Organising the mess in online news comments

Bachelor Project: Barghorn Jérémy and Ennassih Yann

Abstract

Online debate platforms, especially on news websites, often offer a great diversity of topics addressed and related opinions. But they precisely lack a summarized overall view on the trends that emerge from the comments. The need resides therefore in providing better readability for the common reader in order to improve his engagement and ability to quickly understand the issues at stake. Moreover, the main topics addressed on such platforms give a good insight on the content of the discussion and the extraction of arguments would enable the reader to precisely catch the core oppositions on a given subject. This requires summarizing the opinions and themes discussed in those online debates in a concise and visual way, preferably within a single process. This challenge meets a real-world demand; such a project is an opportunity for the media to boost their content, and on a broader scale, it could also benefit government institutions to better inform and to improve citizen participation in public debates.

1 Introduction

The goal of this project is to assemble existing, efficient and open-source technologies into a complete automated pipeline that provides the desired comments summarization. The research and tests were carried out in strict accordance with these criteria of automation and paid-API independence.

The models involved in this work are exclusively natural language processing NLP pre-trained models and sentence transformers. This includes embedding encoders, clustering and classification algorithms as well as large language models. Most models were run using the Hugging Face transformers pipelines [1]. And word embeddings were performed with the SBert `all-MiniLM-L6-v2` sentence transformer [2]. The model yields vectors of dimension 384 and was chosen for its performance and ease of use.

The main work was led around this difficulty of argument extraction, that is the retrieval of the argumentative key points of a comment or main justifications of a claim. "Argument mining" is an existing field of NLP that aims to provide this behavior. But no available and easy-to-integrate tools were found for this project's scope. The aim was then to find "workarounds" using available methods to achieve this challenging goal of argument extraction. A multitude of ready-to-deploy summarization and clustering processes were used for this purpose.

Finally, the extracted arguments had to be formatted into a compact visual layout without general loss of information. This paper strictly follows the end-to-end build process of the pipeline, from the datasets creation to the arguments extraction and topics summarization.

2 Dataset

The first major part of the project was to create a dataset. The project began from scratch, with a search conducted to gather sufficient data for establishing the foundations. The main selection criterion was a sufficient quantity and quality of comments in cases where model training would be necessary. In order to be as general as possible, the data had to come from a reliable media outlet with active readers publishing comments, developing opinions and expressing views.

2.1 Origin of the dataset

There weren't many datasets on the internet (Kaggle) that had data quantity, data quality and were cost free. That is why the New York Times was quickly settled on, as it has made available two datasets containing all the comments and their related information for relatively recent articles. The first one "New York Times Comments" [3] is made of comments for articles published between Jan-May 2017 and Jan-April 2018 and the second one "New York Times Articles & Comments (2020)" [4] contains articles between Jan-Dec 2020. The two datasets were very similar and contained a large number of features from which the most relevant were selected in order to combine them among the years.

Dataset	Articles	Comments
NYT 17/18	approx. 9k	approx. 2M
NYT 20	approx. 16k	approx. 5M
NYT merged	26 121	6 863 978

Table 1: Number of comments and articles.

Dataset	Articles	Comments
NYT 17/18	16	34
NYT 20	11	23
NYT merged	6	7

Table 2: Number of features for comments and articles.

In the end, two files were created, one containing data on articles and the other on comments. Both are available in a `.tsv` or in a `.parquet` format. A standardization process was carried out beforehand. The identifier per article was recreated so that it would be consistent between articles, and the link to the comment was made to facilitate the search for comments from an article or vice versa.

Feature	Description
ID	Unique article identifier (e.g. NYT-ART-0000000001)
word_count	Article word count
headline	Article headline
keywords	List of keywords describing the content of the article
pub_date	Publication date
abstract	One sentence article description or null

Table 3: Features of the articles table.

Feature	Description
commentID	Unique comment identifier
articleID	Article to which the comment is related
commentBody	Text of the comment
replyCount	Numbers of replies to this comment
recommendations	Integer representing a rating of the comment
userLocation	Location the user entered
parentID	Reference to the parent comment in case of reply

Table 4: Features of the comments table.

2.2 Dataset processing

In addition to the creation of the two abundant datasets, comments were classified among 7 emotions together with a confidence score in [0,1] using the `emotion-english-distilroberta-base` model and the Hugging Face pipeline [5]. This labeling allows a first insight on each comment and yields an initial comment differentiation that is always useful for further data analysis.

