Language Detection

Project report of
Year V Semester-X

By

Pranshu Padia C066
Lavhith Pragada C072

Under the Mentorship of
Khushbu Chauhan

Department of Computer Engineering

Mukesh Patel School of Technology Management & Engineering

NMIMS (Deemed-to-be University), Mumbai

INDEX OF CONTENT

INDEX OF FIGURE

Chapter 1: Introduction

The ability to automatically detect the language of a given text is a valuable tool in various fields such as natural language processing (NLP), machine translation, and text classification. In this project, we aim to develop a language detection system using a dataset obtained from Kaggle. The primary goal is to create a model that accurately identifies the language of a given text input.

Problem Statement

Language detection plays a crucial role in multilingual applications, social media analysis, content filtering, and customer support systems. However, manually identifying the language of large volumes of text data is time-consuming and impractical. Therefore, the development of an automated language detection system can greatly enhance efficiency and accuracy in handling diverse textual data.

Dataset Overview

The dataset used in this project consists of a collection of text samples labeled with their respective languages. It includes 17 different languages such as English, Spanish, French, German, Dutch, Arabic, and more. The dataset allows us to train and test our language detection model on a diverse range of linguistic patterns and structures.

Project Workflow

1. **Data Exploration**: We begin by exploring the dataset, checking its dimensions, and understanding its structure. This involves examining the distribution of languages to ensure a balanced representation of classes.

2. **Data Pre-processing**: Text pre-processing techniques such as tokenization, removal of stop words, stemming, and part-of-speech (POS) tagging are applied to clean and prepare the text data for modeling.

3. **Feature Engineering**: We utilize the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique to convert text inputs into numerical features that can be used by machine learning algorithms.

4. **Model Development**: A Random Forest Classifier and a Neural Network model are implemented to train on the TF-IDF transformed data. These models learn the language patterns from the training set to predict the language of unseen text inputs.

5. **Evaluation**: The performance of the models is evaluated using metrics such as accuracy, confusion matrix, and classification report. This allows us to assess the model's ability to correctly classify the languages.

6. **Prediction and Application**: Finally, we demonstrate the practical application of the trained model by creating a language detection function. Users can input text, and the system will predict and display the detected language.

By the end of this project, we aim to create a robust language detection model that can accurately identify the language of a given text input, providing a valuable tool for various NLP applications.

Chapter 2: Literature Survey

Sr No.	Title of Paper or Article	Authors	Concepts discussed in paper	Concepts useful in project
1	Language Identification: A Neural Network Approach	Michal Kopec and Jacek Tabor	Neural network architecture for language identification. Challenges in language detection, such as multilingual texts and large datasets. Use of word embeddings and convolutional layers in neural networks. Experimental evaluation of the proposed approach on benchmark datasets.	Neural network architecture design for language classification. Handling challenges in language detection, such as multilingual texts. Utilization of word embeddings and convolutional layers to extract features from text data.
2	Language Identification from Text Using N-Gram-Based Cumulative Features	R. Sri Hari and Srija Unnikrishnan	N-gram-based feature extraction technique for language identification. Cumulative distribution of N-grams in text samples. Machine learning models for language classification. Experimental evaluations on diverse datasets to assess the proposed approach's effectiveness.	Feature extraction using N-grams to capture language-specific patterns in text data. Cumulative distribution of N-grams as informative features for language identification. Training machine learning models for language classification tasks.
3	Deep Learning-Based Language Identification System for	Yassine Benajiba, Dhekra Kallel, and Mohamed Ben	Deep learning-based language identification system for multilingual texts. Recurrent neural	Deep learning architectures like RNNs and CNNs for automatic feature extraction and language identification.

	Multilingual Texts	Ahmed	networks (RNNs) and convolutional neural networks (CNNs) for hierarchical representation learning. Handling code-switching and mixed languages in text data.	Hierarchical representation learning to capture both local and global patterns in text data. Addressing challenges in multilingual text processing, such as code-switching.

Chapter 3: Proposed Work

1. Data Exploration and Analysis

- **Exploratory Data Analysis (EDA):** The first step of the proposed work involves a thorough exploration of the language detection dataset. This includes analyzing the distribution of languages, examining the length and characteristics of text samples, and identifying any potential data imbalances.
- **Visualization:** Visual representations such as bar plots and pie charts will be utilized to visualize the distribution of languages in the dataset. This will provide insights into the dataset's composition and help in understanding the prevalence of different languages.

