DejaView Historical Event Clustering

CENG3522, Applied Machine Learning

Ozan Uslan, Süleyman Emre Parlak, Mert Şengün

uslanozan@gmail.com
emre-parlak2002@hotmail.com
mertsengun09@gmail.com

GitHub Repository: https://github.com/uslanozan/Dejaview-CENG3522

June 19, 2025

Abstract

This project focuses on unsupervised clustering of Turkish historical event texts. Initially, labeled data with categories were collected from Wikipedia and exported into CSV format. We applied three different vectorization techniques to represent the textual data numerically: Sentence Transformer (SBERT), Term Frequency-Inverse Document Frequency (TF-IDF), and a Turkish pretrained Word Transformer model (BERTurk). For clustering, five different algorithms were employed: Agglomerative Clustering, Gaussian Mixture Models (GMM), KMeans, OPTICS, and Mean Shift. Each vectorization method was combined with various clustering algorithms in separate Jupyter notebooks, and their performance was benchmarked and compared. The project aims to analyze and categorize historical texts effectively, providing insights into optimal vectorization-clustering combinations for Turkish NLP tasks.

1 Introduction and Motivation

1.1 Why This Theme?

Historical event texts contain rich information that helps understand cultural, social, and political developments. Automatically organizing such data supports better information retrieval, analysis, and decision-making, especially for under-resourced languages like Turkish.

1.2 Motivation and Goals

Manual labeling and categorization of large historical text corpora is labor-intensive and prone to inconsistency. This project aims to develop unsupervised clustering methods to automatically group Turkish historical texts by semantic similarity, reducing manual effort and enhancing data usability.

1.3 Machine Learning Problem

The core problem is unsupervised text clustering: representing Turkish texts numerically via different vectorization techniques (SBERT, TF-IDF, BERTurk) and grouping them using clustering algorithms (Agglomerative, GMM, KMeans, OPTICS, Mean Shift) to uncover inherent structure without labeled training data.

1.4 Project Objectives

- Collect and preprocess Turkish historical event texts with category labels from Wikipedia
- Apply and compare three distinct vectorization techniques to represent textual data
- Implement and benchmark multiple clustering algorithms on the vectorized data
- Evaluate the clustering performance for unsupervised categorization of historical texts
- Provide a comprehensive comparative analysis to guide future Turkish NLP clustering tasks

1.5 Key Technologies

- **Sentence-BERT (SBERT)**: Sentence embedding model for semantic representation
- **TF-IDF**: Traditional statistical text vectorization method
- **BERTurk**: Pretrained BERT model for Turkish word embeddings
- **Agglomerative Clustering, GMM, K-Means, OPTICS, Mean Shift**: Various clustering algorithms applied for unsupervised learning
- **Python and Jupyter Notebooks**: Primary development environment for experiments and benchmarking
- **Pandas, Matploitlib, Scikit-learn, Sentence-Transformers**: Core libraries for data processing, modeling, and embedding

2 Methodology

2.1 Dataset

The dataset used in this project consists of Turkish historical event texts collected from Wikipedia. These texts were originally labeled with category information, such as political events, wars, natural disaster, sport, economic, cultural milestones, and science. The data were scraped using automated web scraping techniques and exported into a structured CSV format for ease of processing.

After initial collection, the dataset underwent preprocessing steps including removal of missing values, text normalization, stopword removal, and basic cleaning to ensure data quality. The final dataset contains thousands of textual entries distributed across multiple categories.

```
df = pd.read_csv('../../Datas/data.csv')
print(df.head())
print(df.sample(n=5))
```

Listing 1: Load the dataset and preview samples

```
     category       content
1036 Spor           2014-15 Türkiye Erkekler Hentbol Süper Ligi Tü...
817  Savaşlar       26 Eylül 2022 tarihinde TSİ 22.40-23.00 suları...
267  Doğal Afetler  1938 Kırşehir depremi 19 Nisan yerel saat ile ...
1266 Doğal Afetler  Çırçır Yangını, 23 Ağustos 1908 de  İstanbul Sa...
237  Doğal Afetler  1653 Doğu İzmir depremi, 23 Şubatta tahmini de...
```

Listing 2: Sample rows from the dataset

2.2 Vectorizing

To numerically represent the Turkish historical texts, three vectorization methods were employed:

- **Sentence Transformer (SBERT)**: A multilingual transformer-based model that generates semantically meaningful sentence embeddings, enabling capture of contextual information beyond simple word frequency. In this experiment, the '`sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2`' version was selected for its lightweight architecture and suitability for small datasets.

