2023 3rd Asian Conference on Innovation in Technology (ASIANCON)
Pune, India. Aug 25-27, 2023

Automated Landslide Event Archive Generation from News Articles

Subhayan Bhattacharya
Dept. Of Comp. Sc. and Eng.
Jadavpur University
Kolkata, India
subhayanb.cse.rs@jadavpuruniversity.in

Rishideep Chatterjee
Dept. of Comp. Sc. and Eng.
RCC Institute of Information Technology
Kolkata, India
rishi1a2@gmail.com

Sarbani Roy
Dept. Of Comp. Sc. and Eng.
Jadavpur University
Kolkata, India
sarbani.roy@jadavpuruniversity.in

***Abstract*—Landslides are a type of natural disaster characterised by ground movements in the form of rockfalls, debris falls, mud-flow and slope failures. It is one of the primary natural disaster and has been studied since as early as 1807[1]. Disaster management is a stream of research in Information Technology, Political Science, Legal and Ethical Studies, Geology and so on which has garnered a lot of attention in the recent past. A primary dependency of such research is on archival data related to landslides. As a part of this work, a novel framework for generating landslide archive is proposed. The source of this data archive are online news articles from dependable news publishers. The data archive contains comprehensive information about the date, location, severity, and meteorological causes for landslides. The end-to-end framework contains a user-friendly web interface for running easy, concise, and customisable queries with near real-time responses.**

***Index Terms*—landslide, machine learning, data mining, archive generation, event identification**

I. Introduction

Landslide management has been researched for a few decades now. However, with an increase in the frequency and magnitude of landslides, due to factors like earthquakes, heavy rainfall, deforestation, urbanisation and so on, there has been a spike in the need for feasible and efficient landslide detection and management systems.

There are four main aspects of landslide management, mitigation, preparedness, response, and rehabilitation. All of these are dependent on the study of archival data, in order to identify patterns for causes of landslides, to identify hot spots for landslides, and to plan quick response and rehabilitation. Unfortunately, there is a lack of such archival data that provides extensive and concise information that can be used for such studies.

A primary challenge of building an archive is the variety of information sources and format. Natural disasters are reported through multiple channels, such as online social networks, news articles and coverage, and government reports. There are challenges in processing information from each of these sources. Online social network information is crowd sourced, and the authenticity and completeness of the information is questionable. Government reports contain a comprehensive report, but is published with a substantial delay from the original event, can be biased, and the retrieval is difficult to automate. News articles also provide incomplete data, but is inherently unbiased, and has a much smaller delay, if any, from the original event. Thus, for the purpose of this work, news articles have been considered as the primary source of information, along with some open source APIs for supporting information.

This paper provides a framework for automated landslide archive generation, based on news articles. The end-to-end framework presented in this paper has a user-friendly web interface that allows running queries based on time and specific location of the landslide, and provides the information in a visualisable format.

[1]https://shorturl.at/nvxAJ

II. Literature Review

Event Detection is a well-researched topic and there exists ample literature on this. Event can be detected from different types of data, such as text, image, video, time-series and so on, collected from multiple sources such as online social networks, news reports, government reports and so on.

The paper by Sayyadi et. al. [1] detect events from news articles based on a label clustering method. Tong et. al. [2] identify events from news articles by combining both the textual as well as image data, taking advantage of the multimodal nature of news articles. News articles are available in vernacular languages also, and processing such news articles for detecting events is an interesting research effort. Balouchzahi and Shashirekha [3] provide a linear support vector machine based approach for detecting events from news articles in Indian languages. Similar efforts have been made in other Non-English languages as well, such as detecting protest event from German news articles in the paper by Wiedemann et. al. [4]. The work of Ganesan et. al. [5] uses geo-tagged data from Twitter for event detection with precise location information. Hasan et. al. [6] proposes a framework based on Twitter for real time detection of both major and minor events based on an incremental clustering method.

There are different types of events that can be detected as well, from different sources such as online social networks and news articles. Sims et. al. [7] focuses on identifying events that has happened within a work of literature. Wu et. al. [8] focuses on identifying earthquake event from text features collected from

979-8-3503-0228-8/23/$31.00 ©2023 IEEE

news groups. Abdelhaq et. al. [9] depends on burst features from twitter data to detect local events such as emergency situations and public events. There also exists open domain event detection frameworks such as the work by Chen et. al. [10].

There are groups of work focused on identifying natural disaster events specifically as well. Imran et. al. [11] highlights the opportunities and challenges of identifying natural disasters utilising the multi-modal data available on online social networks. Zhang et. al. [12] focuses on early warning and public information during natural disaster events using online social networks as a platform. A method for detecting the severity of natural disasters using online social networks is provided by Kankanamge et. al. [13]. Filippo [14] proposes a deep learning based technique for landslide event detection. A semi-automatic method based on some human intervention and google earth is proposed by Subiyantoro et. al. [15]. Ofli et. al. [16] proposes yet another landslide detection method based on image processing.

