Automated Information Extraction on Gun Violence News Articles

Samuel Fleischer fls@seas
Ranwei Hu rhu@seas
Rishab Jaggi rjaggi@seas
Tyler Larkworthy tlarkwor@seas

Abstract

We show the results of applying NLP techniques to perform automated information extraction on articles in the Gun Violence Archive. We first present a review of the current literature and provide examples of input unstructured text and target gold labels. We then describe three models used to extract the location and date of gun violence incidents as well as the number of people killed and injured. After establishing a naive baseline, we implement models that follow a published precedences. We also show three extended models with better F_1 scores for extracting address and number of victims. Finally, we present an error analysis and provide potential other areas for performance improvement.

1 Introduction

Gun violence is a large and growing problem in the United States. Unfortunately, its causes are poorly understood and there is a lack of bipartisan support to address the issue. Emotions and political biases often get in the way of productive discussion about potential solutions. Not only do political agendas stand in the way of objective research into the area, but the expense of data collection can also be a limiting factor. We chose this project as we believe it is important to use technology in order to objectively understand, explain, and resolve this issue without opinions and political thought influencing the results. We think that this project has the potential to move forward the conversation in an effort to fight to reduce gun violence in the United States. Modern day NLP tools present a variety of objective measures and techniques we can use for performing advanced information extraction and analysis. The result of this technology can be monumental in moving towards gun control legislation.

The first step in preventing gun violence is collecting data on incidents in order to understand patterns like location and demographic trends. To do this we need a large set of reliable data that can be studied to understand the different forms and effects of gun violence. Manually annotated data leads to great results but is too much of a long and involved process. Thus, we desire a system that can autonomously ingest news articles on gun violence incidents and deduce key data points that can then be aggregated for analysis.

NLP techniques like NER have proved to be very accurate in extracting information like this related to gun violence incidents. Previous work has shown that NLP technologies are perfectly suited for this task of extracting information from the unstructured articles published daily by newspapers (Pavlick et al., 2016). Tools like sentiment analysis, named entity recognition and information extraction prove to be very successful.

We can formalize the problem to be, given a news article about a gun violence incident and some metadata relating to the article, such as the publish date, autonomously extract and report the date and location of the incident, as well as the number of people killed and number of people injured. An example input and output for the system is provided below.

2 Literature Review

(Pavlick et al., 2016) describe the motivation for the Gun Violence Database and how the authors began the automation of data collection using NLP tools. Their paper first discusses how data on gun violence incidents is collected and annotated manually by humans and then how that data collection can be automated using NLP

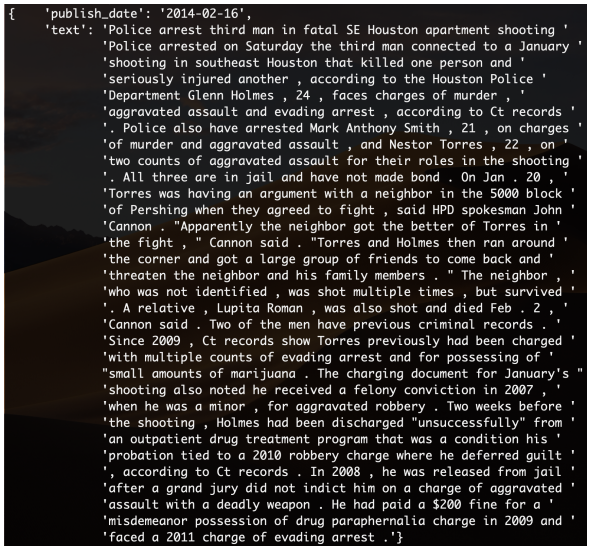

Figure 1.1: Example input

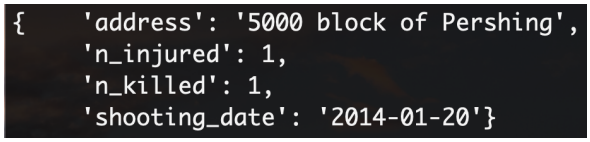

Figure 1.2: Example output

technology. In order to build up the original database, newspapers and articles are crawled and then classified by a machine learning high-recall text classifier. These classified documents are then passed on to human annotators that are able to verify the label. The human participants further annotated the articles by marking or highlighting important pieces of information relating to the incident, like the name of the shooter/victim(s), type of weapon used, location, etc. This results in an easily queryable, curated dataset. Next, the paper describes how the long and involved process of manual annotation can be streamlined using statistical NLP techniques. Modern NLP technologies exist for accurate Information Retrieval (getting the articles on incidents), NER and Event Detection (determine the key participants and details of the incident), Parsing and Coref (connecting articles relating to the same incident), and more. The authors argue that each piece of the original pipeline for populating the GVDB has been studied as its own NLP problem and that solutions readily exist to automate these processes. This is really important since the lack of good data (and funding for studies) has limited our ability to understand and tackle the issue of gun control. This paper shows that there is a promising method of extracting this important data with great accuracy, consistency and scalability using NLP.