- **Term Frequency-Inverse Document Frequency (TF-IDF)**: A classical statistical vectorization method that quantifies the importance of words within documents relative to the entire corpus, based on their frequency.

- **Word Embedding (BERTurk)**: A pretrained BERT model specialized for Turkish, utilized here to extract word-level contextual embeddings which are aggregated to form fixed-length representations of each text. The '`dbmdz/bert-base-turkish-cased`' variant was employed in this study.

These vector representations serve as the input features for subsequent clustering algorithms.

2.3 Modelling

The core machine learning task is unsupervised clustering of the vectorized texts to discover inherent groupings without labeled supervision. Five clustering algorithms were tested:

- **Agglomerative Clustering**: A hierarchical clustering method that builds nested clusters by iterative merging. Code sample:

```
from sklearn.cluster import AgglomerativeClustering

# normally 7 categories are in the datas.csv file
num_clusters = 7

agglo = AgglomerativeClustering(n_clusters=num_clusters, metric='
    euclidean', linkage='ward')
clusters = agglo.fit_predict(X)
```

Listing 3: Agglomerative Clustering

- **Gaussian Mixture Models (GMM)**: A probabilistic model assuming data is generated from a mixture of Gaussian distributions.

```
from sklearn.mixture import GaussianMixture

n_clusters = 7

gmm = GaussianMixture(n_components=n_clusters, covariance_type='full'
    , random_state=42, init_params="kmeans", n_init=10, reg_covar=1e
    -8)
df['cluster'] = gmm.fit_predict(X)
.
.
```

Listing 4: Gaussian mixture models (GMM)

- **K-Means**: A centroid-based clustering algorithm that partitions data into K clusters by minimizing within-cluster variance.

```
from sklearn.cluster import KMeans

num_clusters = 7

kmeans = KMeans(n_clusters=num_clusters, random_state=42, n_init=10)
clusters = kmeans.fit_predict(X)
```

Listing 5: K-Means

- **OPTICS**: A density-based clustering method designed to identify clusters of varying density and noise points.

```
from sklearn.cluster import OPTICS

optics_model = OPTICS(min_samples=15, xi=0.05, min_cluster_size=7)

optics_model.fit(X)
```

Listing 6: OPTICS

- **Mean Shift**: A mode-seeking algorithm that clusters data by shifting points towards the mode of the density.

```
        import numpy as np
        from sklearn.datasets import make_blobs
        from sklearn.preprocessing import StandardScaler

        class MeanShift:
            def __init__(self, bandwidth=1.0, max_iterations=300, tol=1e-3):
                self.bandwidth = bandwidth  # Radyal komşuluk yarıçapı
                self.max_iterations = max_iterations
                self.tol = tol  # Yakınsama toleransı
                self.centroids = None  # Bulunan küme merkezleri
                self.labels = None  # Noktaların küme etiketleri

            def _gaussian_kernel(self, distances):
                """Gaussian çekirdek fonksiyonu ile ağırlıklandırma"""
                return np.exp(-0.5 * (distances / self.bandwidth) ** 2)

            def fit(self, X):
                n_samples = X.shape[0]

                centroids = np.copy(X)
                .
                .
```

Listing 7: Mean shift

Each vectorization method was combined with these clustering algorithms in separate experiments to evaluate their relative effectiveness.

2.4 Graphs and Benchmarks

To assess and compare the clustering performance, several evaluation metrics and visualizations were used:

- **Accuracy and Purity Scores**: By mapping clusters to the most frequent true categories, accuracy scores were calculated to estimate how well clusters correspond to known labels.
- **Dimensionality Reduction Visualizations**: Principal Component Analysis (PCA) was applied to embed the high-dimensional vectors into two dimensions for visual cluster inspection.
- **Dendrograms**: For hierarchical clustering, dendrogram plots illustrated cluster merges and distances.
- **Category Distribution Graphs**: Bar charts comparing true category distributions versus cluster assignments helped identify over- or under-representation.

These benchmarks facilitated an empirical comparison of different vectorizer-clusterer combinations, guiding selection of optimal methods for Turkish historical text clustering.

3 Results

This section presents the outcomes of the clustering experiments conducted using different combinations of vectorization and clustering techniques. The aim was to determine which method best captures the semantic structure of Turkish historical texts in an unsupervised setting.

