponding selected text, which represents the relevant portion of the text.

	textID	text	selected_text
0	102f98e5e2	Happy Mother`s Day hahaha	Happy Mother`s Day
1	033b399113	Sorry for the triple twitter post, was having ...	Sorry for the triple twitter post, was having ...
2	c125e29be2	thats much better than the flu syndrome!	thats much better
3	b91e2b0679	Aww I have a tummy ache	tummy ache
4	1a46141274	hey chocolate chips is good. i want a snack ...	good.
5	45208fede8	_Honi Might be cute to do a little picture boo...	be cute to do a little picture

DATA EXPLORATION

WORD CLOUD

POSITIVE SENTIMENT



NEGATIVE SENTIMENT

NEUTRAL SENTIMENT

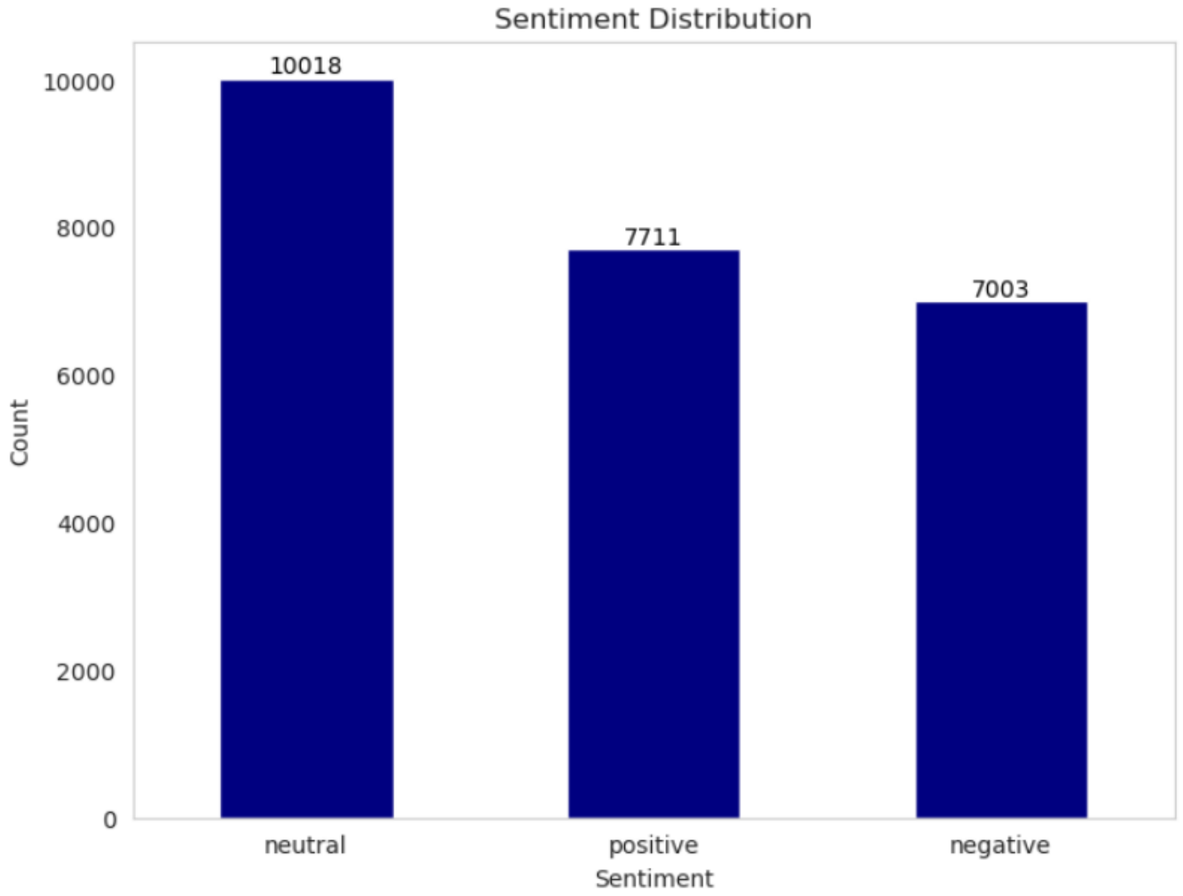
DATA EXPLORATION

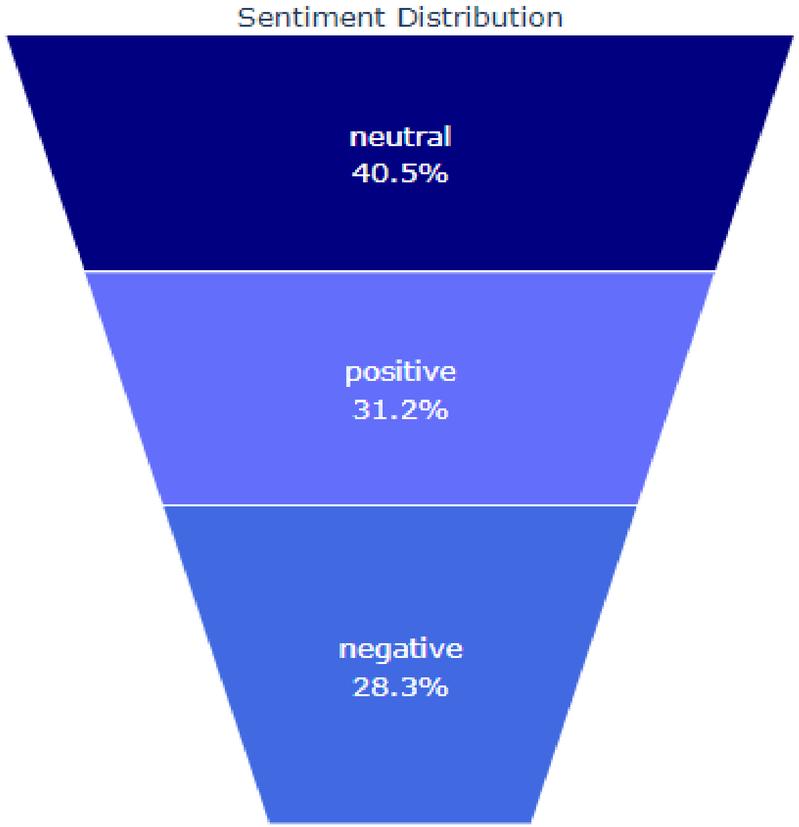
WORD CLOUD

The Wordclouds for positive and negative tweets exhibit distinct patterns. In positive sentiment, prevalent terms like "love," "good," and "thank" convey appreciation and positivity. Conversely, negative sentiment is characterized by terms such as "sad," "work," and "miss," reflecting a more somber or negative tone.

