SI630 Project Proposal

Winter 2025

Elijah Cantu

Abstract

This project aims to develop an intelligent system for identifying and filtering news content from search engine results. With the rise of doomscrolling—the habit of continuously consuming negative news—search engines often contribute to this behavior by surfacing such content. To address this issue, I will train a natural language processing (NLP) classifier to categorize search results as either news or non-news based on their textual content. The classifier will be deployed via a Chrome extension that interacts with Google search results, automatically hiding news articles using CSS. By leveraging NLP and machine learning techniques, this project seeks to create a less engaging and more balanced search experience, prioritizing non-news results. The outcome will be a practical, browser-integrated tool that helps users reduce exposure to news content and mitigate doomscrolling habits.

1 Introduction

In today's digital landscape, search engines serve as primary gateways to information. However, they often amplify doomscrolling—the compulsive consumption of negative news—by surfacing a high volume of news content. While staying informed is essential, excessive exposure to distressing news can contribute to anxiety and information fatigue. To address this issue, this project proposes an intelligent system that filters news content from search engine results, enabling users to have a more balanced and less overwhelming browsing experience.

This project leverages Natural Language Processing (NLP) and machine learning techniques to develop a classifier that distinguishes between news and non-news search results based on textual content. The classifier will be trained on a binary dataset consisting of labeled search results—categorized as either "news" or "non-news." Once trained, the model will be integrated into a Chrome extension that interacts with Google search results, automatically hiding news articles using CSS. By filtering out news content at the search engine level, this system aims to reduce the visibility of news articles, making doomscrolling less enticing while ensuring access to non-news information remains unaffected.

The motivation behind this project is twofold. First, it enhances user experience by allowing individuals to engage with search results without being overwhelmed by news content. Second, it provides a practical tool for those who wish to limit their exposure to news without completely disconnecting from the internet. If successful, this approach could be expanded to support additional filtering options, offering users more control over their online information consumption.

By addressing the issue of excessive news exposure at the search engine level, this project contributes to a healthier digital ecosystem—one where users have more agency over the content they interact with.

2 Problem Definition

The problem this project seeks to address is the detection and removal of news content from search engine results to reduce doomscrolling and improve user experience. Doomscrolling, the compulsive consumption of negative news, has become a widespread issue exacerbated by search engines that frequently prioritize news content. Users searching for general information may find themselves overwhelmed by news articles, often leading to unintended and excessive engagement with distressing content.

The challenge lies in distinguishing between news and non-news results based on their textual content. This system will process the titles and brief descriptions of search results returned by Google and classify each result into one of two categories:

- **News**: Articles that report on current events, politics, business, health, and other newsworthy topics.
- **Non-News**: Webpages that do not primarily serve as news articles, such as blogs, opinion pieces, research papers, forums, product pages, and other informational content.

The input to the system will be the titles and descriptions of search engine results from Google. The output will be a classification for each result: either ”news” or ”non-news.” The system will be implemented as a Chrome extension that interacts with Google’s search results page, automatically detecting and filtering out news articles by hiding them using CSS.

What the Problem is Not

- The system will not assess the credibility, accuracy, or bias of news articles. It will solely focus on identifying whether a result is classified as news or non-news based on its textual content.
- The project does not aim to suppress or censor news altogether but rather to provide users with the option to reduce their exposure to news content during searches.
- The system will not engage in sentiment analysis, fact-checking, or any deeper content verification beyond the classification task.

Success Definition The success of the project will be evaluated using the following metrics:

1. **Accuracy**: The percentage of search results correctly classified as news or non-news. A high accuracy rate indicates the effectiveness of the classifier.
2. **Precision, Recall, and F1-Score**: These metrics will be computed for both categories to assess classification performance.
    - **Precision** measures how many of the identified news results are actually news.
    - **Recall** evaluates how many actual news results were successfully identified.
    - **F1-Score** balances precision and recall to provide a single metric for performance.

By successfully filtering out news articles from search results, this project aims to provide users with a more controlled and less overwhelming search experience while preserving access to non-news content.

3 Data (1 point)

3.0.1 Data

This project utilizes the **3DLNews2** dataset, a large-scale collection of U.S. local news articles spanning nearly three decades (1995–2024). The dataset consists of over **8 million URLs**, with a refined subset of more than **4 million filtered news article URLs**. The data was gathered via an extensive three-month scraping process from **Google and Twitter search results**, using a rigorous filtering pipeline to exclude non-news links.

