CYSHIELD

Hot keywords extraction and topic modeling

Dec 23, 2023

Submitted by :

Eng. Zakaria Abdallah

Table Of Content

Introduction

Imagine sifting through a mountain of text, not for the familiar peaks of recurring themes, but for the fleeting sparks of words bursting into sudden popularity. This domain within NLP, aptly named "hot keyword extraction," delves into the dynamic realm of trending topics and real-time buzz. It's like tracking the hottest hashtags on Twitter or the most searched terms on Google, but with added layers of complexity.

These "hot keywords" are the ephemeral fireflies of language, illuminating trends, events, and conversations that flare up and fade within specific timeframes. Identifying them requires nimble algorithms that can navigate the ever-shifting currents of human discussion. They must distinguish between the fleeting sparks of a meme and the sustained glow of a genuine cultural shift. They must adapt to the chameleon-like nature of language, where slang terms morph overnight and new phrases emerge from the digital ether.

But the challenges are worth the rewards. Hot keyword extraction offers a real-time snapshot of the collective mind, a fleeting glimpse into what captivates and preoccupies us at any given moment. It informs marketing campaigns, guides editorial decisions, and even sheds light on societal concerns and emerging trends. From understanding the latest consumer craze to monitoring potential crises, these ephemeral keywords offer a powerful lens through which to view the ever-evolving landscape of human communication.

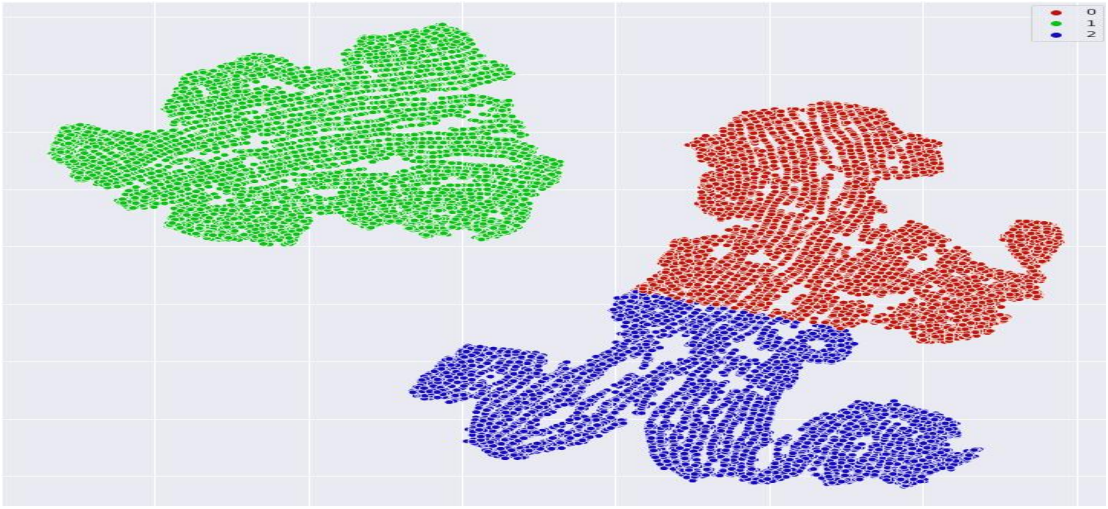
So, the next time you see a trending hashtag or a new phrase lighting up your social media feed, remember the intricate dance of algorithms and linguistics behind the

scenes. They're the silent heroes, extracting the sparks of hot keywords and illuminating the dynamic tapestry of human conversation.

Feature extraction acts as the bridge between raw text and its deeper meaning. Imagine documents as bags of words, filled with terms that might tell us about their content. This step meticulously selects significant features, discarding the common fluff like pronouns and prepositions. Techniques like TF-IDF (Term Frequency-Inverse Document Frequency) weigh each word based on its prevalence within a document and across the entire corpus, highlighting informative terms that truly stand out.

