News search engine

Faculty of Engineering of the University of Porto, Portugal

Vítor Cavaleiro
up202004724@edu.fe.up.pt

Rodrigo Figueiredo
up202005216@edu.fc.up.pt

Diogo Fonte
up202004175@edu.fe.up.pt

Sofia Rodrigo
up202301429@edu.fe.up.pt

ABSTRACT

This report intends to document the process of collection, preparation and processing of a specific unstructured news dataset, for the development of a search system. The sourced data is obtained from trustworthy sources such as Components and ML Resources.

The data preparation phase was conducted throughout a pipeline, structuring and cleaning the information from the CSV files. Subsequently, a detailed analysis was conducted, leveraging graphical representations to enhance the understanding of the dataset's inherent patterns. This first part was finalized with a statement of the prospective search tasks that were going to be worked with the following phases.

The subsequent step in this process involved indexing and filtering the information to facilitate efficient retrieval through queries in Solr. Each search task has been then thoroughly evaluated using precision-recall curves, comparing two different retrieval setups.

The final part of the report is focused on improving the previous steps, where the principal goals are set on semantic search and additional data processing, providing a new and more accurate evaluation, which concludes a complete and profound study of the provided dataset.

KEYWORDS

Dataset, data, news, pipeline, indexing, query

1 INTRODUCTION

In today's rapidly evolving information landscape, the ability to access and extract relevant information from the vast ocean of digital data has become paramount. The constant flow of information, generated by a multitude of sources, from social media platforms and news websites to academic journals and research repositories, underscores the pressing need for robust information retrieval tools.

This article introduces a cutting-edge news search engine, specifically designed to meet the challenges of information retrieval. By combining a data preparation and cleaning methodology, a further understanding of it through graphic analyzers and an indexing and retrieval tool such as Solr, this search engine promises to deliver a tailored and insightful user experience, making it an indispensable tool for individuals and researchers.

The report is divided in three principal sections. The first one gathers the data preparation process, which includes how the datasets were selected, what they contain and the explanation of the pipeline carried out to obtain data prepared for its analyzing. This first milestone lays the groundwork for a simpler information needs retrieval.

The second and third section involve information indexing using Json schemas and retrieval mechanisims, as well as a profound evaluation of the different proposed queries and their results.

2 DATA PREPARATION

Data preparation is the process of preparing raw data so that it is suitable for further processing and analysis. Key steps include collecting, cleaning, and labeling raw data into a form suitable for exploring and visualizing the data. Using specialized data preparation tools is important to optimize this process [4].

2.1 Data collection

Data collection is the first important step to produce a reliable search system. It involves choosing and exploring the content of the dataset, as well as exploring its quality.

2.1.1 Dataset choice. Before selecting the dataset, it was essential to reach a consensus on the theme. After deliberating the pros and cons of various topics, we concluded that news would be a highly suitable subject for an information search system, given the possibility of finding multiple datasets meeting our desired quality criteria.

Once the theme was decided, we initiated the search for datasets that met the necessary requirements, particularly concerning data size and quality. The chosen dataset encompassed 204,000 articles from 18 American publications, collected from Components [6]. However, in order to incorporate data from another source and introduce some complexity, we opted for an additional dataset, this time consisting of 2,555 documents sourced from the BBC website, collected from ML Resources [5].

The first dataset has an MIT license, while the second one is open for academic purposes.

2.1.2 Dataset content. The first dataset contains 204,135 articles from 18 American publications. Includes date, title, publication, article text, publication name, year, month, and URL (for some). Articles mostly unevenly span from 2013 to early 2018, with a smattering pre-2013.

The second one consists of 2225 documents from the BBC news website corresponding to stories in five topical areas from 2004-2005, divided in 5 categories: business, entertainment, politics, sport and tech, and includes raw text data, from where we extracted the needed attributes.

To merge the datasets, we standardized both of them into the same format, ensuring the following attributes:

- `title`: Title of the article

- `author`: Author of the article
- `date`: Date of publication in format YYYY-MM-DD
- `content`: Textual content of the article
- `publisher`: Publisher of the article
- `source`: From where the article was extracted
- `category`: Describes the topic or subject matter of the article.
- `url`: Source webpage of the article

2.1.3 Data quality. The data was sourced from Components and ML Resources, two well-known and reputable platforms. It also has a large volume, is diverse and well documented.

There were some null values and missing data for some fields, but everything was handled the way it should and the data was standardized into a format we specified.

2.2 Data preparation

Data preparation is the principal phase of this first milestone. Data ingestion and data cleaning ease the management of indexing tools that will be concreted afterwards.

2.2.1 Data ingestion. It is important to clean, reduce or filter the data before its analysis and processing.

Firstly, the first dataset that comes in a .db file is exported to a JSON file using SQlite studio. Then, a python script converts through simple functions and dataframes the JSON file to a CSV file.

A similar process takes place with the BBC database. In this case, the data is collected from BBC and BBCSports articles written in various txt files, separated in 5 folders according to the categories. Afterwards, another python script combines them into a CSV file.

Finally, we bind both CSV files into one, which will be the final format of the data for further usage.

2.2.2 Data cleaning. With the help of the pandas library of python, we manipulate the CSV file throughout dataframes.

For the first dataset we start by removing irrelevant parameters such as the first column, the *id*, the *digital* and the *section* in order to transform our data into a more useful information source. Moreover, we eliminate redundant columns such as *year* and *month* since it is already detailed in the *date* parameter. Another important change was to rename some columns to more descriptive words for the actual meaning of the columns. *Publication* parameter was changed to *publisher*, *category* to *source*, and *section* to *category*.

The second dataset was exported as raw text files divided into categories, so we extracted the relevant attributes from the text and merged all the categories into one csv file, leaving it in the same format as the first dataset, so we could merge both datasets.

After merging both datasets, we had one csv file with all the data and there were some things we needed to do in order to clean the data. The first thing was removing all duplicates, as well as replacing empty strings with NaN. We also decided to remove every row with missing title or content as that is valuable information that needs to be present. There was also a problem with the author names, as a lot of them were coming with "\n" before and after their names, so we removed that as well.

2.3 Data analysis

At the start of this stage we decided to create a new column refering to the keyphrases of each article. We used the rakt-nltk library for that.