DATA EXPLORATION

SENTIMENT DISTRIBUTION





DATA EXPLORATION

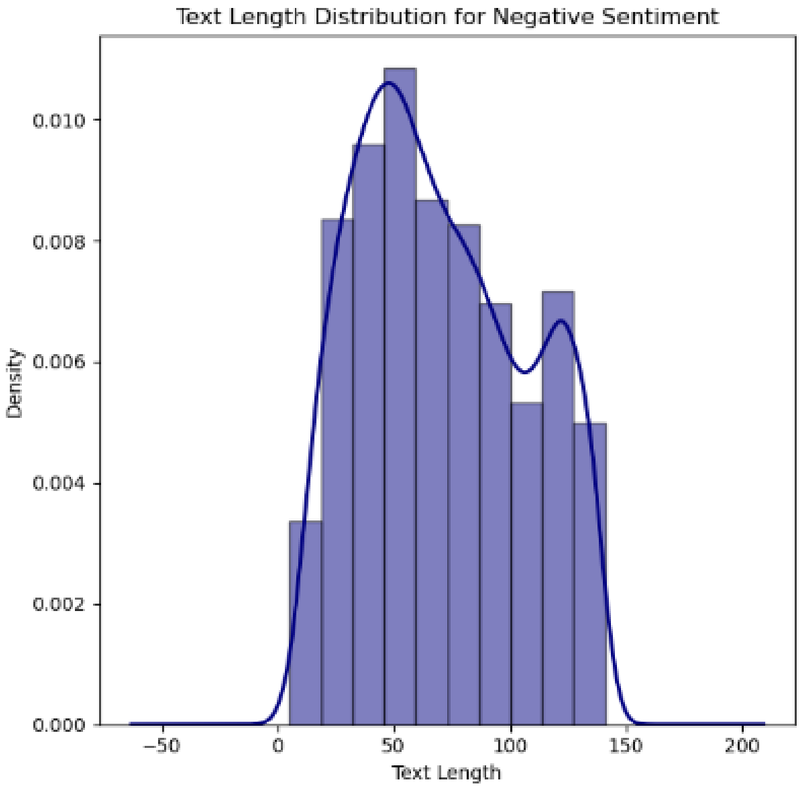
SENTIMENT DISTRIBUTION

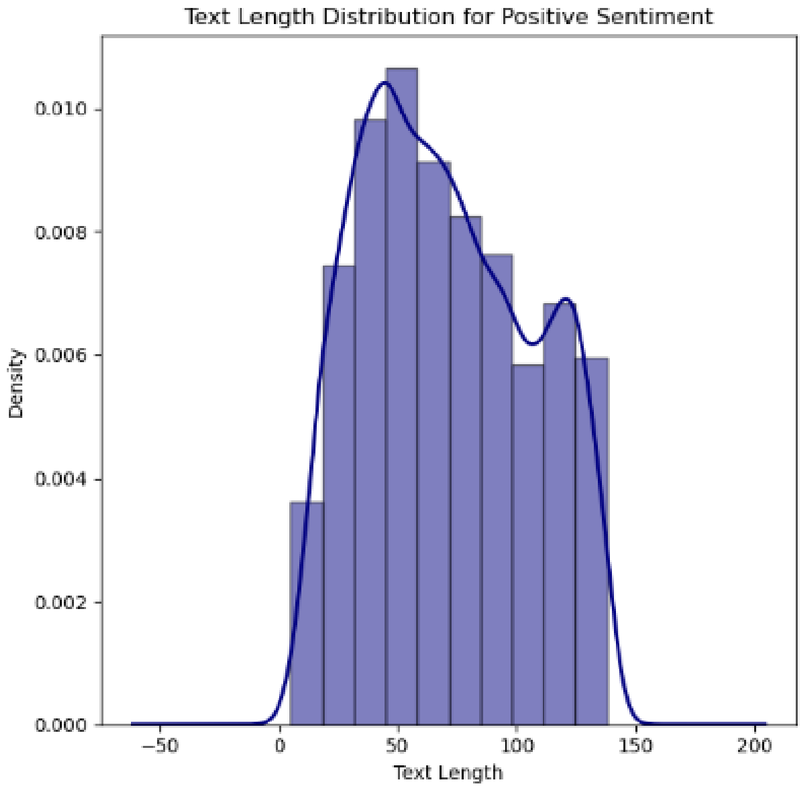
The count plot graph showcases the distribution of sentiments within the dataset. It reveals a comparable count between positive and negative sentiments, while the neutral sentiment demonstrates a noticeably higher count.

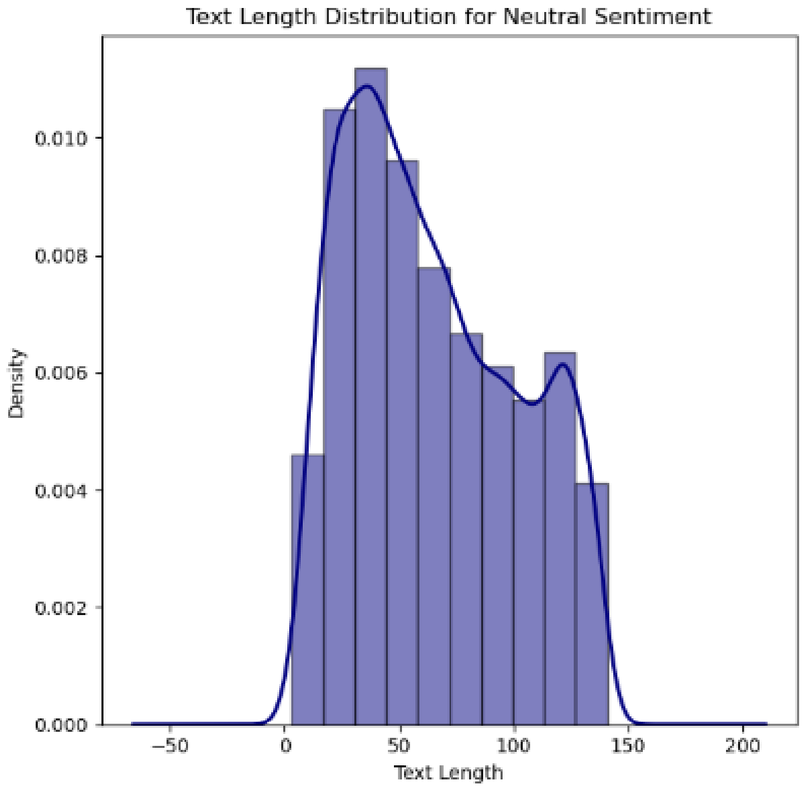
The funnel chart offers valuable insights into the sentiment distribution within the dataset. It highlights a slight imbalance, with the neutral class exhibiting a higher prevalence compared to the relatively balanced distribution between positive and negative sentiments

DATA EXPLORATION

TEXT LENGTH DISTRIBUTION







DATA EXPLORATION

TEXT LENGTH DISTRIBUTION

A closer investigation into the distribution of text lengths across different tweet categories reveals a notable similarity among them. This indicates that most classes exhibit a comparable pattern in terms of text length.

