Milestone 3: Baseline report

First Author
Vishal Paul
Computer Science Department
State University at Buffalo
vpaul@buffalo.edu

Second Author
Subramani Ramadas
Computer Science Department
State University at Buffalo
sramadas@buffalo.edu

Third Author
Christina Dahn
CompLing Department
State University at Buffalo
cdahn2@buffalo.edu

1 Abstract

Social unrest in the form of protests has been prevalent in society for a long period of time. Over the years, this has become more widespread and frequent. Farmers' protests in India, and covid vaccine protests in the USA are some of the latest events to be added to this group. Effectively predicting these events beforehand can tremendously help the Government and local authorities to take action and save losses. In this paper, we propose a novel technique by feeding protest data from existing sources to multiple Gated Recurrent Units. Our results show that recurrent networks are the best models to be used for time series analysis tasks. Moreover, historical data are good indicators in predicting future events.

2 Introduction

Social and civil unrest are physical acts that occur in public places and have a big impact on the daily lives of common people. While protests and demonstrations are non-violent in nature, they have an impact on the stability of societies. For example, the 2020-2021 farmer's protest against the Three Farm Act of the Indian government had a nationwide effect. If there had been a way for the government to predict such events beforehand, then it could have greatly helped them in devising a mitigation plan to reduce damages and losses. Furthermore, event prediction can help in making decisions in the area of national security to allocate and mobilize the resources needed.

Hence, in this project we are taking up this use case to monitor and predict social unrest that might arise following the latest events. We are using the Armed Conflict Location and Event Data Project (ACLED) dataset to get the details of events around the globe. ACLED collects real-time data, such as dates, actors, locations and fatalities of all reported political violence and protest events around the world. With the help of the ACLED data and various Deep Learning tools in Natural Language Processing, we intend to make predictions about the number of future protests taking place in India.

3 Related Work

Since social unrest has been an issue of worldwide interest, many data sources already exist for forecasting such events. This area has also seen multiple papers being published (The EMBERS Architecture for Streaming Predictive Analytics, Andy Doyle, 2014).

The EMBERS (Early Model Based Event Recognition using Surrogates) Architecture is an open-source project which collects 4.6M messages per day from sources like Twitter, HealthMap, and Google Flu Trends. After the data ingestion, the enrichment step involves various text analytic methods such as tokenization, lemmatization, part of speech tagging, and named entity extraction. Traditional models like Logistic Regression, Geo-Temporal Clustering, and Keyword-Based Counting are applied independently of each other to predict the occurrence of a societal event.

The instigation of social unrest on social media platforms usually involves different stages of mobilization on the part of the user (Korolov, 2016). One of the most renowned techniques for predicting social unrest is the use of hashtags for clustering Twitter messages and selecting the top K out of them.

The detection of relevant tweets included 2 steps: Filtering out off-topic tweets and classification of mobilization stages for the remaining tweets. Logistic regression was applied to estimate the probability of protests for a period of time and the results showed a significant correlation between the mobilization in tweets and the occurrence of a protest.

Furthermore, promising results were obtained by including heterogeneous data sources and using LSTM models that effectively exploit sequential data (Leveraging Heterogeneous Data Sources for Civil Unrest Prediction, Lu Meng and Rohini K Srihari, 2019). The Cov-LSTM model gave better results for predictions made for social unrest in Egypt in every evaluation period compared to other base rate models.

This has inspired our work to study social unrest prediction using Deep Learning models which have the ability to overcome the shortcomings of traditional machine learning models.

4 Methods/Model

Subtask 1: Information Extraction

For this task, we needed to extract information from a subset of text, in most cases one or two sentences summarizing a reported event. The challenge was to extract specific categories from the text: number of fatalities, event type, sub event type, actor 1, inter 1, actor 2, inter 2, interaction and location.

As a baseline, we performed Named Entity Recognition on the Notes column with keyword prediction. Since the task produced losses were significantly higher than expected, we instead opted to perform Named Entity Recognition with tfidf for a classification of entities.

For the implementation of the deep learning architecture, we used a classification model with BERT. Figure 1 shows the model's architecture. The input were TSV files, compiled into a corpus. After preprocessing the corpus, which included tokenization, the removal of punctuation and transforming the input into lowercase, we used a pre-trained BERT transformer model and finally used the crossentropy loss function and Adam Optimizer.

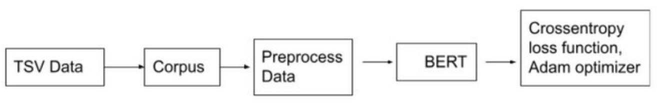
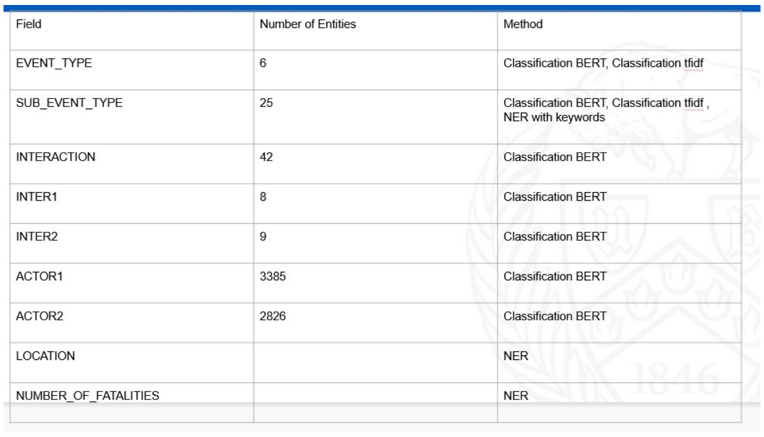


Figure 1: Model Architecture Deep Learning

Figure 2 shows an overview of the different model architectures used for the deep learning architecture. For event type and sub event type, we used tf-idf vectorization for encoding the input features for classification. After comparing the results with a BERT model for classification we decided to proceed with a BERT model for the other categories. The location and number of fatalities were encoded using natural entity recognition.

