Twitter Event Discovery

Dayou Wu[*]
Bohan Liu[*]
Kaiyue Zhang[*]
[dayouwu2,bohan3,kaiyuez2]@illinois.edu
University of Illinois, Urbana-Champaign
Urbana, Illinois, USA

Figure 1: Image of Twitter network with events generated with Microsoft New Bing and DALL-E 3

ABSTRACT

In today's digital age, the ability to rapidly detect and analyze global events through social media is invaluable for timely decision-making and informed public discourse. This research project proposes a novel framework for the detection of events from Twitter data, by leveraging the abilities of large language models to identify named entities and using a similarity search method to align with the comprehensive event records maintained in the Global Database of Events, Language, and Tone (GDELT). This alignment allows for a richer analysis by combining the immediacy and user-driven content of Twitter with the structured, global perspective of GDELT. The project addresses critical challenges such as the identification of relevant tweets which include noise and misinformation, the integration of disparate data formats, and the correlation of event data across platforms.

[*]All authors contributed equally to this research.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

CCS CONCEPTS

• **Information systems** → **Similarity measures**; **Language models**; **Information extraction**; **Content analysis and feature selection**.

KEYWORDS

Information Retrieval, Event Discovery, Named Entity Recognition, Large Language Models

ACM Reference Format:
Dayou Wu, Bohan Liu, and Kaiyue Zhang. 2024. Twitter Event Discovery. In . ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Social media platforms such as Twitter act as a rich source of real-time information. Understanding the event reference in social media text data is beneficial to big data tasks, like user-based recommendation systems. Although such platforms provide the opportunity for the quick spread of large amounts of information, it is very challenging to detect events from Tweets. The challenge mainly lies in the following aspects:

- **Volume**: Twitter generates a massive amount of data every day, with millions of tweets posted every hour. Processing this amount of data requires an efficient and robust algorithm.
- **Variety**: Tweets vary in quality, relevance, and accuracy. They can include slang and abbreviations, adding complexity to the Natural Language Processing (NLP) tasks.

- **Contextual Ambiguity**: The concise nature of tweets (limited to 280 characters) often leads to ambiguous content that can be challenging to interpret. Users might use sarcasm, irony, or colloquial expressions that are difficult for algorithms to understand correctly.
- **Noise and Redundancy**: A significant amount of tweets might be repetitive or off-topic. Efficiently filtering out noise and focusing on informative content is crucial for accurate event detection.

Addressing these challenges requires sophisticated techniques in information retrieval, NLP, machine learning, and scalable computing. In order to make our event detection framework efficient, robust, and trustworthy, it is important to handle the Twitter data carefully. There are several challenges:

- How to clean the dataset and only leave useful information?
- How to extract information from noisy and ambiguous tweets?
- How to identify events from the extracted information?

In this research, we aim to address the above challenges through a robust event detection framework. First, our framework cleans the noisy Twitter dataset to filter out spam, redundant, and irrelevant information. Second, we perform Named Entity Recognition (NER) on filtered data by leveraging the power of Large Language Models (LLMs). Finally, we align the Tweets with existing event databases such as the Global Database of Events, Language, and Tone (GDELT) [12] by performing similarity matching.

2 RELATED WORKS

2.1 Large Language Models

The language modeling task involves interpreting an input text sequence, the context, and calculating the text/token density at requested locations[29]. Traditional methods use numerical or deep models to fit the n-gram probability [3] with a target corpus [5, 15]. Then, with the advent of the transformer architecture [25], transformer-based language models [7, 16, 22] became popular. Variations in transformer-based LMs, after tuning or prompting, have shown promising results in NLP tasks, like semantic classification, text generation, natural language inference (NLI), NER, and so on [9, 10]. Then, with the advent of the GPT-3 and subsequent work in Reinforcement Learning from Human Feedback (RLHF) [4, 20], the ability of transformer-based language models grew significantly and developed the capacity to handle multi-round conversation and mimic human assistants. Such capacity has brought new possibilities in processing complex text content at scale. In our study, we experimented with the latest Llama-3 model [23, 24] on its ability to extract useful information for event extraction.