The articles, in turn, were useful to separate the comments into several clusters along similar headlines. High dimensional data graphical representation was made possible with the help of the TensorFlow embedding projector [6]. A qualitative analysis then proved the clustering to be much more effective when using the headline embeddings than the average keywords embeddings. Comments were subsequently split into 138 clusters using the `util.community_detection` fast clustering method of SBert with a cosine similarity threshold of 0.5.

Those clusters were then processed to remove all hyperlink HTML tags as well as username tags in case of a reply. This was done to further standardize the datasets but is actually not necessary as the removed elements possibly contain useful information. The reply attribute of a comment is still kept by a simple null check of its parentID value, so this data processing does not prevent future work that needs to focus on entire reply sets.

2.3 Final dataset

The main research and implementation work was performed on the 138 comments clusters (see 2.2). The average number of comments per cluster is 8000 comments, with the smallest cluster containing about 80 comments and 330 000 for the biggest one. The clusters together hold a total of 2 687 612 comments related to 7165 articles.

An example of some clusters can be found in the following table. The top three were the most used sets for fine-tuning and any extraction work.

Cluster	Theme	Comments nb.
0	Covid	330 000
3	Joe Biden	140 000
132	George Floyd	10 501
133	Book reviews	2 100

Table 5: Comments clustered by theme.

The attributes of each cluster entry were kept similar to the original exhaustive dataset of comments (see 2.1), including now the label and score emotions classification.

New features	Description
label	classification among 7 emotions
score	classification precision ∈ [0, 1]

Table 6: Additional features for the comments.

2.4 Datasets for fine-tuning

In addition to the New York Times dataset used, two new datasets were created in order to fine-tune models for argument and topic extraction. These files contain two features : Question and Answer. A suitable size for training a large Flan-T5 language model with 780 million parameters is 15 000 questions with answers. Based on this information the comments were used to create the questions and then a fast way to label the data (find the answer) was needed. The Openai API was a suitable solution, as its descriptive power was good enough to provide a convincing result. The model used to provide the answers was the `gpt-3.5-turbo`.

The first set of instructions was designed to teach the model to transform a comment, which may be very long, to a short, concise argument (1 sentence) containing the essential information. The second set teaches the model to find the main topics (ideally 3) from a short, concise argument.

The comments used where taken from the cluster containing articles about Joe Biden, as they mostly contain well-structured arguments on a variety of subjects. The comments are then filtered by length, removing the ones that were shorter than a sentence. All replies were removed too, in order to take only the initial comments. Finally, a minimum recommendation threshold of 20 was used to obtain only comments of sufficient quality.

Each datapoint was then sent to the Openai API to retrieve the desired output. The `gpt-3.5-turbo` API takes a system behavior (which will impact the way the model answers) and a sentence as input and returns the response. For the arguments dataset, the system behavior was `You are a 1 short sentence argument extractor. Only provide the answer.` and the template sentence added in front of each comment was `Extract the main argument of this comment: [comment].` In a similar manner for topics, the system behavior was set to `Answer only in the format of a python list containing 3 elements.` and the template sentence added was `Given this argument extract the 3 main topics? [argument].` The `gpt-3.5-turbo` model costs $0.002 for 1k tokens. The dataset of comments to arguments, once transformed, consumed 2.8M tokens (questions and answers included) at a cost of $5.6. The arguments to topics dataset, once transformed, consumed 1.5M tokens at a cost of $2.9.

3 Argument extraction

Formally, from [7], Argument Mining is a “field of corpus-based discourse analysis that involves the automatic identification of argumentative structures in text”. However, given this task, state of the art models do not offer the option of an efficient deployment yet. So to get as close as possible to this desired behavior, several methods were performed and analyzed on our datasets, from extractive summarization to large language models.

3.1 Conventional approach

As word2vec transformations are very powerful for clustering and similarity comparison in NLP, the first approach taken was to determine if word embeddings are already capable of encoding the argumentative layout of a comment. A qualitative study with the comments dataset and the `all-MiniLM-L6-v2` transformer [2] was conducted to tackle this matter. Comment embeddings were projected in 3D and 2D spaces, using the PCA, UMAP, T-SNE algorithms in TensorFlow [6], in order to identify potential natural clusters; and most importantly, their precision when it comes to hold the argumentative core structures of the comments. In addition, iterative K-means clustering was used to recursively split the comments into small chunks to highlight potential embedder abilities to catch argumentation rather than main keywords above a certain similarity threshold. This analysis did however not reveal any of these behaviors.