However, there exists very limited literature on automated archive creation for different natural disasters. Yoshioka et. al. [17] proposes one such automated method for generating archives of natural disaster based on news articles. The recent history of Tsunamis and its importance in reducing risks of future Tsunamis has been discussed in the work by Imamura et. al. [18]. Thus, this paper contributes in filling this gap of automated archive generation frameworks that is tested using the landslide event, but can be extended to any natural disaster in future.

III. Methodology

This section provides the methodology used to generate the news article archive based on a given keyword. Fig. 1 represents the schematic flow diagram for the methodology proposed in this paper.

Algorithm 1 provides the algorithm for the driver method. The input to Algorithm 1 is the keyword, which for the purpose of this paper is "landslide", and the output of the algorithm is a set of news articles parsed and filtered for the keyword. This output is updated in a database to generate the archive.

Line 1 of Algorithm 1 uses the gnews library to search for all news articles related to a given keyword. Line 2 to 9 iterates over the list of articles, parsing each article individually to extract the metadata from the news article. Line 3 parses the raw HTML content of the article to get the title, publication date, and content of the article. Line 4 uses the BERT QA model with some pre-configured questions to extract the exact date and location of the landslide, and how many people were affected. Line 5 uses the Stanford NER tagger to identify the location from the title and content of the news article. Line 6 uses a pre-trained model to classify the severity of the event. Line 7 uses regular expressions to identify the number of people affected in the landslide. Line 8 gets the summary of the news article using LSA Summarizer. Line 9 gets the date of the landslide from the news article content using dateparser library. Line 10 combines the results obtained from BERT QA model and other methods. Line 11 uses the latitude, longitude, and event date to get the meteorological data to complete the metadata.

Algorithm 1: Create Archive of NEWS Articles on Landslide

Input : keyword
Output: a list of filtered and parsed News article metadata

1 $articles \leftarrow get\_articles\_using\_gnews(keyword)$;
2 **for** $article \in articles$ **do**
3 $\quad title, pub\_date, content \leftarrow get\_content(article)$;
4 $\quad bert\_response \leftarrow get\_details\_bert\_qa\_model(content)$;
5 $\quad location \leftarrow get\_location(title, content)$;
6 $\quad severity \leftarrow get\_severity(content)$;
7 $\quad affected \leftarrow get\_affected(title, content)$;
8 $\quad summary \leftarrow get\_summary(content)$;
9 $\quad date \leftarrow get\_date(content)$;
10 $\quad date, location, affected, lat, long \leftarrow combine\_result(bert\_response, location, affected, date)$;
11 $\quad met\_data \leftarrow get\_metereological\_data(lat, long, date)$
12 **end**

Algorithm 2: *get_content*

Input : news article
Output: title, publication date and content of the article

1 $title \leftarrow article['title']$;
2 $pub\_date \leftarrow article['date']$;
3 $link \leftarrow article['link']$;
4 $html\_data \leftarrow requests.get(link)$;
5 $content \leftarrow$ all paragraphs from $html\_data$ using BeautifulSoup ;
6 **return** $title, pub\_date, content$

Algorithm 2 parses the article to retrieve key information. It is an expansion of Line 3 of Algorithm 1. In Line 1, 2, and 3 of Algorithm 2 the title, publication date and hyperlink to the online article are retrieved from the article respectively. In Line 4, retrieves the HTML content of the hyperlink from Line 3. In Line 5, the BeautifulSoup4 package of python is used to get all the paragraphs from the HTML content from Line 4 as content. In Line 5, the title, publication date and content are returned.

Algorithm 3 is an expansion of Line 4 of Algorithm 1. The BERT QA Model takes a context and a set of questions and answers those questions based on the given context. In this case, the context is the content section of the news articles retrieved in Algorithm 2. The questions are "What event