2.2 Named Entity Recognition

Named Entity Recognition (NER) is a critical task in the field of NLP that involves identifying and classifying named entities in text into predefined categories such as names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. NER serves as a foundation for many NLP applications including information retrieval, machine translation, and question answering.

Traditional NER involves rule-based methods and hand-crafted features. Some advancements involving the use of Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) [8, 18], are known to outperform traditional NER methods.

Recent advancements in language models have provided us with state-of-the-art NER systems that involve the use of transformer-based language models [7, 11, 16] which are pre-trained on vast copra and fine-tuned for specific NER tasks.

2.3 Event Extraction

Event Extraction task involves identifying and classifying both triggers and arguments of an underlying event in the text content [14]. Recent works on Event extraction with BERT-like models [17, 26] require task-specific or task-agnostic tuning. The capacity of such models is limited by the performance of the size of a pre-trained language model (PLM) and the coverage of the dataset used for tuning the pretrained model. In order to handle complex and noisy Twitter data, we use zero-shot LLM prompting to tackle the problem of coverage and model capacity.

Recent advancements in language modeling at scale have brought many transformer-based language models [6, 7, 13, 16] with immense parameters.

3 METHOD

In this project, we created a event detection pipeline for the Twitter Dataset as shown by the diagram (**Figure2**). The detailed components are listed below.

3.1 Overview

To further advance the capability of implementing event detection in free text environments like Twitter, building on our previous work, our Twitter-based event detection model will employ the pipeline illustrated in **Figure2**. Upon acquiring the raw Twitter dataset, the model first undergoes two phases of data cleansing. The initial phase standardizes the format of the text acquired, while the subsequent phase eliminates prevalent spam content within the texts. This spam not only exhibits high repetitiveness but its voluminous, meaningless nature significantly impacts subsequent model analyses, increasing computational complexity and consuming additional storage space. Following this, our model leverages a large language model from spaCy to conduct named entity recognition (NER). By deploying a LLM that has been fine-tuned on the Twitter dataset, our model enhances its capability to identify entities within the dataset. After extracting entities from the texts, the model integrates a corresponding timespan's GDELT dataset, which encompasses extensive event information including participants and links to reports; the specific contents will be detailed later. This data is then transformed into vector embeddings using Langchain OpenAI embedding, followed by storage and querying in a vector database (MongoDB). The model conducts a similarity search based on the semantic similarity between GDELT-linked texts and the identified entities. Ultimately, events surpassing a certain threshold are selected as the relevant events pertaining to the Twitter content described.