Data Source & Acquisition The dataset was obtained from the **3DLNews2 project**, which is publicly accessible via **Globus**. The dataset authors, **Gangani Ariyarathne and Alexander C. Nwala**, granted permission to use both the dataset and preprocessing scripts(Ariyarathne and Nwala, 2024). The dataset is structured into different categories based on media types:

- **Newspapers** (9,441 sources)
- **Radio Stations** (2,449 sources)
- **TV Stations** (886 sources)
- **Broadcast Networks** (1,310 sources)

For this project, **only the Google search results portion of the dataset** is used, focusing on the **news and non-news classification task**. The dataset includes metadata such as **publication date, media name, geographic location, and source (Google or Twitter)**.

Preprocessing Steps The raw dataset includes millions of URLs, requiring extensive filtering and cleaning:

1. **Dereferencing & Normalization**
    - URLs were resolved to their final representations to eliminate redirects.

    - Lowercasing and removal of trailing slashes were performed for consistency.

2. **Filtering Non-News URLs**
    - URLs not belonging to a known **news media domain** were removed.
    - URLs with a **path depth of 0** (e.g., homepages) were discarded.
    - URLs with a **path depth of 3+** were retained, as these are more likely to be actual articles.
    - URLs with a **path depth ¡3** were kept only if they contained common word-boundary separators (-, _, .), indicating structured article URLs.

3. **Text Extraction Titles** and **meta descriptions** were extracted, as these match the format of search result snippets in Google.

4. **Binary Labeling for Training**
    - Entries labeled **"is_news_article" = true** were categorized as **news**.
    - Entries labeled **"is_news_article" = false** were categorized as **non-news**.
    - The final dataset is a **binary classification dataset** with labels:
        - **1 (News Article)**
        - **0 (Non-News)**

Dataset Statistics A summary of the dataset after preprocessing:

Table 1: Number of News Article URLs Collected and Filtered from Google

Media Type	Collected	Filtered
Newspapers	4,992,262	2,367,322
Radio	1,069,333	202,668
TV	888,243	523,613
Broadcast	837,599	470,223
Total	**7,787,437**	**3,563,826**

This dataset forms the foundation for training a **binary classifier** to automatically determine whether a **Google search result** is a **news article or not**. A **balanced subset** will be used for training to avoid bias toward either category.

4 Related Work

The problem of clickbait detection has been widely studied in the context of Natural Language Processing (NLP) and machine learning. Several approaches have been proposed to identify clickbait headlines, leveraging various techniques such as traditional machine learning classifiers, deep learning models, and semantic embeddings.

1. Clickbait Post Detection using NLP for Sustainable Content

In their work, (Ganapati Raju N. V., 2023) focus on building a system to detect clickbait posts on media platforms using NLP and machine learning techniques. They apply pre-processing techniques such as tokenization, lemmatization, and stemming to extract essential features from headlines. These features are then used to train supervised classifiers like logistic regression and random forests. The evaluation of the model involves common metrics, including accuracy, precision, recall, and F1 score. The proposed system aims to help users identify clickbait and thereby reduce exposure to potentially harmful content. This approach highlights the importance of feature extraction in clickbait detection but primarily focuses on traditional classifiers rather than newer, more complex models.

2. Clickbait Classification and Spoiling Using Natural Language Processing

(Adhitya Thirumala, 2023) approach clickbait detection from two distinct angles: classification and content "spoiling." In their first task, they classify clickbait into three categories using binary classifiers. In the second task, they explore using question-answering models and large language models (LLMs) to "spoil" the clickbait, i.e., reveal the misleading or exaggerated claim in the article body. Their results suggest that while their classification models performed better than existing baselines, their spoiler generation model did not outperform traditional extractive approaches. This study introduces an interesting dual-task framework but highlights the challenge of generating accurate spoilers, which complicates the detection process.

3. Identification of Clickbait News Articles using SBERT and Correlation Matrix

(Supriya and Kumar, 2023) propose a novel method to identify clickbait headlines by exploiting the dissimilarity between headlines and the body of the article. They use the Sentence-BERT (SBERT) model to extract features from both the headline and the associated article paragraphs. A correlation matrix is then used to select relevant sentences from the paragraph for classification. Their results show that a support vector machine (SVM) classifier with these concatenated embeddings achieves a performance of 0.84 accuracy, outperforming existing state-of-the-art models. This work introduces the idea of analyzing the relationship between headlines and article content, which provides valuable insight into how misleading headlines might diverge from the underlying article, setting it apart from more traditional feature-based approaches.

