Duplicate Question Pairs Detector

Duplicate Question Pairs Detector is an NLP and machine learning project that predicts whether two input questions are duplicates. The project includes data preprocessing, feature extraction (including token-based and fuzzy matching features), and prediction using a pre-trained machine learning model. The model and associated vectorizer and stopwords are stored as model.pkl, cv.pkl, and stopwords.pkl, respectively. This project utilizes the Quora Question Pairs dataset for training and evaluation.

Table of Contents

Overview

This project leverages NLP techniques and machine learning to identify duplicate question pairs. By preprocessing text (e.g., lowercasing, decontracting, and punctuation removal), extracting various features (such as common words, token similarity, and fuzzy match scores), and applying a trained model, the system accurately classifies whether two questions are duplicates. The training and validation of the model are based on the Quora Question Pairs dataset.

Project Components

- **app.py**
  Implements a Streamlit interface that:
  - Loads the pre-trained model from model.pkl.
  - Accepts two questions as input.
  - Uses helper functions to extract features and create a query vector.

    - Displays the prediction result ("Duplicate" or "Not Duplicate").

- **helper.py**
  Contains the following key functions:
    - **Text Preprocessing (preprocess)**: Cleans and normalizes input text by lowercasing, removing special characters, decontracting words, and stripping HTML tags.
    - **Feature Extraction**:
        - **Basic Features:** Such as the length of the questions and total/common words.
        - **Token-Based Features:** Ratios based on common non-stopwords, common stopwords, and token positions (first/last word match).
        - **Length-Based Features:** Including difference in token count and longest common substring ratio.
        - **Fuzzy Matching Features:** Using fuzzywuzzy library functions (QRatio, partial ratio, token sort, and token set ratios).
    - **Query Vector Creation (query_point_creator)**: Combines all the extracted features with bag-of-words (BoW) representations from the CountVectorizer (cv.pkl), producing the input for the prediction model.
- **Model Files**
    - model.pkl: Contains the trained machine learning model.
    - cv.pkl: A serialized CountVectorizer (or similar) used for generating BoW features.
    - stopwords.pkl: Contains a list of stopwords used during preprocessing.
- **ipynb File (Training & Exploration)**
  The Jupyter Notebook (initial_with_advance_feature.ipynb) contains the data exploration, feature engineering, and model training routines. This notebook was used to generate and save the model.pkl, cv.pkl, and stopwords.pkl files. It includes:
    - Data cleaning and preprocessing steps.
    - Feature extraction experiments.
    - Model training and validation routines.
    - Hyperparameter tuning and performance evaluation.

Installation

1. **Clone the Repository:**

bash

CopyEdit

```
git clone https://github.com/yourusername/duplicate-question-detector.git
cd duplicate-question-detector
```

2. **Create a Virtual Environment:**

bash

CopyEdit

```
python -m venv venv
source venv/bin/activate   # On Windows: venv\Scripts\activate
```

3. **Install Dependencies:**

bash

CopyEdit

```
pip install -r requirements.txt
```

Ensure that your requirements.txt includes:

   - streamlit
   - beautifulsoup4
   - distance
   - fuzzywuzzy
   - numpy
   - scikit-learn

4. **Place the Required Files:**
   - Ensure model.pkl, cv.pkl, and stopwords.pkl are in the project root directory.

Usage

1. **Run the Streamlit Application:**

bash

CopyEdit

```
streamlit run app.py
```

2. **Interact with the Application:**
    - Enter two questions into the provided text fields.
    - Click the "Find" button to see the prediction result.

Code Structure

graphql

CopyEdit

```
duplicate-question-detector/
|
├── app.py              # Streamlit interface to run duplicate question prediction.
├── helper.py           # Utility functions for text preprocessing and feature extraction.
├── model.pkl           # Trained machine learning model.
├── cv.pkl              # CountVectorizer or similar, for BoW feature extraction.
├── stopwords.pkl       # List of stopwords.
├── initial_with_advance_feature.ipynb  # Notebook for model training and experimentation.
├── requirements.txt    # List of Python dependencies.
└── docs/
    ├── Duplicate_Question_Pairs_Detector_Documentation.md  # Detailed documentation.
    └── Duplicate_Question_Pairs_Detector_Documentation.pdf   # PDF version (converted).
```

Model Training (ipynb)

The Jupyter Notebook (initial_with_advance_feature.ipynb) includes:

- **Data Exploration:** Initial data analysis and visualization.
- **Preprocessing Steps:** Implementation of text cleaning and tokenization.
- **Feature Engineering:** Development and testing of various feature extraction functions.

- **Model Training:** Training of the machine learning model, including hyperparameter tuning.
- **Model Evaluation:** Performance metrics and validation.
- **Model Serialization:** Saving the final model (model.pkl), vectorizer (cv.pkl), and stopwords (stopwords.pkl).

For further details on training and experiments, please refer directly to the notebook.

Dataset

The project uses the **Quora Question Pairs** dataset. This dataset is widely used for identifying duplicate questions, making it ideal for training and evaluating models on tasks involving question similarity and duplicate detection.

Contributing

Contributions to this project are welcome. Please follow these steps:

1. Fork the repository.
2. Create a new branch for your changes.
3. Commit your changes with clear commit messages.
4. Open a pull request for review.