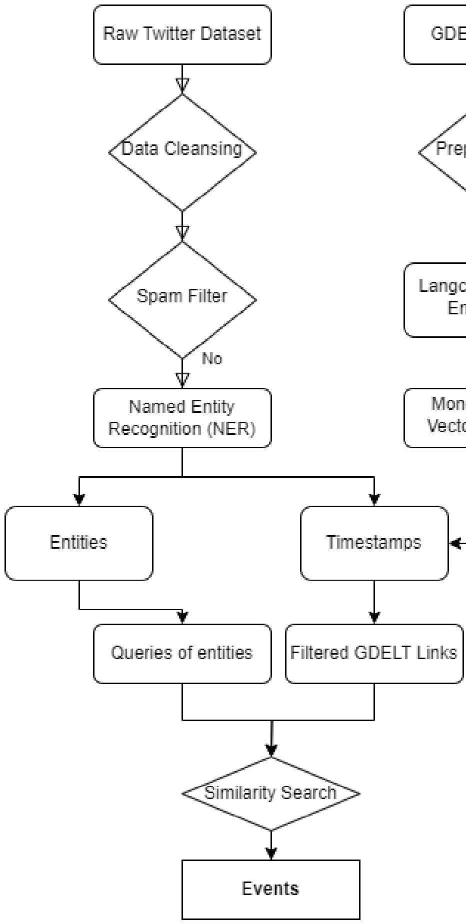

Figure 2: Schematic of the model pipeline

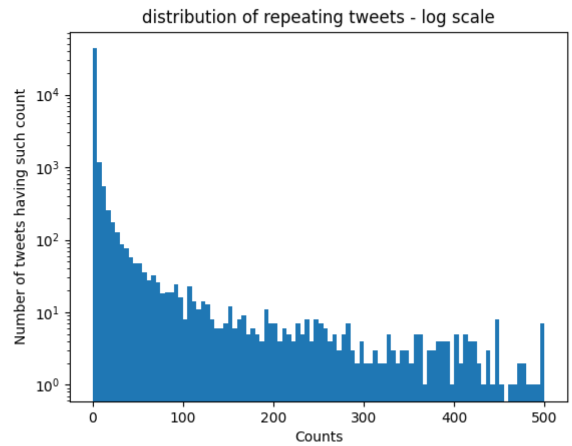

Figure 3: Tweet repeat count distribution - log scale, outlier omitted for zooming

3.2 Data Cleansing

To enhance event detection capabilities on Twitter, an initial focus is placed on data cleansing, which is essential due to Twitter's inherent design limiting tweets to 30 words, often not allowing for meaningful information. The presence of numerous emojis, slang, and retweets further disrupts textual structure and coherence, challenging the effectiveness of Large Language Models (LLMs). Thus, pre-processing Twitter text is crucial, consisting of two main parts: processing the text itself and filtering spam and spammers.

Like all text pre-processing, our model initially removes emojis, non-standard characters, numbers, and converts all text to lowercase. Subsequently, based on the nltk toolkit, stop words from the English lexicon are also removed. Following this, we noticed repetitive posting behavior from single users, indicative of potential spamming. Drawing from literature on the impact of spam on Twitter[19], it was deemed necessary to further filter out these spam users and their messages.

Contrary to traditional studies[28] that use the difference between a user's out-degree (content they initiate) and in-degree (replies, retweets, comments) as a threshold for detecting spam, our model introduces a novel formula that improves upon existing graph-based spam detectors.

Let $E(u, t)$ be the engagement level of user u with topic t, and $N(u, t')$ be the engagement level of user u with any other topic t' where $t' \in T \setminus \{t\}$. The difference in median engagements can be defined as:

$$\Delta = \text{Median}(E(u,t)) - \text{Median}\left(\bigcup_{t' \in T \setminus \{t\}} N(u,t')\right)$$

We propose that a user who follows many topics and frequently posts under various topics should not be considered a spammer, even if they rarely interact with other users. Thus, we defined the following formula based on this delta value to set thresholds for filtering out spam information.

This data cleansing approach was tested across multiple Twitter datasets, with the results as shown in Table 1 below. These results affirm that our pre-processing effectively reduces dataset size and computational costs while maintaining text quality and minimizing disruption from noise and spam.

Though we have reduced the data size through preprocessing, it still contains too much information and make it expensive to run LLM applications. Therefore, we need to reduce the redundancy in our dataset. Among the 5494548 tweets posted in January 2023, a vast majority of them are retweets, and therefore duplication in the dataset. After de-duplication based on the full-text content of the tweet, the dataset contains 46867 distinct entries. As shown in Figure 3, most of the data mass has less than 5 repeats in the dataset, which means the tweet was not retweeted often, indicating a lesser importance and public interest. Therefore, we remove such tweets and gather the rest 3438 popular tweets from January 2023 for processing.

