PROJECT REPORT

For

Sentiment Analysis Using Transformer And LSTM

23/04/2024

Prepared by

Specialization	**SAP ID**	**Name**
AIML(H) B-2	500097852	Sk Mamud Haque
AIML(H) B-2	500097457	Utkarsh Rastogi

UNIVERSITY OF TOMORROW

School Of Computer Science
UNIVERSITY OF PETROLEUM & ENERGY STUDIES,
DEHRADUN- 248007. Uttarakhand

Table of Contents

1 Introduction

1.1 Purpose of the Project

The purpose of the project "Sentiment Analysis using Transformer and LSTM" is to develop an advanced sentiment analysis model that combines the strengths of transformer and LSTM (Long Short-Term Memory) architectures. The goal is to achieve improved accuracy and efficiency in sentiment analysis tasks, particularly in analyzing and understanding the sentiment expressed in textual data. By using transformers, the project aims to achieve more accurate sentiment analysis results compared to traditional methods. The project may involve tasks such as data preprocessing, fine-tuning the transformer model on a sentiment analysis dataset, and evaluating the model's performance. The ultimate goal is to develop a robust and efficient sentiment analysis system that can classify the sentiment of text into categories like positive, negative, or neutral.

1.2 Target Beneficiary

The target beneficiaries of this project include researchers, data scientists, businesses, and organizations that rely on sentiment analysis for various purposes. This project aims to provide a more accurate and efficient sentiment analysis tool that can be used in applications such as social media monitoring, customer feedback analysis, market research, and more Customer service departments can use sentiment analysis to classify and prioritize feedback, improving response times and customer satisfaction. Researchers can analyze large text data to understand public opinion on various topics for academic or market research. Social media influencers can gauge audience reception to their content, enabling them to create more engaging posts.

1.3 Project Scope

The project scope includes the development and implementation of a sentiment analysis model using transformer and LSTM architectures. This involves collecting and preprocessing textual data, training the model on labeled datasets, and evaluating its performance. The scope also includes the potential integration of the model into existing systems or applications for real-world use cases. It can also help in managing brand reputation by monitoring and responding to negative feedback effectively. In market research, sentiment analysis can provide insights into public opinion on new products, marketing campaigns, or industry trends. Social media monitoring using sentiment analysis can help businesses and influencers understand audience perceptions and adjust their strategies accordingly. Additionally, sentiment analysis can be applied in political analysis to gauge public opinion on political issues and government policies. In healthcare, it can analyze patient feedback to improve patient care. In finance, sentiment analysis can analyze market sentiment and news sentiment for investment decisions. It also has applications in academic research and the entertainment industry, where it can analyze audience reactions to movies, TV shows, and music to understand audience preferences.

1.4 References

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems (pp. 5998-6008).

[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] 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.

[4] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... & Brew, J. (2019). HuggingFace's transformers: State-of-the-art natural language processing. arXiv preprint arXiv:1910.03771.

[5] Zhang, Y., Yang, Q., Zhang, S., Chen, J., & Du, H. (2020). Sentiment analysis using ensemble classifiers based on BERT and ERNIE. IEEE Access, 8, 89997-90005.

2. Project Description

2.1 Reference Algorithm

The reference algorithm for "Sentiment Analysis using Transformer and LSTM" involves a hybrid approach that combines the transformer architecture for contextual understanding and the LSTM architecture for sequential learning. This approach leverages the transformer's ability to capture long-range dependencies and the LSTM's capability to model sequential data, resulting in a more effective sentiment analysis model.

Here is a detailed explanation of how BERT can be used for sentiment analysis:

1. Preprocessing: Tokenize the input text using BERT's tokenizer and add special tokens ([CLS] at the beginning and [SEP] at the end) to mark the start and end of a sentence.
2. Input Representation: Convert the tokenized text into numerical representations using BERT's tokenizer. BERT uses WordPiece embeddings to represent words as vectors.
3. Model Architecture: Use a pre-trained BERT model (e.g., BERT-base or BERT-large) as the base model. BERT consists of multiple layers of transformer blocks, each containing self-attention mechanisms. On top of the base BERT model, add a classification layer (e.g., a fully connected layer) to predict the sentiment of the input text.
4. Fine-Tuning: Fine-tune the pre-trained BERT model on a sentiment analysis dataset. During fine-tuning, the weights of the BERT model are adjusted using backpropagation to minimize a loss function that measures the difference between the predicted sentiment and the actual sentiment labels in the dataset.
5. Inference: Use the fine-tuned BERT model to predict the sentiment of new input text. The model outputs probabilities for each sentiment class (e.g., positive, negative, neutral), and the class with the highest probability is selected as the predicted sentiment.