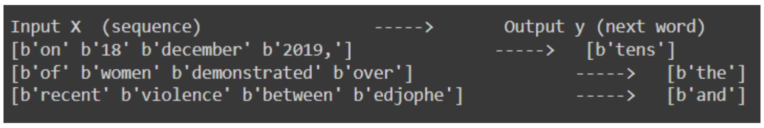


Field	Number of Entities	Method
EVENT_TYPE	6	Classification BERT, Classification tfidf
SUB_EVENT_TYPE	25	Classification BERT, Classification tfidf , NER with keywords
INTERACTION	42	Classification BERT
INTER1	8	Classification BERT
INTER2	9	Classification BERT
ACTOR1	3385	Classification BERT
ACTOR2	2826	Classification BERT
LOCATION		NER
NUMBER_OF_FATALITIES		NER

Figure 2: Comparison of Methods for each field

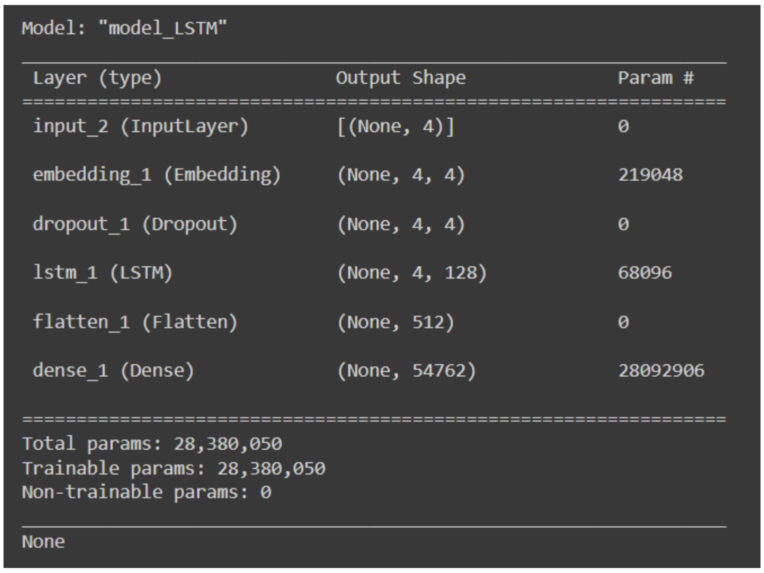
Subtask 2: Summary Generation
Given keywords from ACLED dataset, the task is to generate ACLED style text. For this problem, we created a language model on the training data provided. We created a neural network of Embedding, LSTM and Dense Layers.

We created a corpus of all the keywords. From the corpus, we created x and y label for the Neural Network where x is a tensor of 4 words and y is the next predicted word. We trained the model with these xtrain and ytrain and then predicted the text for test keywords. X and Y are encoded as well for the model. The model architecture is shown as below. The model was then trained for 100 epochs.



```
Input X  (sequence)                -----> 	Output y (next word)
[b'on' b'18' b'december' b'2019,']         ----->   [b'tens']
[b'of' b'women' b'demonstrated' b'over']           ----->   [b'the']
[b'recent' b'violence' b'between' b'edjophe']      ----->   [b'and']
```

Figure 3: Sample encoded x and y



```
Model: "model_LSTM"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 4)]               0

 embedding_1 (Embedding)     (None, 4, 4)              219048

 dropout_1 (Dropout)         (None, 4, 4)              0

 lstm_1 (LSTM)               (None, 4, 128)            68096

 flatten_1 (Flatten)         (None, 512)               0

 dense_1 (Dense)             (None, 54762)             28092906

=================================================================
Total params: 28,380,050
Trainable params: 28,380,050
Non-trainable params: 0
_________________________________________________________________
None
```

Figure 4: LSTM Model Architecture

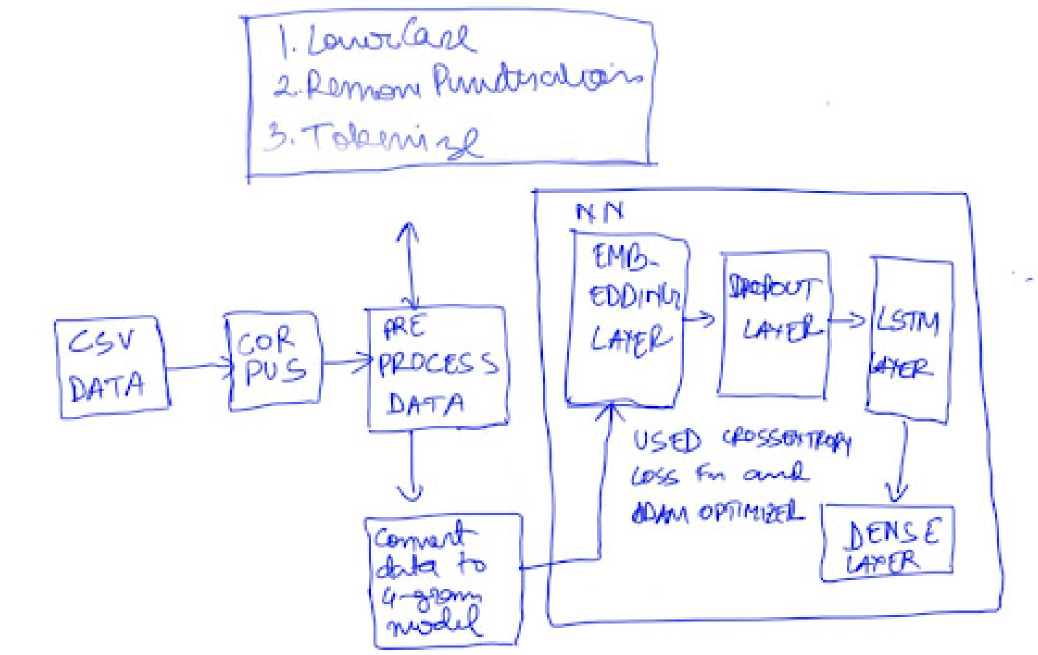


Figure 5: Model Architecture

Subtask 3: Event prediction from text
Gated Recurrent Unit(GRU) is a type of gating mechanism in recurrent neural networks that aims to solve the vanishing gradient problem. Since it is a type of Recurrent Neural Network (RNN), it can be applied to time series related tasks in order to capture the internal state to process the sequences of input.

Upon evaluation of the density distribution for the data for each day, we decided on a maximum sequence length of 90 and a minimum sequence length of 10.