Another methodology was to perform summarization techniques on the comments. Both `facebook/bart-large-cnn` [8] and `google/pegasus-cnn-dailymail` [9] models were tested for this purpose. For most of the longest comments, these methods were able to provide a good summary, but results were often too extractive and focused on a few sentences without including the argumentative content of the comment.

Nevertheless, in these attempts to satisfactorily extract arguments, it is important to keep in mind the limitations of a qualitative study.

3.2 Approach using LLM's

After the more conventional approach with basic tools and given the fact that there was no existing pretrained model available and ready to use, the decision to try Large Language Models or LLM's was taken. LLM's are a suitable option because they have become increasingly accessible, have the ability to answer abstract questions and cover a more varied field. Because their training is based on huge datasets, they also have significant descriptive power which, if properly applied, could help us achieve the desired result.

From the most advanced models available, numerous prompts were tested to obtain an appropriate and consistent result on a number of questions. They were tested on the different Flan-T5 variants (large, xl, xxl) [10] and on the Stanford Alpaca [11] model.

Model	Parameters	Size
Flan-T5 large	780M	1.8k
Flan-T5 xl	3B	1.8k
Flan-T5 xxl	11B	1.8k
Stanford Alpaca	7B	52k
gpt-3.5-turbo	175B	unknown

Table 7: Size comparison between different models in terms of number of parameters and size of instruction set.

The approach worked not as well as expected since major difficulties were encountered in forcing the model's output to be consistent even if the prompts used were the ones that came directly from the instruction set. For example, the result depended too much on the comment used, and the model opted to summarize or simply extract some parts instead of giving an actual argument. The output format was inconsistent, and the short sentence requirement was never respected. Sometimes the model tended to hallucinate, incorporating erroneous or irrelevant information into the output.

The same tests were carried out on the `gpt-3.5-turbo`, but this time with good results. The argument covered the important points well, summarized the comment in a short sentence and was consistent on several inputs. The result is not surprising given that the model has 15 to 200 times more parameters and has several fine-tuning and reinforcement learning overlays that the others don't.

The concern was that the pipeline should not be dependent on a paid API, because if a real application is envisaged, sending user comments and information to a private organization could be a problem. A model similar in size to the Flan-T5 large or xl would be more suitable, as it could be easily deployed on a normal infrastructure while being less costly.

3.3 Adopted solution

Finally after a long period of testing the decision to try to fine-tune a Flan-T5 was taken. A dataset has been specially created (see 2.4) to teach a new instruction to the existing model to specialize it in argument extraction. Fine-tuning was carried out in less than 3 hours on a Flan-T5 large, as this was the largest model that could be loaded on a single `NVIDIA A100-SXM4-40GB GPU`. In the end, we obtained an LLM that has all the vocabulary and the ability to construct well structured sentences, while applying it to argument extraction. We performed the same tests as on the other models and the results were very convincing. Outputs are not as good as those of the `gpt-3.5-turbo`, but they by far outperformed all other non-tuned versions. The model may have a tendency to write long sentences if the comment is long, but there is no more concern about hallucinations or inconsistency in the output. Moreover the result is always shorter than the initial comment and is able to find the main argument. The fine-tuned version is also able to detect sarcasm in certain cases and provides an argument that is in line with the meaning of the comment. The template sentence used to trigger the argument extraction is `Extract the main argument of this comment: [comment].`

4 Topic extraction and summarization

Extracted arguments now facilitate the construction of a hierarchical structure of the topics. The goal to achieve here is to build a tree-like layout where topics contain multiple arguments linked to their original comment. This last layer completes the pipeline and provides a visual and summarized presentation of a set of comments to the reader.

4.1 Prompt engineering

A non-trivial part of this pipeline subtask is to extract a few main topics from arguments, as they need to encapsulate all the relevant information in a very short number of words. A first available model `valurank/MiniLM-L6-Keyword-Extraction` [12] enabled a simple keywords extraction. This offers great output consistency but includes irrelevant words for the desired purpose of topic extraction.

Topic extraction prompts were then given to the Flan-T5 [10] and the Alpaca 7B [11] models. The prompts were strongly inspired by the 52k instructions fine-tuning set of Alpaca and oriented to constrain the outputs to a certain consistent format. The extraction was performed on single argument inputs and argument lists, as well as on the original comment bodies. The Alpaca model offers a good trade-off between topic length and relevancy, yet the generated tokens are still not as consistent as desired along multiple runs.