In order to extract more information about the chosen dataset and with the help of python libraries such as matplotlib.pyplot or seaborn, we have developed different data analysis plots to represent some parameters. Each of the following tries to obtain some characterization of the data to help us understand the information we are handling:

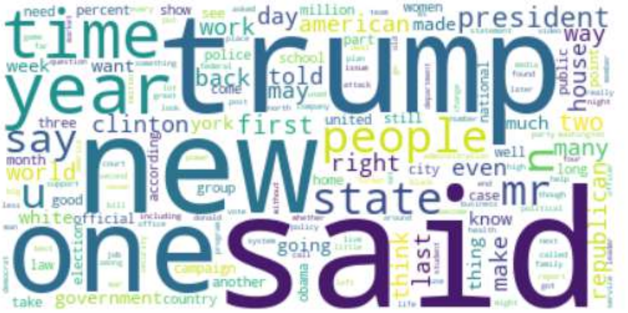

Figure 1: Articles wordcloud

Figure 1 represents a wordcloud, which is generated from article content using keyword extraction to identify the most common words or phrases. We could conclude that our data is mostly dedicated to political issues.

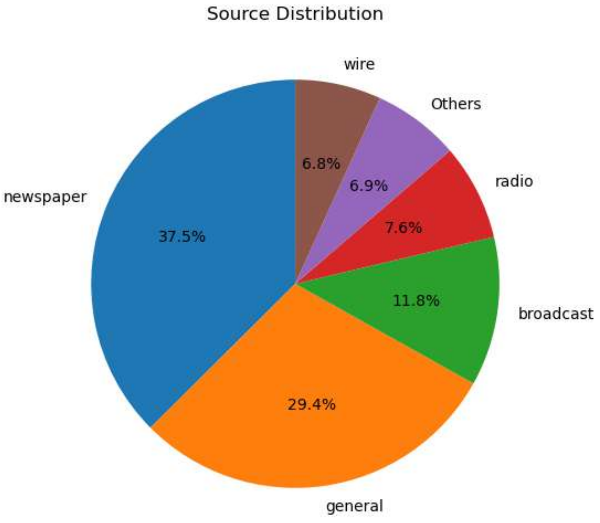

Figure 2: Sources pie chart

Figure 2 represents a pie chart, which visualizes the proportion of articles from each publication source within the dataset. The plot clearly emphasizes the newspaper as the most used source of news release. The general category includes articles that were published in several platforms (newspaper, website, etc).

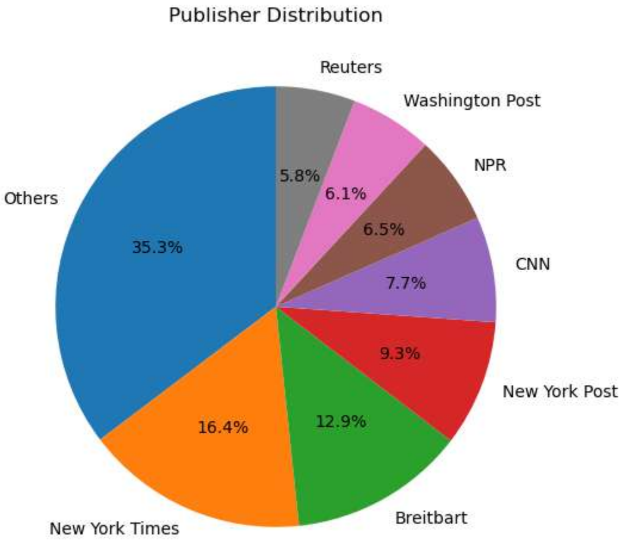

Figure 3: Publishers pie chart

Figure 3 represents another pie chart, which shows the percentage of articles from each publisher within the dataset. The most common ones are The New York Times, Breitbart and The New York Post.

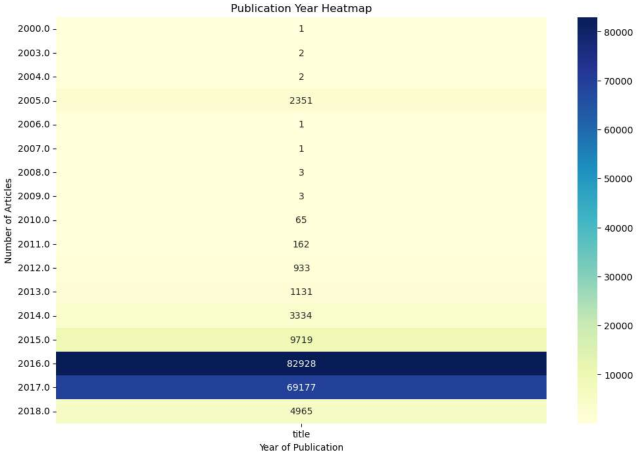

Figure 4: Heatmap of articles per year

Figure 4 represents a heatmap, a multivariable plot generated to visualize the number of articles published per year. 2016 and 2017 are clearly the years that cover the most amount of written articles.

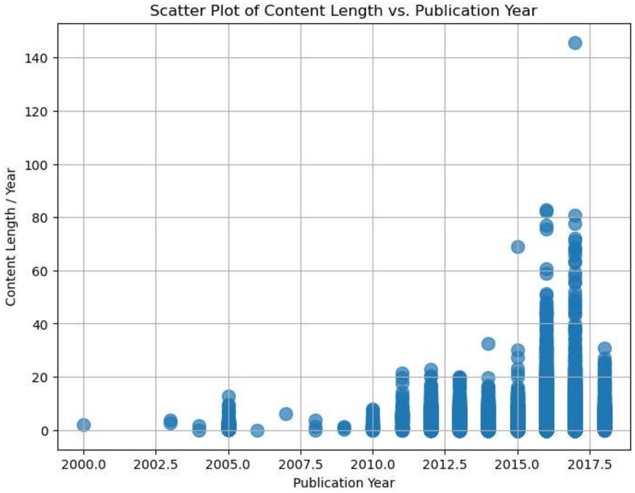

Figure 5: Scatter plot of content length per year

Figure 5 describes a scatter plot, which explores the correlation between the length of the articles and the publication date. It exposes 2015, 2016 and 2017 as the years with longer articles and also with the most amount, just as shown in the previous plot.

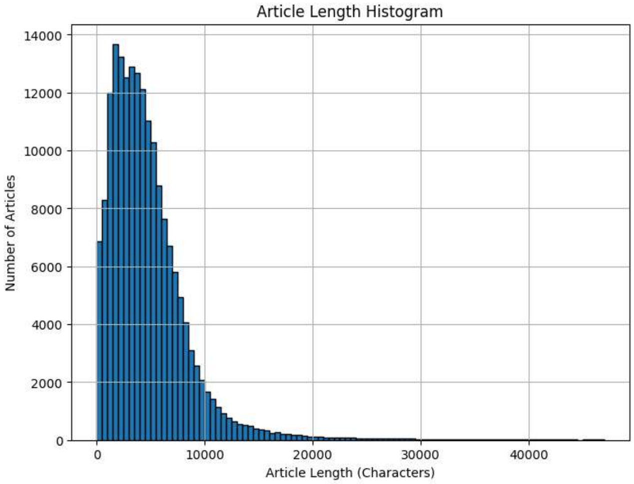

Figure 6: Article Length Histogram

Figure 6 represents a histogram of the articles length, and it can be observed that it has a right-skewed distribution with the majority of articles having a length of less than 10000 characters.