4. 3DLNews: A Three-decade Dataset of US Local News Articles

(Ariyarathne and Nwala, 2024) introduce the 3DLNews dataset, a comprehensive resource for studying local news in the United States over nearly three decades (1996-2024). The dataset consists of almost 1 million URLs collected from more than 14,000 local media outlets, including newspapers, TV stations, and radio stations, spanning all 50 states. The URLs were scraped from Google and Twitter search results, specifically targeting news content by using carefully constructed search queries and multi-step filtering techniques. This process helped remove non-news articles and ensured that the dataset primarily consists of relevant news content.

For my project, 3DLNews is invaluable as it provides a large-scale collection of US local news articles, making it an ideal foundation for training a binary classifier to distinguish between news and non-news Google search results. Its diverse media sources and comprehensive temporal coverage make it particularly well-suited for analyzing the nature of local news over time and improving classification accuracy in identifying news articles. Additionally, the dataset's multi-step filtering process and inclusion of rich metadata offer a robust resource for fine-tuning models and evaluating their performance in real-world scenarios.

5. News Classification Based on Headlines

(Rana et al., 2014) explores the use of text mining techniques to classify news based on headlines. The authors focus on processing and analyzing short news snippets using methods such as tokenization, stop-word removal, and stemming. They discuss several classification algorithms, including K-Nearest Neighbors (KNN), Naive Bayes, Support Vector Machines (SVM), and Decision Trees, evaluating them on their ability to classify headlines accurately. Their approach emphasizes the challenge of effectively capturing the essence of a news story through limited information (the headline alone). While their methods provide a solid foundation for headline-based classification, they focus solely on classifying headlines without distinguishing between news and non-news content. In contrast, my approach is a binary classifier that categorizes search result headlines as either news or non-news articles, aiming to filter out news content from search engine results to provide a more balanced browsing experience.

5 5. Methodology

Data Preprocessing

1. **Text Cleaning**: The text from Google search results (including titles and descriptions) will be cleaned by removing unwanted characters like HTML tags, special symbols, punctuation, and any numbers. This ensures the dataset contains only the relevant text necessary for the classification task.

2. **Tokenization**: After cleaning, the text will be split into individual tokens (words or phrases). This step breaks the text into smaller chunks that the machine learning model can analyze individually.

3. **Stopword Removal**: Common stopwords (e.g., ”the,” ”and,” ”is”) that do not contribute significantly to the meaning of the text will be removed. This step helps reduce the amount of noise in the dataset, focusing on more meaningful words.

4. **Lemmatization**: Each word in the text will be lemmatized, meaning it will be reduced to its base form. For example, ”running” will be converted to ”run,” helping standardize the text and reduce redundancy.

5. **Feature Extraction**: To prepare the text for machine learning, the tokens will be converted into numerical features using methods like **TF-IDF (Term Frequency-Inverse Document Frequency)** or potentially **word embeddings** in future iterations. These techniques capture the significance of words in relation to their frequency in a document compared to the entire dataset.

Model Training (Outline)

1. **Data Splitting**: The dataset will be split into training and testing sets. Typically, I will use an **80/20** split, where 80% of the data will be used to train the model, and 20% will be used for testing. This ensures a representative distribution of the data for both training and evaluation.

2. **Training Process**: I will train the classifier using the preprocessed data. During training, the model will learn to associate features (words or tokens) from the search result text with their corresponding labels (news or non-news).

5. Evaluation and Results

5.0.1 5.1 Evaluation Setup

The evaluation will be conducted through a structured process involving data splitting, baseline comparisons, and performance measurement using key classification metrics.

5.0.2 5.2 Baseline Comparisons

To measure the effectiveness of my classifier, I will compare it against two baselines:

1. **Random Baseline**: A naive model that assigns labels randomly with equal probability (50% for news, 50% for non-news). This represents a lower bound for performance.

2. **Majority Class Baseline**: A simple model that always predicts the majority class in the dataset. Given that news articles are more frequent in the dataset, this baseline will predict all inputs as "news." This will provide a meaningful reference point for evaluating the advantage of my trained classifier.

5.0.3 5.3 Experimental Setup

I will split the dataset into **80% training** and **20% testing** to ensure generalization. The test set will be kept unseen during training.

5.0.4 5.4 Performance Results

After training, I will evaluate my classifier's performance against the baselines using the test set. The results will be summarized in the following table:

Model	Accuracy	Precision	Recall	F1-Score
Random Baseline	TBD	TBD	TBD	TBD
Majority Baseline	TBD	TBD	TBD	TBD
My Classifier	**TBD**	**TBD**	**TBD**	**TBD**

Table 2: Performance comparison of different models.

5.0.5 5.5 Discussion of Results

I anticipate that my classifier will **significantly outperform both baselines**, achieving a higher accuracy and F1-score. The evaluation will help determine how well my model distinguishes between news and non-news headlines.