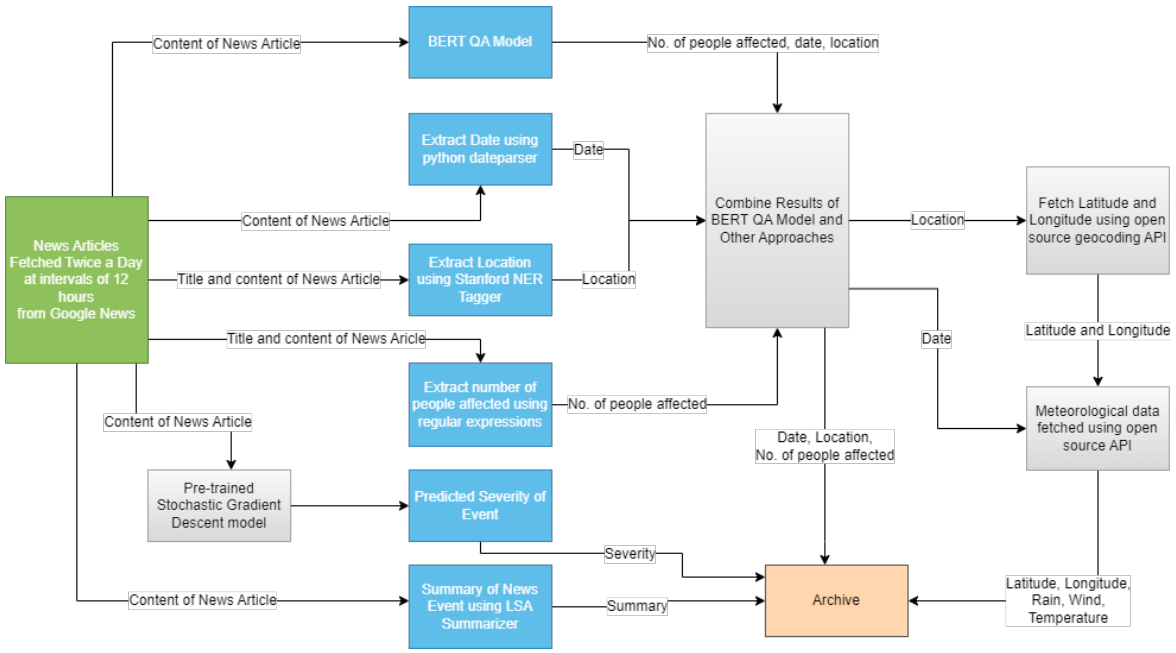


Fig. 1: Schematic Representation of Methodology

Algorithm 3: *get_details_bert_qa_model*

Input : content section of news article
Output: date, location, and number of affected people

1 $model \leftarrow$pre-trained model bert-large-uncased-whole-word-masking-finetuned-squad ;
2 $questions \leftarrow$ pre-configured set of questions ;
3 **for** $question \in questions$ **do**
4 $\quad ans \leftarrow$ answer predicted by $model$;
5 $\quad$ **if** ans *not found* **then**
6 $\quad\quad ans \leftarrow$ "N/A"
7 $\quad$ **end**
8 **end**
9 **return** $answers$

occurred?", "What was the exact date of the landslide?", "What was the exact location of the landslide?", "How many people died in the landslide?", "How many people were injured in the landslide?", "How many people overall were affected by the landslide?", and "How severe was the landslide?". Algorithm 3 takes as input the content portion of the news article and outputs the date, location and number of affected people as answers to these questions. Line 1 loads a pre-trained model and Line 2 loads the questions mentioned above. Line 3 to 8 iterates over each question. Line 4 checks for the predicted answer using the model defined in Line 1. In Line 5 to 7, if an answer is not found in Line 4, then the answer is set to "N/A". Line 9 returns the answers.

For Line 5 of Algorithm 1, Stanford NER tagger is used to get words tagged as location. Initially, words from the title of the News article is searched for words tagged as "LOCATION". If such words are found, the last one of such words is returned. If no such word is found in the title, then the content of the news article is searched for words tagged as "LOCATION" and if such words are found, then the last of such words is returned. If no such word is found in either title or content of the news article, then an empty string is returned.

Algorithm 4 corresponds to Line 6 of Algorithm 1. It takes as input the content section of the news article and provides the predicted severity based on that content. A Stochastic Gradient Descent model is trained using manually annotated data to classify the content section of the news article into classes High, Low, and Uncertain. Other conventional machine learning models were considered for the purpose, but the performance of Stochastic Gradient Descent Model is better than other contemporaries. A detailed performance analysis of the models is provided in Section IV.

Algorithm 4: *get_severity*

Input : content section of the news article
Output: severity of the event

1 $model \leftarrow$ pre-trained model;
2 $predict \leftarrow$ prediction of severity based on $model$ and $content$;
3 **return** $severity$

Algorithm 5 provides the steps for obtaining the number of people affected in the event. The input to the algorithm is the title and content section of the news article. Line 1 initialises count to -1. Line 2 to 9 checks if there are any keywords related to death, such as die, died, deaths and so on, present in the title, and if yes, if there is the number of people who

Algorithm 5: *get_affected*

Input : content section and title of the news article
Output: number of people affected in the event

```
1  count ← −1 ;
2  if title contains any keyword related to death then
3      if title contains count of people died then
4          count ← number of people died
5      end
6      else
7          count ← "undetected number of people died"
8      end
9  end
10 else if title contains any keyword related to injury then
11     if title contains count of people injured then
12         count ← number of people injured
13     end
14     else
15         count ← "undetected number of people were injured"
16     end
17 end
18 else if content contains any keyword related to death then
19     if content contains count of people died then
20         count ← number of people died
21     end
22     else
23         count ← "undetected number of people died"
24     end
25 end
26 else if content contains any keyword related to injury then
27     if content contains count of people injured then
28         count ← number of people injured
29     end
30     else
31         count ← "undetected number of people were injured"
32     end
33 end
34 if no count is found then
35     count ← "not found"
36 end
37 return count
```

died present in the title. If not, Line 10 to 17 checks if there are any keywords related to injury, such as injured, injuries and so on present in title, and if yes, if there is the number of people who were injured present in the title. If not, the same steps are repeated for the content of the article in Line 18 to 25 and Line 26 to 33 respectively. The search is done using regular expressions. In line 34 to 36, if no such information is found in either title or content, count is set to "not found". Line 37 returns the detected count.