2.4 Domain data model

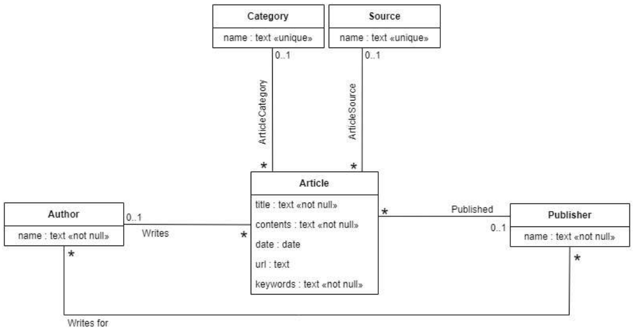

Figure 7: Domain data model [2]

The data model comprises four primary classes that represent key entities within the domain: Article, Author, Publisher, and Category. These classes encapsulate essential aspects of the domain and are interrelated to facilitate effective data organization.

The Article class serves as the central entity and embodies written content within the domain. It possesses several attributes that define articles, including a title, content, publication date, URL, and keywords. Articles have associations with the Author, Publisher, and Category classes, enabling the linkage of articles to authors, publishers, and categories, as appropriate.

Authors represent individuals responsible for creating articles and the sole attribute for authors is their name. An Author class is associated with the Publisher class, illustrating the collaborative relationship between authors and publishing entities.

Publishers signify the organizations or entities responsible for publishing articles, and the only attribute is the name of the publisher. This class maintains a connection with the Author class, demonstrating the affiliation between authors and their respective publishers.

Categories are used for classifying articles based on various themes or topics, and their only attribute is also the category name. Categories are directly connected to the Article class, enabling the categorization of articles into relevant topics.

Sources are connected to the articles as they specify where a given article comes from. Their only attribute is the source name.

2.5 Data processing pipeline

The whole process of this first milestone is developed in Jupyter Notebook.

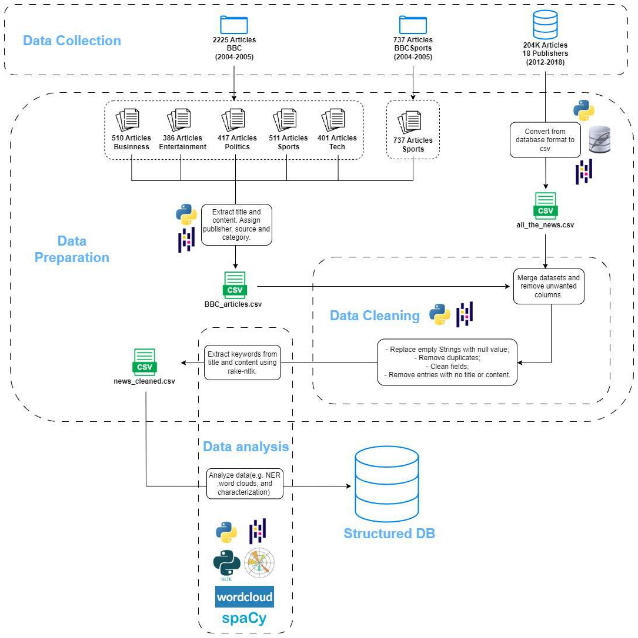

Figure 8: Data flow diagram of the pipeline [2]

2.6 Prospective search tasks

The search system that is going to be developed in the next milestones requires a previous task of information needs search. These are some examples:

- Find news articles where Trump spoke on the immigration crisis
- Find news about LeBron's good performances in games his team won
- Find articles related to homicides investigated by the FBI in 2017
- Find news articles regarding the conflicts between republicans and democrats about gun ownership

3 INFORMATION RETRIEVAL

Information retrieval is the activity of obtaining information system resources throughout content-based indexing, that are relevant to an information need from a collection of those resources. [7]

3.1 Tools

For this milestone, the use of Solr [1] and Docker containers [3] was essential for building our information retrieval system. Solr provides tools for indexing and querying long textual fields. It has also powerful search capabilities, is highly flexible and can index a wide range of document types. This software incorporates as well advanced text analysis and tokenization features. The utilization of Docker containers ensures consistency, scalability, and efficiency across various environments.

3.2 Indexing

To know which fields we needed to index, we focused on the domain data model and the prospective search tasks subsections. In addition, we have created a schema that contains different field types associated with each field depending on its characteristics. Table 1 describes this correspondance. The creation of a new field *Code* was necessary for the retrieval part.

Field	Field type
Code	Code id
Title	Synonym text
Author	Char text
Date	Date
Content	Content text
Publisher	Char text
Category	Synonym text

Table 1: Description of field type correspondance

Every field type has a Solr class, an index analyzer and a query analyzer. Each analyzer has different filters and a tokenizer, which in our case will be always the Standard Tokenizer from Solr.

The following items describe the several field types created and the filters that are applied to each one of them.

- **Char text**
  - ASCII Folding Filter: this filter converts alphabetic, numeric, and symbolic Unicode characters which are not in the Basic Latin Unicode block (the first 127 ASCII characters) to their ASCII equivalents, if one exists.
  - Lower Case Filter: converts any uppercase letters in a token to the equivalent lowercase token. All other characters are left unchanged.
  - Mapping Char Filter: this char filter is used for changing one string to another (for example, for normalizing á to a)
- **Synonym text**
  - ASCII Folding Filter
  - Lower Case Filter
  - Synonym Graph Filter: this filter maps single or multitoken synonyms, producing a fully correct graph output.
  - Porter Stem Filter: this filter applies the Porter Stemming Algorithm for English. The results are similar to using the Snowball Porter Stemmer with the language="English" argument.
- **Content text**
  - ASCII Folding Filter
  - Lower Case Filter
  - Synonym Graph Filter
  - Porter Stem Filter
  - English Possessive Filter: this filter removes singular possessives (trailing 's) from words.
  - Hyphenated Words Filter: this filter reconstructs hyphenated words that have been tokenized as two tokens because of a line break or other intervening whitespace in the field test. If a token ends with a hyphen, it is joined with the following token and the hyphen is discarded.
  - Stop Filter: this filter discards or stops analysis of, tokens that are on the given stop words list. A standard stop words list is included in the Solr conf directory, named stopwords.txt, which is appropriate for typical English language text.
- **Date**
  No filters are applied since it is a date and Solr has a class named TrieDateField for it.

3.3 Retrieval and Evaluation

In this subsection, we describe four different information needs built from the prospective search tasks. We will detail each one of them, as well as the query that we used, the boosts that were applied and the results we obtained.

Since the number of retrieved documents was high, we analyzed some of them for each query to determine which ones were more relevant. Using the given spreadsheet, we wrote their code in qrels.txt files.