Table 1: Dataset Volumes Before and After Cleansing

Twitter Datasets	Before Cleansing (million)	After Cleansing (million)
French Election	3.882	0.891
Israel Hamas Conflict	1.831	0.457
South China Sea	5.494	1.131
Russia Ukraine War	4.572	0.930

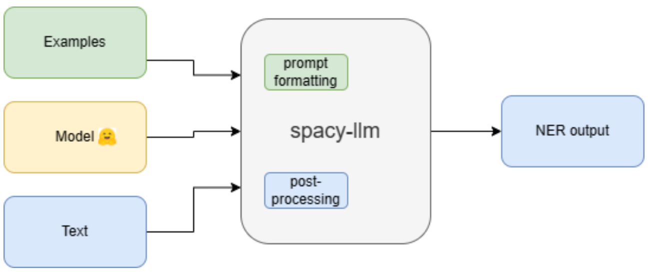

Figure 4: Named Entity Recognition Pipeline with spacy-llm[2] and Llama-8b-Instruct-awq [1]

3.3 Named Entity Recognition

For NER on the tweets, we use few-shot prompting based on spacy-llm library and Llama-8b-instruct-awq. The NER pipeline is illustrated in Figure 4. The pipeline takes in a LLM, fewshot example(s) and text content to process. Then spacy handles the prompt formatting and postprocessing of the generated text. Example prompt and interaction is shown in Table 2. Batch inference is used to achieve efficient inference. In our setup we used 24 batchsize and 512 maximum generated tokens to balance performance and efficiency. Left padding was used for batching as Llama is a decoder-only language model.

3.4 Similarity Matching

Through Named Entity Recognition (NER), we identified many critical entities in the text, including locations, people, and organizations mentioned. These details often allow us to pinpoint events occurring within a specific time frame. This time window is crucial as the same location or person could be associated with multiple different events. We operate under the assumption that a user's Twitter message about an event appears only after the event has occurred, and the interest in this event typically diminishes over time unless a related event reignites interest, which would then be considered a separate event and matched accordingly. Thus, we have set a one-month time window for each tweet to search back for event information for matching purposes. However, this method is currently heuristic-based and will be revisited in our discussion section.

With the entity and timestamp data available, we attempt to match it with the Global Database of Events, Language, and Tone (GDELT) to find corresponding event information. The process involves setting the time range from one month before the timestamp of the first Twitter message to one month before the timestamp of the last. Then, each GDELT report link is converted into vector embedding via LangChain's OpenAI Embedding and stored in a vector database, along with other metadata about the GDELT event, such as participants, involved countries, organizations, and time. These vector-based embedding serve as indexing methods to facilitate rapid similarity searches.

For each Twitter message, we first filter the content within the specified time window based on metadata timestamps and the tweet's timestamp. Next, we convert the tweet's entities into a query and perform similarity matching with GDELT report links. During this process, the top-ranked links and their corresponding scores are displayed. We chose event report links for matching because most links in GDELT incorporate the report article's title as part of the link—for example, 'https://www.lifenews.com/2023/01/11/illinois-legislature-passes-radical-bill-to-kill-more-babies-in-abortions/'. However, this approach has its uncertainties; not all reports use this link format. In the discussion section, we will propose an alternative solution, which involves scraping the text content of the links for matching. Currently, this approach faces challenges as many GDELT links are inaccessible or have expired, which limits its effectiveness.

For all matching results and their corresponding scores, we tested various score thresholds to evaluate their effectiveness, and the outcomes are documented in §4.

4 EVALUATION

4.1 Data Collection

To evaluate the performance of our framework, we collected Tweets from January 2023 and GDELT events in the same timeframe. Since there are no existing datasets that can be used to evaluate our framework, we had to perform annotation through human labor. For a given Tweet, we identify its associated event(s) in the GDELT database, and label the Tweet with the URLs and named entities in the Tweet that are relevant to the event. For example, given this Tweet:

> Why did 97 Democrats vote against a ban on selling American oil to China?

This Tweet suggests that there was a vote conducted by the House and the theme of the vote was whether to ban sales of American oil to China. Through digging in the GDELT database, we found the news article titled "House votes to ban sales from Strategic Petroleum Reserve to China" [21]. We then label the Tweet with the URL to the article and the named entities: "Democrats", "American", and "China".