(Chang and Manning, 2012) discuss the use of pattern-based rules for date/time extraction. The authors describe using text regular expressions, compositional rules, and filtering in order to best extract dates, times, and ranges of dates. Three types of patterns are used: token patterns, string patterns, and time patterns. The authors tested the performance of their system on the TempEval-2 task, achieving an F1 score of 0.92.

(Arulanandam et al., 2014) discuss crime information extraction, primarily location extraction. The goal of their paper is to be able to find all sentences that contain a location, and then to determine whether that sentence is the crime location through using a model.The method described included first obtaining a corpus of relevant news articles, tokenizing the words, and then running existing NER software on the tokens to find tokens representing locations. Then, the sentences including location tokens are tagged as either CLS (crime location sentence) or NO-CLS (not a crime location sentence) through using binary features such as sentHasCrimeTerm, sentHasCityLoc and a CRF model. Thus, the crime locations are in the sentences tagged as CLS. With CLS labels, this method received an F-score of 0.87, and with non-CLS labels, this method received an F-score of 0.93.

3 Experimental Design

3.1 Data

The raw data comes from the website https://www.kaggle.com/jameslko/gun-violence-data. Each event states the address, number killed, number injured, shooting date, the source article, and more. We used each event to get the training data. In particular, we extracted the contents of each article using a python library called newspaper, and compiled the event id, title, publish date, address, number killed, number injured, and the shooting date into a JSON object. See Appendix B for data extraction code.

Taking this filtered JSON file, we then bifurcated the data into the features and the labels

where the features have title and publish date, and the labels have number address, killed, number injured, and shooting date.

We currently have 2338 samples. With the current splitting scheme, we are allocating 70% of the data to training, 9% to validation, and 21% to testing.

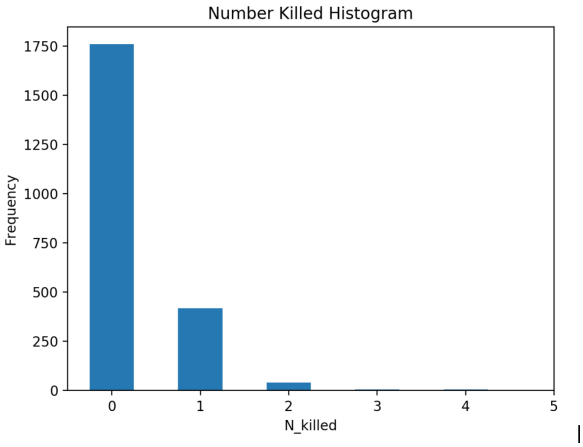

Figure 3.1: A histogram of the number killed field reveals that the vast majority of entries are 0

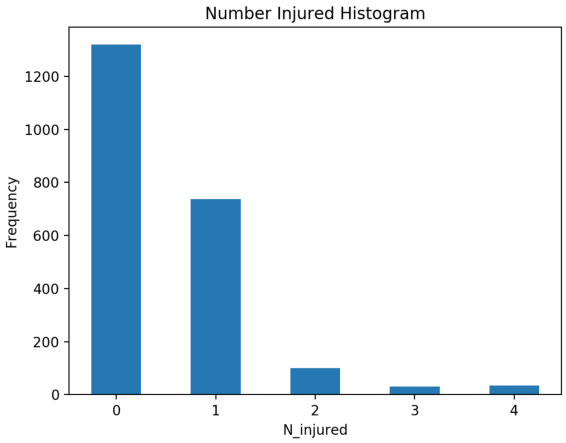

Figure 3.2: A histogram of the number injured field shows that, similar to the number killed field, most entries are 0. The distribution is slightly less skewed, however.

3.2 Evaluation Metric

To evaluate our models, we used both accuracy and macro F1 scores. Accuracy scores were calculated as a simple proportion of assigned labels that were correct. Since all fields involved multi-label classification, we used macroaveraging to calculate F1 scores. This involved calculating precision and recall for each category, then taking the harmonic mean of these scores. The F1 score was then calculated using the following equation:

$$F_1 = \frac{2PR}{P+R}$$

Several publications, including (Chang and Manning, 2012) and (Arulanandam et al., 2014) used the F1 score metric to assess their models.