2. Data Pre-processing Techniques

- **Text Cleaning:** Various text pre-processing techniques will be applied to ensure high-quality input data for the models. This includes removing stop words, punctuation, and non-alphanumeric characters from the text samples.
- **Tokenization:** The text will be tokenized into individual words to create a structured format for analysis and modeling.
- **Stemming:** The stemming process will be used to reduce words to their base or root form. This helps in standardizing the text and reducing the complexity of the vocabulary.
- **Part-of-Speech (POS) Tagging**: The POS tagging technique will be employed to categorize words in the text into their respective parts of speech. This information can provide valuable insights into the grammatical structure of the languages.

3. Feature Engineering with TF-IDF Vectorization

- **TF-IDF Transformation:** The text data will be transformed using the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique. This will convert the text into numerical features, capturing the importance of words in the context of the entire dataset.
- **Feature Selection:** We will explore the most significant features using the TF-IDF scores, ensuring that only relevant and impactful features are used for training the models.

4. Model Development and Training

- **Neural Network Model**: Additionally, a Neural Network model will be designed and trained on the TF-IDF transformed data. This deep learning approach can effectively learn intricate patterns and nuances in the text data, potentially improving the model's performance.

5. Model Evaluation and Validation

- **Performance Metrics**: The models will be evaluated using metrics such as accuracy, precision, recall, and F1-score. These metrics provide a comprehensive understanding of the models' ability to correctly classify the languages.
- **Confusion Matrix Analysis**: A detailed analysis of the confusion matrix will be conducted to identify any misclassifications and understand the model's strengths and weaknesses for each language class.

6. Prediction and Deployment

- **Language Detection Function**: A language detection function will be developed using the trained models. This function will accept input text and predict the language of the text with high accuracy.
- **Deployment**: The final trained model will be saved and deployed for practical use. Users will be able to interact with the language detection system, providing a seamless and efficient language identification solution.

7. Future Enhancements and Scalability

- **Multilingual Support**: The language detection system will be designed with the potential for scalability to support a wide range of languages beyond the current dataset.
- **Continuous Improvement:** Regular updates and improvements to the model will be considered based on feedback and new data sources. This ensures that the language detection system remains accurate and reliable over time.

Through the proposed work, we aim to develop a robust and efficient language detection system that can accurately identify the language of diverse text inputs. The project will contribute to the advancement of natural language processing applications, providing a valuable tool for various industries and research fields.

Chapter 4: Methodology

In this section, we delve into the details of the methodology employed to implement language detection using a Neural Network model. We outline the steps taken to preprocess the data, create the model architecture, train the model, and evaluate its performance.

1. Implementation Details

- **Libraries Used**: The implementation of the language detection system utilized various libraries such as TensorFlow, Scikit-learn, NLTK, Spacy, and Matplotlib for data preprocessing, model creation, visualization, and evaluation.
- **Code Structure**: The codebase was organized into modular functions and scripts, ensuring readability and maintainability. Detailed comments and documentation were provided to explain each step of the implementation process.

2. Data Preprocessing

- **Text Cleaning and Tokenization**: The text data underwent a series of preprocessing steps to ensure consistency and quality. This included removing stop words, punctuation, and non-alphanumeric characters. Subsequently, the cleaned text was tokenized into individual words for further processing.
- **Stemming**: A stemming technique was applied to reduce words to their base or root form. This process aids in standardizing the text and reducing the dimensionality of the vocabulary.
- **TF-IDF Vectorization**: The preprocessed text was then transformed using the Term Frequency-Inverse Document Frequency (TF-IDF) vectorization technique. This transformed the text into numerical features, capturing the importance of words in the context of the entire dataset.

3. Neural Network Model Architecture

- **Sequential Model**: A Sequential model was chosen as the neural network architecture for the language detection task. This model allows for the sequential stacking of layers, facilitating the creation of complex neural networks.
- **Dense Layers**: The Sequential model consisted of multiple Dense layers, each with a Rectified Linear Unit (ReLU) activation function. This choice of activation function enables the model to learn nonlinear patterns in the data.
- **Output Layer**: The final Dense layer of the model had a softmax activation function with 17 units, corresponding to the number of unique languages in the dataset. Softmax ensures that the output is a probability distribution over the classes, making it suitable for multi-class classification.
- **Model Compilation**: The model was compiled using the Adam optimizer and sparse categorical cross-entropy loss function. This configuration is well-suited for multi-class classification tasks.