We used several systems to test the information needs: firstly, we just used the indexed fields described in the previous subsection and afterwards we explored the following:

- **Fields boosts**: enhance the relevance of specific fields.
- **Term boosts**: elevate the importance of certain words.
- **Proximity searches**: emphasizes the closeness of terms.
- **Wildcards/Fuzziness**: allows for flexibility in matching variations of a term.

To evaluate the queries' results we take into account two main concepts: *Precision*, which is the accuracy of retrieved relevant documents, and *Recall*, that quantifies the proportion of relevant documents retrieved. We also consider the following evaluation metrics:

- **Average precision (AP)**: value obtained for the set of documents existing after each relevant document is retrieved.
- **Precision at 10 (P@10)**: value obtained from the number of recommended documents that are relevant divided by the number of recommended documents.

QUERIES

The following comparisons will be based on two different retrieval setups: the first one only uses a simple schema without filters and the second one uses our schema and several boosts that we described previously.

- **Q1**: Find news articles where Trump spoke on the immigration crisis
  - Arguments and boosts

Tag	Value
q	Trump immigration
qf	title^2.5 content
fl	all fields
bq	title:Trump^3 title:\"Trump speak 2\" 5^2.5 content:Trump^2.5 content:\"Trump speak 2\" 5^2

Table 2: Description Q1 arguments

  - Precision metrics and P-R curves

Metric	Simple value	Value with boosts
AP	0.53	0.71
P@10	0.5	0.8

Table 3: Precision metrics for Q1

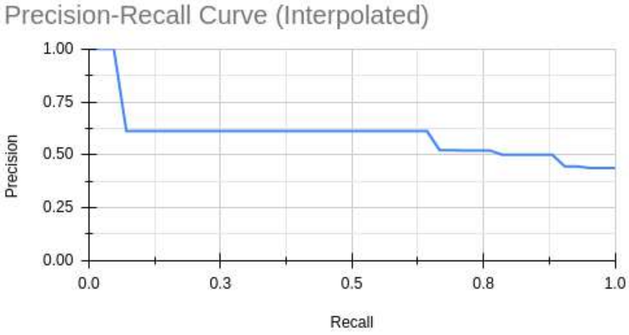

Figure 9: P-R curve for Q1 with simple schema

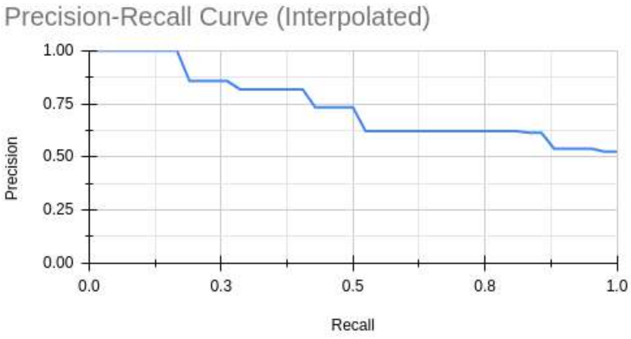

Figure 10: P-R curve for Q1 with boosts

  - Discussion
    As we can see in table 3, the average precision and the precision at 10 is higher for the second setup, meaning there are more relevant documents on the first positions for the second setup. Through figures 9 and 10 we can check that the second setup presents more relevant documents on the first results as the graph keeps constant at precision 1 for longer, and it also doesn't decrease that fast comparing to the first setup.

- **Q2**: Find news about LeBron's good performances in games his team won
  - Arguments and boosts

Tag	Value
q	Lebron good game win
qf	title content^1.5
fl	all fields
bq	title:\"Lebron win\" 5^2 content:\"Lebron bad\" 5^0.1 content:\"Lebron fail 2\" 5^0.1 content:\"Lebron points\" 5^2 content:\"Lebron win 2\" 5^2 content:scored 2^2
pf	title content^3

Table 4: Description Q2 arguments

  - Precision metrics and P-R curves

Metric	Simple value	Value with boosts
AP	0.7	0.89
P@10	0.4	0.6

Table 5: Precision metrics for Q2

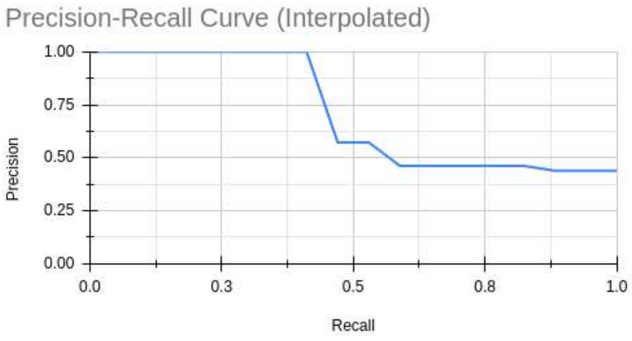

Figure 11: P-R curve for Q2 with simple schema

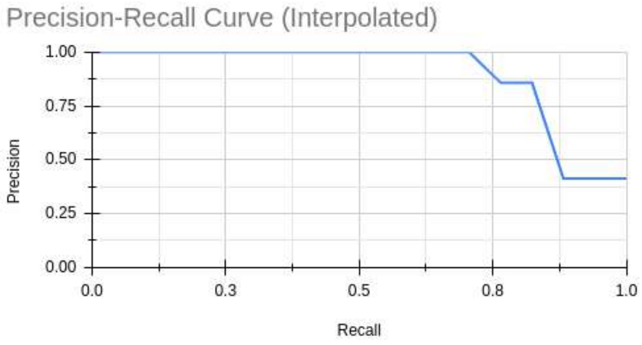

Figure 12: P-R curve for Q2 with boosts

- Discussion
  As we observe in table 5 and figures 11 and 12, results are improved less than in query 1, but the second setup keeps the graph constant at precision 1 longer, meaning it has more relevant results on the top places. Precisions at 10 aren't that good but it isn't that important as we have few total relevant documents, and most of them are presented on the top.

