Big Data Text Summarization - 2017 Westminster Attack

CS4984/CS5984 - Team 4 Project Report

Authors:

Aaron Becker
Colm Gallagher
Jamie Dyer
Jeanine Liebold
Limin Yang

Instructor:

Dr. Edward A. Fox

Virginia Tech
Blacksburg, VA 24061
December 15, 2018

Table of Contents

List of Figures

List of Tables

1 Abstract

Automatic text summarization, a process of distilling the most important information from a text document, is to create an abridged summary with software. Basically, in this task, we can regard the "summarization" as a function which takes a single document or multiple documents as an input and has the summary as an output. There are two ways that we can manage to create a summary: extractive and abstractive. The extractive summarization means that we select the most relevant sentences from the input and concatenate them to form a summary. Graph-based algorithm like TextRank, Feature-based models like TextTeaser, Topic-based models like Latent Semantic Analysis (LSA), and Grammar-based models could be viewed as approaches to extractive summarization. Abstractive summarization aims to create a summary similar to humans. It keeps the original intent, but uses new phrases and words not found in the original text. One of the most commonly used models is the encoder-decoder model, a neural network model that is mainly used in machine translation tasks. Recently, there is another combination approach that combines both extractive and abstractive summarization, like Pointer-Generator Network, and the Extract then Abstract model.

In this course, we're given both a small dataset (about 500 documents) and a big dataset (about 11,300 documents) that mainly consist of web archives about a specific event. Our group is focusing on reports about a terrorist event – Attack Westminster. It occurred outside the Palace of Westminster in London on March 22, 2017. The attacker, 52 year-old Briton Khalid Masood, drove a car into pedestrians on the pavement, injuring more than 50 people, 5 of them fatally. The attack was treated as "Islamist-related terrorism".

We first created a Solr index for both the small dataset and the big dataset, which helped us to perform various queries to know more about the data. Additionally, the index aided another team to create a gold standard summary of our dataset for us. Then we gradually delved into different concepts and topics about text summarization, as well as natural language processing. Specifically, we managed to utilize the NLTK library and the spaCy package to create a set of most frequent important words, WordNet synsets that cover the word, words constrained by part of speech (POS), and frequent and important named entities. We also applied the LSA model to retrieve the most important topics. By clustering the dataset with k-means clustering, and selecting important sentences from the clusters using an implementation of the TextRank algorithm, we were able to generate a multi-paragraph summary. With the help of named entity recognition and pattern-based matching, we confidently extracted information like the name of the attacker, date, location, nearby landmarks, the number killed, the number injured, and the type of the attack. We then drafted a template of a readable summary to fill in the slots and values. Each of these results individually formed a summary that captures the most important information of the Westminster Attack.

The most successful results were obtained using the extractive summarization method (k-means clustering and TextRank), the slot-value method (named entity recognition and pattern-based matching), and the abstractive summarization method (deep learning). We evaluated each of the summaries obtained using a combination of ROUGE metrics as well as named-entity coverage compared to the gold standard summary created by team 3. Overall, the best summary was obtained using the extractive summarization method, with both ROUGE metrics and named-entity coverage outperforming other methods.

2 Literature Review

The task of summarization can take on many forms. A summarization program can be tasked to summarize single documents, multiple documents, or a document with a query viewpoint. Furthermore, the summaries generated can be of several types, including indicative summaries (a description of the document, doesn't contain informative content), informative summaries (includes informative content, ideally includes all main ideas of the document), keyword summaries (words and / or phrases from the document(s)), and headline summaries (a summary reduced to one sentence) [15].

The two main methods for summarization are extractive and abstractive [19]. Extractive summaries are generated by selecting relevant words and phrases from the input and piecing them together, while abstractive summaries are generated by paraphrasing the input and potentially using new words and phrases not used in the input. One method of extractive summarization is the Graph Based method, which typically uses the TextRank algorithm [18, 17]. With TextRank, words or phrases are nodes in a graph and the relation between them are edges in the graph. The graph is then solved using something like the PageRank algorithm [20]. Another extractive method is the Feature Based model, where features are extracted from sentences and words to be used in determining importance [4]. Such features include sentence position in the document, length of the sentence, word frequency, and font style. Another method is the Topic Based model, where Latent Semantic Analysis is used to calculate the topic of the document and then the sentences are evaluated to determine what topics are included [21].

The encoder-decoder model is the main model used for abstractive summarization. The encoder is used to convert documents to latent representations, and the decoder generates a summary using these representations [11]. There are many challenges involved with this model, including the challenge of setting focus on important sentences or keywords, handling important but infrequent words, and handling long documents [15]. The extractive and abstractive approaches can be combined in a Pointer-Generator Network [25]. Another method is the Extract then Abstract model, where an extractive method is used to select sentences from the documents, then use the selected sentences as the input for an abstractive model [9].