3.3 Simple baseline

The simple baseline consisted of basic predictions. For the address, SpaCy NER was run on the text, and the first FAC/LOC labelled entity was returned. For the date, the day before the published baseline was returned. For the number injured and number killed, the value 0 was returned. As seen in Table 1, predicting 0 actually produced reasonably high scores. This suggests that many of the events actually have 0 individuals killed or injured.

	Address	Date	Number Killed	Number Injured
Accuracy	0.1176	0.4118	0.8529	0.6471
Macro F1	0.0274	0.2387	0.4603	0.2619

Table 1: Performance of baseline model on test data

4 Experimental Results

4.1 Published baseline

Since gun violence information extraction is a very specific and novel task, there was not much literature describing detailed approaches for our tasks. Most articles regarding gun violence information extraction either gave a brief paragraph or two for each type of data extracted, for many different types of data such as date or location, or simply mentioned an existing framework (such as a library) that was used to extract the data. Since we are extracting four pieces of gun violence related information from the text, namely number injured, number killed, data of event, and location of event, we required essentially four different approaches, one for each piece of information. Due to the lack of specific information for each separate piece of information, some of the approaches selected had to be inferred these brief descriptions.

For date extraction, we used the article by (Chang and Manning, 2012). The approach discussed mainly described pattern based rules, such

as regular expressions, in order to determine dates. We applied this idea to our texts; notably, the model also looks at days of the week in order to calculate the day of the shooting. For crime location extraction, we adapted the approach from (Arulanandam et al., 2014), using CRF with (modified) features to classify each sentence with a tagged location as CLS or not-CLS. For both number killed and number injured, we applied a similar approach; for each number in the text, we used CRF with some features to classify the sentence as having 0-4 injured/killed (temporarily limited). Then, the sentence in the text with the largest label is returned. Since papers did not provide much insight on approaches at all for extracting these numbers, other than for example a brief mention of model algorithm, this approach was also adapted from the (Arulanandam et al., 2014) paper.

	Address	Date	Number Killed	Number Injured
Accuracy	0.1471	0.6765	0.8235	0.4412
Macro F1	0.0084	0.5185	0.4516	0.2444

Table 2: Performance of the published baseline model on test data

4.2 Extensions

4.2.1 Date model extension

An issue with the published-baseline model was that it still picked the first date that met a regex pattern. It is often the case that the first date is not necessarily the date of the incident. Thus, for the extension, we fetched all candidate dates in the model and let a Logistic Regression classifier pick the highest probability candidate. We had four features in the date model extension:

- Number of crime words
- Number of court terms
- Number of misleading terms
- Type of date (0 if absolute and 1 if relative)

All counts were done on a per sentence basis.

4.2.2 Address model extension

For the address model, we decided to add regex patterns in addition to the CRF model. A shortcoming of the published-baseline model was that the returned address did not fit the format that the gold labels expect. For example, the Named Entity Recognizer may recognize Maple Street as a valid street, even though the gold label expects 4000 block of Maple Street as the correct label. By adding regex patterns, the classifier can more accurately return addresses in the correct format.

4.2.3 Killed and injured model extension

For the killed/injured model we decided to adopt a document-based Support Vector Classifier instead of using a CRF on the individual sentences. The issue with the published-baseline model was that we did not have killed/injured labels at the sentence level, and instead only had data on the entire document. We could have scanned for words or numbers that matched with the number in the gold label, but there was a risk of generating false positives. Thus, by training a model on the entire document, we could more accurately determine the number of victims without the risk of generating false positives.

For features, we counted the number of death terms, injured terms, prison terms, single terms, and plural terms. We also counted the number of unique individuals using NER.

	Address	Date	Number Killed	Number Injured
Accuracy	0.1176	0.6765	0.8529	0.6471
Macro F1	0.0274	0.5185	0.4603	0.3710

Table 3: Performance of extended model on test data

4.3 Error analysis

Address errors:

- Misclassification: 50%
- Pattern Failure: 25%
- Multiple locations (includes police blotter): 25%

There are three categories of Address misclassifications. The first category is simply a classifier mistake. Some examples include the locations of where the suspects are arrested/live, or where the victims lives, or the location of the 911 call. In addition, sometimes NER fails and the model returns an empty string. The second category is pattern failure. For this error, the model returns the correct address but it is not in the same format as the gold label. This type of error was more commonplace in the published-baseline model, but the addition of the regex extension helped reduce the number of errors of this type. The

final category is multiple locations. This type of error is universal across all models and it arises from the fact that some of these articles are police blotters that include multiple incidents from different places. It is also not uncommon to see articles that describe a string of related crimes that have occurred over multiple places.