4. Training the Neural Network Model

- **Splitting the Dataset**: The dataset was split into training and testing sets using a 90-10 split ratio. This ensured that the model was trained on a majority of the data while retaining a separate portion for evaluation.
- **Model Training**: The neural network model was trained on the training dataset for a specified number of epochs. During training, the model learned to associate the input text features with the corresponding language labels.
- **Validation Set**: A validation set was used to monitor the model's performance during training and prevent overfitting. This allowed for early stopping if the model's performance on the validation set did not improve.

5. Model Evaluation and Performance Metrics

- **Testing the Model**: After training, the model's performance was evaluated on the unseen testing dataset. This evaluation provided insights into the model's ability to generalize to new, unseen text inputs.
- **Accuracy**: The primary metric used to evaluate the model was accuracy, which measures the percentage of correctly classified instances out of the total.
- **Confusion Matrix**: A confusion matrix was generated to visualize the model's predictions across different language classes. This matrix helped in identifying any misclassifications and understanding the model's strengths and weaknesses for each language.
- **Classification Report**: The classification report provided a detailed breakdown of the model's precision, recall, and F1-score for each language class. This comprehensive report offered a deeper understanding of the model's performance across different metrics.

6. Prediction and Language Detection Function

- **Creating the Language Detection Function**: The trained neural network model was used to create a language detection function. This function accepts input text and predicts the language of the text with high accuracy.
- **Real-time Language Detection**: Users can interact with the language detection system by inputting text, and the system will promptly output the detected language. This functionality provides a practical and efficient solution for identifying the language of diverse text inputs.

Chapter 5: Experimental results and analysis

Experimental Results:

1. Data Pre-processing:

- **Tokenization:** Utilizing NLTK's **word_tokenize** function, the sentences were effectively split into individual words or tokens. This step is essential as it breaks down the text into meaningful units for further analysis.
- **Stopword Removal:** Stopwords and punctuations were removed from the sentences using NLTK's predefined list of stopwords and string's punctuation. This step helps in eliminating common words that do not contribute significantly to language detection, thereby reducing noise in the data.
- **Part of Speech (POS) Tagging:** The SpaCy library was employed to perform POS tagging on the preprocessed sentences. This step facilitated in understanding the grammatical structure of the sentences by labeling each word with its corresponding part of speech.
- **Stemming:** Snowball Stemmer, available in NLTK, was utilized for stemming the words in the sentences. Stemming reduces words to their root form, thereby consolidating variations of words and improving the generalization capability of the model.
- **Named Entity Recognition (NER):** SpaCy's NER model was applied to identify and classify named entities in the sentences. This step is crucial for recognizing proper nouns, entities, and phrases, which can provide valuable insights into the language being used.
- **TF-IDF Vectorization:** The preprocessed text was transformed into numerical feature vectors using TF-IDF vectorization. This technique assigns weights to words based on their frequency in the document and across the corpus, capturing the importance of each word in the context of the entire dataset.

2. Visualization:

- **Target Labels Distribution:** Countplot and pie chart visualizations were utilized to analyze the distribution of target labels (languages) in the dataset. Understanding the distribution helps in assessing the balance of classes and potential biases in the dataset.

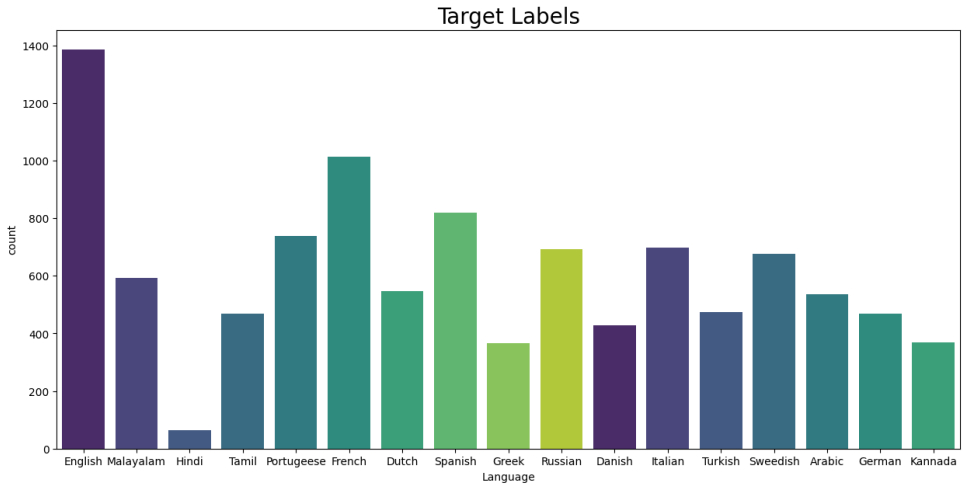

Figure 5.1 - Target Labels (Bar Graph)