3 Design and Implementation

As a group of 5 we worked on the steps of different types of units to create a summary of the "Attack Westminster". Since we started with varying backgrounds in Python and the NLTK toolkit, every group member started with Unit 1 to familiarize themselves with these tools before rewriting the code to use PySpark and be able to be run on the hadoop cluster.

3.1 A Set of Most Frequent Important Words

Initially, we found the most frequent important words without any kind of cleaning or modifying of the input sentences. We used NLTK word_tokenize and collections.Counter() to get our first results. These results were not what we desired. Instead of important words we unsurprisingly got the following tokens ",", ".", "the", "and", "of", "to", "in", etc.

Those results were filled with many stop words. To prevent this, we added a library of stop words in English as well as our own customized stop words ("'s", "said", "could", "also", "news",

Table 1: Results Unit 1 - Comparison of Small and Big Data Set and Bigrams

Small Dataset		**Big Dataset**		**Bigrams Big Dataset**
Word	**Count**	**Word**	**Count**	**Bigram**
police	1983	police	45011	westminster bridge
attack	1728	attack	39338	police officer
westminster	1327	westminster	30792	khalid masood
london	1176	people	26519	prime minister
people	1047	london	24099	keith palmer
parliament	815	parliament	18396	metropolitan police
masood	685	one	15705	theresa may
bridge	651	bridge	14446	posted mar
one	627	masood	13110	houses parliament
man	499	may	11057	palace westminster
may	480	officer	10722	westminster attack
security	469	us	10597	police officers
attacker	466	man	10532	emergency services
officer	466	security	10259	attack westminster
terrorist	442	attacker	10178	house commons
us	429	car	10154	terrorist attack
two	417	would	10003	armed police
incident	414	terrorist	9906	pc keith
terror	414	terror	979	22 2017
car	409	incident	9390	security services

“–”, “...”). After ignoring these words, we got a very good list of most frequent words for the small data set of Hurricane Sandy. To get even better results we lower cased every word and replaced “.” with “ ” so that words at the end of a sentence would count as well. Otherwise “police.” would be considered a different word than “polic”.

When we got the new event, “Attack Westminster” we slightly modified the Python script for the new data set by changing our customized stop words to be (“’s”, “said”, “could”, “also”, “news”, “–”, “...”, ““”, “””). Additionally we modified the script to use the ASCII encoding instead of UTF-8 encoding because we cleaned the small data set of “Attack Westminster” with jusText before running Unit 1 on it. The results for Unit 1 are shown in Table 1. The results for both data sets were very similar. These results are good for a first rough impression on the event but don’t help to creating a complete summary. To get more meaningful results we also searched for the most frequent bi-/trigrams. In these results we found many important terms including the whole name of the attacker and Prime Minister, and the location of the event. The results of the most frequent bigrams of the big data set are presented in Table 1. A complete version of Unit 1 results

Table 2: Results Unit 1 - Trigrams Obtained with Zepplin script

Trigram	**Count**
pc keith palmer	121
posted mar 22	72
mar 22 2017	72
minister theresa may	68
prime minister theresa	68
told buzzfeed news	60
outside houses parliament	60
westminster terror attack	48
pedestrians westminster bridge	47
the metropolitan police	47

is attached in the source code folder.

The results of Unit 1 present a good overview of what the event is about but it alone does not create an effective summary. This unit helped us get familiar with our dataset, helped us obtain better results in later units, and taught us how to work with tools that we would continue to use throughout the semester.

In addition to the Python scripts, we wrote code for Unit 1 on Zeppelin which used PySpark. We were able to get most frequent words, bigrams and trigrams from that as well. The results were the same but the script ran faster than our original Python script. The results of the small data set with the Zeppelin script were very good except for some noise in the data. With the exception of the sixth most frequent trigram, all of the trigrams are important for the event.

3.2 A Set of WordNet Synsets That Cover the Words

In this unit our goal was to find a better and more meaningful list of important words. We worked with the code of Unit 1 and modified it. For our first attempt to obtain these results, we added all synonyms of most frequent words to build a larger important word set which was not particularly better than our list and had no effect on frequency of synonyms.

For our second attempt we sorted the words used of each synset lemma name for the 500 most frequent words. That yielded much better results which are presented in Table 3.

We used lemmatization. For lemma words like attack and attacks you can see that they were combined for processing because the results are the same. Compared to Unit 1 lots of words are the same but we also found some new words like law and assailant which helped us to get a better understanding of the event and the use of the NLTK toolkit.

