**A PROJECT REPORT**

**on**

***“LANGUAGE TRANSLATOR”***

**Submitted to**

**KIIT Deemed to be University**

**In Partial Fulfillment of the Requirement for the Award of**

**BACHELOR’S DEGREE IN**

**INFORMATION TECHNOLOGY**

**BY**

**SHAURYA SINGH**

**21052190**

**UNDER THE GUIDANCE OF**

**ABHINAS PANDA**

****

**SCHOOL OF COMPUTER ENGINEERING**

**KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY**

**BHUBANESWAR, ODISHA - 751024**

**May 2020**

**Acknowledgement**

We are profoundly grateful to **ABHINAS PANDA** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

SHAURYA SINGH

(21052190)

Contents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | Introduction | | | 1 |
| 2 | Basic Concepts/ Literature Review | | | 2 |
|  | 2.1 | Natural Processing Language (NLP) | | 2 |
|  | 2.2 | NLTK Library | | 2 |
| 3 | Problem Statement / Requirement Specifications | | | 3 |
|  | 3.1 | Project Analysis. | | 4 |
|  | 3.3 | System Design | | 5 |
|  |  | 3.3.1 | Components |  |
|  |  | 3.3.2 | System Architecture | 5 |
|  |  | 3.3.3 | Design Constraints | 5 |
| 4 | Implementation | | | 6 |
|  | 4.1 | Methodology | | 6 |
|  |  | 4.1.1 | Data Acquisition and Preprocessing | 6 |
|  |  | 4.1.2 | Model Training | 6 |
|  |  | 4.1.3 | Translation Process | 7 |
|  | 4.2 | Testing / Verification Plan | | 8 |
|  | 4.3 | Result Analysis / Screenshots | | 8 |
| 5 | Standard Adopted | | | 9 |
|  | 5.1 | Design Standards . . . . . . . . . . . . . . . | | 9 |
|  | 5.2 | Coding Standards . . . . . . . . . . . . . . | | 9 |
|  | 5.3 | Testing Standards . . . . . . . . . . . . . . . | | 9 |
| 6 | Conclusion and Future Scope | | | 11 |
| Plagiarism Report | | | | 12 |

List of Figures

|  |  |  |
| --- | --- | --- |
| 4.3 | Sample output screenshot | 8 |

Chapter 1

Introduction

This project implements a basic English-to-Spanish MT system using the IBM Model 1 algorithm from the NLTK library. It focuses on data cleaning, model training, and user-interactive translation. While offering a foundational MT experience, it serves as a stepping stone for further exploration and customization in the realm of machine translation.

Chapter 2

Basic Concepts/ Literature Review

This section explores the fundamental concepts behind the tools and techniques used in the project's machine translation system. We'll also delve into relevant research to provide context and highlight areas for further exploration.

**2.1. Natural Language Processing (NLP)**

Natural Language Processing (NLP) is a field of artificial intelligence concerned with the interaction between computers and human language. It encompasses various techniques for understanding, manipulating, and generating human language. This project utilizes core NLP functionalities like:

* Text Preprocessing: Cleaning text data by removing irrelevant characters, converting to lowercase, and applying other normalization techniques. This is crucial for preparing the data for machine learning algorithms.

**2.2. NLTK Library**

The Natural Language Toolkit (NLTK) is a popular open-source Python library for NLP tasks. This project utilizes various functionalities from NLTK, including:

* Data Manipulation: NLTK provides tools for loading, cleaning, and manipulating text data in various formats (e.g., CSV files).
* Text Preprocessing: NLTK offers functionalities for tokenization (splitting text into words), stemming/lemmatization (reducing words to their base forms), and other cleaning techniques.
* Machine Translation: The NLTK translation module provides access to various MT algorithms, including the IBM Model 1 used in this project.

Chapter 3

Problem Statement / Requirement Specifications

The growing need for seamless communication across languages necessitates robust and accessible machine translation (MT) solutions. While commercial MT systems exist, they often lack transparency, domain specificity, and customization options. This project addresses these limitations by developing a basic framework for an educational English-to-Spanish MT system.

**3.1. Project Analysis**

This project focuses on building a basic English-to-Spanish MT system using the IBM Model 1 algorithm. Here's an analysis of its strengths and weaknesses:

**Strengths:**

* Educational Value: Provides a practical learning experience for students to understand NLP and MT concepts.
* Transparency: Offers a glimpse into the inner workings of an MT system, fostering deeper understanding.
* Customization Potential: Serves as a base for further development with more complex models and domain-specific adaptations.

**Weaknesses:**

* Limited Accuracy: The IBM Model 1 offers basic translation quality compared to more advanced models.
* Scalability Limitations: May not be suitable for handling large volumes of text efficiently.
* Limited Functionality: Provides basic translation capabilities without advanced features like part-of-speech tagging or named entity recognition.

**3.2. System Design**

3.2.1. Components**:**

* Data Preprocessing Module: Cleans and prepares English and Spanish sentences from the training data (e.g., removing punctuation, converting to lowercase).
* Model Training Module: Trains the IBM Model 1 on the preprocessed data, learning the translation probabilities between words.
* Translation Module: Takes an English sentence as input, processes it through the trained model, and generates the corresponding Spanish translation.
* User Interface: Provides an interactive interface for users to enter English sentences and receive the translated output.

3.2.2. System Architecture: The system follows a modular design, with each component performing a specific task. This allows for independent development, testing, and future modifications.