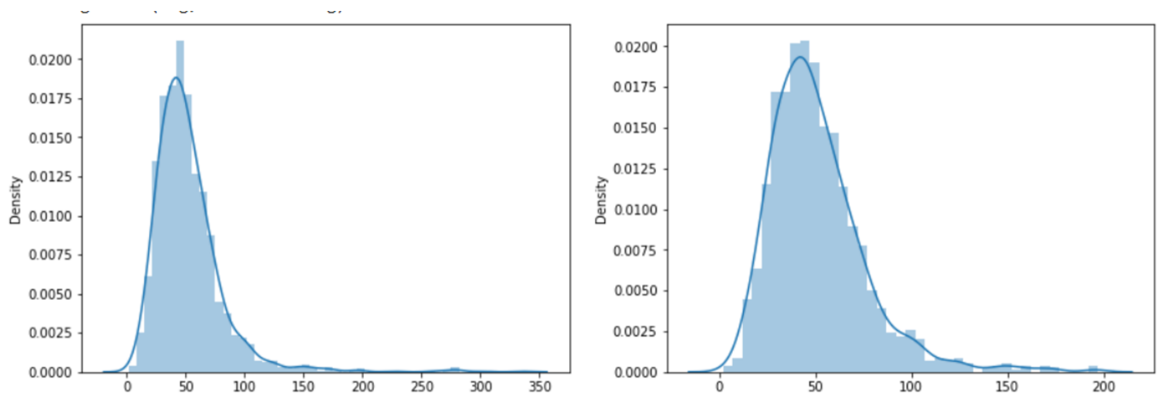
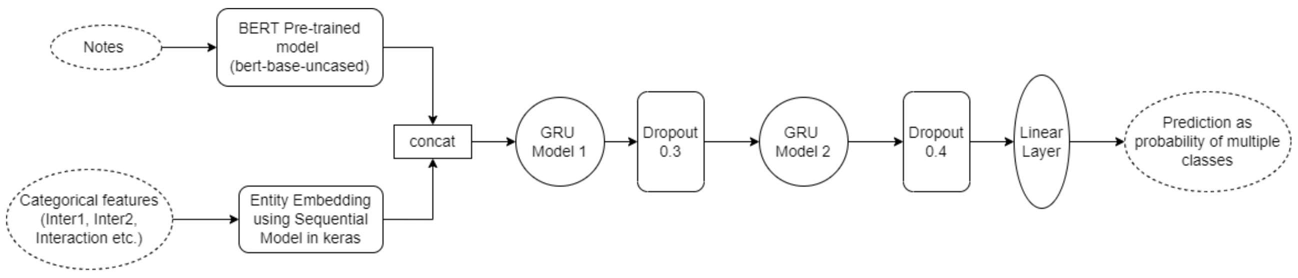


Figure 6: Density Distribution of count of data for days in the dataset

Using the PyTorch library, we implemented two GRU Models to transform and process the input data sequence for dates. The input notes feature is given a 768-dimensional embedding using a pre-trained BERT transformer model. The output of the BERT model is further concatenated with the entity embedding of categorical features.



Notes
BERT Pre-trained model (bert-base-uncased)
Categorical features (Inter1, Inter2, Interaction etc.)
Entity Embedding using Sequential Model in keras
concat
GRU Model 1
Dropout 0.3
GRU Model 2
Dropout 0.4
Linear Layer
Prediction as probability of multiple classes

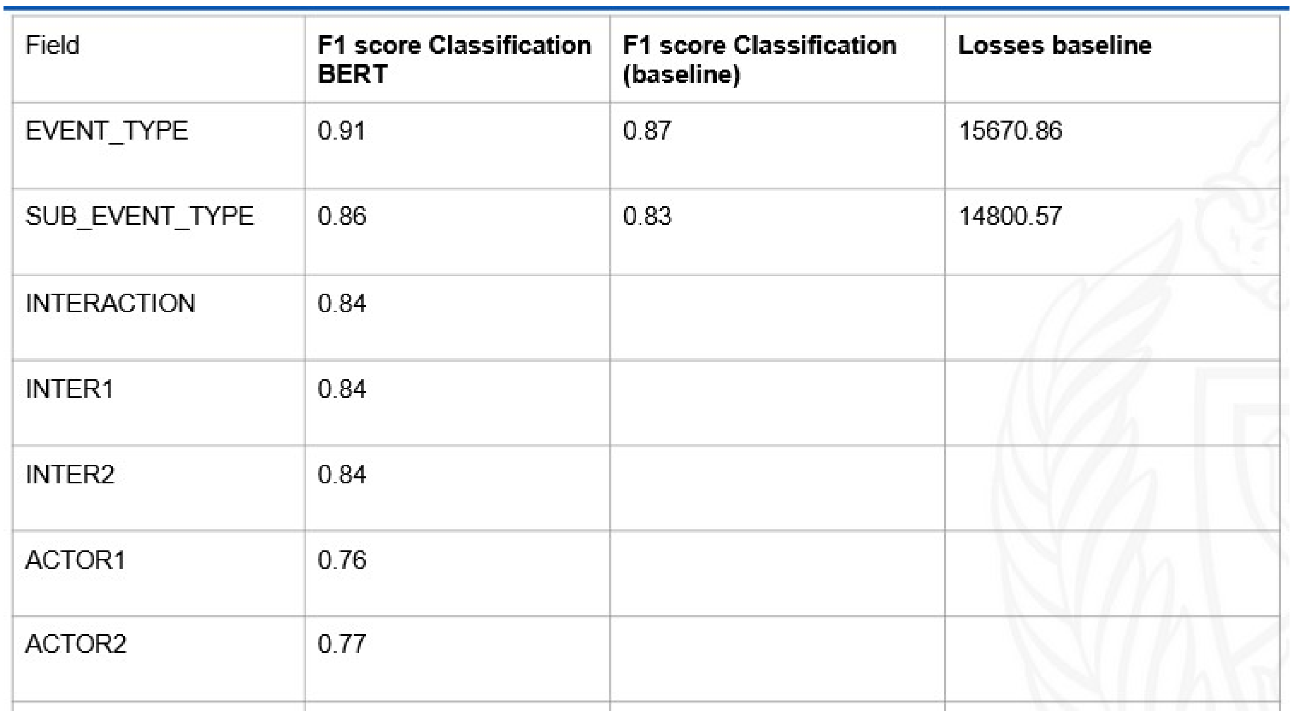
Figure 7: Dual-GRU Model Architecture

The concatenated embedding output is loaded into iterable datasets using PyTorch DataLoader for a batch size of 16. The Dual-GRU model, as shown in figure, takes in a batch and uses two GRU networks to make the prediction. This model is designed to handle predictions for different lead times while a constant number of 3 past days is used as the input to make prediction for the nth day. Each GRU network is comprised of two hidden layers. Moreover, a dropout of 0.3 ad 0.4 are applied to the output of the first and second GRU networks respectively. Finally a linear layer takes in the output of second GRU network and predicts the probability of the multiple classes for the count of protests.