Since human annotation is extremely time-consuming, we labeled 40 Tweets as our test set.

Table 2: Example of Expected and Actual Generation

Section	Content
Input	You are an expert Named Entity Recognition (NER) system. Your task is to accept Text as input and extract named entities. Entities must have one of the following labels: LOCATION, NATIONALITY, ORGANISATION, PERSON. If a span is not an entity label it: '==NONE=='. Q: Given the paragraph below, identify a list of entities, and for each entry explain why it is or is not an entity: Paragraph: Jack and Jill went up the hill. Answer: 1. Jack | True | PERSON | is the name of a person 2. Jill | True | PERSON | is the name of a person 3. went up | False | ==NONE== | is a verb 4. hill | True | LOCATION | is a location Paragraph: ukrainian government issued official blacklist americans brazilians others accused kremlin assets due opposition us proxy war including tulsi gabbard tucker lula da silva rand paul john mearsheimer Answer:
Expected Output	1. The | False | ==NONE== | is a determiner 2. issued an official blacklist | False | ==NONE== | is an action 3. of | False | ==NONE== | is a preposition 4. Brazilians | True | NATIONALITY | is a nationality 5. and others | False | ==NONE== | not specific 6. it | False | ==NONE== | is a pronoun 7. accused | False | ==NONE== | is a verb 8. of | False | ==NONE== | is a preposition 9. being | False | ==NONE== | is a verb 10. assets | False | ==NONE== | is a noun 11. due to our opposition to the | False | ==NONE== | opposition is not entity 12. US proxy war | True | ==None== | is a noun phrase 13. including | False | ==NONE== | is a verb 14. Tulsi Gabbard | True | PERSON | is the name of a person.
Actual Generation	1. ukrainian | True | NATIONALITY | refers to a country (ukraine) 2. government | True | ORGANISATION | refers to a governing body 3. americans | True | NATIONALITY | refers to a group of people 4. brazilians | True | NATIONALITY | refers to a group of people 5. others | False | ==NONE== | refers to an undefined group 6. accused | False | ==NONE== | is a verb 7. kremlin | True | ORGANISATION | refers to the Kremlin, a building in Moscow 8. assets | False | ==NONE== | is a noun referring to property 9. due | False | ==NONE== | is a preposition 10. opposition | True | ORGANISATION | refers to a group opposing something 11. us | True | NATIONALITY | refers to the United States 12. proxy | False | ==NONE== | is a noun referring to an agent 13. war | False | ==NONE== | is a noun referring to a conflict 14. including | False | ==NONE== | is a preposition 15. tulsi gabbard | True | PERSON | refers to a particular person 16. tucker | False | ==NONE== | is a name without a defining label 17. lula da silva | True | PERSON | refers to a particular person 18. rand paul | True | PERSON | refers to a particular person 19. john mearsheimer | True | PERSON | refers to a particular person """ from transformers import AutoModelForTokenClassification, AutoTokenizer # Load the pre-trained model and tokenizer model = AutoModelForTokenClassification.from_pretrained("atis-bert-named-entity-recognition") tokenizer = AutoTokenizer.from_pretrained("atis-bert-named-entity-recognition") # Define the entity labels entity_labels = ["LOCATION", "NATIONALITY", "ORGANISATION", "PERSON"] def recognize_entities(text): inputs = tokenizer.encode_plus(text, add_special_tokens=True, max_length=512, return_attention_mask=True, return_tensors='pt', truncation=True, padding='max_length') outputs = model(**inputs) results = [] for i, output in enumerate(outputs): for idx, score_vector in enumerate(output): for idx2, score in enumerate(score_vector): score ='
Pipeline Extractions	[('ukrainian', 'NATIONALITY'), ('government', 'ORGANISATION'), ('americans', 'NATIONALITY'), ('brazilians', 'NATIONALITY'), ('kremlin', 'ORGANISATION'), ('opposition', 'ORGANISATION'), ('us', 'NATIONALITY'), ('tulsi gabbard', 'PERSON'), ('lula da silva', 'PERSON'), ('rand paul', 'PERSON'), ('john mearsheimer', 'PERSON')]

4.2 Evaluation

We employ the F_1 metric to evaluate the effectiveness of our framework. The F_1 Score is calculated as in the following formula:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

We evaluated our framework with different similarity thresholds and report our result in Figure 5. The best F_1 Score we achieved was **0.75**, with a recall of **0.9474** and a precision of **0.6207**.