BERT has shown state-of-the-art performance in various sentiment analysis tasks and is widely used in research and industry for this purpose. The flexibility and effectiveness of transformer-based models like BERT make them a popular choice for sentiment analysis tasks requiring a deep understanding of contextual information in text.

2.2 Data/ Data structure

The project requires labeled datasets containing text samples annotated with sentiment labels (e.g., positive, negative, neutral). The data structure includes preprocessing steps such as tokenization, padding, and encoding to prepare the data for training the transformer and LSTM models. Additionally, the project may involve using pre-trained word embeddings or transformer embeddings to represent the input text data.

The data structure for the project "Sentiment Analysis using Transformers" would typically involve the following components:

1. Input Text Data: The raw text data that needs to be analyzed for sentiment. This could be in the form of reviews, social media posts, or any other textual data.
2. Preprocessed Data: The preprocessed data after tokenization and encoding. This includes converting the text into numerical representations that can be fed into the transformer model.
3. Transformer Model: The transformer model (e.g., BERT, RoBERTa, GPT) used for sentiment analysis. This includes the architecture of the model, the weights learned during training, and any additional layers added for classification.

Overall, the data structure of the project involves managing the input text data, preprocessing it for use with the transformer model, training the model on a labeled dataset, and using the fine-tuned model for sentiment analysis on new data.

2.3 SWOT Analysis

The SWOT analysis for "Sentiment Analysis using Transformer and LSTM" includes:

1. Strengths:
   - Utilizes state-of-the-art transformer models like BERT, which are known for their superior performance in natural language processing tasks.
   - Can handle complex contextual information in text, leading to more accurate sentiment analysis results.
   - Has a wide range of applications in various industries such as marketing, customer service, and market research.
   - Can provide valuable insights into customer feedback and public opinion.
2. Weaknesses:
   - Requires a large amount of labeled data for training the transformer model, which can be time-consuming and expensive to acquire.
   - Fine-tuning transformer models for specific sentiment analysis tasks requires expertise in machine learning and natural language processing.
3. Opportunities:
   - Can be enhanced by incorporating domain-specific knowledge or fine-tuning on domain-specific datasets to improve performance in specific industries or use cases.
   - Can be integrated into existing systems and applications to automate sentiment analysis tasks and provide real-time insights.
4. Threats:
   - Competing approaches and models in the field of sentiment analysis may offer alternative solutions with different trade-offs.
   - Rapid advancements in transformer models and NLP techniques could make current approaches obsolete or less competitive in the future.

Overall, the project has significant strengths in leveraging advanced transformer models for sentiment analysis, but it also faces challenges related to data availability, expertise required for implementation, and competition in the field of NLP.

2.4 Project Features

Features of the project "Sentiment Analysis using Transformers" include:

1. Hybrid Model: Utilizes both transformer and LSTM architectures for sentiment analysis, combining their strengths for improved performance.
2. Customization: Allows for customization of the model architecture and hyperparameters based on specific requirements and use cases.
3. Integration: Can be integrated into existing systems or applications for automated sentiment analysis tasks.
4. Real-Time Analysis: Capable of providing real-time or near-real-time sentiment analysis results.
5. Evaluation Metrics: Utilizes metrics such as accuracy, precision, recall, and F1 score for evaluating the model's performance.