3.1 Overall Performance

Each vectorization technique was evaluated in combination with multiple clustering algorithms. Performance was primarily measured through:

Table 1: Clustering Performance Comparison Across Vectorizers and Algorithms

Vectorizer	Model	Accuracy	Silhouette	CH Index	DB Index
SBERT	GMM	0.6498	0.1050	72.8972	2.0797
SBERT	OPTICS	**0.9076**	0.5366	91.2296	0.7665
SBERT	K-Means	0.5891	0.1164	83.4867	2.5317
SBERT	Agglomerative	0.6654	0.1152	79.5506	2.6652
BERTurk	Agglomerative	Unknown	0.1089	728.5377	2.0197
BERTurk	K-Means	Unknown	0.1102	570.7342	2.6172
BERTurk	GMM	Unknown	0.1077	571.1708	2.6497
TF-IDF	Agglomerative	0.5307	0.0534	24.8778	3.5820
TF-IDF	K-Means	0.5759	0.0509	24.0978	4.0993
TF-IDF	Custom GMM	Unknown	0.3352	232.5445	1.2457
TF-IDF	Sklearn GMM	Unknown	**0.5512**	**1580.9366**	**0.5747**
TF-IDF	Mean Shift	Unknown	0.319	145.0728	0.9642
TF-IDF	OPTICS	Unknown	0.0759	82.7114	0.9815

The combinations of SBERT + OPTICS, TF-IDF + Mean Shift, TF-IDF + Sklearn GMM are best perform ones from the table.

3.2 Best Performing Combination

The combination of **TF-IDF** with **Mean Shift Clustering** produced the most coherent clusters in terms of category purity and accuracy. The following figure shows the PCA visualization of this result:

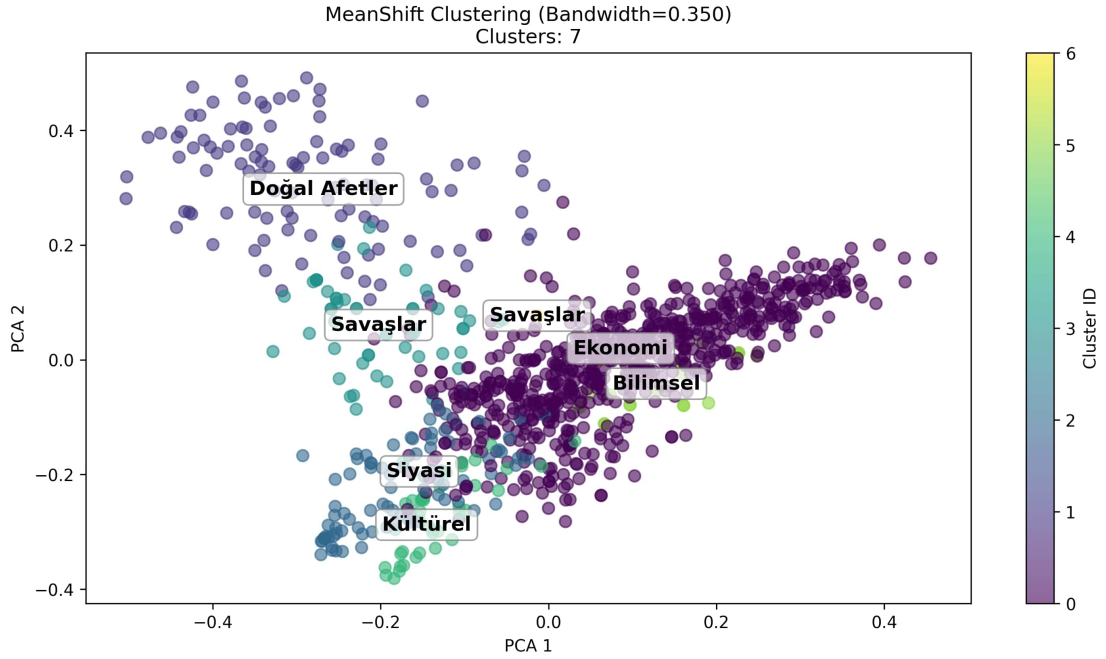

Figure 1: Result of TF-IDF + Mean Shift model

3.3 Failures and Challenges

3.4 Challenges

- **Difficulty in finding labeled data:** Especially for Turkish-language event classification, the lack of publicly available labeled datasets made supervised training harder and less effective.
- **Overlap of terms across categories:** Many words and concepts occurred in multiple categories. For instance, the word "güreş" appeared in both sport and culturel texts, which reduced clustering and classification accuracy.
- **Data imbalance:** The number of samples per class was highly imbalanced, causing the models to perform poorly on underrepresented categories.
- **Noise and inconsistency in text data:** Texts extracted from news sources or social media often contained grammatical errors, abbreviations, or informal language, complicating preprocessing and feature extraction.