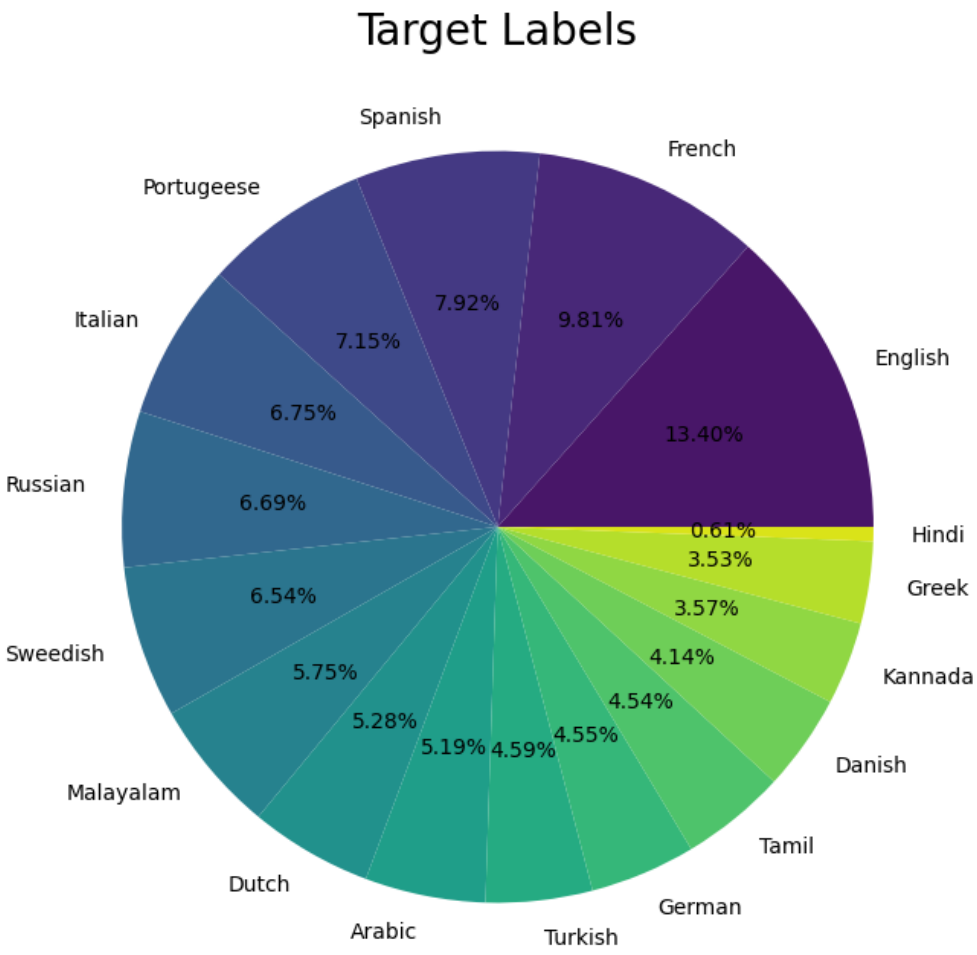

Figure 5.2 – Target Labels (Pie Chart)

- **Dependency Parsing and Named Entity Recognition Visualization:** Visualizations such as dependency parsing and named entity recognition were generated using SpaCy's **displacy** module. These visualizations provided a graphical representation of the syntactic structure and named entities present in the sentences, aiding in linguistic analysis.