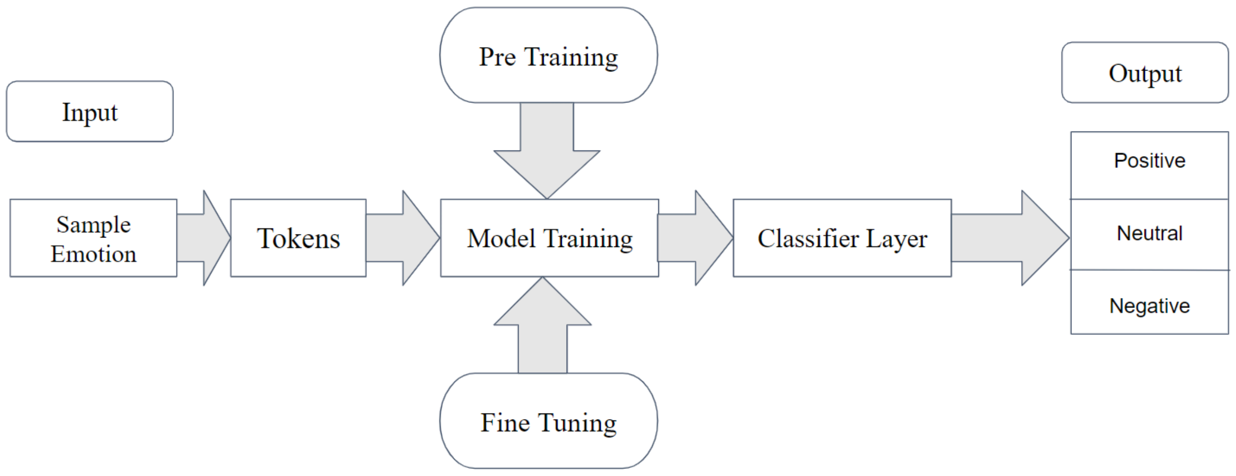
These features collectively make the project a powerful tool for sentiment analysis, with the ability to adapt to different domains and provide valuable insights from textual data.

2.5 User Classes and Characteristics

- Researchers: Interested in exploring advanced NLP techniques and improving sentiment analysis algorithms.
- Data Scientists: Involved in developing and deploying sentiment analysis models for various applications.
- Businesses: Looking to gain insights from customer feedback and social media data for decision-making.
- Developers: Responsible for integrating the sentiment analysis model into applications and systems.
- End Users: Benefit from more accurate sentiment analysis in applications such as social media platforms, customer service systems, and market research tools.

The project offers application flexibility, being applicable to various industries and use cases, allowing for customization and integration into existing systems. Performance evaluation is a key aspect, focusing on metrics such as accuracy, precision, recall, and F1 score to assess the model's effectiveness. Domain knowledge is beneficial for users to fine-tune models and interpret results in context, making collaboration between data scientists, domain experts, and stakeholders crucial for success. The project needs to be scalable to handle large volumes of text data and capable of real-time or near-real-time processing for applications requiring timely insights.

2.6 Design Diagram



3. System Requirements

3.1 User Interface

The user interface for "Sentiment Analysis using Transformer and LSTM" can be a web-based application, a command-line interface (CLI), or a graphical user interface (GUI). The interface allows users to input text data for sentiment analysis and view the analysis results. It may include features for selecting the sentiment analysis model, customizing model parameters, and visualizing sentiment analysis results. The interface would allow users to input text data for sentiment analysis and view the results, including the predicted sentiment labels and associated probabilities. It may also include features for data visualization, model customization, and performance evaluation.

3.2 Software Interface

The software interface of the project involves interactions between the sentiment analysis model, the user interface, and any external systems or services. This interface includes the implementation of the transformer and LSTM models, data preprocessing pipelines, and model evaluation components. It may use libraries and frameworks such as TensorFlow, PyTorch, and Hugging Face Transformers for model development and deployment. This interface would handle the preprocessing of text data, feeding the data into the sentiment analysis model, and returning the sentiment analysis results to the user interface for display. It may also include APIs or libraries for integrating the sentiment analysis functionality into other applications or services.

3.3 Database Interface

The project may use a database to store and manage labeled datasets, model checkpoints, and other relevant data. The database interface involves interactions with the database management system (DBMS) to store and retrieve data. It includes operations such as storing labeled datasets, retrieving data for model training, and storing model evaluation metrics. The database interface would involve interactions with the database management system (DBMS) to store and retrieve data. This interface would handle tasks such as storing labeled datasets, retrieving data for model training, and storing model checkpoints and evaluation metrics.