5 Results

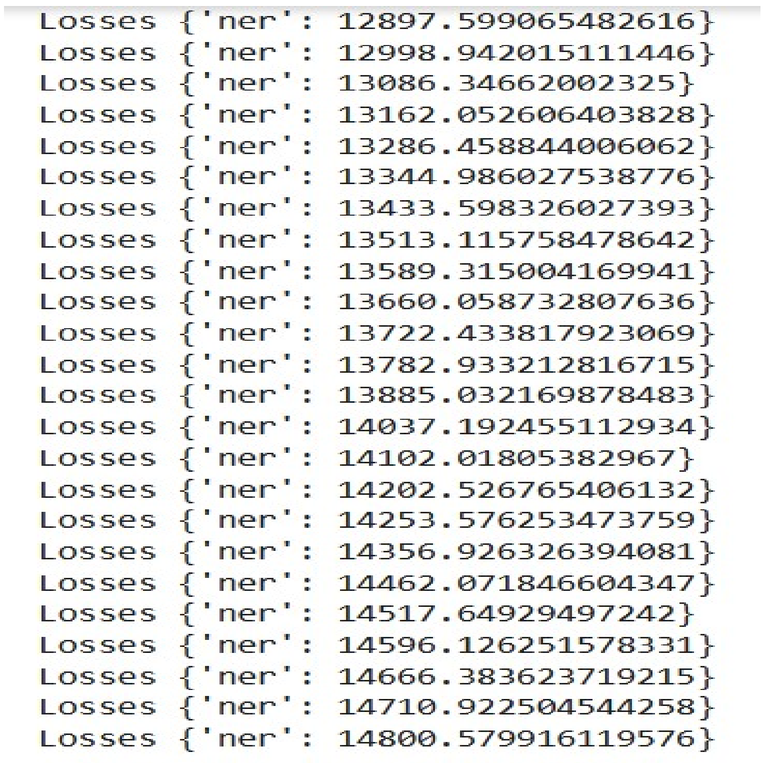
Subtask 1: Information Extraction

The following are the results for the baseline and the deep learning models for subtask 1 (see Figure 8 below): we calculated the F1 score for all the categories. All show an 80 percent accuracy on the F1 score. Event type showed an accuracy of 91 percent, which might be due to the low number of categories (8). The NER model with keyword classification showed enourmous losses, which are shown in Figure 9.



Field	F1 score Classification BERT	F1 score Classification (baseline)	Losses baseline
EVENT_TYPE	0.91	0.87	15670.86
SUB_EVENT_TYPE	0.86	0.83	14800.57
INTERACTION	0.84		
INTER1	0.84		
INTER2	0.84		
ACTOR1	0.76		
ACTOR2	0.77		

Figure 8: Results baseline and deep learning models



```
Losses {'ner': 12897.599065482616}
Losses {'ner': 12998.942015111446}
Losses {'ner': 13086.34662002325}
Losses {'ner': 13162.052606403828}
Losses {'ner': 13286.458844006062}
Losses {'ner': 13344.986027538776}
Losses {'ner': 13433.598326027393}
Losses {'ner': 13513.115758478642}
Losses {'ner': 13589.315004169941}
Losses {'ner': 13660.058732807636}
Losses {'ner': 13722.433817923069}
Losses {'ner': 13782.933212816715}
Losses {'ner': 13885.032169878483}
Losses {'ner': 14037.192455112934}
Losses {'ner': 14102.01805382967}
Losses {'ner': 14202.526765406132}
Losses {'ner': 14253.576253473759}
Losses {'ner': 14356.926326394081}
Losses {'ner': 14462.071846604347}
Losses {'ner': 14517.64929497242}
Losses {'ner': 14596.126251578331}
Losses {'ner': 14666.383623719215}
Losses {'ner': 14710.922504544258}
Losses {'ner': 14800.579916119576}
```

Figure 9: Losses NER model with keywords

The classification model trained on BERT showed a higher accuracy compared to the simpler model. While the difference is not very pronounced and within a few percentages points, if this were scaled up to a significantly bigger dataset the difference could be decisive.

Subtask 2: Summary Generation

The following are the results for our baseline model for Subtask 2:

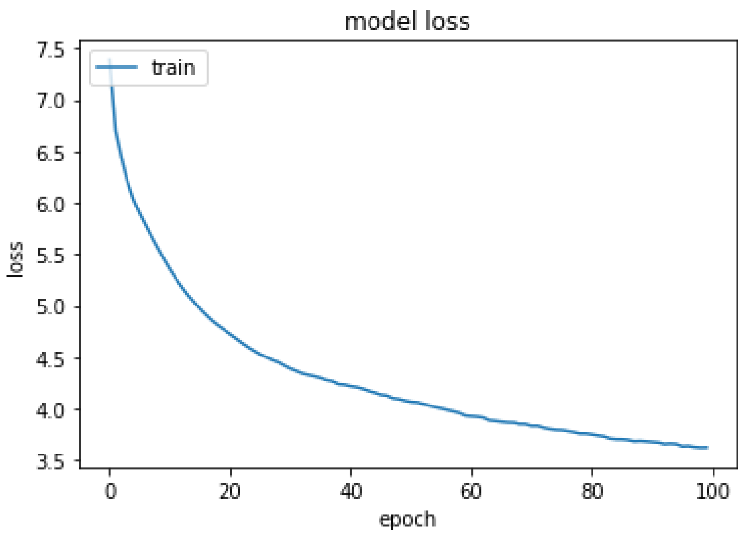


Figure 10: Training Loss

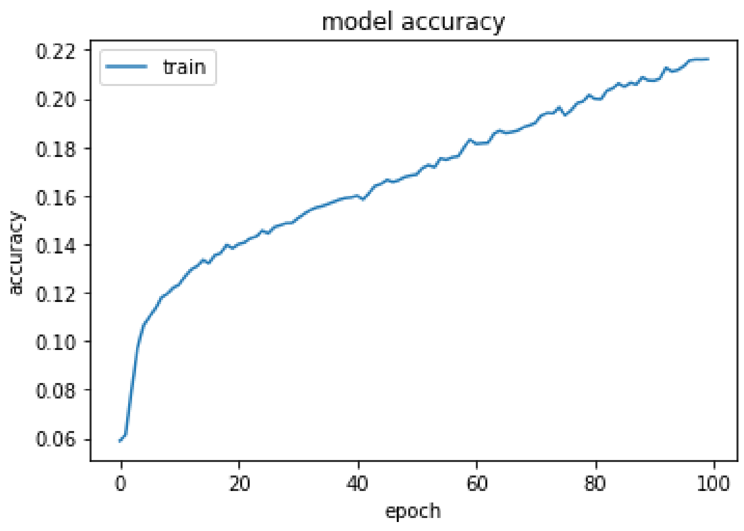


Figure 11: Training Accuracy

We calculated perplexity, rogue (precision, recall and f-measure), and meteor score. The graph shows data for 100 samples from the train text.

