Commodity Tracing

Project Overview

The Commodity Tracing Model is designed to trace the flow of specific commodities through various stages of the supply chain. It processes data from declarations, invoices, and CRN records to identify and track the movement of goods. The model uses natural language processing (NLP) techniques to clean and translate product descriptions, extract keywords, and calculate text similarity to ensure accurate matching of products across different datasets.

Data Sources

The model runs on preprocessed data which is aggregated from the following data files:

- `hdm_data.csv`
- `invoice_base.csv`
- `account_details.csv`
- `declarations.csv`

These files should be placed in the appropriate directories as referenced in the config.py.

Features

- **Data Processing:** Handles large datasets from declarations, invoices, and HDM records.
- **NLP Cleansing:** Cleans and translates product descriptions using custom NLP techniques.
- **Keyword Extraction:** Uses YAKE (Yet Another Keyword Extractor) to identify relevant keywords from product descriptions.
- **Text Similarity:** Calculates cosine similarity between product descriptions using Sentence Transformers.
- **Supply Chain Analysis:** Iteratively analyzes the supply chain to trace the flow of commodities across multiple levels.
- **Custom Configuration:** Allows for flexible configuration of product-specific parameters, such as declaration codes, HDM ADG codes, and filtering terms.

Configuration Parameters

The config.json file contains the configurable parameters for the pipeline. Key sections include:

- **product_name:** The name of the product being traced (e.g., "petrol").
- **declaration:** Configuration for processing declaration data.
- **start_date:** Start date for filtering declarations.
- **end_date:** End date for filtering declarations.
- **decl_codes:** List of declaration codes specific to the product.

- **hdm:** Configuration for processing HDM data.
- **start_date:** Start date for filtering HDM records.
- **end_date:** End date for filtering HDM records.
- **adg_codes:** List of HDM ADG codes specific to the product.
- **invoices:** Configuration for processing invoice data.
- **start_date:** Start date for filtering invoices.
- **end_date:** End date for filtering invoices.
- **filtering:** Custom filtering terms for product descriptions.
- **user_description_regex_terms:** Additional regex terms for filtering descriptions.

Installation

Follow these steps to set up the project on your local machine:

1. Clone the repository:

```
git clone https://github.com/your-repo/commodity_tracing.git
```

2. Ensure Python 3.11 is installed:

This project requires Python 3.11 You can download and install it from the official Python website: https://www.python.org/downloads/.

3. Create a virtual environment

After installing Python 3.11, create a virtual environment for the project:

For Windows:

```
python -m venv env
env/Scripts/activate
```

For macOS/Linux:

```
python3.11 -m venv env
source env/bin/activate
```

4. Install dependencies

After activating the virtual environment, install the required dependencies:

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

5. Prepare Data

Ensure that the input data files (such as decl_nov.csv, hdm_nov.csv, invoices_base_petrol.csv, acc_details_2023_nov.csv) are placed in the data/provided directory.

6. Run the main script

To run the main script, execute the following command:

```
python commodity_tracing.py
```

This will execute the model codes according to the settings defined in config.json.

Output

The model will generate output files in the data/generated directory, including:

1. Filtered declarations, invoices, and HDM records and translated and cleaned product descriptions.

- The data for each supply chain level will be saved in the folder of the corresponding level.

2. Summary tables for monetary values, quantities, and average prices.

The csv files will contain tables for monetary values, quantities, and average prices as well as level information for the selected commodity.

Code Structure

- **commodity_tracing.py:** The main script that runs the commodity tracing model.
- **config.py:** Configuration class that loads and manages settings from config.json.
- **custom_translation/:** Directory containing custom NLP cleansing and translation modules.
- **data/:** Directory for input and output data files.
- **provided/:** Input data files.
- **generated/:** Output data files.

Example

To trace the flow of petrol through the supply chain:

1. Set the product_name to "petrol" in the config.json file.
2. Provide the relevant declaration codes, HDM ADG codes, and filtering terms.
3. Run the model to generate the output files.