3.4 Protocols

The project may use various protocols for communication between different components, such as HTTP for web-based interfaces, RESTful APIs for inter-application communication, and protocols specific to the transformer and LSTM models (e.g., TensorFlow Serving for TensorFlow-based models). Additionally, the project may use standard data formats like JSON or CSV for exchanging data between components. Additionally, the project may use standard data formats like JSON or CSV for exchanging data between components.

4. Non-functional Requirements

4.1 Performance Requirements

- Speed: The sentiment analysis process should be fast and efficient, capable of analyzing large volumes of text data in a reasonable amount of time.
- Scalability: The system should be able to scale to handle increasing amounts of text data without compromising performance.
- Accuracy: The sentiment analysis model should achieve high accuracy in predicting sentiment labels to ensure reliable results.
- Real-Time Processing: The system should support real-time or near-real-time processing for applications requiring timely insights.

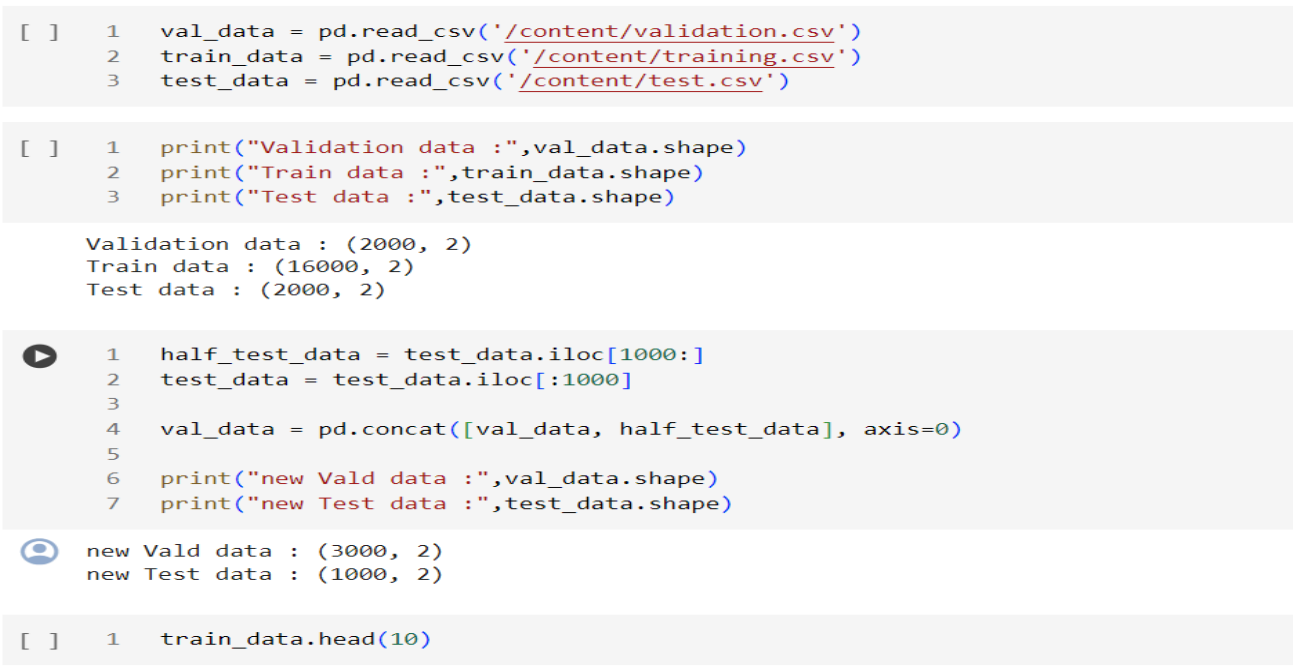
4.2 Security Requirements

- Data Privacy: The system should adhere to data privacy regulations and ensure that sensitive user data is protected.
- Authentication and Authorization: Access to the system and its data should be controlled through authentication and authorization mechanisms.
- Data Encryption: Data transmission and storage should be encrypted to protect against unauthorized access.
- Secure APIs: Any APIs exposed by the system should be secure and protected against malicious attacks.

4.3 Software Quality Attributes

- Reliability: The system should be reliable, providing consistent and accurate results under varying conditions.
- Maintainability: The codebase should be well-organized and documented, making it easy to maintain and update.
- Usability: The user interface should be user-friendly and intuitive, allowing users to easily interact with the system.
- Performance Efficiency: The system should use resources efficiently, minimizing resource consumption while maximizing performance.
- Portability: The system should be portable across different platforms and environments, allowing for easy deployment and integration.