Table 3: Results Unit 2 - Small Data Set

Word	Final Results with Lemma	Frequency
policing	1.3417	0.0387
police	1.3347	1.3338
law	1.3264	-0.0076
attack	1.1601	1.1569
attacks	1.1601	0.1792
westminster	0.9063	0.9063
london	0.7912	0.7912
parliament	0.5547	0.5541
people	0.4853	0.4878
bridge	0.4307	0.4331
policeman	0.3855	0.0775
officer	0.3854	0.3080
officers	0.3854	0.2279
attacker	0.3690	0.3169
assailant	0.3690	0.0531

3.3 A Set of Words Constrained by POS

In this unit, we aimed to find more information about our dataset by taking a similar approach to our work in units 1 and 2, but looking specifically for the most frequent nouns and verbs. By filtering our results to these parts of speech, we hoped to find out what specific entities were involved, and any actions that occurred which could help determine what happened.

To accomplish this, we used the NLTK library's `pos_tag` method to tag our cleaned dataset by Part of Speech. One item we noticed when doing this was that this part of speech tagger relied on word casings, and surrounding text to determine the part of speech for each token. So, rather than passing it a lower-cased version of the text without stopwords, we retrieved better results (particularly with nouns) by passing the tagger the tokenized cleaned text. We then filtered out stopwords, and lower-cased them to count the dataset and get our most common nouns and verbs.

Overall, the results for both the small and large dataset were very similar, and were overall fairly descriptive of the event that took place. The results for both the small and big dataset are shown in Table 4.

3.4 A Set of Frequent and Important Named Entities

In this unit, we worked to determine the most common named entities found in our dataset. We utilized the spaCy library's named entity recognizer to help determine the named entities [26]. SpaCy's named entity recognizer uses a pretrained English model based on the OneNote 5.0 corpus to determine which phrases are considered named entities. In our case, we chose to use the

Table 4: Results Unit 3 - Most Common Nouns and Verbs in Small and Big Dataset

Nouns Big Dataset		**Nouns Small Dataset**		**Verbs Big Dataset**		**Verbs Small Dataset**	
police	41606	police	1842	told	8127	told	364
attack	37720	attack	1658	killed	7856	killed	335
westminster	30760	westminster	1325	including	6281	including	292
people	26514	london	1151	injured	5942	injured	260
london	24096	people	1042	arrested	5446	arrested	225
parliament	17530	parliament	780	posted	4907	shot	207
bridge	14410	masood	658	died	4814	died	203
masood	13105	bridge	646	know	4506	posted	184
officer	10722	man	486	shot	4399	released	179
man	10532	security	469	confirmed	4198	know	178
security	10259	officer	465	made	4165	made	175
car	10154	attacker	455	released	4061	known	173
attacker	10052	car	407	added	3769	stabbed	161
terror	9171	incident	387	saw	3718	confirmed	160
officers	9146	terror	383	say	3605	saw	159
incident	8831	palmer	352	going	3570	used	159
may	7851	terrorism	351	following	3407	according	154
minister	7696	officers	349	see	3375	going	152
palmer	7445	may	330	stabbed	3368	added	146
terrorism	7405	minister	311	left	3345	left	145

`en_core_web_md` model, as it had a better F-Score and Precision score than the `en_core_web_lg` model, and was almost 1/8th the size, ensuring faster run time. The named entities are shown in Table 5. These named entities were then used for slot value extraction in Unit 8.

SpaCy has several different types of Named Entities that could be extracted; some of the most important types to our group included:

- GPE (Geographic Political Entity) - Countries, cities, or states
- LOC (Location) - Non-GPE locations, mountain ranges, or bodies of water
- PERSON - People, including fictional
- DATE - Absolute or relative dates or periods
- CARDINAL - Numerals that do not fall under another type
- NORP - Nationalities or religious or political groups

- TIME - Times smaller than a day
- ORG - Companies, agencies, institutions, etc.
- LAW - Named documents made into laws

Table 5: Unit 5 Results - Set of Most Frequent Named Entities

Category	**Named Entities**									
GPE	london	932	westminster	663	birmingham	226	britain	221	uk	212
PERSON	masood	454	khalid masood	191	keith palmer	161	palmer	127	rowley	94
DATE	wednesday	221	today	220	thursday	184	yesterday	137	may	121
ORG	parliament	254	isis	139	islam	137	bbc	114	the house of commons	107
TIME	morning	56	this morning	47	night	41	last night	37	afternoon	36
FAC	westminster bridge	375	the palace of westminster	122	parliament square	87				
NORP	british	314	muslim	191	islamic	142	islamist	106	muslims	90
LOC	europe	32	west	15	premise	11	the river thames	10	the middle east	9
LAW	the terrorism act	8	the terrorism act 2000	6	start	5	sharia	3		

3.5 A Set of Important Topics

In order to aid in generating a better summary of our dataset, in this unit we ran Latent Semantic Analysis (LSA) (using the Gensim Python library [23]) to help identify important topics and concepts occurring throughout our dataset. Given the nature of the Westminster Attack we believed that the data set would be focused on a very small set of large topics. We thought this because compared to the events of other teams, the Westminster Attack was a very specific event that lasted under two minutes as opposed to multiple events occurring over a large span of time.