- **Q3**: Find articles related to homicides investigated by the FBI in 2017
  - Arguments and boosts

Tag	Value
q	homicide FBI
qf	title^2 content
fl	all fields
fq	date:[2017-01-01T00:00:00Z TO 2017-12-31T23:59:59Z]
bq	title:homicides^2.0

Table 6: Description Q3 arguments

  - Precision metrics and P-R curves

Metric	Simple value	Value with boosts
AP	0.62	0.81
P@5	0.2	0.6

Table 7: Precision metrics for Q3

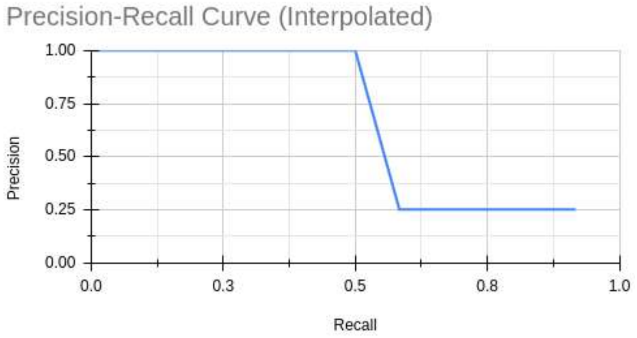

Figure 13: P-R curve for Q3 with simple schema

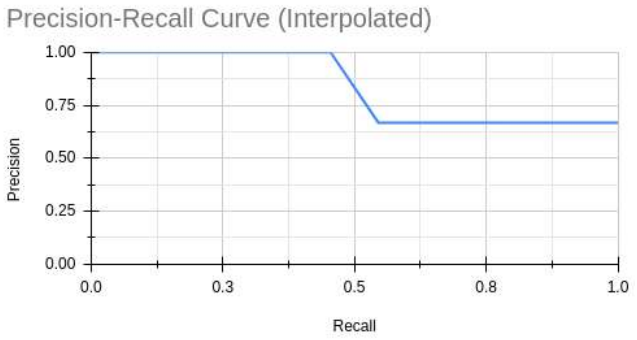

Figure 14: P-R curve for Q3 with boosts

  - Discussion
    For this query, as there are only 12 relevant documents in total, we chose to have the precision at 5 metric. From figures 13 and 14 we can see that the first positions have relevant documents which is a good sign, but the second setup shows more relevant documents at the last positions, that's why the graph doesn't decrease as much as the first setup. From table 7, we can check that the precisions are good, with the second setup presenting a better result for the reasons mentioned above. The precision at 5 isn't that good for the first setup as the few relevant documents presented are all gathered on the first positions, but then the following documents are mostly not relevant, making the precision go down a lot.

- **Q4**: Find news articles regarding the conflicts between republicans and democrats about gun ownership
  - Arguments and boosts

Tag	Value
q	Republicans Democrats "gun ownership"
qf	title^2.5 content
fl	all fields
bq	title:gun^1.5 content:gun^1.5 content:conflict^1.5 content:Republican 2^2 content:Democrat 2^2 content:congress^2

Table 8: Description Q4 arguments

  - Precision metrics and P-R curves

Metric	Simple value	Value with boosts
AP	0.83	0.87
P@10	0.7	0.8

Table 9: Precision metrics for Q4

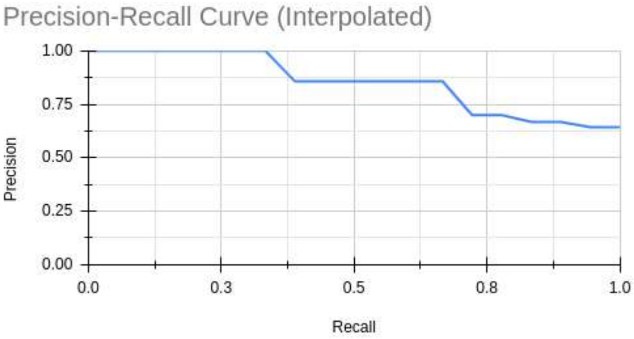

Figure 15: P-R curve for Q4 with simple schema

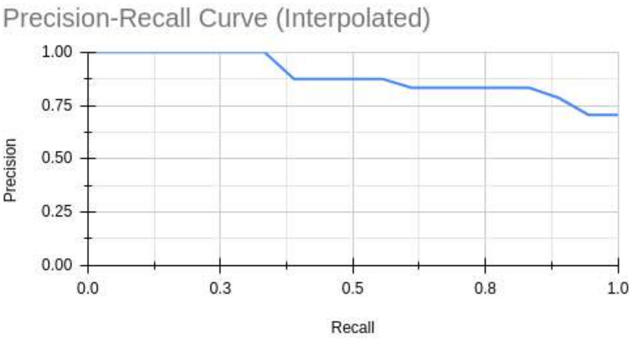

Figure 16: P-R curve for Q4 with boosts

  - Discussion
    As we can see in table 9 and figures 15 and 16, boosts slow down the precision decrease but do not improve final retrieval results. For this query the results were very similar as we can see in table 9, with the second setup being better by just a bit. From figures 15 and 16 we can confirm that they are very similar, presenting relevant documents on the first positions, but the second setup doesn't decrease as much, meaning the total amount of relevant documents presented is higher than the first one.

4 SEARCH SYSTEM

For this last part of the project, we did not implement a user interface and used Solr's one. In the following section, we are going to describe some improvements in our search system.

4.1 Improvements

We have made four relevant improvements to our previous search system.

4.1.1 P-R curve formula correction. In the previous part we were calculating the precision and recall incorrectly since we were dividing every step by the relevant results found up to that point, instead of dividing by all relevant results.

4.1.2 Term and field boosts. For the previous information retrieval we were using different field boosts for each query, which was going against the principle of having just one search system, so for this evaluation all queries were evaluated using the same boosts.

4.1.3 Semantic search. In order to improve our search system, we analyzed several ideas and selected those most aligned with the context of our dataset. The principal idea we chose to explore further was the implementation of a semantic search approach.

Semantic Search refers to the ability of a system to understand the context and intent behind a user's query, rather than simply relying on keyword matching. This method, often referred to as 'dense vector search,' holds the promise of comprehending the nuances of user intent and context, thereby enabling our system to identify documents that share semantic relations with the query rather than relying solely on keyword matching. This strategic shift towards semantic search aligns with our goal of enhancing search accuracy and relevance within our dataset.

Using the course tutorial and the Sentence Transformer package of python, along with an extended article about Semantic search [9], we added a new field and field type to our schema, in order to update the dataframe's pipeline with embeddings. The field type is also Solr's class DenseVectorField with a vector dimension of 384 and using the same knnAlgorithm as in the given tutorial, but in our case we add new parameters.

4.1.4 Additional data processing. In addition to implementing semantic search, we've integrated an additional data processing step using 'clean-text' from PyPi. [8] This process helps handling the issue of dirty content found across the web. By leveraging 'clean-text,' we normalize the text representation of scraped data, effectively transforming corrupted inputs into clean, readable outputs. For instance, it converts encoded characters, removes unwanted elements such as special characters or irrelevant formatting, and enhances the overall text coherence. This Python package employs a suite of tools including ftfy, unidecode, and customized regular expressions, ensuring a robust and comprehensive approach to text normalization.