CODE



```
[ ]  val_data = pd.read_csv('/content/validation.csv')
     train_data = pd.read_csv('/content/training.csv')
     test_data = pd.read_csv('/content/test.csv')
```

```
[ ]  print("Validation data :",val_data.shape)
     print("Train data :",train_data.shape)
     print("Test data :",test_data.shape)
```

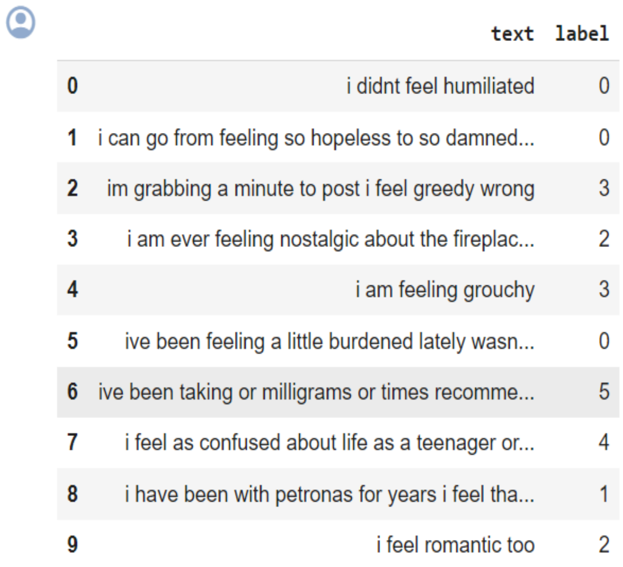
```
Validation data : (2000, 2)
Train data : (16000, 2)
Test data : (2000, 2)
```

```
half_test_data = test_data.iloc[1000:]
test_data = test_data.iloc[:1000]

val_data = pd.concat([val_data, half_test_data], axis=0)

print("new Vald data :",val_data.shape)
print("new Test data :",test_data.shape)
```

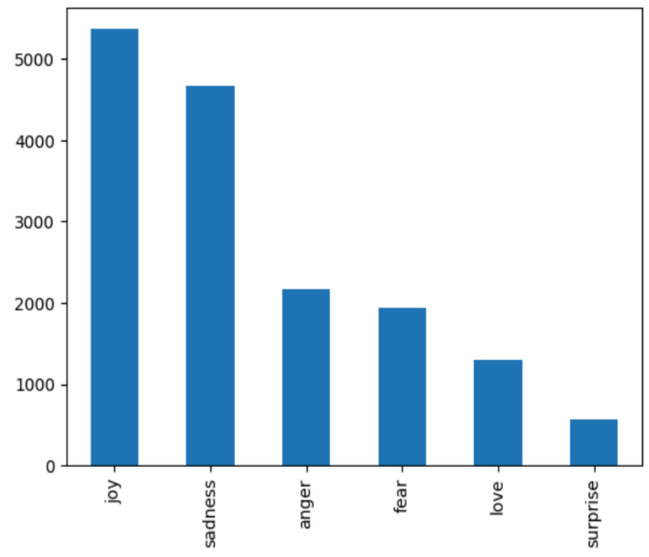
```
new Vald data : (3000, 2)
new Test data : (1000, 2)
```

```
[ ]  train_data.head(10)
```