On another note, these models hardly understand number constraints given in the prompt, e.g. `Generate the 3 topics of this text.` A further step in this work should be to delve into constrained beam search, implementing for example a custom subclass of `Constraint` of Hugging Face [13] to force the model to generate some specific tokens.

4.2 Fine-tuning

In the same way as for part 2.3 after some unsuccessful testing on various prompts the decision was taken to fine-tune a second Flan-T5 large model in order to do topic extraction. The input that can be used to trigger the model to fulfill this task is : `Given this argument extract the 3 main topics? [argument].` The output will be in the format of a python list since this was the best solution to force the model to be consistent. The boundary of 3 topics is not always respected by the Flan-T5 if the input arguments are long but the output is convenient enough since these topics are clustered and only the ones that appear the most will be selected at the end to be displayed.

4.3 Topic clustering

In the previous work of topic extraction, the following process was also tested : at each model input, a prompt with a topic list and new arguments were given with the task to update that list. This task would enable us to return a final list of topics for any given set of arguments but it turned out to be too complex, even for state of the art models such as `GPT 3.5`.

Once the topic extraction instructions were created (see 2.4) and the output format set, the final step was to perform topic clustering in order to reduce the number of topics covered into a short set of main themes. The TensorFlow projector [6] and the K-means algorithm were used for this purpose. But there remained one challenge : how to annotate each cluster with the best terms, i.e. how to choose the pipeline's top layer words that will be displayed to the reader ? Once each cluster is obtained, a pairwise cosine similarity can be computed between the topics, mapping each topic to a score. The topic with the biggest score is then selected to describe the entire cluster. A similar process is already integrated in the final retained solution.

For the end pipeline the method selected to cluster the main topics was the community detection from the SBERT Sentence Transformer library. The `util.community_detection` takes a similarity threshold that was set to 0.60 and the minimal community size is adjusted dynamically based on the number of topics in input. The best community size is computed with different starting points based on the amount of data to handle and then a binary search is performed in order to show between 5 and 10 main topics. Besides the efficiency of this method, it also yields each cluster with topics sorted by similarity scores in decreasing order. The two most "centered" topics were then chosen to describe each community, completing therefore the pipeline.

Another studied but not used promising approach is to perform an agglomerative clustering on the topics, which directly returns a tree-like structure : the user only needs to decide the number of different topics to display and the pipeline then runs through the tree to the layer with the desired number of clusters.

5 Web-application : presentation and features

The ultimate aim of this research project was to propose a web application that would enable all the work carried out to be visualized. Once the data pipeline had been defined and all the components required to process comments were functional, everything was put together in a `Streamlit` web-page. The aim of this website was to present as faithfully as possible a concrete implementation of our project to an existing newspaper. This means that it is possible to select an article from those available, read the comments and see directly what our pipeline has provided in order to summarize this large mass of information. We can compare side by side the input and the result to see that with the new tool we are able to faster understand the main ideas coming from the comments in a shorter time period.

5.1 Pre-processing and on demand computation

Given that the amount of articles and comments available to us is huge, not everything was preprocessed in terms of argument or topic extraction. By going through the web-app there are some clusters (Cluster 3 and Cluster 132) that were pre-processed and others that are not. However, in order to be consistent the data pipeline is available for all clusters. That's why it's possible to load pre-trained models into memory (RAM or G-RAM) and perform the analysis live on the CPU or GPU (the app will automatically select the best available resources). Sliders can be used to select the number of comments to be processed, as this can be very time-consuming. This feature also allows us to see how our pipeline dynamically adapts to the data received.

5.2 Data pipeline implementation

Firstly, the application has access to various articles in the form of clusters (see 2.2) and allows their contents to be viewed. A user can select a specific article or the entire cluster to continue his analysis. Depending on the selection some information is displayed : article identifier, keywords, publication date, abstract if it exists and a bar chart showing an emotion analysis of the cluster and/or the article (see 2.2). In the center of the page there are the comments related to the user's selection. If pre-processing was done on this cluster or if the user triggered the processing of some data, a section showing the main topics appears. These main topics are derived from the adaptive community detection performed on the topics (see 4.3). Topics were generated from arguments with the `Flan-T5-topics` (see 4.2) model and arguments were generated from comments with the `Flan-T5-arguments` (see 3.3) model. The user can then select one or more main themes and the corresponding information is displayed (Arguments, Comments and Emotions). The data within a theme can then be sorted according to emotions to provide a better understanding of the information. Note that the community detection step of the pipeline is never pre-processed.