For Line 8 of Algorithm 1, a text parser and LSA text summariser is used. It takes as input the content of the news article and returns the summary of the article in 2 lines.

Algorithm 6: *get_date*

Input : content section of the news article
Output: date of the event

```
1  remove punctuation from content;
2  for each word w ∈ content do
3      date ← w parsed using dateparser library ;
4      if date is valid and in the past and not more than a month old then
5          return date
6      end
7  end
8  return ""
```

Algorithm 6 corresponds to Line 9 of Algorithm 1. It takes as input the content of the news article and returns the date of the event if it is found, otherwise it returns an empty string in Line 8. In Line 1, all punctuation and special symbols are removed from content. Line 2 to 7 iterates over each word of the content. In Line 3, the word is parsed using the dateparser library of python and stored in date. In Line 4 to 6, if the date is valid and in the past, and it is not older than a month, the date is returned. This is to ensure that if there are other irrelevant dates in the text, they can be avoided.

Algorithm 7 combines the results obtained from BERT QA Model and through other methods. It takes as input the responses from BERT QA model, the location detected in Algorithm 4 and the number of people affected detected in Algorithm 5. The algorithm outputs the event date, location of the event, number of people affected in the event, and the latitude and longitude of the location. Line 1 gets the event date from the response of the BERT QA model. In Line 2, location is set to the output of Algorithm 4. In Line 3 to 5, if location is empty, it is set to the response of the BERT QA model. In Line 6, affected is set to the output of Algorithm 5. In Line 7 to 9, if affected is "not found", it is set to the response of the BERT QA model. Line 10 uses an open source geocoding API [2] to get the latitude and longitude of the location. Line 11 returns the event date, location of the event, number of people affected in the event, and the latitude and longitude of the location.

[2]https://positionstack.com/documentation

Algorithm 7: *combine_results*

Input : output of algorithm 3, algorithm 4, and algorithm 5

Output: date, location, number of people affected, latitude and longitude of the event

$date \leftarrow$ date from BERT Question Answer model responses;
$location \leftarrow$ location from Algorithm 4 ;
if *$location$ is empty* **then**
    $location \leftarrow$ location from BERT Question Answer model responses
end
$affected \leftarrow$ number of people affected from Algorithm 5;
if *$affected$ is "not found"* **then**
    $affected \leftarrow$ number of people affected from BERT Question Answer model responses
end
$latitude, longitude \leftarrow$ geocoded latitude and longitude from $location$ using open source geocoding API ;
return $date, location, affected, latitude, longitude$

For Line 11 of Algorithm 1, an open source weather API [3] is used to fetch meteorological data for the event date and location (latitude and longitude). From the meteorological data fetched using the API, the average temperature, total rain, and maximum wind speed are returned.

Once all the information is obtained using Algorithm 1 through 7, the data is then saved into a sqlite database. The query contains the type of event (limited to landslide), the time window of the event (granularity of year) and the location of the event and filters the results from the database using these parameters and renders to UI.

IV. Results

This paper has proposed a framework for creating an archive of landslide events from online news reports. The proposed framework combines BERT QA model, novel methods, and open source APIs in extracting metadata from news articles to form the archive. The BERT QA model is configured with questions about the date of the landslide, location of the landslide, and the number of people affected in the landslide. A pre-trained BERT QA model is used to make the detection.

For predicting the severity of the landslide event, 800+ news articles were annotated manually into classes High, Low, and Uncertain. Six classifiers (namely, Multi Layer Perceptron Classifier, Decision Tree Classifier, Random Forest Classifier, Support Vector Classifier, K Nearest Neighbour Classifier, and Stochastic Gradient Descent Classifier) were considered. The training data was split into 70-15-15 for training, validation, and testing respectively. Table II shows the different metrics (Accuracy, Precision, Recall, F1 Score, Area Under Curve (AUC) Score, Execution time, and Memory Utilisation) of the six classifiers considered. Fig. 2 shows the confusion matrices for the same classifiers, where the classes High, Low, and Uncertain are represented as 0, 1, and 2 respectively. Fig. 3 shows the ROC (Receiver Operating Characteristic) curves for each of the classifiers for each of the three classes using one-vs-rest classifications.