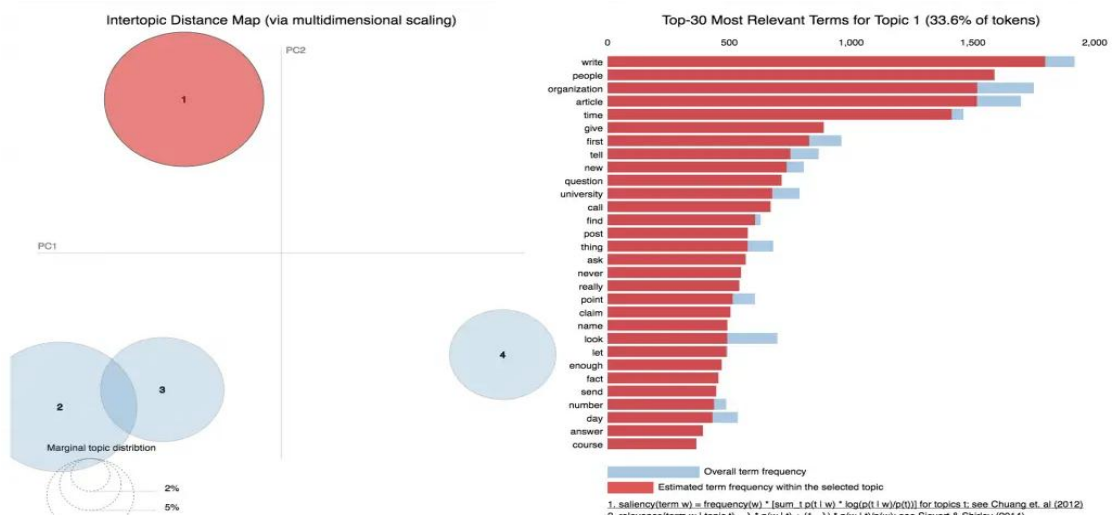
Now, equipped with these features, we embark on the journey of topic modeling. Here, two distinct approaches emerge:

- K-means clustering: Picture sorting documents into pre-defined groups based on their feature similarity. Imagine placing these bags of words on a map, clustering those with similar keywords close together. While straightforward, k-means might struggle with nuanced topics or overlapping themes.



- Latent Dirichlet Allocation (LDA): Enter the probabilistic world of LDA, where documents are seen as mixtures of hidden "topics." Each topic comprises a set of features with a particular distribution, and each document belongs to multiple topics with varying degrees. Think of this as a layered,

probabilistic tapestry, where documents blend different thematic threads in unique ways.



So, which approach takes the crown? It depends on your data and goals. K-means offers simplicity and clear-cut clusters, while LDA unveils latent structures and explores the probabilistic interplay of themes. Often, these techniques can even be combined, with k-means refining topics discovered by LDA

No matter the chosen path, NLP feature extraction and topic modeling open up a world of possibilities. From analyzing customer reviews to understanding scientific literature, these techniques empower us to extract knowledge, identify trends, and navigate the vast ocean of text with newfound comprehension.

Data description

In this task I used 2 different datasets

TRC Victims:

- Types of crimes: The data includes information on a wide range of politically motivated crimes that occurred during apartheid in South Africa, including killings, torture, detention without trial, and forced disappearances.

- Victims: The data includes information on over 21,000 victims, including their names, ages, genders, and locations. In some cases, the data also includes information about the victims' occupations, political affiliations, and family members.
- Perpetrators: The data often identifies the perpetrators of the crimes, who were often members of the security forces or government-backed vigilante groups.
- Dates and locations: The data includes information on the dates and locations of the crimes, which can help to paint a picture of the geographical and temporal patterns of apartheid-era violence.
- Sources: The document includes references to other documents that provide more information about the incidents, such as the TRC hearings and reports. This can be helpful for researchers who want to learn more about specific cases or events.

USHMM English Oral Testimonies Dataset:

- **Dataset Summary**

This is a collection of approximately 1,000 English Oral Testimonies at the United States Holocaust Memorial Museum (USHMM). The oral testimonies were

collected during the late-twentieth and early twenty-first centuries. These were converted from PDFs into raw text with Tesseract. The text was post-processed with a Python script to convert it into segments of dialogue. Because this process was automated, mistakes may remain. Occasionally, headers and footers appear in the middle of the dialogue. If found, submit an issue and these can be corrected. This dataset was created during William J.B. Mattingly's postdoc at the Smithsonian Institution's Data Science Lab which had a cross-appointment with the USHMM. This dataset is being used for text classification, named entity recognition, and span categorization.

- **Languages**

These testimonies are strictly in English, but they were given by non-native speakers. This means foreign language words and phrases may appear throughout the testimonies.