5.3 Single article or multi document summarization

Since the implemented pipeline is very general it is easy to feed it with multiple articles as input and then wait for the community detection to find the main topics. This feature is available on the app if the `Select all articles in cluster` checkmark is selected. The amount of data to handle can be huge for example Cluster 1 about Covid-19 contains 375 000 comments and we thus have approximately 1M topics to cluster. This step is currently not parallelized or optimized and can take some time. If the Cluster 3 about Joe Biden is selected and all the 210 articles and 140 000 comments are processed, the 5 main topics found are : `['Biden administration / Biden not as sharp as Clinton', 'Presidential behavior / nominee to defeat Trump', 'Voting', 'Joe Biden blew it / Joe Biden as a creep', 'Donald Trump / Voting for Democratic candidate']` Note that for a main topic the two first topics of the community are displayed if they are different.

5.4 Backend

In the case where the arguments and/or topics are not pre-processed there is always a possibility to load the models `Flan-T5-arguments` and `Flan-T5-topics` in the background to analyze the data. This can be done in the `Control` tab of the web-app. One model takes approximately `6GB` of G-RAM on a `NVIDIA TITAN Xp 12GB` and is able to process 1 comment or argument per second.

6 Conclusion

The aim of this semester project was to find a way of making the comments section under newspaper articles more user-friendly and readable. Using natural language processing and avoiding human assistance in the pipeline, we developed a methodology that can handle huge data amounts in order to extract relevant arguments and summarize the discussed content into main topics.

Four main themes were tackled during these 14 weeks. First, there was the need to create a dataset meeting the requirements for the rest of the project. Secondly, we had to develop methods for extracting arguments that were previously non-existent. Then the knowledge acquired in the field of argument extraction was reused and applied to topic extraction. Finally, in order to better visualize this gigantic mass of data and to verify that our implementation was functional, a website was developed to bring together all the elements mentioned above.

The results obtained are difficult to evaluate, since the extraction of arguments or topics can only be tested subjectively by reading the mass of comments and doing the analysis work manually. However, after numerous tests and discussions with the project supervisors, we have come to the conclusion that the results are satisfactory for the task in hand and the final rendering shows off all the work done.

This research work presents one way of doing things, but many other decisions could have been made throughout the thought process, and many points are open to improvement. The project can be seen as proof that the means currently available are sufficient for a reliable, effective implementation applicable to the real world in real time.

The future limitations we can envisage with this approach is the political neutrality of training sets. Indeed, training sets influence the way models behave, and will emphasize information. In addition, we may wonder to what extent a newspaper will be able to use this real-time analysis to choose its hedlines and strongly influence the public debate. This information will make it possible to write only articles that arouse certain emotions or target a certain type of reader, and we will need to think about the ethics of this tool since it is initially designed for the reader.

7 Acknowledgements

We would like to express our sincere appreciation to the project supervisors, Dr. Rémi Lebret, Dr. Léo Laugier and Stéphane Massonet, for their support throughout our bachelor project. Their expertise and help have been instrumental in shaping the direction of our research and ensured its successful completion.

References

1. Face, H. *Pipeline for machine learning models* https://huggingface.co/docs/transformers/main_classes/pipelines.
2. SBert. *384 dimensional word embedding* https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2.
3. Kesarwani, A. *New York Times Comments* https://www.kaggle.com/datasets/aashita/nyt-comments.
4. Dornel, B. *New York Times Articles Comments (2020)* https://www.kaggle.com/datasets/benjaminawd/new-york-times-articles-comments-2020.
5. Hartmann, J. *Emotion English DistilRoBERTa-base* 2022. https://huggingface.co/j-hartmann/emotion-english-distilroberta-base.
6. TensorFlow. *High-dimensional data visualization* https://projector.tensorflow.org/.
7. Chakrabarty, T., Hidey, C., Muresan, S., Mckeown, K. & Hwang, A. *AMPERSAND: Argument Mining for PERSuAsive oNline Discussions* 2020. arXiv: 2004.14677 [cs.CL].
8. Lewis, M. *et al.* BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *CoRR* **abs/1910.13461.** arXiv: 1910.13461. http://arxiv.org/abs/1910.13461 (2019).
9. Zhang, J., Zhao, Y., Saleh, M. & Liu, P. J. *PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization* 2019. arXiv: 1912.08777 [cs.CL].
10. Chung, H. W. *et al. Scaling Instruction-Finetuned Language Models* 2022. arXiv: 2210.11416 [cs.LG].
11. Taori, R. *et al. Stanford Alpaca: An Instruction-following LLaMA model* https://github.com/tatsu-lab/stanford_alpaca. 2023.
12. Valurank. *Keywords extraction based on SBert MiniLM-L6 model* https://huggingface.co/valurank/MiniLM-L6-Keyword-Extraction.
13. Face, H. *Transformers generation utilities : constrained beam search* https://huggingface.co/docs/transformers/main/en/internal/generation_utils#transformers.Constraint.