Compare to the state-of-the-art technology on Twitter[27], our model outperforms their result in event-detection tasks by around 20 percent increase in precision. It marks that our method could perform well on political-related Twitter Datasets. The performance on more general tasks still needs to be evaluated in the future.

5 DISCUSSION

5.1 Dataset

During the annotation of the test dataset, we identified several points that need to be addressed.

- Even though some Tweets contain named entities, the tweets are not necessarily associated with events. For example, here is a popular Tweet we encountered during annotation:

  RT @elonmusk If you don't think there's at least a tiny chance you're an NPC ... you're an NPC

  This Tweet is clearly not associated with any events. However, "@elonmusk" could be identified as a named entity, so we would need to handle these situations carefully.
- Some Tweets contain named entities that are not related to the actual event. For example, consider the following Tweet:

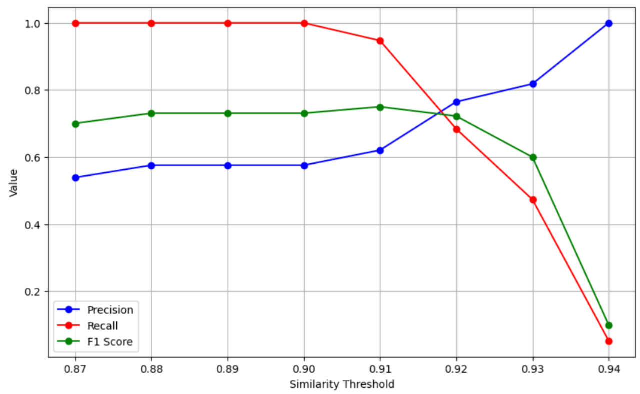

Figure 5: Precision, Recall and F1 Score at different Similarity Thresholds

> Why does Greta Thunberg never protest in China?
> This Tweet suggests that Greta Thunberg protested somewhere, but not in China. In fact, the event associated with this tweet is about a protest in Germany. This suggests that we should emphasize the timestamp more than the named entities. We could also introduce negative named entities to potentially improve both the precision and the recall of our event detection.

5.2 Time Window

In this project, we have simplistically set a one-month time window for events based on consensus. However, the actual duration highly depends on factors such as the size of the event, its impact, and the occurrence of related events. Discussions about an event can vary widely, lasting from a day to a year. In subsequent research, we plan to design various models to assess the general time window for events and to predict adjustments to the event's time window based on factors like size and participants.

By implementing a more universal time window as discussed, we could not only reduce the computational costs associated with processing the entire dataset but also enhance the robustness of our model. This would allow it to perform well not only on more general datasets but also on more detailed (fine-grained) datasets.

5.3 Similarity Threshold

To reach the best result, an optimal similarity threshold is required for the whole dataset. If the threshold is set too low, it allows for the retrieval of a large quantity of events, potentially encompassing nearly all events. However, many of these events may be inaccurately identified. Conversely, setting a high threshold can yield a high accuracy in event detection, but the overall recall rate or event coverage rate becomes markedly low, compromising the model's robustness. In our dataset, we found that a balanced threshold between these extremes achieves the best F-1 score.

In future research, we plan to further investigate the optimal threshold value. Given that each dataset involves different events, individuals, and locations, a substantial amount of data will need to be analyzed to establish a relatively balanced threshold. This iterative process aims to refine the balance between precision and recall, optimizing the effectiveness of event detection in diverse datasets.