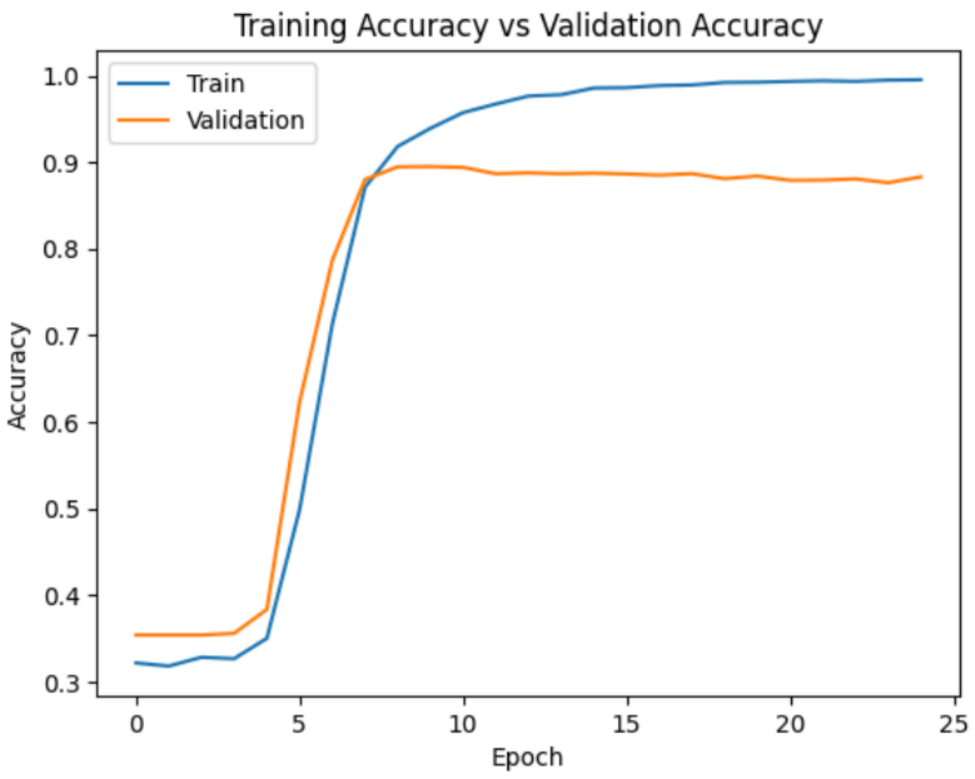
Output



	text	label
0	i didnt feel humiliated	0
1	i can go from feeling so hopeless to so damned...	0
2	im grabbing a minute to post i feel greedy wrong	3
3	i am ever feeling nostalgic about the fireplac...	2
4	i am feeling grouchy	3
5	ive been feeling a little burdened lately wasn...	0
6	ive been taking or milligrams or times recomme...	5
7	i feel as confused about life as a teenager or...	4
8	i have been with petronas for years i feel tha...	1
9	i feel romantic too	2



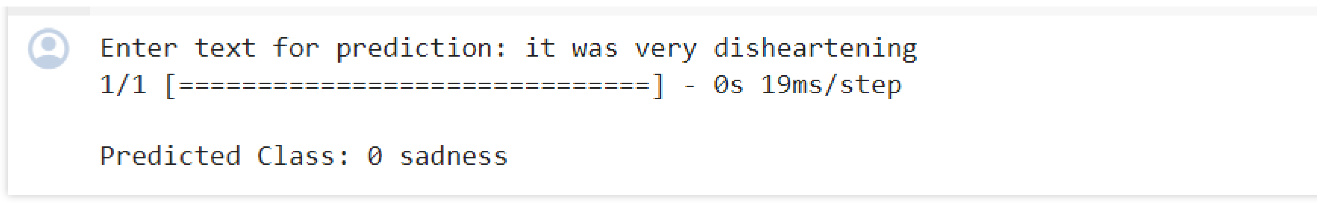
```python
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('Training Accuracy vs Validation Accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```



Code:

```python
def get_text(text):
    tokenizer3 = Tokenizer()
    tokenizer3.fit_on_texts(text)
    word_index3 = tokenizer3.word_index

    stemmed_wordss = [stemmer.stem(word) for word in word_index3.keys()]

    tokens_list = tokenizer2.texts_to_sequences([stemmed_wordss])[0]

    for i in range(len(tokens_list)):
        for j in range(length_of_longest_sentence - len(tokens_list)):
            tokens_list.append(0)
    return tokens_list

user_input = input("Enter text for prediction: ")
user_test = get_text([user_input])
user_test = np.array(user_test)
user_test = user_test.reshape(1, len(user_test))
user_predictions = model.predict(user_test)
predicted_class = np.argmax(user_predictions)
print("\nPredicted Class:", predicted_class, labels_dict.get(predicted_class))
```

Output



```
Enter text for prediction: it was very disheartening
1/1 [==============================] - 0s 19ms/step

Predicted Class: 0 sadness
```