8 Appendix

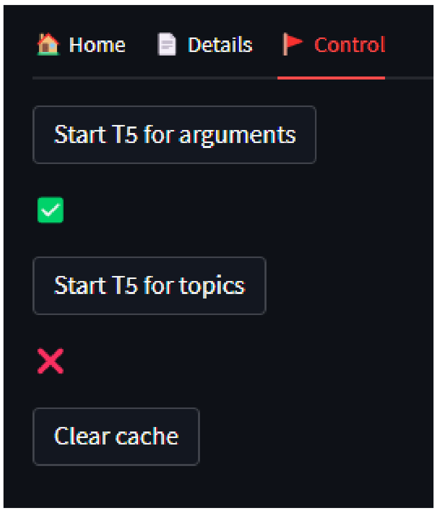
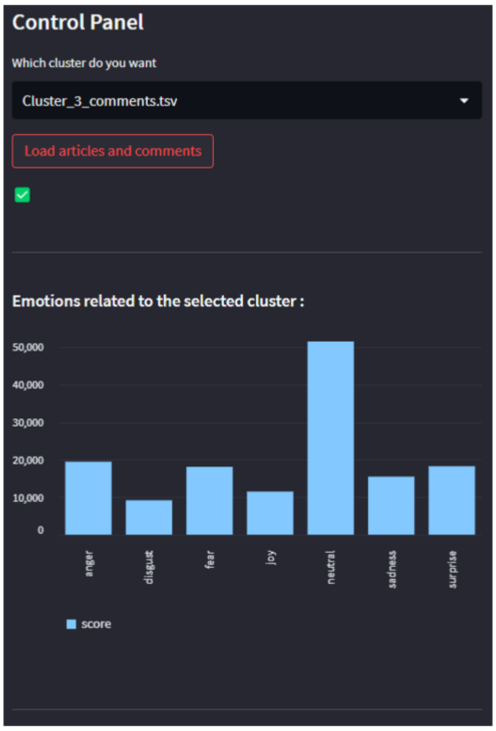


Figure 1: Control panel allowing to start the Flan-T5 models in memory.



Control Panel

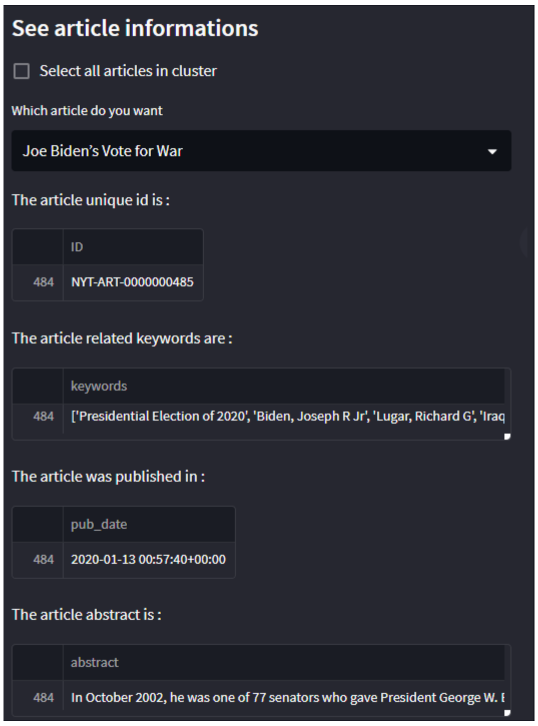
Which cluster do you want

Cluster_3_comments.tsv

Load articles and comments

Emotions related to the selected cluster :

Figure 2: Control panel showing article information.