- **Dataset Structure**
  - **Data Fields**
    - **rg:** String, the RG number used by the USHMM to identify specific items in a collection.
    - **sequence:** Integer, the unique ID for the dialogue row.
    - **text:** String, the actual piece of dialogue.
    - **category:** String, can be a question or an answer.

TF-IDF and K-means

1. Abstract:

Topic Modeling/Text Classification Series, which demonstrates using TF-IDF and K-Means clustering for topic modeling on textual data from the TRC's Detainee No More report.

2. Code Breakdown:

- **Data Loading:**
    - load_data(file): Reads a JSON file and returns the data.
    - write_data(file, data): Writes data to a JSON file with indentation.
- **Data Cleaning:**
    - remove_stops(text, stops): Removes punctuation, stop words, digits, and extra spaces from text.
    - clean_docs(docs): Applies remove_stops to each document in a list.
- **TF-IDF Vectorization:**
    - Define a TfidfVectorizer with parameters:
        - lowercase: Convert text to lowercase.
        - max_features: Maximum number of features (words) to consider.
        - max_df: Maximum document frequency for a word to be considered.
        - min_df: Minimum document frequency for a word to be considered.

        - ngram_range: Range of n-grams to consider (1-3 in this case).
        - stop_words: Stop word list ("english").
    - Fit the vectorizer to the cleaned documents and get the tf-idf vectors and feature names.
- **Topic Modeling with K-Means Clustering:**
    - KMeans model with k=10 clusters (true_k).
    - Fit the model to the tf-idf vectors.
    - Find the top 10 keywords for each cluster based on the cluster centroids.
    - Save the results to a text file, including cluster labels and top keywords.

3. Improvements and Additions:

- The code could be more modular by separating preprocessing, vectorization, and clustering into functions.
- Consider experimenting with different TF-IDF parameters and KMeans hyperparameters.
- Evaluate the quality of the clusters using metrics like silhouette score or adjusted Rand score.
- Explore visualizing the topics using word clouds or other techniques.

Topic Modeling with Gensim LDA on USHMM Data

1. Abstract:

This code focuses on topic modeling of textual data from the United States Holocaust Memorial Museum Detainee No More report using Gensim library. It involves data pre-processing, text cleaning, dictionary creation, TF-IDF removal, and finally, applying Latent Dirichlet Allocation (LDA) to identify latent topics.

2. Code Breakdown

- **Data Loading and Preprocessing:**
  - Data is loaded from a JSON file ("ushmm_dn.json") containing texts as a list of lists.
  - Preprocessing includes:
    - Lemmatization using SpaCy to convert words to their base form.
    - Tokenization and removal of stop words using Gensim's simple_preprocess.
    - Generation of bigrams and trigrams using Gensim's Phrases and Phraser classes.
- **TF-IDF Removal:**
  - Gensim's TfidfModel is used to remove words with low TF-IDF scores (defined by low_value).
  - The corpus is updated with new bag-of-words representations after removing these low-scoring words.

- **LDA Model Training and Evaluation:**
  - An LDA model with 10 topics is trained using the updated corpus and dictionary.
  - A test document is then used to demonstrate topic prediction.
  - The top topics for the test document are extracted and sorted based on their probability scores.
- **Model Saving and Future Exploration:**
  - The trained model is saved to a file for further analysis or use.
  - The code provides a framework for further exploration, such as:
    - Experimenting with different LDA parameters like number of topics, alpha, or passes.
    - Evaluating the model using coherence measures like perplexity or UMass coherence.
    - Visualizing the topics using word clouds or other techniques.
- **Additional Notes:**
  - The code uses SpaCy for lemmatization, but other NLP libraries like NLTK could be used.
  - The specific threshold values for bigram/trigram generation and TF-IDF removal might require tuning based on the data and desired results.
  - This code provides a basic example of LDA topic modeling, and further refinement and analysis are recommended for deeper insights.

Overall conclusion

Overall this code demonstrates a workflow for topic modeling on textual data using Gensim and LDA. It provides a starting point for exploring latent topics within the USHMM Detainee No More report and can be adapted for similar tasks with other text datasets. For topic modeling and feature extraction from text articles, you can't go wrong with both but for a better understanding:

- **K-means** is usually much better and faster when it comes to shorter text articles and it has proved its reliability within the example provided.
- **LDA** is a really good when it comes to long text with many topics inside them as it can easily mange to extract the related topics inside the articles.