DATA EXPLORATION

DO WE HAVE ANY NULL VALUES IN OUR DATASET?

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24732 entries, 0 to 24731
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   textID         24732 non-null  object
 1   text           24732 non-null  object
 2   selected_text  24732 non-null  object
 3   sentiment      24732 non-null  object
dtypes: object(4)
memory usage: 773.0+ KB
None
```

TRAINING DATA

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2748 entries, 0 to 2747
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   textID         2748 non-null   object
 1   text           2748 non-null   object
 2   selected_text  2748 non-null   object
dtypes: object(3)
memory usage: 64.5+ KB
None
```

TEST DATA

There are no null values present in the dataset, indicating a complete and reliable data source for further analysis.

DATA EXPLORATION

DO WE HAVE ANY DUPLICATES IN OUR DATASET?

	textID	text	selected_text	sentiment
count	24732	24732	24732	24732
unique	24732	24732	20298	3
top	28ac06f416	good luck with your auction	love	neutral
freq	1	1	174	10018

The dataset does not contain any duplicate values, ensuring data integrity and eliminating the need for duplicate handling.

DATA PREPARATION

TEXT PRE-PROCESSING IN NLP FOR OUR MODEL

- **Converting the text to lowercase**
  - Converting the text to lowercase ensures that the text is uniformly represented in lowercase letters. This step helps in treating words with different capitalizations as the same word.
- **Tokenizing the words**
  - We will utilize the BERT tokenizer to tokenize the text. This involves breaking the sentence into words or sub word units and assigning numerical values to each token, enabling the model to process the text as numerical input.

DATA PREPARATION

TEXT PRE-PROCESSING IN NLP FOR ALTERNATIVE MODELS

In the subsequent section, we will delve into the NLP text processing steps employed for the alternative models under examination. The selection of these models was predicated upon their prowess in effectively handling and analyzing textual data. Through our exploration of diverse approaches, our objective is to harness the potential of NLP techniques to optimize the outcomes of text analysis and information extraction. This comprehensive investigation aims to exploit the inherent capabilities of NLP in order to unlock deeper insights and maximize the value derived from textual data analysis.

DATA PREPARATION

TEXT PRE-PROCESSING IN NLP FOR ALTERNATIVE MODELS

The text pre-processing steps for our alternative models include the following:

- Converting the text to lowercase
- Removing punctuation
- Removing stopwords
- Stemming
- Tokenizing the words
- TF-IDF Calculation
- Vectorization

DATA PREPARATION
STOP-WORDS

WHAT ARE STOP-WORDS?

Stop words are common words that are often considered irrelevant or insignificant in natural language processing (NLP) tasks. These words are frequently used in a language but typically do not carry significant meaning or contribute much to the overall understanding of a text

	STOP-WORDS
0	if
1	Too
2	WIll
3	were
4	VERY
5	of

DATA PREPARATION

STEMMING

WHAT IS IT?

Stemming is a text preprocessing technique that involves reducing words to their base or root form. It helps group words with similar meanings but different forms, reducing vocabulary size and improving computational efficiency.

Original Text	Stemmed Text
good luck with your auction	good luck with your auction
Hmm..You can`t judge a book by looking at its cover	hmmyou cant judg a book by look at it cover
They`re mental.	theyr mental
m suck	suck
not so good mood..	not so good mood

DATA PREPARATION

TF-IDF CALCULATION FUNCTIONS

WHAT IS TF-IDF?

TF-IDF (Term Frequency-Inverse Document Frequency) is a measure used to determine the importance of a word in a collection of documents. It combines the term frequency (how often a word appears in a document) with the inverse document frequency (how rare a word is in the entire document collection) to identify significant words for tasks like information retrieval and text analysis.

WHY TF-IDF?