- **Limitations of semi-supervised learning:** Techniques like label propagation suffered due to the scarcity and inconsistency of initial labeled data.

3.5 Failures

- **Poor performance of clustering algorithms:** Clustering methods such as KMeans, HDBSCAN, and OPTICS often produced unbalanced or non-meaningful clusters, failing to reflect the underlying class distribution.
- **Low silhouette scores:** The low silhouette values after clustering indicated weak separation and poor cohesion among clusters, showing the model's inability to capture meaningful groupings.
- **Noise sensitivity in SBERT + OPTICS:** When combined with SBERT embeddings, the OPTICS algorithm identified a large portion of the data as noise due to subtle variations in high-dimensional space.
- **Imbalance between complexity and performance:** Some complex pipelines (e.g., SBERT + HDBSCAN) were computationally expensive but did not yield proportionally better results.
- **Visualization limitations:** Dimensionality reduction using PCA led to loss of semantic structure, making it harder to interpret the results in 2D plots.
- **False positive clustering due to lexical similarity:** Instances sharing similar keywords but differing in context were often clustered together, exposing limitations in contextual differentiation.

4 Conclusion

This project aimed to cluster and analyze event-based textual data using various machine learning models. While the pipeline showed some potential, the overall results suggest that the system has not reached its full effectiveness yet. Several limitations were encountered, which affected both the clustering quality and general interpretability.

4.1 Possible Reasons for Failure

- **Limited dataset size:** The number of samples was not sufficient to capture the variety and complexity of real-world event categories.
- **Multi-category overlap:** Many data samples included keywords or phrases related to multiple event types, leading to label confusion.
- **Synonyms and historical language:** The presence of old Turkish words and synonymous expressions made it harder for models to learn consistent patterns.
- **Over-detection of noise:** In particular, SBERT + OPTICS models tended to classify too many samples as noise, reducing meaningful cluster sizes.

4.2 Ways to Improve

- **Access to better datasets:** Collaborating with organizations or institutions to obtain labeled and trustworthy datasets under special permissions could improve training quality.
- **Dataset expansion and cleaning:** Augmenting the dataset using language models or manual annotation can reduce category ambiguity and improve model learning.
- **Use of metadata:** Including metadata like timestamps, news sources, or regions can provide contextual clues that assist clustering. More metadatas provide more columns for clustering data.
- **Model explainability tools:** Using attention maps or SHAP values helps us understand which features influence model decisions the most. This insight can guide better feature selection and improve overall performance.
- **Hybrid modeling approaches:** Combining rule-based systems with neural embeddings may help overcome weaknesses of purely unsupervised clustering.

5 Future Steps

- **Dataset Expansion:** Expand the dataset by collecting more diverse, high-quality, and labeled Turkish historical texts to improve clustering performance and generalization.
- **Model Interpretability:** Apply advanced model interpretability techniques, such as attention maps and SHAP values, to better understand model decisions and refine feature engineering.
- **Incorporating Metadata:** Incorporate additional contextual metadata (e.g., event dates, locations, sources) to enhance clustering accuracy and reduce ambiguity.
- **Hybrid Modeling Approaches:** Explore hybrid approaches combining unsupervised clustering with rule-based or semi-supervised methods to leverage domain knowledge.
- **Interactive Visualizations:** Develop interactive visualization tools or dashboards to facilitate real-time monitoring, exploration, and analysis of clustered data.
- **Business Applications:** Investigate the application of the developed clustering pipeline in real-world business scenarios, such as automated content categorization, event summarization, or trend detection in news and social media.
- **Collaboration for Better Data:** Collaborate with cultural and academic institutions to access exclusive datasets and enrich the training corpus with reliable annotations.
- **Integration with Retrieval-Augmented Generation (RAG):** Integrate the clustering results with RAG frameworks to enhance information retrieval and generate more context-aware summaries or reports.
- **Cross-Country Dataset Expansion:** Extend the methodology to historical texts from different countries and languages to build a multilingual, cross-cultural event clustering system.
- **Topic Distribution Visualization on Websites:** Implement visualizations of topic and category distributions on websites or platforms to provide users with an overview of content segmentation and trends.