NEWS ARTICLE CLUSTERING PROGRAM

OVERVIEW

This program clusters news articles based on their textual similarity, enabling researchers and analysts to identify thematic patterns across news publications without manually reading every article.

The program processes articles from multiple publishers (BBC, CNN, CNA, ST) across five different themes:

1. Cybercrime & Digital Fraud
2. Forensic Science & Criminal Investigations
3. Medical Fraud & Malpractice
4. Media Misinformation & Fake News
5. Organised Crime & Drug Trafficking

It focuses on articles with high relevance scores (3-5) and applies natural language processing (NLP) techniques to group similar articles into meaningful clusters.

FEATURES

- Multi-publisher support: Processes articles from different news sources
- Theme-based processing: Organizes clustering by thematic domains
- Adaptive clustering: Automatically determines optimal number of clusters using silhouette analysis
- Text preprocessing pipeline: Comprehensive text cleaning and normalization
- Hierarchical output: Generates both detailed article-level results and cluster summaries
- Intelligent cluster labeling: Labels clusters with theme-specific prefixes (A-E) and extracts representative keywords
- Robust error handling: Gracefully handles missing files and falls back to simplified approaches when needed

REQUIREMENTS

- Python 3.6+
- Required libraries: pandas, numpy, scikit-learn, nltk, seaborn, openpyxl

INSTALLATION

Run the following command to install required packages:

pip install pandas numpy scikit-learn nltk seaborn openpyxl

You may also need to download NLTK resources:

```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
```

USAGE

1. Clone this repository
2. Ensure your file structure matches the expected format (see File Structure section)
3. Run the program: python news_article_clustering.py
4. Follow the prompts to select which theme to process (or select 'all')
5. Review the generated output files

FILE STRUCTURE

The program expects the following file structure:

```
Project Directory/
├── news_article_clustering.py
├── 01 - Cybercrime & Digital Fraud/
│   ├── 01 - BBC_Cybercrime_Articles_ProcessedUpdated.xlsx
│   ├── 01 - CNN_Cybercrime_Articles_ProcessedUpdated.xlsx
│   ├── 01 - CNA_Cybercrime_Articles_ProcessedUpdated.xlsx
│   ├── 01 - ST_Cybersecurity_Articles_ProcessedUpdated.xlsx
│   └── 02 - Articles/
│       ├── article1.txt
│       ├── article2.txt
│       └── ...
├── 02 - Forensic Science & Criminal Investigations/
│   └── ...
└── ...
```

Each article text file should follow this format:

- First line: article title
- Content starts after line 10

TECHNICAL DETAILS

Text Preprocessing:

The program employs a comprehensive preprocessing pipeline:

1. Text combination: Combines article title (with double weight) and content
2. Normalization: Converts to lowercase
3. Cleaning: Removes URLs, special characters, and numbers
4. Tokenization: Breaks text into individual words
5. Stop word removal: Eliminates common words with low semantic value
6. Lemmatization: Reduces words to their base form to consolidate similar terms
7. TF-IDF vectorization: Converts text to numerical vectors that represent term importance while accounting for document frequency

TF-IDF Vectorization parameters:

- max_features=5000 # Limit features to 5000 most important terms
- min_df=2 # Ignore terms that appear in fewer than 2 documents

- max_df=0.85 # Ignore terms that appear in more than 85% of documents
- ngram_range=(1, 2) # Include both single words and word pairs (bigrams)

Optimal Cluster Determination:

Finding the optimal number of clusters is a critical step in delivering meaningful results. The program uses silhouette analysis:

1. Iterates through a range of potential cluster numbers (2 to min(20, dataset_size/5))
2. For each number of clusters, it:
   - Creates a K-means model
   - Computes the silhouette score (a measure of cluster separation and cohesion)
3. Selects the number of clusters with the highest silhouette score

The silhouette score measures how similar each point is to its own cluster compared to other clusters. Values range from -1 to 1, with higher values indicating better-defined clusters.

The silhouette coefficient for each sample is calculated as:
(b - a) / max(a, b)
where:

- a is the mean distance between a sample and all other points in the same cluster
- b is the mean distance between a sample and all other points in the next nearest cluster

K-means Clustering:

The program uses K-means clustering to group articles:

1. Initialization: Uses k-means++ initialization to select initial centroids that are distant from each other
2. Assignment: Assigns each article to the nearest centroid
3. Update: Recalculates centroids based on the mean of all points in each cluster
4. Iteration: Repeats steps 2-3 until convergence or max_iter is reached

K-means parameters:

- n_clusters: Determined by silhouette analysis
- init='k-means++': Smart initialization of centroids
- max_iter=300: Maximum number of iterations
- n_init=10: Number of times algorithm will run with different initializations
- random_state=42: For reproducibility

Cluster Analysis:

After clustering, the program:

1. Assigns theme-specific cluster names (e.g., "A1", "B3", "E7")
2. Extracts top keywords for each cluster from centroid vectors
3. Generates consolidated master lists with cluster assignments
4. Creates cluster summaries with key statistics

OUTPUT FILES

For each processed theme, the program generates:

1. Consolidated Masterlist (XX_consolidated_masterlist.xlsx): All articles with cluster assignments and keywords
2. Cluster Summary (XX_cluster_summary.xlsx): Overview of each cluster with statistics and sample articles

UNIQUE AND INNOVATIVE FEATURES

- Adaptive prefix system: Clusters are labeled with theme-specific prefixes (A-E based on theme ID)
- Weighted title incorporation: Titles are given extra weight in the processing pipeline to better influence cluster formation
- Bigram inclusion: Uses both single words and word pairs for more contextually meaningful clusters
- Graceful degradation: Falls back to simpler models if optimal clustering fails
- Parameter tuning: Automatically adjusts TF-IDF and clustering parameters based on dataset properties
- Silhouette optimization: Uses mathematical measure of cluster quality rather than arbitrary cluster numbers

LICENSE

AUTHOR

Your Name