- TF-IDF can help identify important words within each text by considering their frequency and rarity in the dataset.
- It can reduce the impact of common and uninformative words that appear across multiple texts, allowing the model to focus on more meaningful terms.

DATA PREPARATION

TF-IDF CALCULATION FUNCTIONS

MATHEMATICAL REPRESENTATION OF TF-IDF:

TF(t, d) = (number of times t appears in d) / (total number of terms in d)

IDF(t) = log (N / (1+df))

TF - IDF (t, d) = TF(t, d) * IDF(t)

where:

- *t represents a term (word) in a document.*
- *d represents a specific document.*
- *N is the total number of documents*
- *df is the number of documents with term t*

DATA PREPARATION

TF-IDF CALCULATION FUNCTIONS

STEPS INVOLVED IN CALCULATING TF-IDF:

- Count the frequency of each word within each text.
- Determine the rarity of each word across the entire text collection.
- IDF is calculated by taking the logarithm of the ratio between the total number of texts and the number of texts containing a specific word.
- Multiply the term frequency (TF) of each word in a document by its inverse document frequency (IDF).
- Save the calculated TF-IDF values.

DATA PREPARATION

VECTORIZATION

WHAT IS IT?

- Text vectorization is the conversion of raw text into numerical representations, enabling mathematical operations and the use of machine learning algorithms.

- Through vectorization, important features are extracted from the comments, such as the importance of words using TF-IDF. This process aids in sentiment analysis by enhancing the model's understanding and ability to identify sentiment.

- Vectorization can reduce the dimensionality of the data. By representing the text as vectors, we can potentially reduce the number of features while still preserving important information.

DATA PREPARATION

PRE-PROCESSING CONSIDERATIONS FOR BERT MODEL

- **Contextual Information Loss:** Removing punctuation, stopwords, and performing vectorization in BERT can result in the removal or distortion of crucial contextual information, leading to decreased accuracy.
- **Valuable Cues:** BERT already incorporates punctuation and stopwords during training, so removing them may discard important cues that contribute to understanding.
- **Word Variation Loss:** Stemming eliminates word variation, which is essential for BERT's comprehension of context, potentially compromising accuracy.
- **Unnecessary TF-IDF:** BERT learns contextualized word representations, making TF-IDF calculation redundant. Separate vectorization techniques may not align well with BERT's dense word embeddings.

DATA PREPARATION

CONCLUSION

- To ensure a comprehensive evaluation, we conducted experiments involving different natural language processing techniques.

- During our analysis, we observed that certain pre-processing steps have the potential to modify or remove contextual information, which can adversely impact the performance of our model.

- Therefore, in the model selection phase, we aim to address this challenge by exploring and selecting the most suitable model that can effectively overcome these limitations and enhance the performance of BERT.

MODEL SELECTION

ALTERNATIVE APPROACHES ATTEMPTED

STANDARD LOGISTIC REGRESSION:

- Sentiment analysis datasets, including the provided tweets, frequently consist of high-dimensional features. However, standard logistic regression may encounter difficulties in effectively managing such complex data, which can result in problems such as overfitting or diminished performance.

- In sentiment analysis, it is crucial to comprehend the sequential structure of language and capture the interdependencies among words. However, standard logistic regression treats individual words or features independently and fails to consider the contextual relationships between words.

MODEL SELECTION

ALTERNATIVE APPROACHES ATTEMPTED

SUPPORT VECTOR MACHINES:

- SVMs are not well-suited for sentiment analysis because they face challenges in handling large datasets and high-dimensional features, impacting their scalability.

- SVMs may face limitations in effectively handling the inherent noise and variability present in text data, which can impact their ability to accurately classify sentiment.

- SVMs, which rely on linear decision boundaries, may encounter difficulties in capturing the non-linear relationships present in sentiment analysis tasks.

MODEL SELECTION

BERT

WHY BERT?

- BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained language model developed by Google.
- BERT captures contextual information and understands the intricacies of sentiment expressed in a sentence.
- It is pretrained on a large corpus of text data, providing a strong foundation in language understanding.
- Fine-tuning BERT on sentiment analysis tasks improves its performance and reduces the need for extensive labeled data.