See article informations

Select all articles in cluster

Which article do you want

Joe Biden's Vote for War

The article unique id is :

	ID
484	NYT-ART-0000000485

The article related keywords are :

	keywords
484	['Presidential Election of 2020', 'Biden, Joseph R Jr', 'Lugar, Richard G', 'Iraq

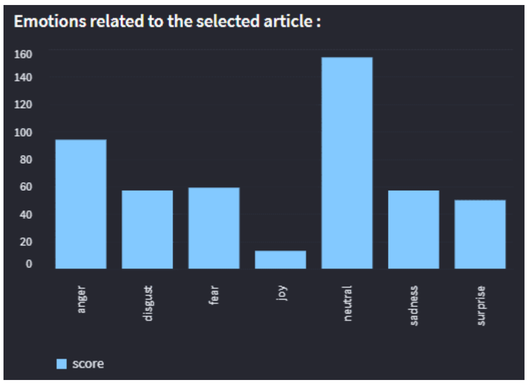
The article was published in :

	pub_date
484	2020-01-13 00:57:40+00:00

The article abstract is :

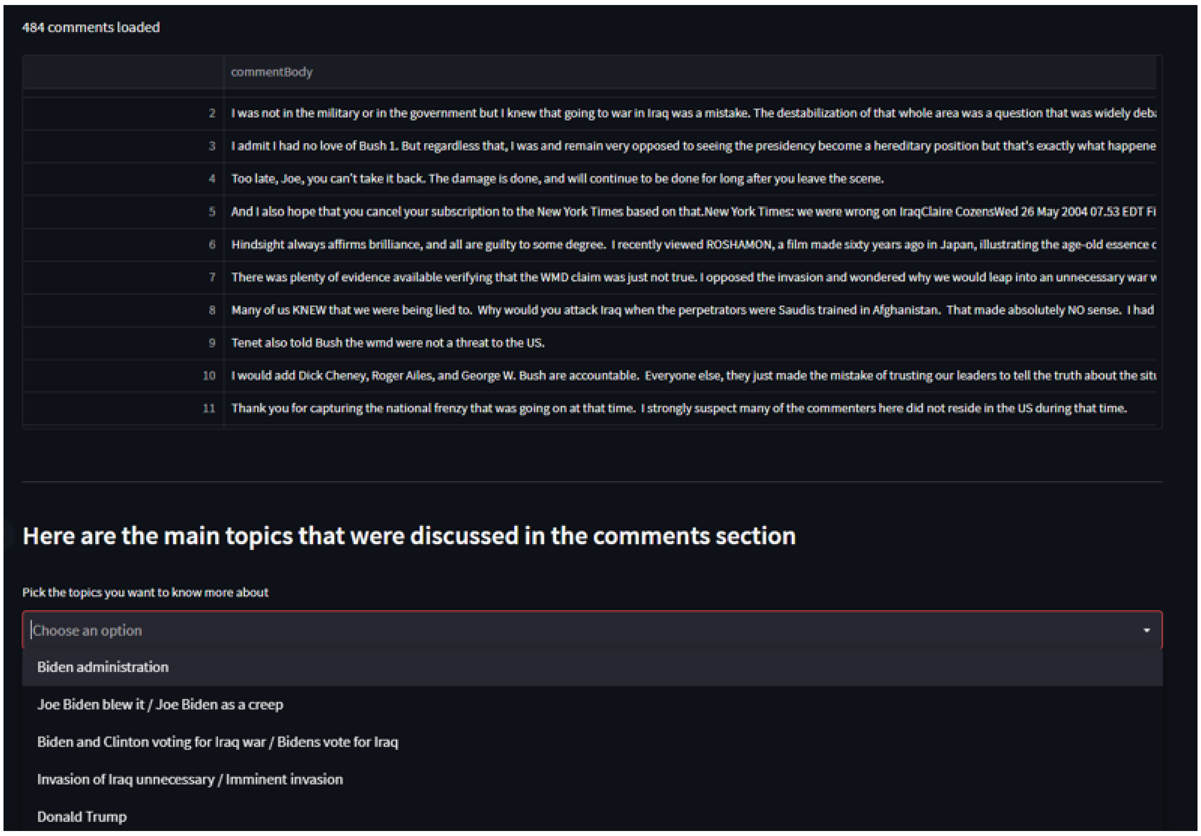
	abstract
484	In October 2002, he was one of 77 senators who gave President George W. I

Figure 3: Control panel showing article information.



Emotions related to the selected article :

Figure 4: Control panel showing article information.