5.4 GDELT Data Retrieval

In this project, we utilized the links from GDELT reports as the material for matching with Twitter entities. This approach is based on the heuristic that many news articles and reports tend to include the title of the article as part of their web links. However, this assumption is not always correct; there are also instances where web pages use unique IDs to mark links, such as 'https://apnews.com/cb608d31f591fd06ec2fd753f122f6cb', from which we cannot directly infer the embedded information via the URL content.

Furthermore, it is common for article titles not to fully incorporate the names of individuals involved in the events. Most titles opt for hook words or impressive expressions, and specific content details might be obscured. For instance, a report about the UK's support for Ukraine might not mention the UK Prime Minister in the title, whereas the Twitter message might only include words like 'Prime Minister' and 'support'. This can lead to misses and false matches during similarity matching. Thus, accessing the full content of the articles would be ideal.

However, we encountered numerous issues in attempting to scrape this content, such as many links recorded on GDELT being inaccessible or expired. Additionally, anti-scraping measures and advertisements on websites significantly impede our automated scraping efforts. If we could find an effective way to mine textual information from web pages, we would have a richer set of data for similarity searches, thereby enhancing the overall effectiveness and robustness of our program.

5.5 LLM for NER

As shown in Table 2 there are two main vulnerability in the method:

- When batch inference is used, the model does not produce end-of-sequence token until token limit is reached. Such approach is inefficient, and more importantly, could hurt the extraction performance. The model could generate another round of user-agent interaction that is similar to the prompted one, causing false detections in the outpur. Therefore, an adaptive token limit and content-length-aware batching is needed for better performance and efficiency.
- The current pipeline's performance highly relies on the quality of the prompt. The false detections (government, opposition) shown in the table could be caused by that the ORGANIZATION class being too generic, and the lack of examples for this class. However, a detailed prompt that incorporates multiple materialization of multiple class of entities could be hard to construct succinctly. As context length is a valuable resource in LLM inference, a prompt too long is also not desirable. A study on the trade-off between long prompt and efficient inference would be a next step.

Meanwhile, given the strong capacity of LLMs in interpreting complex text data, the NER prompt we used could be further extended toward a chain of extraction that combines entity extraction and event extraction, which eventually leads to more distinctive information for clustering.

6 CONCLUSION

In conclusion, our model introduces a novel method for rapidly and accurately filtering events through embedding and named entity recognition. This approach differs from the state-of-the-art Event Detection with Clustering of Wavelet-based Signals (EDCoW)[27] strategy, allowing for more precise event discovery in shorter periods. However, our model is primarily suited for coarse-grained event detection cases and may not perform as well in identifying highly detailed events or events that do not involve a significant number of entities.

Furthermore, we believe that further adjustments to the time window, similarity threshold, and deeper exploration of GDELT content could enhance the robustness of our model and improve its performance in finer-grained tasks. Additionally, our model avoids issues encountered with Word Co-occurrence models like EDCoW, where results can be ambiguous and do not directly lead to conclusive events. By providing an intuitive output of event-related report links, our model facilitates easier assessment of event relevance, offering better prospects for downstream LLM applications and greater scalability.

REFERENCES

[1] [n. d.]. casperhansen/llama-3-8b-instruct-awq · Hugging Face. https://huggingface.co/casperhansen/llama-3-8b-instruct-awq. (Accessed on 05/10/2024).

[2] [n. d.]. Large Language Models · spaCy Usage Documentation. https://spacy.io/usage/large-language-models. (Accessed on 05/10/2024).