MODEL SELECTION

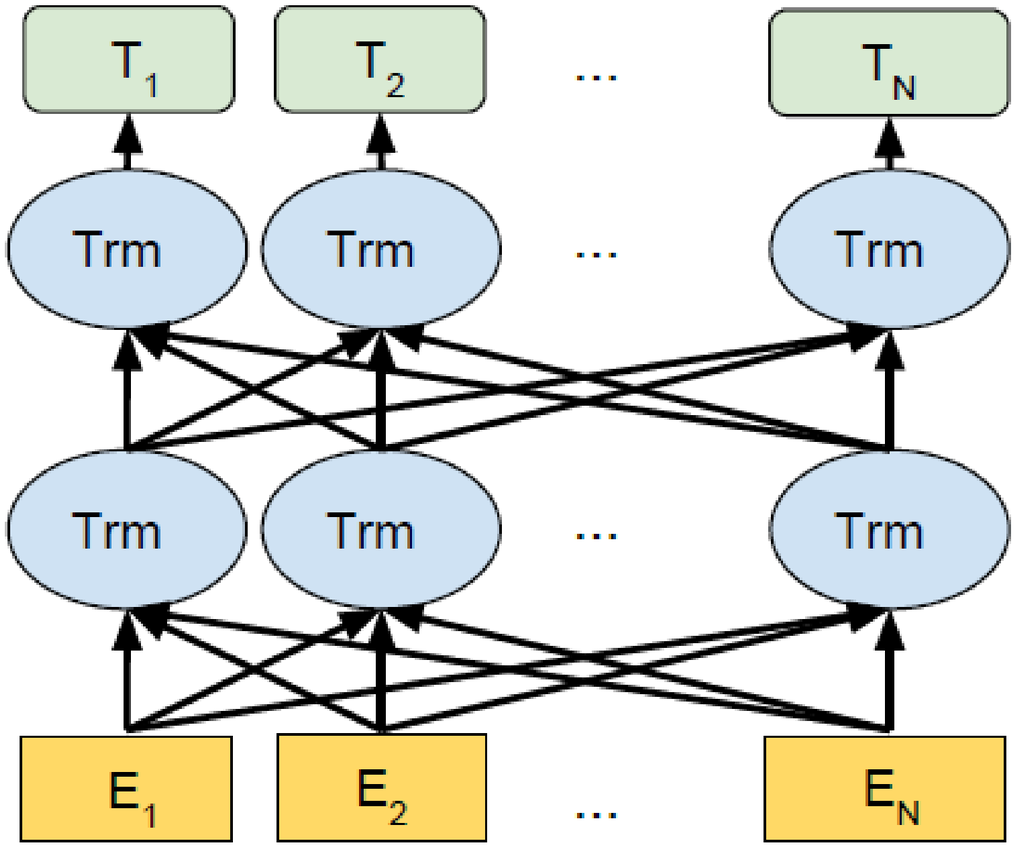
BERT

Utilizing BERT from the Transformer Library:

- We utilized the BERT model provided by the Transformer library for our model implementation.
- We employed the BERT tokenizer, which allowed us to leverage pre-trained tokenizers and ensure effective encoding of the input.
- Due to resource limitations, we utilized the BERT uncased version instead of BERT-large, which could have potentially provided even better results. Nonetheless, we leveraged the capabilities of the BERT model to effectively analyze and classify the sentiment of the input data.

MODEL SELECTION

BERT MODEL



MODEL SELECTION

WHY TRANSFORMERS?

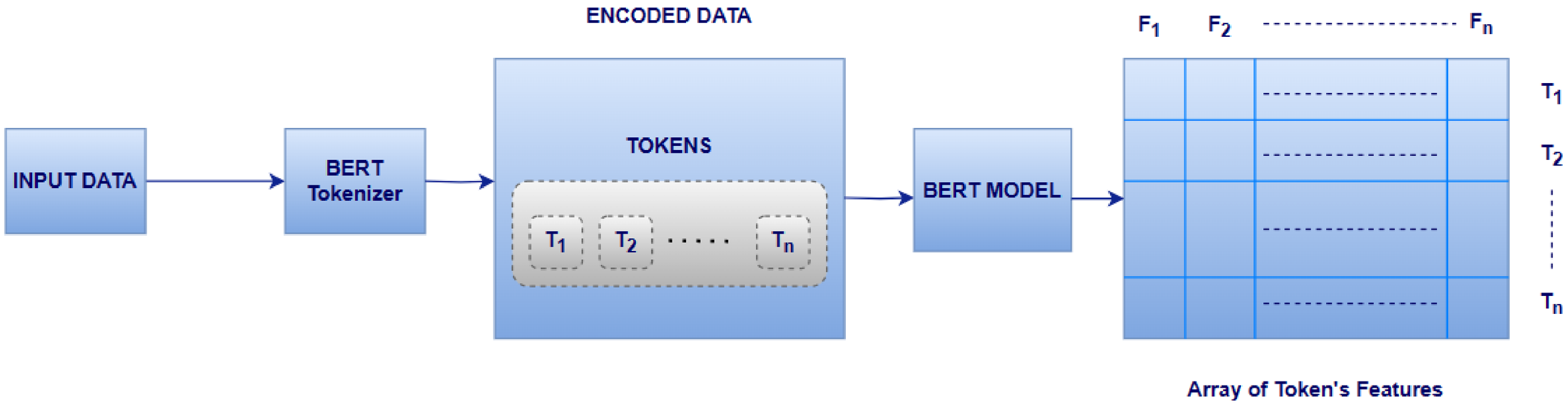
- **State-of-the-art models:** Transformers provide cutting-edge models that have achieved superior performance on various NLP tasks. They are pre-trained and ready to use, saving time and effort in model development.