After a thorough evaluation of potential enhancements for our search system, we discarded the idea of adding additional information sources since our dataset is already very complete and has a considerable size. While the dataset has a substantial size, this posed practical challenges, not only complicating some data processing steps but also led to significant time constraints, making certain operations unfeasible. The new data processing step we decided to add aggravated this situation and we basically needed to decide between reducing the dataset size or forgoing the additional processing step to maintain system feasibility and efficiency. Based on this, we decided to substantially decrease the dataset size and maintain the new data processing step. In order to achieve this we made a small verification to ensure we wouldn't eliminate any important new that was related to any of our queries.

4.2 Evaluation

The two systems that are going to be compared both use semantic search, one without re-raking and one with it. The re-ranking changes the order of the results by applying field boosts.

Since we started using semantic search, it was possible to have inputs more similar to a natural search instead of just a query with specific words, so we changed all queries based on that.

Regarding the process of choosing the relevant documents for the qrels files, we noticed that the process we were following for the last information retrieval was wrong, so we made it differently this time. We retrieved one hundred documents for each query and chose the most relevant documents from there, with a limit of 20. In addition to doing the process wrong, we also calculated the recall the wrong way, resulting on graphs displaying inconsistent information, so we also fixed that for this evaluation.

- **Q1**: trump speaking on immigration crisis
(20 relevant documents)
  - Precision metrics and P-R curves

Metric	Without boosts	With boosts
AP	0.45	0.62
P@10	0.4	0.7

Table 10: Precision metrics for Q1

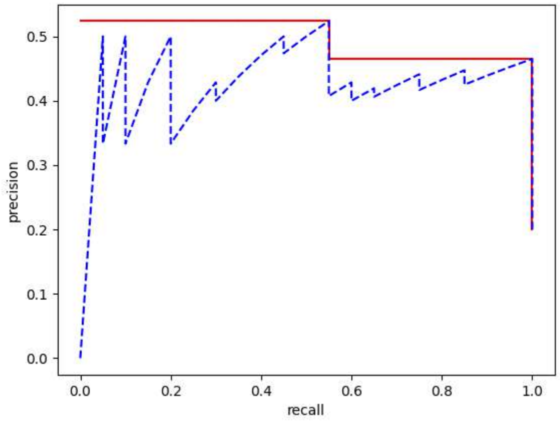

Figure 17: P-R curve for Q1 without boosts

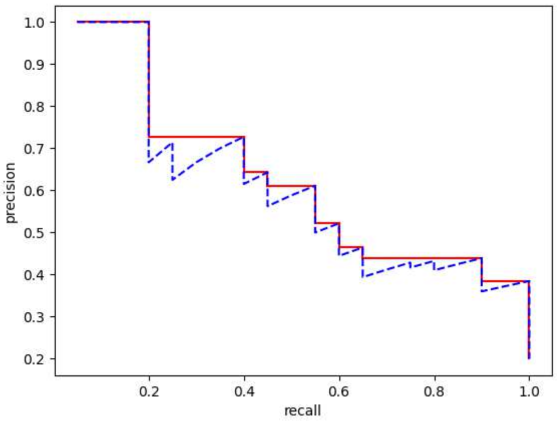

Figure 18: P-R curve for Q1 with boosts

- As we observe in figure 18, for the system with boosts all the relevant documents were found within the retrieved documents as the recall ends at 1. It is also possible to observe that the first 5 positions contain relevant documents as the precision stays at 1 till the recall is 0.2, which is a good indication. The precision also doesn't drop that fast, and the values for average precision and precision at 10, as seen in table 10, are satisfactory. On the other hand, for the system without boosts, although the precision starts at around 0.5, as seen in figure 17, it keeps almost constant till the end, while presenting all relevant documents on the retrieved ones, as well. The values for the average precision and precision at 10 are lower than the ones from the system with boosts, as seen in table 10, mainly because of the bad start, where the first retrieved document wasn't relevant. We can determine that the system with boosts performs better as it presents relevant documents in the first positions and maintains a higher accuracy for longer than the other system.

- **Q2**: lebron good performance in games he won (12 relevant documents)
  - Precision metrics and P-R curves

Metric	Without boosts	With boosts
AP	0.39	0.39
P@10	0.5	0.3

Table 11: Precision metrics for Q2

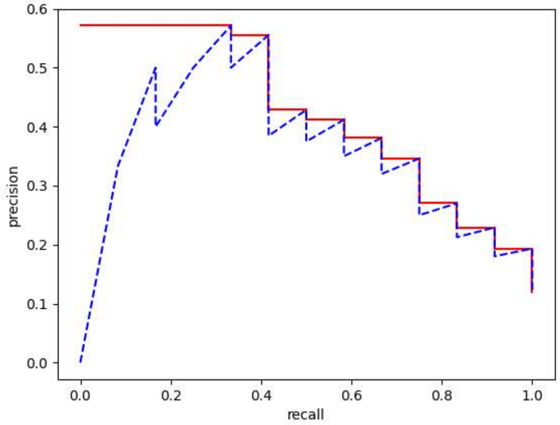

Figure 19: P-R curve for Q2 without boosts

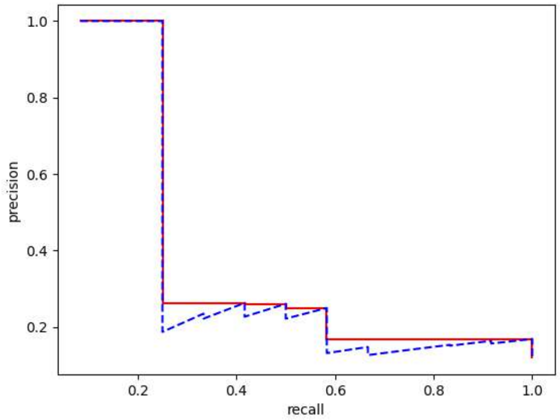

Figure 20: P-R curve for Q2 with boosts

- For this query the results depict a different picture with respect to the first one, as both systems get the same AP and the one without boosts gets a better P@10, as seen in table 11. By looking at figures 19 and 20 we can check that even though the system with boosts has relevant documents on the first positions, the accuracy has a big drop when the recall is still low, meaning that there was a big sequence of documents retrieved that weren't relevant, before finding another relevant one, which isn't ideal. For the one without boosts, the first places don't have relevant documents but the accuracy drops slower. A positive aspect for both systems is that all relevant documents were found within the retrieved documents. Unlike the first query, the system with boosts didn't perform better. The reason could be the number of relevant documents, as there are only 12, since this is a really specific query.