As can be seen from Table II, Fig. 2, and Fig. 3, Stochastic Gradient Descent (SGD) classifier has the most promising results across all metrics, except Precision (Support Vector Machine has a marginally higher value) and Execution Time (Decision Tree has a lower value). Thus, for predicting the severity of landslides using the content of the news articles in real time, Stochastic Gradient Classifier has been used.

This paper also presents approaches for identifying date, location, and number of people affected from the news article using BERT QA Model. The same are identified using python dateparser, Stanford NER tagger, and regular expressions respectively as well. The number of people affected is identified using regular expressions. The results of these approaches and the responses obtained from BERT QA are shown in Table II. As can be seen from the table, the results from both approaches are comparable.

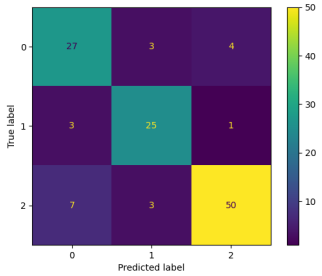
Fig. 4 shows the extracted metadata from the original news article using the methodology described in Section III. As can be seen from the figure, the extracted metadata captures all information available for the event from the landslide and accentuates it with additional information sourced from different open source APIs. This comprehensive metadata archive helps identify hotspots for landslides, helps identify patterns in contributing factors such as rain, temperature and wind speed and captures the progression of such events in general as well.

Fig. 6a shows the landing page of the web portal with user friendly UI/UX to help run easy and customised queries for fetching entries from the archive. Keyword and Location are compulsory inputs. Timeline is an optional input. By default, for all events matching the query parameters, the title of the news article, the date of the landslide and the location of the landslide are displayed. Additionally, the other information like summary, predicted severity, number of people affected, latitude and longitude, and meteorological data can be shown. Queries are run based on the inputs provided. A sample query and it's output are shown in Fig. 5 .The timeline generated is shown in Fig. 6b.

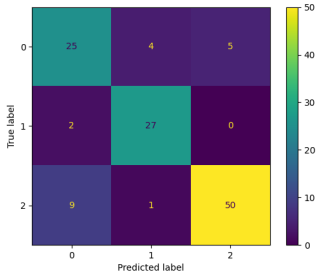
V. Conclusion

As a part of this paper, a novel framework is presented that can automatically and continuously scrape news articles from google news based on a given keyword (landslide), and parse and filter the news article to generate an archive of the comprehensive metadata. The paper also provides a web-based portal to conveniently and efficiently query the event archive based on location and time period. This data archive

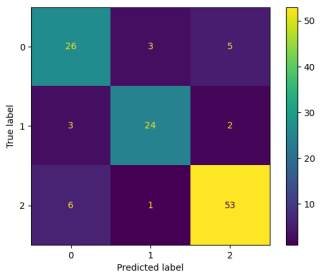
[3] https://www.weatherapi.com/api-explorer.aspx