The table displays average scores from all the sample data. We generated each sentence using the keywords as input and calculated the scores from actual text (from training data).

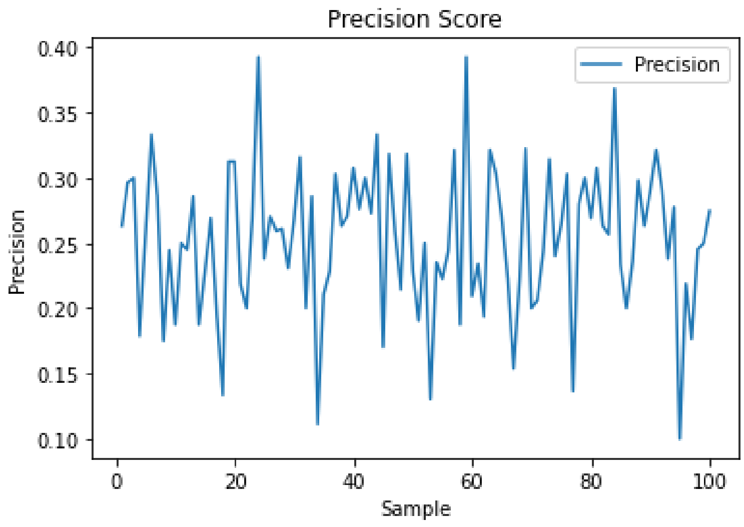


Figure 12: Precision Score

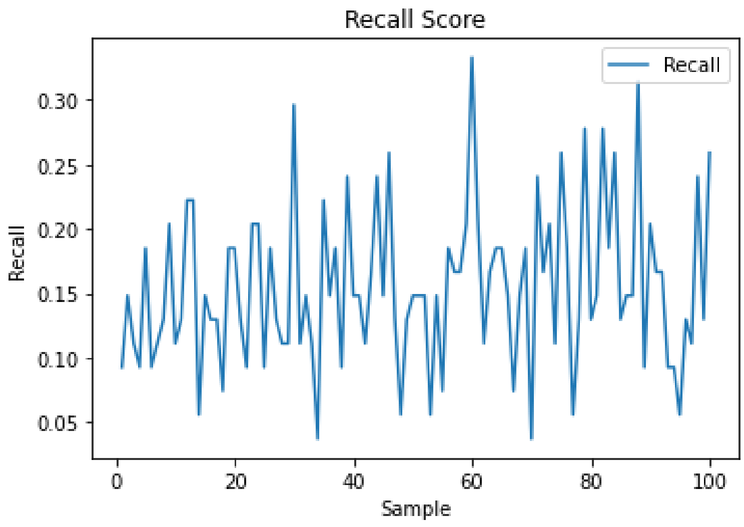


Figure 13: Recall Score

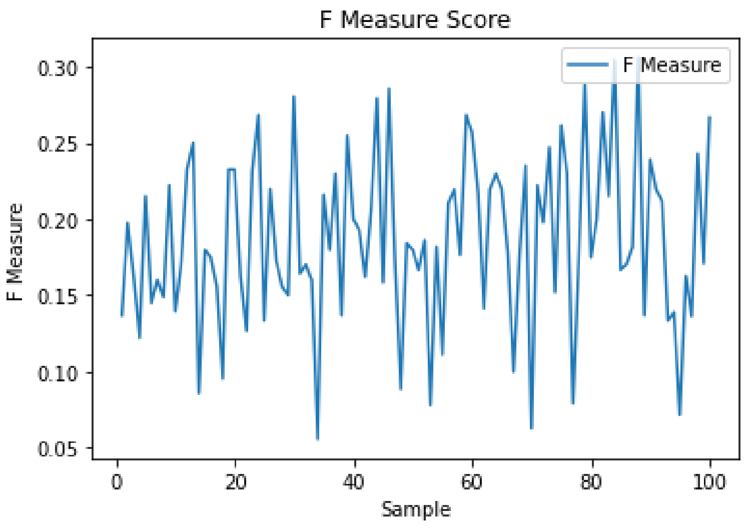


Figure 14: F-Measure Score

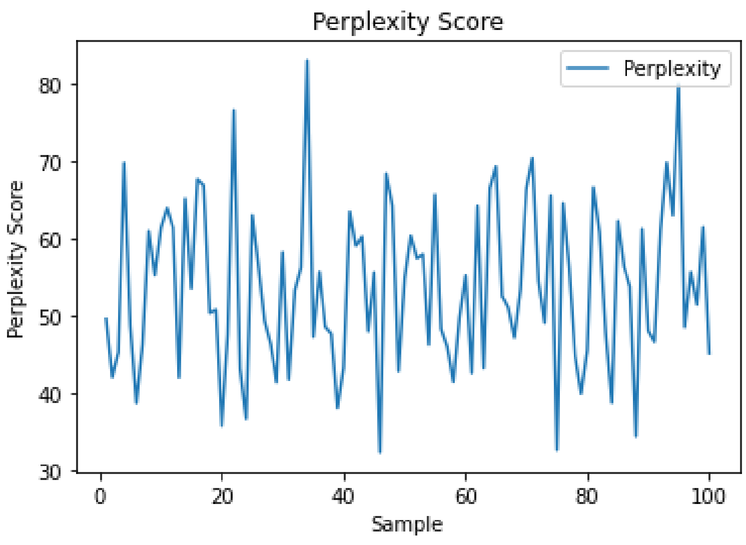


Figure 15: Perplexity Score

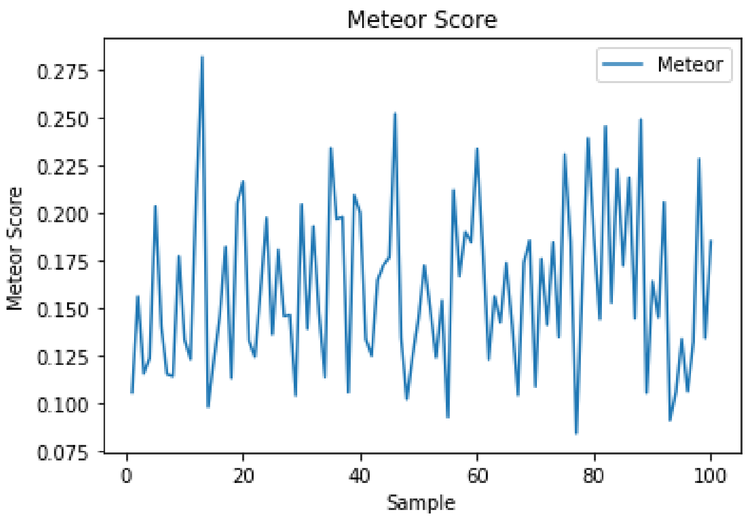
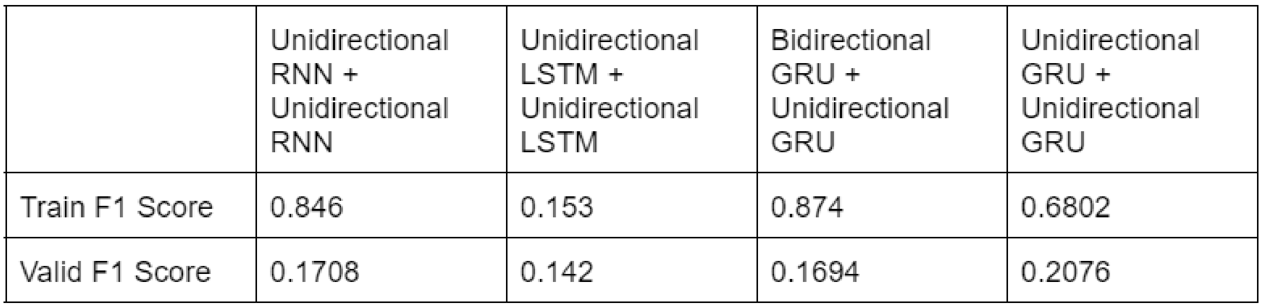


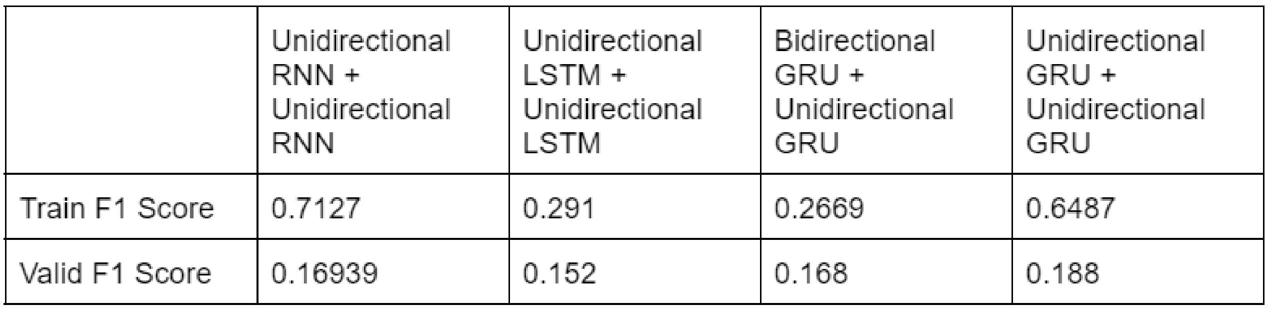
Figure 16: Meteor Score

Subtask 3: Event prediction from text
The event prediction from text task was performed using the ACLED dataset of around 77000 records for India. Our data had multiple datapoints for a single day. Entire data for every single day was collated to give 1368 days worth of data. 80% of the data was used for training and the rest 20% for validation. Each data point corresponds to a day and is labeled by the number of protests that happened on that day.



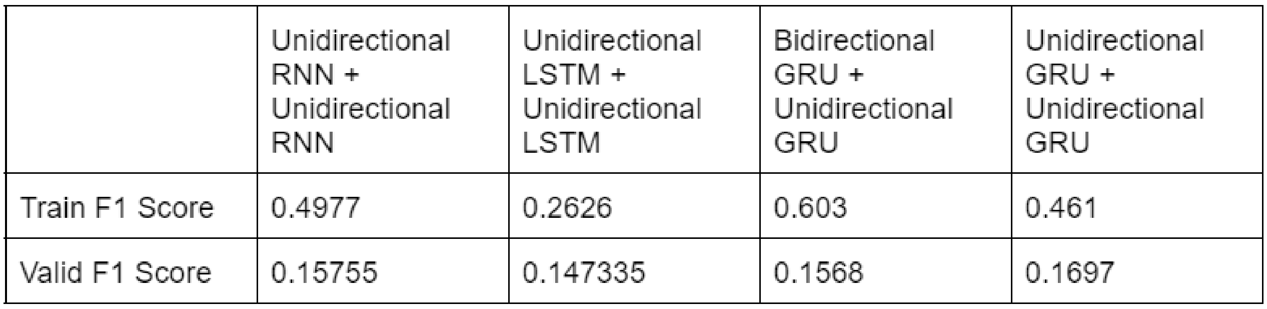
	Unidirectional RNN + Unidirectional RNN	Unidirectional LSTM + Unidirectional LSTM	Bidirectional GRU + Unidirectional GRU	Unidirectional GRU + Unidirectional GRU
Train F1 Score	0.846	0.153	0.874	0.6802
Valid F1 Score	0.1708	0.142	0.1694	0.2076

Figure 17: Lead Time: 3 days



	Unidirectional RNN + Unidirectional RNN	Unidirectional LSTM + Unidirectional LSTM	Bidirectional GRU + Unidirectional GRU	Unidirectional GRU + Unidirectional GRU
Train F1 Score	0.7127	0.291	0.2669	0.6487
Valid F1 Score	0.16939	0.152	0.168	0.188

Figure 18: Lead Time: 5 days



	Unidirectional RNN + Unidirectional RNN	Unidirectional LSTM + Unidirectional LSTM	Bidirectional GRU + Unidirectional GRU	Unidirectional GRU + Unidirectional GRU
Train F1 Score	0.4977	0.2626	0.603	0.461
Valid F1 Score	0.15755	0.147335	0.1568	0.1697

Figure 19: Lead Time: 10 days

The implementation for baseline considered only a couple of categorical features and gave a binary classification of whether a protest occurs or not with a training score of 0.215. However, the combination of multiple recurrent units with the addition of more features performs significantly better than the baseline results. Moreover, the model combining two unidirectional GRUs performed slightly better than other combinations for all the 3 different lead times.