- **Q3**: homicide investigated by fbi in 2017
  (14 relevant documents)
  - Precision metrics and P-R curves

Metric	Without boosts	With boosts
AP	0.15	0.58
P@5	0.1	0.6

Table 12: Precision metrics for Q3

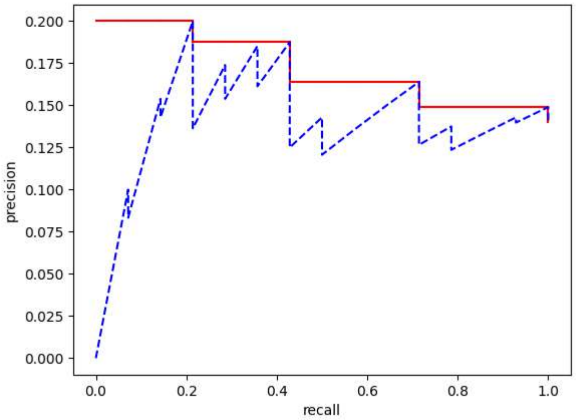

Figure 21: P-R curve for Q3 without boosts

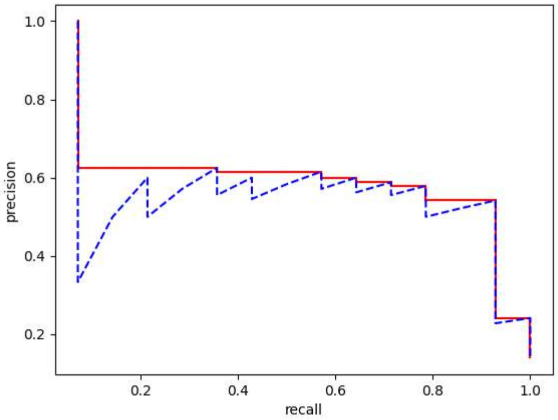

Figure 22: P-R curve for Q3 with boosts

  - In this query, as it can be seen on figures 21 and 22, there is a very big difference between the two systems. While both of them end on recall 1, which is a good signal, the accuracy for the system without boosts is very bad, with most of the top positions with non relevant documents. The re-ranking really helped on that aspect, and the first document is already relevant for the system with boosts, while it also maintains a good precision while the recall goes up, meaning it retrieves a big sequence of relevant documents. From table 12 we can confirm what its visible on the graphs with both the average precision and precision at 10 being really bad for the system without boosts and decent for the one with boosts. This confirms what we had concluded from the evaluation of the first query about the system with boosts performing better. We think that the big difference between the systems is due to the fact that there aren't many relevant documents and usually on news that are related to homicides, it is usually emphasized on the title that he crime was an homicide, so when we boost the title, we are giving more importance to those news, and placing them higher on the ranking.

- **Q4**: conflicts between republicans and democrats about gun ownership
  (20 relevant documents)
  - Precision metrics and P-R curves

Metric	Without boosts	With boosts
AP	0.25	0.27
P@10	0.3	0.3

Table 13: Precision metrics for Q4

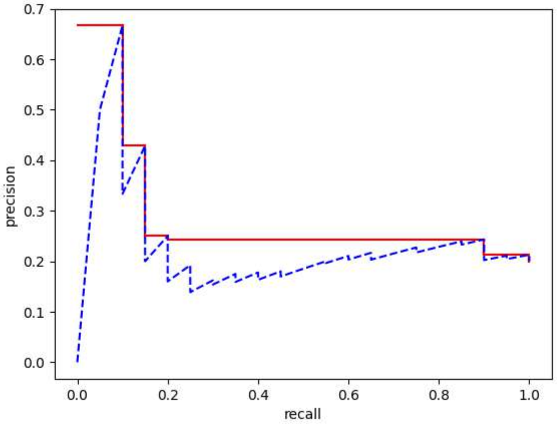

Figure 23: P-R curve for Q4 without boosts

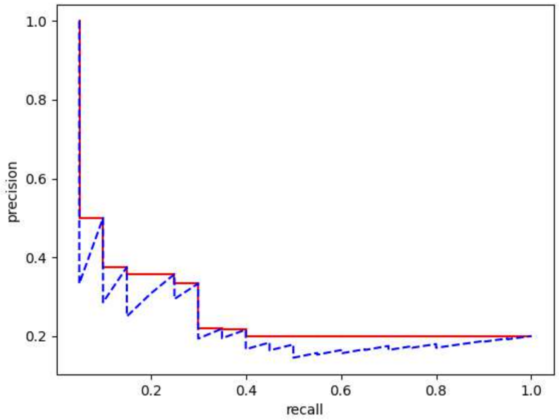

Figure 24: P-R curve for Q4 with boosts

- From table 13 we can already tell that the results for this query were bad, with low values for both the average precision and the precision at 10. Figures 23 and 14 confirm that observation, it's visible that the graph stays constant at a low accuracy for most of the time, meaning the relevant documents are on the last positions of the retrieved documents. There aren't that many differences between the two systems as both perform equally bad. The only difference is that for the system with boosts there is a relevant document at the first position, with the precision starting at 1. We believe that the bad results for this query can be due to the level of detail of the query and how picky we were when selecting the relevant documents, as we only chose the ones that talked about both republicans and democrats, and it was clear that there was some type of conflict between the two regarding the gun ownership. There were a lot of results that matched the query but weren't exactly what we wanted.

5 OVERALL SYSTEM COMPARISON

From the evaluation of all queries using both systems it was clear that the system with boosts performed generally better, getting better average precision and more relevant documents on the top positions. The semantic search that was used for both of them was really a big addition in relation to the previous information retrieval and it made the search system perform much better. In the appendix there is a "Top 10 results" section where we show the top 10 retrieved results for each query and each system in order to make possible the validation of the presented results.

6 CONCLUSIONS

All the objectives set in the first part of the project were successfully accomplished. The satisfaction of those objectives has allowed us to have a better understanding of the datasets we are working with and led to a preparation for the next stages of the project.

Regarding the second part of the project, we can consider we have made possible the prospective search tasks described in the first part so, we can conclude that the main objective has been accomplished. Apart from this, we have to be critical of the results of the evaluation that has been carried out. The values obtained in the different queries that were executed could be improved, but still, the usage of personalized boosts has helped a lot. In every single case, the boost chosen for the query has improved the result.

Considering the retrospective from the previous work, we implemented some new ideas, like semantic search (which was quite important) and we add two more steps of data processing (text cleaning and dataset resizing). We also fixed some issues regarding the evaluation previously carried out.

REFERENCES

[1] -. *Apache Solr Reference Guide.* https://solr.apache.org/guide/solr/latest/index.html (visited on 2023-11-07).
[2] -. *Diagrams.net.* https://app.diagrams.net/ (visited on 2023-10-10).
[3] -. *Docker Guides.* https://docs.docker.com/get-started/overview/ (visited on 2023-11-07).
[4] -. *What is data preparation?* https://aws.amazon.com/what-is/data-preparation/?nc1=h_ls (visited on 2023-10-07).
[5] 2016. *BBC dataset from ML resources.* http://mlg.ucd.ie/datasets/bbc.html (visited on 2023-10-09).
[6] 2019. *News dataset from Components.* https://components.one/datasets/all-the-news-articles-dataset (visited on 2023-10-09).
[7] 2020. *Information Retrieval.* https://www.librarianshipstudies.com/2020/02/information-retrieval.html (visited on 2023-11-13).
[8] 2022. *Functions to preprocess and normalize text.* https://pypi.org/project/clean-text/ (visited on 2023-12-13).
[9] 2023. *Semantic search.* https://medium.com/@maithri.vm/from-keywords-to-meaning-embracing-semantic-fusion-in-apache-solrs-hybrid-search-paradigm-e7be29534ddd (visited on 2023-12-10).