[3] Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-Based *n*-gram Models of Natural Language. *Computational Linguistics* 18, 4 (1992), 467–480. https://aclanthology.org/J92-4003

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. arXiv:2005.14165 [cs.CL]

[5] Ciprian Chelba, Mohammad Norouzi, and Samy Bengio. 2017. N-gram Language Modeling using Recurrent Neural Network Estimation. arXiv:1703.10724 [cs.CL]

[6] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. arXiv:2003.10555 [cs.CL]

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[8] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. arXiv:1508.01991 [cs.CL]

[9] Yizhu Jiao, Sha Li, Yiqing Xie, Ming Zhong, Heng Ji, and Jiawei Han. 2022. Open-Vocabulary Argument Role Prediction for Event Extraction. arXiv:2211.01577 [cs.CL]

[10] Guillaume Lample and Alexis Conneau. 2019. Cross-lingual Language Model Pretraining. arXiv:1901.07291 [cs.CL]

[11] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. arXiv:1909.11942 [cs.CL]

[12] Kalev H. Leetaru. 2023. The GDELT Project. https://www.gdeltproject.org/

[13] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. arXiv:1910.13461 [cs.CL]

[14] Qian Li, Jianxin Li, Jiawei Sheng, Shiyao Cui, Jia Wu, Yiming Hei, Hao Peng, Shu Guo, Lihong Wang, Amin Beheshti, and Philip S. Yu. 2022. A Survey on Deep Learning Event Extraction: Approaches and Applications. arXiv:2107.02126 [cs.CL]

[15] Jiacheng Liu, Sewon Min, Luke Zettlemoyer, Yejin Choi, and Hannaneh Hajishirzi. 2024. Infini-gram: Scaling Unbounded n-gram Language Models to a Trillion Tokens. arXiv:2401.17377 [cs.CL]

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs.CL]

[17] Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable Sequence-to-Structure Generation for End-to-end Event Extraction. arXiv:2106.09232 [cs.CL]

[18] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bidirectional LSTM-CNNs-CRF. arXiv:1603.01354 [cs.LG]

[19] Malik Mateen, Muhammad Azhar Iqbal, Muhammad Aleem, and Muhammad Arshad Islam. 2017. A hybrid approach for spam detection for Twitter. In *2017 14th international Bhurban conference on applied sciences and technology (IBCAST)*. IEEE, 466–471.

[20] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. arXiv:2203.02155 [cs.CL]

[21] Lauren Peller. 2023. House votes to ban sales from Strategic Petroleum Reserve to China. *ABCNews* (2023). https://abcnews.go.com/Politics/house-votes-ban-sales-strategic-petroleum-reserve-china/story?id=96397178

[22] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683 [cs.LG]

[23] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[24] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL]

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[26] Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. 2023. DeepStruct: Pretraining of Language Models for Structure Prediction. arXiv:2205.10475 [cs.CL]

[27] Jianshu Weng and Bu-Sung Lee. 2011. Event detection in twitter. In *Proceedings of the international aaai conference on web and social media*, Vol. 5. 401–408.

[28] Tingmin Wu, Sheng Wen, Yang Xiang, and Wanlei Zhou. 2018. Twitter spam detection: Survey of new approaches and comparative study. *Computers & Security* 76 (2018), 265–284.

[29] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. arXiv:2303.18223 [cs.CL]

A INDIVIDUAL CONTRIBUTIONS

Kaiyue Zhang

Kaiyue Zhang was involved in coming up with the framework. Kaiyue Zhang implemented the code for data cleansing on the raw dataset. Kaiyue Zhang examined the raw Twitter dataset and annotated the test dataset for evaluation.

Bohan Liu

Bohan Liu did examined dataset statistics and scripted data transformations for Data Cleaning. Bohan Liu implemented new functionalities in the Spacy-LLM library to apply latest Llam3-8B-Instruct model and achieve efficient batch inference. Bohan Liu scripted the NER inference script and conducted the experiment on NER with LLM. Bohan Liu integrated the NER result with the twitter dataset for further processing and extractions.

Dayou Wu

Dayou Wu proposed the architechture of the whole event detection model. Dayou Wu generated the Twitter dataset for evaluation and implemented the part of Similarity Search for entity matching with GDELT data. Dayou Wu evaluated the result on several different thresholds and visualized the results.

Received 10 May 2024