6 Discussion and Error Analysis

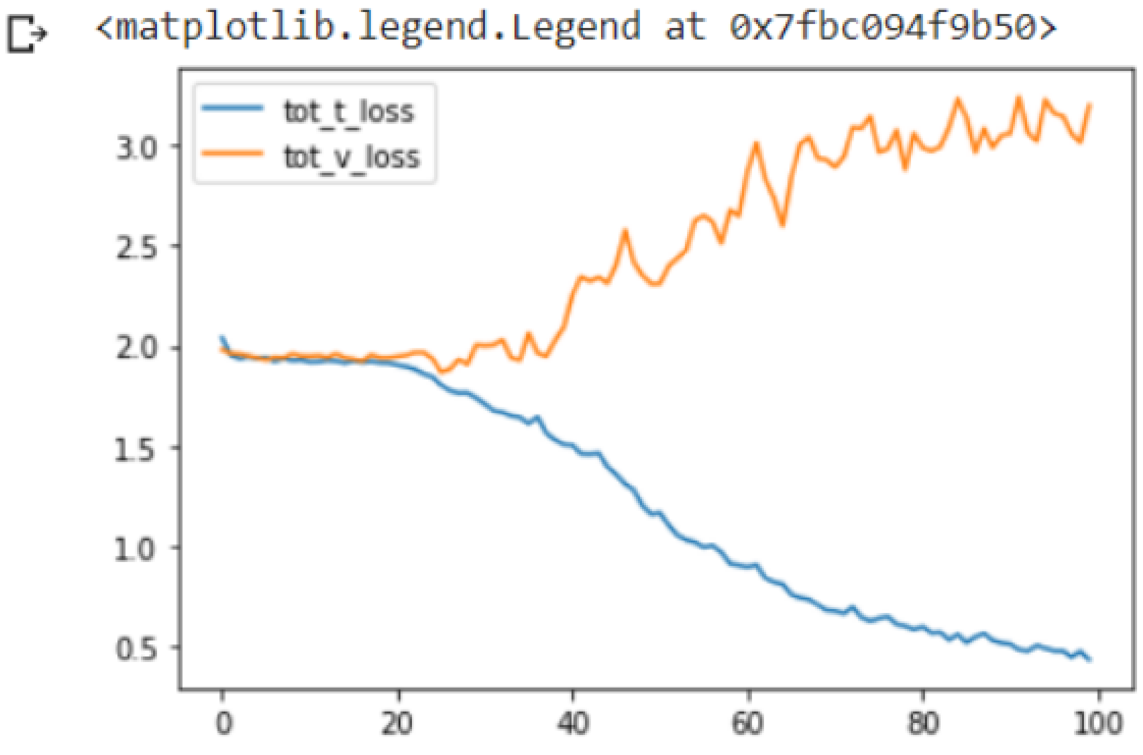
Subtask 1: Information Extraction

We had started out with a NER baseline classification with keywords to understand how the dataset is built and what models would be best suited. For example, for the subevent ”peaceful protests”, we had inserted the keywords ’protests’, ’protested’, ’demonstrate’ and ’hunger strike’. However, the keyword classification resulted in enourmous losses due to the imprecision of the classification and overlap between subevents. After performing several improvements, we decided to use tf-idf vectorization instead.

Subtask 2: Summary Generation
We had also implemented a character based prediction model but the results were not that good as the word based prediction model. Also one of the disadvantages we faced were, it was difficult to fit the model with data for the character based model. Very large datasets of large dimensions were produced for a small amount of text. Hence, we moved forward with word based prediction. The validation loss and accuracy were of the same range as train accuracy and loss because of the following reasons and also it could be improved as stated below: As our model involved creation of 4-word set generation vectors, we could only fit a limited amount of text from the train data. This can be improved with better resources. We had limited resources, hence couldn’t fit much of the training data. The accuracy can also be improved by sampling the data and selecting a variety of features and targets.
For future work and prospects, we can also incorporate Transformer architecture. Transformer architecture works better in Natural Language Generation tasks.

Subtask 3: Event prediction from text

Although the Dual-GRU unidirectional model performs better compared to other model combinations, the resulting F1 scores are lower than expected. While plotting the losses for the Dual-GRU model, we can see that after epoch 30, the models seems to overfit.



```
<matplotlib.legend.Legend at 0x7fbc094f9b50>
```

Figure 20: GRU Losses (Lead Time 3 days)

```
<matplotlib.legend.Legend at 0x7fbbec837e10>
```

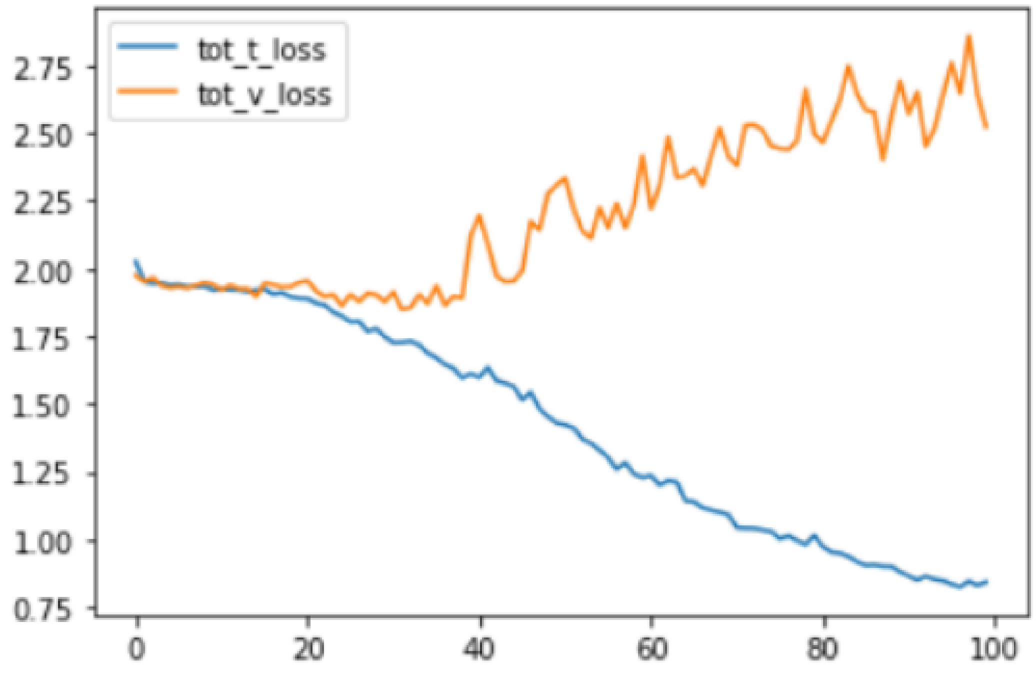


Figure 21: GRU Losses (Lead Time 5 days)

```
<matplotlib.legend.Legend at 0x7fbac083afd0>
```