Date errors:

- Misclassification: 60%
- Multiple incidents (includes police blotter): 20%
- Annotator mistake: 20%

The first type of error is a simple misclassification. There are several types of dates that can appear in an article, such as court hearings, dates of accusations, and dates of arrests. Overnight shootings can also confuse the classifier. Sometimes events took place far in the past and the article does specify the exact date in which the event occurred (ie. the February shooting). Some articles do not even mention the date. The second type of error is multiple incidents. Police blotters often include incidents from different points in the week. Finally, a significant portion of errors come from annotator mistake. It is often the case that an annotator will get the relative date of the incident incorrectly (ie. the annotator does not know when 'last Friday' was).

Killed errors:

- Misclassification: 65%
- Multiple incidents (includes police blotter): 35%

Misclassifications were the most common type of error encountered. Some of the misclassified articles can be attributed to a lack of words that contribute to the 'death terms' feature. For example, there was an article where a man was discovered 'full of gun-shot holes'. In these cases, there is no particular word that immediately suggests that the man died, but readers can infer from context that the man died. Information extraction based on inference is extremely difficult, and we would need to take an entirely different approach to this problem if we were to reduce misclassification error.

5 Conclusions

We struggled to improve upon the performance of the published baseline. We are unable to state whether we reached state-of-the-art considering that our overall set of tasks and data set are fairly novel and a limited amount of literature is available. Still, we can see that we were unable to achieve the F1 score that was attained in (Chang and Manning, 2012). However, this could be because we were not merely trying to extract dates, but extract the particular date on which a shooting occurred. Its also notable the amount by which we were able to improve our accuracy scores from the baseline to the extended models. Further research in this area could explore different models that perhaps utilize dependency paths and attempt co-reference resolution. It is evident though that a fairly straightforward model with the limited features that were implemented was able to reach reasonably competitive performance on extracting such key pieces of data.

Acknowledgments

We would like to thank Professor Chris Callison-Burch for his guidance and mentorship during this project.

References

Rexy Arulanandam, Bastin Tony Roy Savarimuthu, and Maryam A. Purvis. 2014. Extracting crime information from online newspaper articles. In *Proceedings of the Second Australasian Web Conference - Volume 155*, AWC '14, pages 31–38, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Angel X. Chang and Christopher Manning. 2012. SUTime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 3735–3740, Istanbul, Turkey. European Language Resources Association (ELRA).

Ellie Pavlick, Heng Ji, Xiaoman Pan, and Chris Callison-Burch. 2016. The gun violence database: A new task and data set for NLP. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1018–1024, Austin, Texas. Association for Computational Linguistics.

A Appendix: GitHub Link

https://github.com/husinthewei/CIS530P

B Appendix: Data Extraction Code

```
import pandas
from newspaper import Article
import numpy as np
import validators
import json
import argparse

# there are 239,677 pages

parser = argparse.ArgumentParser()

parser.add_argument('--start', type=int, required=True)
parser.add_argument('--end', type=int, required=True)
parser.add_argument('--out', type=str, required=True)

def main(args):
    articles = []

    def parse_url(x):
        article_url = x['source_url']
        iid = x['incident_id']
        if (validators.url(article_url)):
            article_dict = {}
            article = Article(article_url)
            article.download()
            if (article.html != ''):
                article.parse()
                article_dict['incident_id'] = iid
                article_dict['title'] = article.title
                if (article.publish_date):
                    article_dict['publish_date'] = article.publish_date.date().isoformat()
                else:
                    article_dict['publish_date'] = ''
                article_dict['text'] = article.text
                article_dict['shooting_date'] = x['date']
                article_dict['address'] = x['address']
                article_dict['n_killed'] = x['n_killed']
                article_dict['n_injured'] = x['n_injured']
                articles.append(article_dict)

    df = pandas.read_csv('gun-violence-data_01-2013_03-2018.csv')
    df = df.replace(np.nan, '', regex=True)

    df_slice = df[['incident_id', 'source_url', 'address', 'n_killed', 'n_injured', 'date']][args.start:args.end]

    df_slice.apply(parse_url, axis=1)

    #title = 'gv_data_2000.json'
    with open(args.out, 'w') as outfile:
        json.dump(articles, outfile)


if __name__ == '__main__':
    args = parser.parse_args()
    main(args)
```

C Appendix: Presentation Link

https://docs.google.com/presentation/d/1MT2pOGaIjk2FbP_MUGAxjeIieqqsWMGbW7XO-KfhL2o/edit?usp=sharing