Error Analysis

- **False Positives**: Some non-news results may be misclassified as news due to similar phrasing.

- **False Negatives**: Certain legitimate news headlines may be incorrectly marked as non-news, likely due to ambiguous wording.

- **Potential Improvements**: Future iterations could incorporate deep learning models (e.g., transformers like BERT) to improve contextual understanding.

5.0.6 5.6 Conclusion

The evaluation will demonstrate the effectiveness of my NLP-based classifier in accurately identifying news content in search results. The model's performance over baseline methods will validate its potential for deployment in the Chrome extension, which will filter news results effectively to mitigate doomscrolling behavior. Future work will refine the classifier with additional data sources and explore alternative model architectures for improved accuracy.

6 Discussion

The discussion section is where you start to unpack the results for the reader to help them understand what was learned. You may not have many results at the moment (but you should have some!) so you can discuss here what has gone wrong and right in your current setup. For example, maybe you realized you needed more data, or maybe you realized that the SVD was not helping your analysis. The discussion should point the way to what work will be done next.

You can leave this section blank.

6.0.1 7. Work Plan

Week 1: Data Preprocessing and Initial Setup

- **Objective**: Complete text cleaning, tokenization, stopword removal, and lemmatization.
- **Tasks**:
  - Tokenize the text (split into individual words or phrases).
  - Remove common stopwords that don't contribute much to the meaning.
  - Implement lemmatization (reduce words to their base form).
  - Review and verify the preprocessing pipeline with sample data.
  - Prepare the preprocessed data for feature extraction.

Week 2: Feature Extraction and Data Splitting

- **Objective**: Implement feature extraction and split the data into training and testing sets.
- **Tasks**:
  - Set up feature extraction methods such as **TF-IDF** to convert text into numerical features.
  - Ensure the data is ready for model training by preparing feature matrices.
  - Split the dataset into training (80%) and testing (20%) sets.
  - Test the splitting process to ensure a good representative sample for training and testing.

Week 3: Model Training

- **Objective**: Begin model training using preprocessed data and chosen classifier.
- **Tasks**:
  - Select the first classification model(s) to train (e.g., logistic regression, SVM).
  - Train the classifier using the preprocessed data and the extracted features.
  - Implement the training process, ensuring proper handling of the training data.
  - Evaluate initial performance on the test set (though full evaluation metrics will be done later).

Week 4: Evaluation and Refinement

- **Objective**: Evaluate model performance, compare against baselines, and refine the system.
- **Tasks**:
  - Evaluate the classifier's performance using the test set, comparing it against the **Random Baseline** and **Majority Class Baseline**.
  - Record performance metrics such as accuracy, precision, recall, and F1-score.
  - Refine the model and preprocessing steps based on results.
  - Prepare a summary of results for final documentation, including insights from performance comparisons.

7 Acknowledgements

I kinda realized way late into the project that if I made a clickbait classifier, well, how was I going to feed it only news search results. So basically, I was stumped. I had to rewrite my whole project on just being a news article classifier for all search results. The big problem, was I could not find any datasets of non-news search results. If I did not stumble upon Gangani Ariyarathne and Alexander C. Nwala's paper ”3DLNews: A Three-decade Dataset of US Local News Articles” and found their dataset they created, I would have been cooked. Not only is it exactly what I was looking for, there is so much data to train on.

References

Elisa Ferracane Adhitya Thirumala. 2023. Clickbait classification and spoiling using natural language processing. In *Proceedings of the NLP Conference*. Conference Publisher, pages 12–18.

Gangani Ariyarathne and Alexander C. Nwala. 2024. 3dlnews: A three-decade dataset of us local news articles. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*. ACM, New York, NY, USA, pages 1–5. https://doi.org/10.1145/3627673.3679165.

Nikhil Nyalakanti Premsai Kambampati Yeshwanth Kanthali Shivam Pandey K. Maithili Ganapati Raju N. V. 2023. Clickbait post detection using nlp for sustainable content. *Journal of Online Media* 5:23–35.

Mazhar Iqbal Rana, Shehzad Khalid, and Muhammad Usman Akbar. 2014. News classification based on their headlines: A review. In *Proceedings of the 17th IEEE International Multi Topic Conference (INMIC 2014)*. IEEE, pages 1–6. `https://doi.org/10.1109/INMIC.2014.7097339`.

Jyoti Prakash Singh Supriya and Gunjan Kumar. 2023. Identification of clickbait news articles using sbert and correlation matrix. *Journal of NLP and Media Studies* 13(153):1–12. Published: 25 November 2023.