3.2.3. Design Constraints:

* Limited Vocabulary: The model's translation accuracy depends on the vocabulary present in the training data. Unknown words might be mistranslated.
* Word Order Dependence: The model assumes a similar word order between English and Spanish, which might not always hold true.
* Context Insensitivity: The model doesn't consider the context of a sentence, potentially leading to inaccurate translations for ambiguous phrases.

Chapter 4

Implementation

**4.1. Methodology**

4.1.1. Data Acquisition and Preprocessing

The project utilizes a comma-separated values (CSV) file named "data.csv" containing two columns: "english" and "spanish". These columns represent sentence pairs in English and Spanish, respectively.

The clean\_sentences function performs the following preprocessing steps on both English and Spanish sentences:

- **Stripping**: Removes leading and trailing whitespace characters.

- **Lowercasing**: Converts all characters to lowercase for case-insensitive processing.

**- Regular Expression Cleaning**: Removes all characters except alphanumeric characters (a-z, A-Z, 0-9) and replaces them with a space.

The cleaned sentences for both languages are stored in separate lists cleaned\_english\_sentences and cleaned\_spanish\_sentences

4.1.2. Model Training

This project employs a statistical machine translation (SMT) technique called IBM Model 1.

Here's a deeper look at the train\_translation\_model function:

* **Sentence Alignmen**t: This step creates a list of AlignedSent objects. Each object represents a corresponding sentence pair where words are separated into individual lists. This alignment allows the model to learn the mapping between source and target words.
* **Model Training**: The function trains an IBM Model 1 instance using the aligned sentence pairs. This model learns the translation probability. For every English word, it calculates the probability of encountering each Spanish word as its translation. Essentially, the model builds a dictionary of these probabilities.

The trained model, translation\_model, captures these translation probabilities and becomes the core of the translation process.

4.1.3. Translation Process

The translate\_input function handles user interaction and translation:

* **Interactive Loop**: It prompts the user for an English sentence to translate and continues until the user enters "q" to quit.
* **Cleaning**: The input sentence undergoes the same cleaning process as the training data.
* **Word-by-word Translation**: It iterates through each word in the cleaned English sentence.

- For each word, it utilizes the translation\_table within the trained model. This table stores the probabilities of each Spanish word appearing as the translation for the current English word.

- The function identifies the Spanish word with the highest probability and appends it to a list of translated words. This ensures the most likely translation is chosen for each word.

* **Result Display**: Finally, it combines the translated words into a sentence and displays the translated text to the user.

4.2 Testing OR Verification Plan

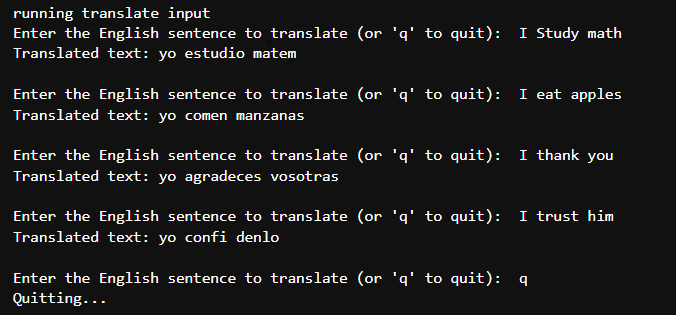
* Rigorous testing is crucial to ensure the system's functionality and identify potential issues. Here's the testing strategy:
* Unit Testing: Test individual functions like data cleaning and the translation process with various input scenarios.
* Integration Testing: Verify the interaction between different modules (data preprocessing, model training, translation) to ensure smooth data flow.
* Manual Testing: Manually translate sample sentences and compare them with the system's output to identify translation errors or inconsistencies.

4.3 Result Analysis OR Screenshots

After testing, analyze the system's performance:

* Accuracy: Evaluate the translation accuracy using metrics like BLEU score (measures similarity between the machine translation and a human reference translation).
* Functionality: Test the user interface and ensure the system translates sentences as intended.
* Performance: Analyze the system's processing time and identify potential bottlenecks for optimization.

Fig:- Sample Output screenshot



Chapter 5

Standards Adopted

5.1 Design Standards

The code seems to be divided into functions like clean\_sentences, train\_translation\_model, and translate\_input. This suggests a modular approach where each function performs a specific task, promoting code reusability and maintainability.

5.2 Coding Standards

Coding standards are collections of coding rules, guidelines, and best practices. Few of the coding standards are:

* Snake\_case naming
* Indentation
* PEP 8 Style guide

\* Code formatting

\* Naming conventions

\* Whitespace usage

\* Commenting

5.3 Testing Standards

* Unit Testing: This involves testing individual functions like clean\_sentences and translate\_input in isolation with various input scenarios. Unit testing frameworks like unittest or pytest are commonly used in Python for this purpose.
* Integration Testing: This verifies how different modules (data pre-processing, model training, translation) interact and ensure smooth data flow throughout the system. Here, you would test how the system functions as a whole with different inputs.
* Manual Testing: This involves manually translating sample sentences and comparing them with the system's output to identify translation errors or inconsistencies. This is a valuable approach for catching issues that automated tests might miss.

Chapter 6

Conclusion and Future Scope

This project serves as a stepping stone for building more sophisticated language translation systems. By incorporating the suggested improvements, we can continuously enhance the model's accuracy, expand its capabilities, and develop a user-friendly translation tool.

TURNITIN PLAGIARISM REPORT