- **Efficient computation:** Transformers utilize a parallelizable architecture, enabling faster training and inference, resulting in lower compute costs and a smaller carbon footprint.

- **Flexible model selection:** Transformers offer a range of model options, from small to large-scale architectures, allowing users to choose the most suitable framework for their specific task. This flexibility enables scalability and adaptability throughout a model's lifecycle.

MODEL SELECTION

BERT MODEL FOR FEATURE EXTRACTION



MODEL SELECTION

BERT TOKENIZER

- The BERT tokenizer is employed to effectively tokenize and preprocess the input texts in preparation for the BERT pre-trained model.

- It performs all the essential preprocessing steps necessary for the model, ensuring compatibility and optimal performance. The tokenizer is versatile and can be applied to both individual strings and lists of texts.

- Its output is a dictionary that contains the encoded tokens and accompanying information, facilitating further processing. This dictionary seamlessly integrates into downstream code or can be directly used as input for the BERT model, enhancing the overall efficiency and effectiveness of the system.

MODEL SELECTION

HOW BERT TOKENIZER WORKS?

Vocabulary Creation:

- BERT Tokenizer starts by creating a vocabulary from a large corpus of text.
- The vocabulary consists of a set of unique tokens, including words, sub words, and special tokens.

Text Tokenization:

- BERT Tokenizer breaks down input text into smaller meaningful units called tokens.
- Tokens can be individual words or sub words generated during vocabulary creation.
- The tokenizer also handles special characters, punctuation, and unknown words.

Special Tokens:

- BERT Tokenizer adds special tokens to the input sequence to provide additional information to the model.
- These tokens include [CLS] (classification), [SEP] (separator), and [PAD] (padding).

MODEL SELECTION

HOW BERT TOKENIZER WORKS?

Token Encoding:

- BERT Tokenizer assigns a unique numerical ID (token ID) to each token in the vocabulary.
- Each input sequence is converted into a sequence of token IDs to be understood by the model.

Padding and Truncation:

- BERT Tokenizer ensures that all input sequences have the same length by padding or truncating them.
- Padding adds [PAD] tokens to the shorter sequences, while truncation cuts off tokens from longer sequences.

MODEL SELECTION

HOW BERT TOKENIZER WORKS?

Attention Masks:

- BERT Tokenizer generates attention masks to indicate which tokens the model should pay attention to.
- These masks help the model distinguish between actual tokens and padded tokens during training and inference.

Output:

- The BERT Tokenizer returns the processed input as a set of token IDs and attention masks.

MODEL SELECTION

PyTorch Tensors

- PyTorch tensors are utilized to represent the encoded data in a format that is compatible with the PyTorch framework, enabling seamless integration and efficient processing.
- These tensors are created after the data is encoded using the BERT tokenizer.
- PyTorch tensors ensure that the encoded data is in a suitable format for the BERT model and facilitate efficient computations and operations within the PyTorch framework.

MODEL SELECTION

HOW BERT MODEL WORKS FOR FEATURE EXTRACTION?

- The tokenized input, obtained using the BERT tokenizer, is fed into the BERT model.
- The BERT model generates contextualized representations for each token in the input sequence.
- These representations capture the contextual information of the tokens based on their surrounding context in the sequence.
- The BERT model then extracts features from these contextualized representations.
- These features represent the contextual information of the tokens and can be used for various downstream tasks.