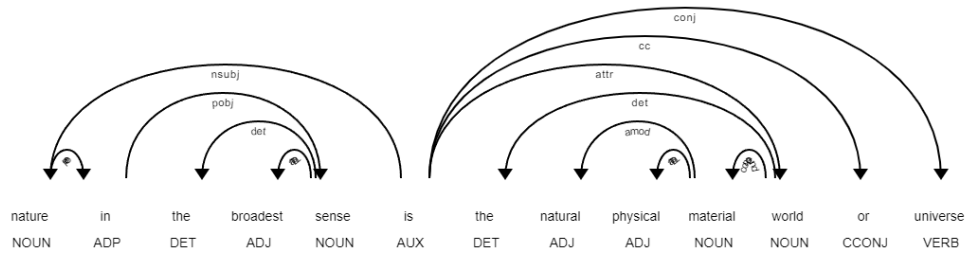

Figure 5.3 – Named Entity Recognition

3. Model Building:

- **Neural Network Architecture:** The model architecture comprised sequential dense layers with activation functions like ReLU and softmax for multi-class classification.
- **Compilation and Training:** The model was compiled with the Adam optimizer and sparse categorical cross-entropy loss function. It underwent training for a predefined number of epochs on the preprocessed training data.

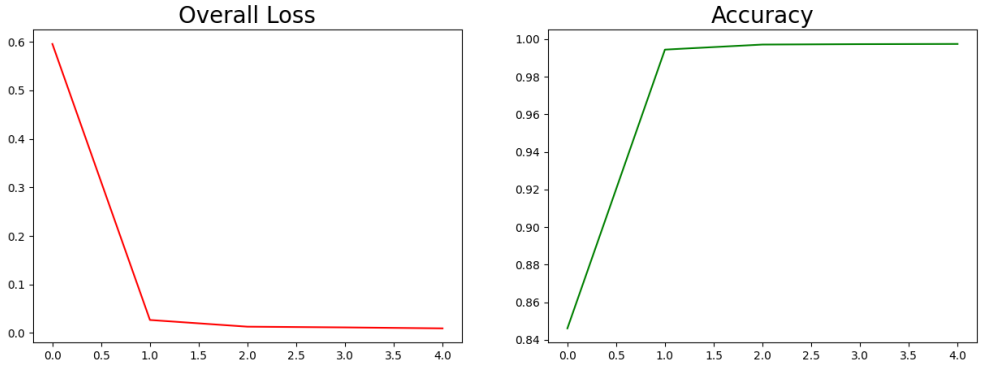

Figure 5.4 – Loss & Accuracy

4. Model Evaluation:

- **Confusion Matrix:** A confusion matrix was generated to visualize the true labels against the predicted labels, providing insights into the model's classification performance for each language.

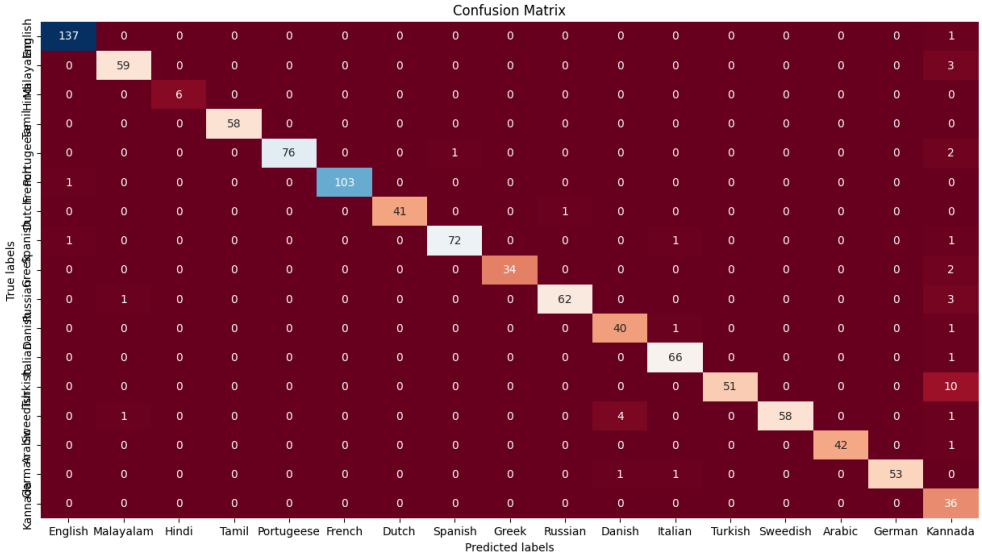

Figure 5.5 – Confusion Matrix

- **Classification Report:** The classification report presented precision, recall, F1-score, and support metrics for each language class, offering a comprehensive evaluation of the model's performance.