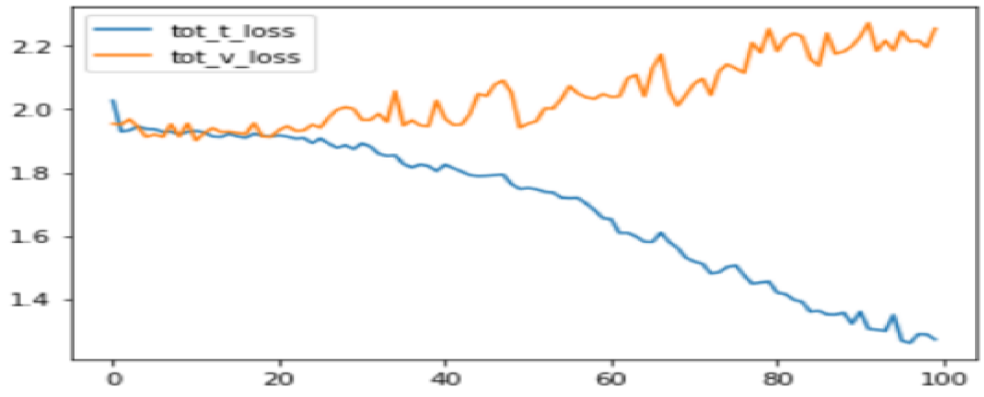


Figure 22: GRU Losses (Lead Time 10 days)

Thus, the model can be improved by incorporating more categorical features and including data from heterogeneous data sources. Future scope of training can include the Twitter data, news articles data and other video or image data. Moreover, the model can be trained for other countries as well so that it can be generalised to predicting the count of protests all over the globe.

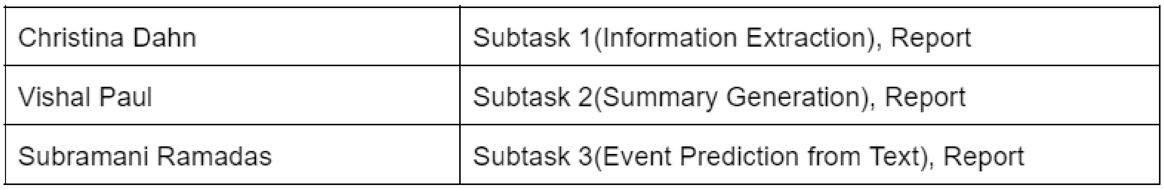
7 Conclusion

Subtask 1: Information Extraction
The task was to extract the number of fatalities, event type, sub event type, actor 1, inter 1, actor 2, inter 2, interaction and location from the summary of reported events. Our approach was to use multi-class classification using BERT. It worked pretty well on the given datasets with F1 scores of over 80 percent. However, the model could be further improved by replacing BERT with GPT2 or GPT3 in the model architecture. To conclude, there are several ways of performing classification on these categories, however a natural entity recognition with classic NLP methods is difficult due to the overlap between categories and the unusual structure of the summary, since the pretrained models, for example the NLTK, often are trained on different sentence structures and categories.

Subtask 2: Summary Generation
The task was to generate text from given a set of keywords. Our approach was to create a set of words and predict the next word using a LSTM model architecture. Our model worked pretty well on the less dataset we were able to provide due to limited resources. If we had access to better compute resources then we would have achieved even better results. But, I believe, as future prospects we can use a different method by using Transformers. For example, we can make us of the T5 transformer to implement the text generation task. To conclude, there are multiple ways to approach the problem, and we selected the one where can have exposure to both NLP and Machine Learning.

Subtask 3: Event prediction from text
Predicting social unrest in the form of protests has been a topic of interest for quite a long time and it is still a work in progress. Accurately predicting unrest events beforehand is a challenging task and while recurrent neural networks is the best way to move forward in solving this problem, incorporating data from multiple sources along with training the model on multiple countries would greatly improve the performance of the model. This work leverages an existing dataset to provide the predictions for various lead times.

8 Work Distribution



Christina Dahn	Subtask 1(Information Extraction), Report
Vishal Paul	Subtask 2(Summary Generation), Report
Subramani Ramadas	Subtask 3(Event Prediction from Text), Report

Figure 23: Work Distribution

9 Bibliography

References

Bird, Steven; Loper, Edward; Klein, Ewan (2009): Natural Language Processing with Python. O'Reilly Media Inc.

Doyle, A. et al (2014)., "The EMBERS architecture for streaming predictive analytics," 2014 IEEE International Conference on Big Data (Big Data), in https://ieeexplore.ieee.org/document/7004477

Korolov, R. et al., "On predicting social unrest using social media," 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), 2016, pp. 89-95, doi: 10.1109/ASONAM.2016.7752218 In https://blender.cs.illinois.edu/paper/unrestprediction.pdf

Meng, Lu; Srihari, Rohini (2019): Increasing Lead Time and Granularity of Civil Unrest Prediction through Time Series Data. In https://milets19.github.io/papers/milets19$_poster_7.pdf$

Ramakrishnan, Naren et al. (2014): Beating the news' with EMBERS: Forecasting Civil Unrest using Open Source Indicators. In https://arxiv.org/abs/1402.7035
https://towardsdatascience.com/from-pre-trained-word-embeddings-to-pre-trained-language-models-focus-on-bert-343815627598

https://towardsdatascience.com/building-rnn-lstm-and-gru-for-time-series-using-pytorch-a46e5b094e7b

https://towardsdatascience.com/gate-recurrent-units-explained-using-matrices-part-1-3c781469fc18