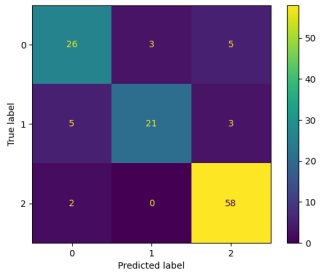
(a) MLP Confusion Matrix



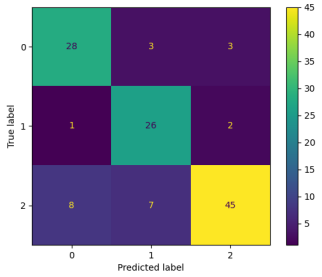
(b) Decision Tree Confusion Matrix



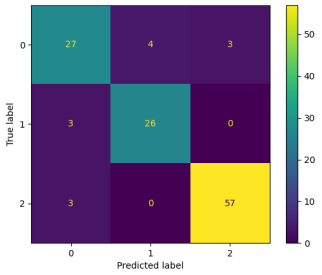
(c) Random Forest Confusion Matrix



(d) SVM Confusion Matrix

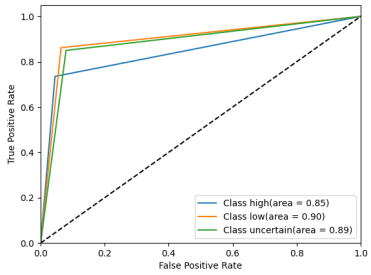


(e) KNN Confusion Matrix

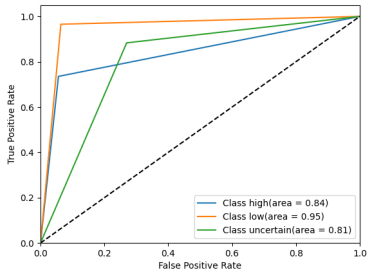


(f) SGD Confusion Matrix

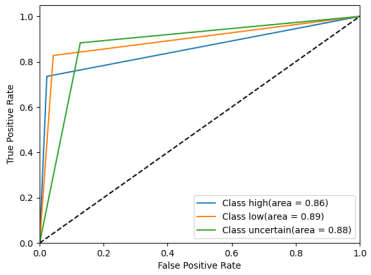
Fig. 2: Confusion Matrices



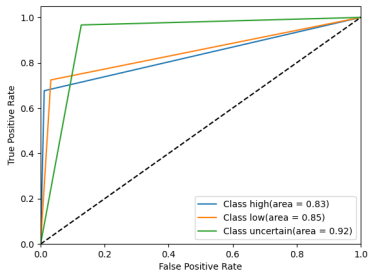
(a) Multi Layer Perceptron ROC and AUC



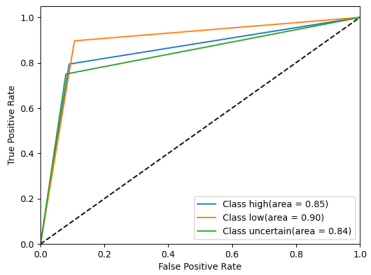
(b) Decision Tree ROC and AUC



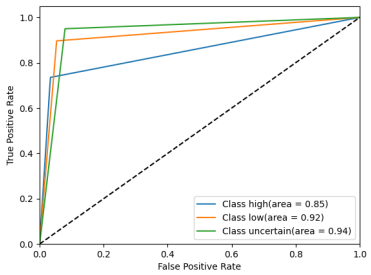
(c) Random Forest ROC and AUC



(d) SVM ROC and AUC



(e) KNN ROC and AUC



(f) SGD ROC and AUC

Fig. 3: ROC and AUC

TABLE I: Comparison of different classifiers tested for severity model training

Classifier	Accuracy	Precision	Recall	F1 Score	AUC Score	Execution Time	Memory Utilisation
Multi Layer Perceptron (MLP)	81.31	87.26	82.11	84.47	0.87	22802.441ms	164.59 MB
Decision Tree	73.98	79.38	86.17	82.32	0.86	1589.596ms	163.91 MB
Random Forest	82.11	88.18	82.92	85.24	0.87	3301.013ms	165.83 MB
Support Vector Machine (SVM)	82.92	89.98	82.92	85.51	0.86	1725.356ms	164.96 MB
K Nearest Neighbout (KNN)	79.67	82.25	79.67	80.41	0.86	1702.560ms	165.68 MB
Stochastic Gradient Descent (SGD)	85.36	89.31	87.81	88.31	0.91	1631.551ms	163.47 MB

TABLE II: Comparison of BERT QA model and Other methods

Content Excerpt	Parameter	Bert QA Response	Result from Other Methods
killed more than 35	*Number of people affected*	More than 35 people have died due to the landslides	Casualties: 35
after severe rainfall at Barra do Sahy, in Sao Sebastiao, Brazil	*Location*	Sao sebastiao	Brazil
More than 120 people were killed in devastating floods and landslides	*Number of people affected*	More than 120	Casualties: 120
Most of the casualties were recorded in the west and north of Rwanda	*Location*	Rwanda	Rwanda
a landslide took place on april 20, 2023, near the chadwick farms	*Date*	April 20 , 2023	2023-04-20
landslides partially buried a house in bumasugu village in buluganya sub county, bulambuli district on may 25, 2023	*Date*	May 25 , 2023	2023-05-25