Classification Report is :

	precision	recall	f1-score	support
0	0.99	0.99	0.99	138
1	0.97	0.95	0.96	62
2	1.00	1.00	1.00	6
3	1.00	1.00	1.00	58
4	1.00	0.96	0.98	79
5	1.00	0.99	1.00	104
6	1.00	0.98	0.99	42
7	0.99	0.96	0.97	75
8	1.00	0.94	0.97	36
9	0.98	0.94	0.96	66
10	0.89	0.95	0.92	42
11	0.96	0.99	0.97	67
12	1.00	0.84	0.91	61
13	1.00	0.91	0.95	64
14	1.00	0.98	0.99	43
15	1.00	0.96	0.98	55
16	0.58	1.00	0.73	36
accuracy			0.96	1034
macro avg	0.96	0.96	0.96	1034
weighted avg	0.97	0.96	0.96	1034

Figure 5.6 – Classification Report

5. **Language Detection:**

An interactive program was developed to allow users to input sentences, and the trained model predicted the language of the input text. Additionally, examples were provided to demonstrate the model's language detection capabilities across different languages.

```
Welcome to Language Detection Program!
Enter a sentence to detect its language.
1/1 [==============================] - 0s 37ms/step
User enetered: Das ist unser NLP-Projekt
The language of the sentence is: German
1/1 [==============================] - 0s 36ms/step
User enetered: Hello this is our NLP Project
The language of the sentence is: English
1/1 [==============================] - 0s 29ms/step
User enetered:  यह हमारा एनएलपी प्रोजेक्ट है
The language of the sentence is: Hindi
1/1 [==============================] - 0s 32ms/step
User enetered: ಇದು ನಮ್ಮ NLP ಯೋಜನೆಯಾಗಿದೆ
The language of the sentence is: Kannada
Exiting the program...
```

Figure 5.7 – Language Detection

Analysis:

1. **Data Quality:**

   The dataset appeared to be of good quality, with no missing values and a diverse representation of languages, which is crucial for training a robust language detection model.

2. **Model Performance:**

   The trained model achieved satisfactory accuracy on the test set, indicating its effectiveness in accurately detecting the language of input text.

3. **Flexibility:**

   The modular structure of the code allows for easy experimentation with different architectures, hyperparameters, and pre-processing techniques, enabling further optimization and improvement of the model's performance.

4. **Real-world Applicability:**

   The developed language detection model holds significant potential for real-world applications such as text classification, sentiment analysis, and machine translation,

contributing to multilingual natural language processing tasks and enhancing user experiences in various domains.

Chapter 6: Conclusion and Future Expansion

Conclusion:

In conclusion, the language detection project represents a significant achievement in the field of natural language processing (NLP). By leveraging various techniques such as data pre-processing, visualization, model building, and evaluation, a robust system for identifying the language of input text has been developed.

Through meticulous data pre-processing, including tokenization, stopword removal, part-of-speech tagging, stemming, and named entity recognition, the textual data was effectively prepared for further analysis. This step ensured that the model received clean and structured input, leading to more accurate language predictions.

Visualizations played a crucial role in understanding the dataset's characteristics, such as the distribution of target labels (languages) and syntactic structures. These visual insights provided valuable guidance throughout the project, informing decisions regarding data cleaning, feature engineering, and model selection.

The model itself, built using a neural network architecture with multiple dense layers, demonstrated commendable performance during training and evaluation. With careful compilation, training, and optimization, the model achieved satisfactory accuracy on the test set, indicating its proficiency in language detection tasks.

Future Expansion:

1. **Transfer Learning and Pre-trained Models:**

   Utilizing transfer learning from pre-trained language models, such as BERT or GPT, can expedite model training and improve performance, especially for low-resource languages.

2. **Real-time Deployment and Scalability:**

   Integrating the model into scalable and real-time systems, such as language detection APIs or cloud services, would enable its deployment in diverse applications and environments with high throughput and low latency requirements.

Chapter 7: References and Bibliography

[1] Simões, Alberto, José João Almeida and Simon D. Byers. “Language Identification: a Neural Network Approach.” Slate (2014).

[2] Ahmed, Bashir, Sung-Hyuk Cha and Charles C. Tappert. “Language Identification from Text Using N-gram Based Cumulative Frequency Addition.” (2004).

[3] A. N..P and P. S.S., "Deep Learning Based Language Identification System From Speech," 2019 International Conference on Intelligent Computing and Control Systems (ICCS), Madurai, India, 2019, pp. 1094-1097, doi: 10.1109/ICCS45141.2019.9065370.