MODEL SELECTION

SENTIMENT CLASSIFICATION USING LOGISTIC REGRESSION

- Logistic regression is computationally efficient, making it suitable for handling large-scale datasets and high-dimensional feature representations produced by BERT.

- It offers transparent outcomes by assigning weights to individual features. These weights signify the significance and impact of each feature in sentiment prediction. This enables us to discern the specific textual components that influence the sentiment classification.

- It exhibits adaptability in handling sentiment analysis tasks that encompass multi-class categorization, enabling accurate prediction of sentiments across diverse categories such as positive, neutral, and negative.

MODEL PERFORMANCE

- We utilize the F1 score to evaluate the performance of our sentiment analysis model. The F1 score is a weighted mean of precision and recall, providing a balanced measure of classification accuracy.
  - It ranges from 0 to 1, where 1 represents perfect precision and recall, and 0 indicates the worst performance.
- The beta parameter in the F1 score calculation determines the weight assigned to recall.
  - When beta is less than 1, precision is given more weight, while a value greater than 1 favors recall.
- We chose to use beta=1 and the 'weighted' average for calculating the F1 score. This approach allows us to provide a balanced evaluation of the model's ability to accurately classify instances from each class.
- By utilizing the 'weighted' average, the F1 score provides a fair assessment of the model's performance across all sentiment categories.

MODEL PERFORMANCE

F1 SCORE

WHY DID WE CHOOSE F1 SCORE?

- The F1 score combines precision and recall, ensuring that the model not only accurately identifies positive, neutral, and negative sentiment (precision), but also captures all instances of these sentiments in the data (recall). This is particularly important in sentiment analysis tasks, where the goal is to accurately capture users' opinions and attitudes.

- Given the consequences of misclassifying sentiment and the need for accurate sentiment capture, the F1 score serves as a suitable evaluation metric for our analysis. It offers a comprehensive assessment of the model's performance in identifying sentiment, considering both precision and recall, and accommodating imbalanced datasets.

MODEL PERFORMANCE

F1 SCORE

The table highlights the F1 scores achieved by each model in the study. The BERT + Logistic Regression model outperforms the alternative models, SVM and Standard Logistic Regression, with a significantly higher F1 score of 0.8749. This exceptional result demonstrates the robustness and reliability of the BERT-based model in accurately classifying the data, solidifying its superiority over the other models. However, it is important to note that the Standard Logistic Regression model exhibits a notable advantage in terms of faster processing time compared to the BERT Logistic Regression model.

Model	F1 Score
BERT Logistic Regression	0.8749
SVM	0.8162
Standard Logistic Regression	0.8079

MODEL PERFORMANCE

In our analysis, we chose not to explore hyperparameters to ensure a consistent and standardized evaluation approach. Moreover, by focusing on the default hyperparameters, we aimed to provide a practical perspective, reflecting real-world scenarios where models are often deployed without extensive fine-tuning. This approach allows us to assess the models' performance in a more realistic setting, where extensive hyperparameter optimization may not always be feasible or practical. By understanding how the models perform with their default configurations, we can gain valuable insights that can guide decision-making in resource-constrained environments

BONUS TASK

- The bonus task in the challenge involves estimating the overlap between predicted words and ground truth selected words.
  - The Jaccard coefficient is used as an evaluation metric for this task.

- We leveraged the spaCy library to perform pre-processing tasks, including spell correction, tokenization, and part-of-speech tagging. Subsequently, we extracted specific word categories such as nouns, adjectives, and verbs, and merged them to create the resulting extracted text.
  - This approach distinguishes our model from one that would have directly used the selected_text for the classification task.

BONUS TASK

Upon analysis, we observed that the selected text retains the contextual meaning of the original sentence. Therefore, employing a transformer-based model, which captures the semantic understanding of the text rather than relying solely on specific word categories (such as 'NOUN', 'VERB', 'ADJ'), would have yielded more favorable results in terms of the Jaccard coefficient.

REFERENCES

- https://towardsdatascience.com/text-preprocessing-in-natural-language-processing-using-python-6113ff5decd8
- https://realpython.com/nltk-nlp-python/
- https://github.com/google-research/bert
- https://www.projectpro.io/article/bert-nlp-model-explained/558

THE TEAM

SHUBHIKA GARG

Master's in Data Science and Engineering

YOSSI TAPIERO

Telecom SudParis

DORIAN LEBRU

Telecom SudParis

JORDAN ALLEMAND

Telecom SudParis