484 comments loaded

	commentBody
2	I was not in the military or in the government but I knew that going to war in Iraq was a mistake. The destabilization of that whole area was a question that was widely deb
3	I admit I had no love of Bush 1. But regardless that, I was and remain very opposed to seeing the presidency become a hereditary position but that's exactly what happene
4	Too late, Joe, you can't take it back. The damage is done, and will continue to be done for long after you leave the scene.
5	And I also hope that you cancel your subscription to the New York Times based on that.New York Times: we were wrong on IraqClaire CozensWed 26 May 2004 07.53 EDT Fi
6	Hindsight always affirms brilliance, and all are guilty to some degree. I recently viewed ROSHAMON, a film made sixty years ago in Japan, illustrating the age-old essence c
7	There was plenty of evidence available verifying that the WMD claim was just not true. I opposed the invasion and wondered why we would leap into an unnecessary war w
8	Many of us KNEW that we were being lied to. Why would you attack Iraq when the perpetrators were Saudis trained in Afghanistan. That made absolutely NO sense. I had
9	Tenet also told Bush the wmd were not a threat to the US.
10	I would add Dick Cheney, Roger Ailes, and George W. Bush are accountable. Everyone else, they just made the mistake of trusting our leaders to tell the truth about the sitı
11	Thank you for capturing the national frenzy that was going on at that time. I strongly suspect many of the commenters here did not reside in the US during that time.

Here are the main topics that were discussed in the comments section

Pick the topics you want to know more about

Choose an option

Biden administration

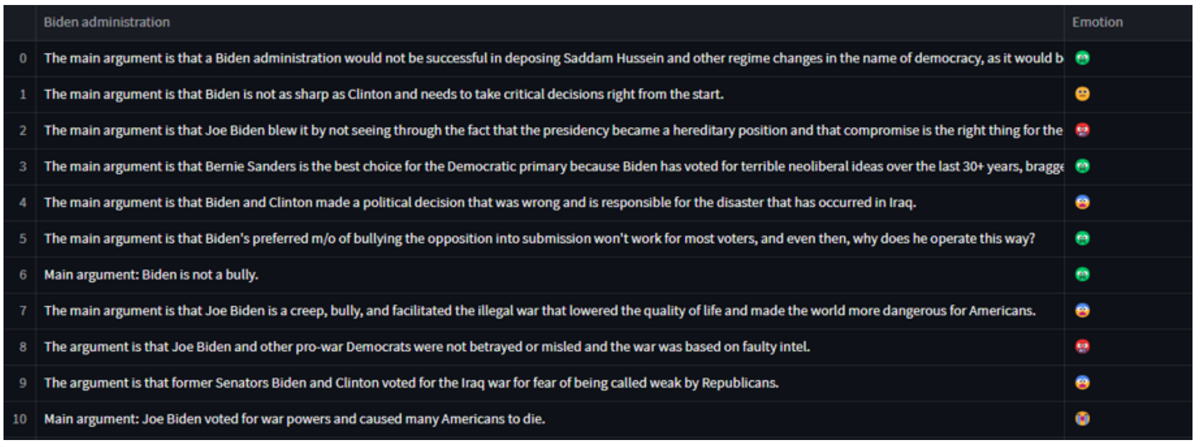
Joe Biden blew it / Joe Biden as a creep

Biden and Clinton voting for Iraq war / Bidens vote for Iraq

Invasion of Iraq unnecessary / Imminent invasion

Donald Trump

Figure 5: Main page showing the comment body for the selected article and the main topics found by the pipeline.



	Biden administration	Emotion
0	The main argument is that a Biden administration would not be successful in deposing Saddam Hussein and other regime changes in the name of democracy, as it would b	🤢
1	The main argument is that Biden is not as sharp as Clinton and needs to take critical decisions right from the start.	😕
2	The main argument is that Joe Biden blew it by not seeing through the fact that the presidency became a hereditary position and that compromise is the right thing for the	😡
3	The main argument is that Bernie Sanders is the best choice for the Democratic primary because Biden has voted for terrible neoliberal ideas over the last 30+ years, bragge	🤢
4	The main argument is that Biden and Clinton made a political decision that was wrong and is responsible for the disaster that has occurred in Iraq.	😨
5	The main argument is that Biden's preferred m/o of bullying the opposition into submission won't work for most voters, and even then, why does he operate this way?	🤢
6	Main argument: Biden is not a bully.	🤢
7	The main argument is that Joe Biden is a creep, bully, and facilitated the illegal war that lowered the quality of life and made the world more dangerous for Americans.	😨
8	The argument is that Joe Biden and other pro-war Democrats were not betrayed or misled and the war was based on faulty intel.	😡
9	The argument is that former Senators Biden and Clinton voted for the Iraq war for fear of being called weak by Republicans.	😨
10	Main argument: Joe Biden voted for war powers and caused many Americans to die.	😭

Figure 6: Section showing the extracted arguments and emotions for the topic ”Biden administration”.