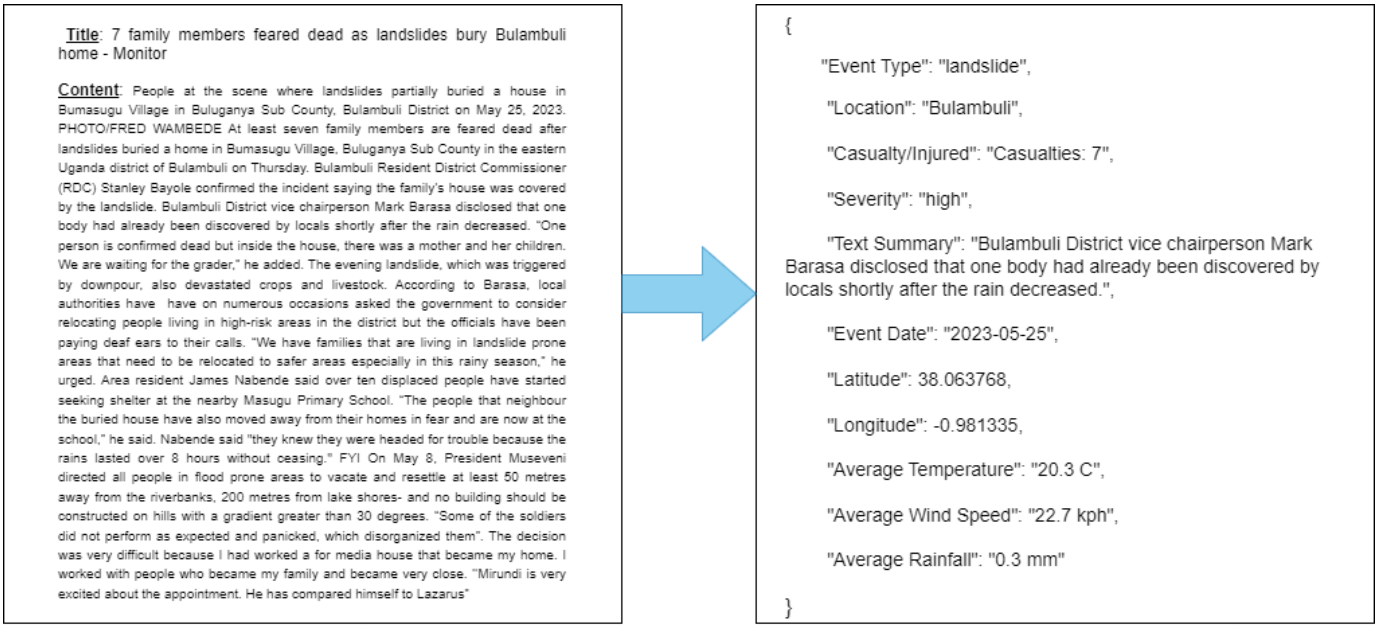


Fig. 4: Transformation of News Article to Metadata

and query mechanism can be further used to identify patterns in similar events, and early detect or predict such landslide events in future. The framework can be easily extended for other events and keywords as well. The primary constraint of this work is missing data from news articles. Efficient methods of recovering missing data from news articles can also be investigated in the future.

REFERENCES

[1] H. Sayyadi, A. Sahraei, and H. Abolhassani, "Event detection from news articles," in *Advances in Computer Science and Engineering: 13th International CSI Computer Conference, CSICC 2008 Kish Island, Iran, March 9-11, 2008 Revised Selected Papers*. Springer, 2009, pp. 981–984.

[2] M. Tong, S. Wang, Y. Cao, B. Xu, J. Li, L. Hou, and T.-S. Chua, "Image enhanced event detection in news articles," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 9040–9047.

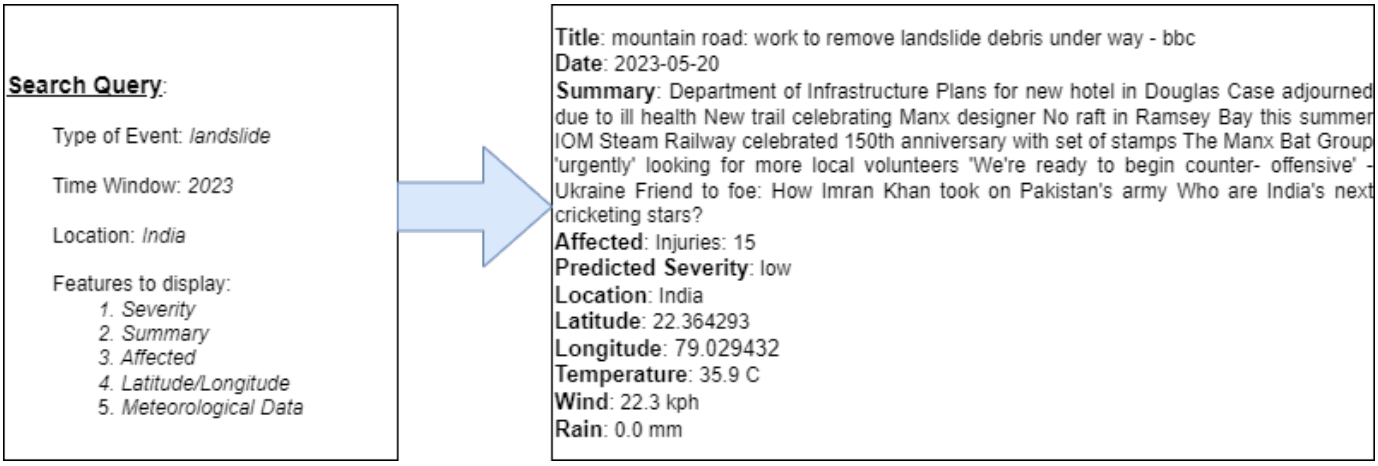
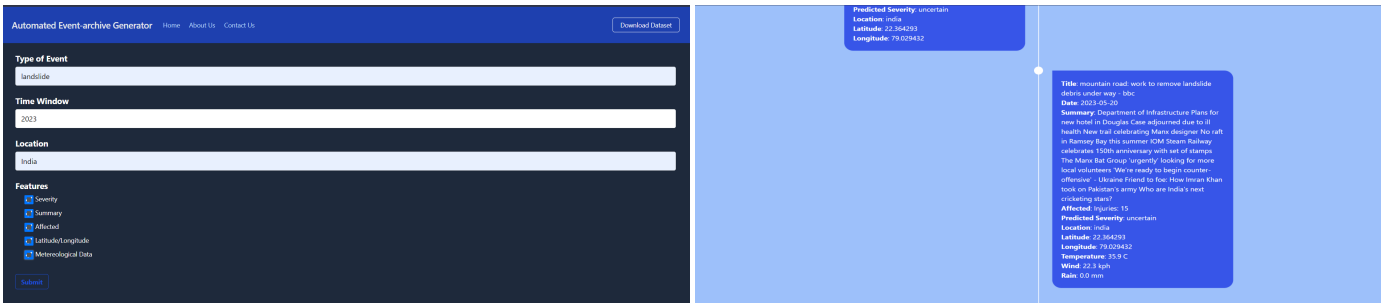