A TOP 10 RESULTS

A.1 Q1

A.1.1 with boosts.

(1) Trump supporters says he speaks for them on immigration
(2) Will Trump's Tough Talk On Immigration Cause A Farm Labor Shortage?
(3) Trump at crossroads on immigration?
(4) Trump tries tricky dismount on immigration
(5) Immigrant Advocates On Trump's Immigration Speech: Told You So
(6) Explaining What Donald Trump Wants to Do Now on Immigration - The New York Times
(7) Could Any Speech on Illegal Immigration Help Trump?
(8) Trump could be softening on harsh immigration proposal
(9) Is Donald Trump Reversing His Stance on Immigration?
(10) Donald Trump: 'I'm not flip-flopping' on immigration

A.1.2 without boosts.

(1) Trump to give immigration speech amid major questions
(2) Trump's immigration speech was spot-on — but ...
(3) Trump postpones immigration speech
(4) Trump supporters says he speaks for them on immigration
(5) Trump's Immigration Disaster
(6) What Was So Shocking About Trump's Immigration Speech?
(7) Trump's hardline immigration rhetoric runs into obstacles — including Trump
(8) Highlights of Donald Trump's Immigration Speech and Mexico Trip - The New York Times
(9) Immigrant Advocates On Trump's Immigration Speech: Told You So
(10) Could Any Speech on Illegal Immigration Help Trump?

A.2 Q2

A.2.1 with boosts.

(1) LeBron James shared a touching moment with Craig Sager during the legendary announcer's final NBA game
(2) ESPN segment shows the craziest parts of LeBron James' insane, Finals-saving block in Game 7
(3) LeBron James passes Michael Jordan, Cavaliers back in Finals with 135-102 win - LA Times
(4) Talk that Spurs should have won title last year rankles LeBron James - LA Times
(5) How mind-blowing Kobe-LeBron trade broke down in 2007
(6) As Warriors Prepare for Game 7 Pressure, LeBron James Says He Doesn't Feel Any - The New York Times
(7) LeBron James sad about Finals loss: 'I've been in a funk' - LA Times
(8) LeBron James leads call to end gun violence: 'We have to do better'
(9) N.B.A. Finals: How the Warriors Stunned the Cavs to Win Game 3 - The New York Times
(10) Cavaliers finally may land LeBron 'a f—ing playmaker'

A.2.2 without boosts.

(1) LeBron James Fast Facts
(2) LeBron James will be answering for this MJ comparison all year
(3) LeBron may or may not be a better player than Jordan, but he's a better man
(4) LeBron James passes Michael Jordan, Cavaliers back in Finals with 135-102 win - LA Times
(5) LeBron James saved the Cavaliers with a superhuman play that will go down as one of the greatest blocks in history
(6) One Thing LeBron James Can't Win: A Comparison With Michael Jordan - The New York Times
(7) LeBron James passes Michael Jordan as most valuable player in NBA history
(8) LeBron James Issues a Timely Reminder of His Greatness - The New York Times
(9) NBA: LeBron James scores 35 points, Cavaliers sweep Raptors - LA Times
(10) THE LEBRON JAMES INTERVIEW: The world's best athlete reveals how his team pulled off the greatest comeback in NBA history

A.3 Q3

A.3.1 with boosts.

(1) FBI: Major Cities Hit by 21.6 Percent Spike in Murders - Breitbart
(2) No, Trump Isn't Under Criminal Investigation by the FBI
(3) Trump Says He Asked Comey Whether He Was Under Investigation By The FBI
(4) FBI joins Portland stabbings investigation
(5) The Trump Official The FBI Was Investigating
(6) FBI Investigates Possible Islamic State Knife
(7) in Virginia
(8) FBI: Murders Up Nearly 11 Percent In 2015; Violent Crime Rose Slightly
(9) FBI Director Comey Explains Reopened Criminal Investigation to FBI Agents - Breitbart
(10) Clintonworld's Top 5 Active FBI Investigations
(11) FBI: Violent crime across US spiked in 2015, murders up nearly 11 percent

A.3.2 without boosts.

(1) FBI: Violent crime across US spiked in 2015, murders up nearly 11 percent
(2) The Trump Official The FBI Was Investigating
(3) FBI: Murders Up Nearly 11 Percent In 2015; Violent Crime Rose Slightly
(4) What the hell is going on at the FBI?
(5) No, Trump Isn't Under Criminal Investigation by the FBI
(6) FBI stats back up Trump's warning on crime
(7) Along With Assault And Arson, FBI Starts To Track Animal Abuse
(8) FBI Director Comey Is Wrong: The Case for Prosecuting Hillary Clinton Is Strong
(9) Behind the scenes of the FBI director hunt
(10) FBI: Suspect in nightclub rampage investigated twice for ties to Islamic extremism

A.4 Q3

A.4.1 *with boosts.*

(1) Republicans Leave Town Without Punishing Democrats For Gun Control Sit-In
(2) Gun Ownership in LGBT Community Continues Surge Under Trump
(3) Why Democrats are excited about today's votes on doomed gun control bills
(4) U.S. House Republican gun bill draws the ire of Democrats
(5) Poll: Gun Rights More Important Than Gun Control
(6) NRA and Republicans find unlikely ally on rollback of gun control rule: science
(7) Democrats' Epic Hypocrisy on Guns and Terror
(8) Orlando, Obama and the truth about guns
(9) Sanders And Clinton Clash On Guns, Health Care In Democratic Debate
(10) U.S. gun rules heighten tension between police, citizens: Obama

A.4.2 *without boosts.*

(1) Democrats Fight Back on Gun Control
(2) House Democrats' Gun-Control Sit-In Turns Into Chaotic Showdown With Republicans - The New York Times
(3) U.S. House Republican gun bill draws the ire of Democrats
(4) Republican senator seeks bipartisan support for gun deal
(5) House Democrats Take a Milder Approach on Gun Control
(6) Democrats' Epic Hypocrisy on Guns and Terror
(7) Democrats' hope for gun control reform: appeal to Trump's 'unpredictable' nature
(8) Special Report: Why Obama and other gun control advocates own stock in firearms makers
(9) The changing politics of gun control
(10) Latest gun control bid falters in Congress, Democrat sit-in ends