Fig. 5: Archive Query Example



(a) Landing Page

(b) Generated Timeline

Fig. 6: Archive Query Web Portal

[3] F. Balouchzahi and H. Shashirekha, "An approach for event detection from news in indian languages using linear svc." in *FIRE (Working Notes)*, 2020, pp. 829–834.

[4] G. Wiedemann, J. M. Dollbaum, S. Haunss, P. Daphi, and L. D. Meier, "A generalized approach to protest event detection in german local news," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, 2022, pp. 3883–3891.

[5] S. Ganesan, P. Rachana, D. Maharjan, S. Panth, and A. P. Kurmi, "Event detection using geo-tagged twitter information," in *2022 International Conference for Advancement in Technology (ICONAT)*. IEEE, 2022, pp. 1–5.

[6] M. Hasan, M. A. Orgun, and R. Schwitter, "Real-time event detection from the twitter data stream using the twitternews+ framework," *Information Processing & Management*, vol. 56, no. 3, pp. 1146–1165, 2019.

[7] M. Sims, J. H. Park, and D. Bamman, "Literary event detection," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3623–3634.

[8] Y. Wu, Y. Lin, Z. Zhou, D. C. Bolton, J. Liu, and P. Johnson, "Deepdetect: A cascaded region-based densely connected network for seismic event detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 1, pp. 62–75, 2018.

[9] H. Abdelhaq, C. Sengstock, and M. Gertz, "Eventweet: Online localized event detection from twitter," *Proc. VLDB Endow.*, vol. 6, no. 12, p. 1326–1329, aug 2013. [Online]. Available: https://doi.org/10.14778/2536274.2536307

[10] Y. Chen, S. Yang, and X. Cheng, "Bursty topics extraction for web forums," in *Proceedings of the Eleventh International Workshop on Web Information and Data Management*, ser. WIDM '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 55–58. [Online]. Available: https://doi.org/10.1145/1651587.1651600

[11] M. Imran, F. Ofli, D. Caragea, and A. Torralba, "Using ai and social media multimodal content for disaster response and management: Opportunities, challenges, and future directions," *Information Processing Management*, vol. 57, no. 5, p. 102261, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306457320306002

[12] C. Zhang, C. Fan, W. Yao, X. Hu, and A. Mostafavi, "Social media for intelligent public information and warning in disasters: An interdisciplinary review," *International Journal of Information Management*, vol. 49, pp. 190–207, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0268401218310995

[13] N. Kankanamge, T. Yigitcanlar, A. Goonetilleke, and M. Kamruzzaman, "Determining disaster severity through social media analysis: Testing the methodology with south east queensland flood tweets," *International Journal of Disaster Risk Reduction*, vol. 42, p. 101360, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2212420919307940

[14] F. Catani, "Landslide detection by deep learning of non-nadiral and crowdsourced optical images," *Landslides*, vol. 18, no. 3, pp. 1025–1044, 2021.

[15] A. Subiyantoro, C. J. Van Westen, B. V. Den Bout, R. A. Yuniawan, and A. R. Mulyana, "Semi-automatic landslide detection using google earth engine, a case study in poi village, central sulawesi," in *2022 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*. IEEE, 2022, pp. 1–4.

[16] F. Ofli, M. Imran, U. Qazi, J. Roch, C. Pennington, V. J. Banks, and R. Bossu, "Landslide detection in real-time social media image streams," *arXiv preprint arXiv:2110.04080*, 2021.

[17] S. Yoshioka, S. Tani, and S. Toda, "Extraction of temporal relation by the creation of historical natural disaster archive," *International Journal of Computer and Information Engineering*, vol. 2, no. 11, pp. 3896–3901, 2008.

[18] F. Imamura, S. P. Boret, A. Suppasri, and A. Muhari, "Recent occurrences of serious tsunami damage and the future challenges of tsunami disaster risk reduction," *Progress in Disaster Science*, vol. 1, p. 100009, 2019.