Initially we ran LSA on the small data set and generated the topics found in Table 6. From these topics we were able to see that the most frequent topic is exactly what we expected. It is the topic describing the details of the attack. Unfortunately the other topics we discovered aren't nearly as good. The second topic shows that even in our cleaned small data set there are many 404 error and display error web pages that persisted through our cleaning efforts.

Even though the topics from the small data set were very poor, we also ran LSA on our cleaned big data set. The top 15 topics generated from here can be seen in Table 7. Fortunately these topics seem much more reasonable as they focus on important people and aspects of the Westminster attack as opposed to the errors, JavaScript, and cookies that can be seen throughout the topics of the small dataset.

Table 6: Unit 6 Results - Set of Topics for the Small Dataset

Topic	**Keyword 1**	**Keyword 2**	**Keyword 3**	**Keyword 4**	**Keyword 5**
0	-0.196*polic	-0.192*attack	-0.158*masood	-0.135*offic	-0.129*parliament
1	0.524*error	0.420*display	0.392*page	0.281*provid	0.280*contact
2	-0.722*reserv	-0.645*right	-0.080*locat	-0.075*noth	-0.068*found
3	0.305*cooki	0.293*site	0.222*use	0.182*avail	0.165*longer
4	-0.265*masood	-0.218*buzzfe	0.146*parliament	-0.144*news	-0.127*cooki
5	0.440*locat	0.436*found	0.401*noth	0.382*look	0.217*search
6	0.642*avail	0.612*longer	0.231*site	-0.130*cooki	0.091*delet
7	-0.497*sun	-0.235*news	-0.171*complaint	-0.152*press	-0.150*standard
8	-0.186*page	0.146*locat	0.133*noth	-0.129*fundrais	0.128*parliament
9	-0.232*page	0.177*palmer	0.163*locat	0.139*noth	-0.132*buzzfe
10	0.267*page	-0.249*comment	-0.220*delet	-0.185*flag	-0.166*cancel
11	-0.202*page	-0.185*comment	-0.154*delet	0.131*edt	-0.127*flag
12	-0.168*edt	0.167*cooki	-0.161*page	-0.149*sun	-0.136*googl
13	-0.186*fundrais	-0.157*justgiv	0.118*email	0.118*e-mail	-0.117*money
14	-0.242*fundrais	-0.192*justgiv	-0.162*rais	-0.134*delet	-0.127*money

Table 7: Unit 6 Results - Set of Topics for the Big Dataset

Topic	**Keyword 1**	**Keyword 2**	**Keyword 3**	**Keyword 4**	**Keyword 5**
0	0.146*polic	0.129*masood	0.115*said	0.109*offic	0.103*parliament
1	0.632*trendolizer	0.251*trend	0.241*cooki	0.211*automat	0.186*data
2	-0.201*masood	-0.155*cristea	0.137*incid	0.122*car	0.116*reuter
3	-0.299*cristea	-0.204*andreea	-0.176*burnaz	-0.134*shine	-0.127*andrei
4	-0.189*editor	0.184*masood	0.177*arrest	-0.164*complaint	-0.161*contact
5	0.235*editor	0.197*contact	0.194*complaint	0.172*ipsoregul	0.171*dissatisfi
6	-0.253*barrier	-0.203*cyclist	-0.169*khater	0.168*wire	-0.149*crash
7	0.156*masood	0.143*barrier	0.140*cristea	0.132*khater	-0.127*palmer
8	-0.281*woman	-0.272*lorriman	-0.244*distress	-0.223*pictur	-0.174*jami
9	-0.407*privaci	-0.282*enjoy	-0.255*featur	-0.210*data	-0.206*better
10	-0.230*masood	0.205*mar	-0.179*share	-0.165*2.4k	-0.163*mailonline.co.uk
11	-0.398*aggreg	-0.187*elect	-0.176*news	-0.162*mar	-0.141*articl
12	-0.447*aggreg	-0.167*news	-0.157*perus	-0.151*snippet	-0.145*conveni
13	-0.374*elect	-0.177*cochran	-0.163*parti	-0.159*melissa	-0.152*kurt
14	0.163*share	0.151*2.4k	0.149*mailonline.co.uk	-0.136*aggreg	-0.122*woman

3.6 An Extractive Summary, as a Set of Important Sentences

In this unit, we aimed to create a summary by extracting and combining relevant sentences from our dataset into a meaningful summary. To accomplish this, we first needed to find an algorithm which would allow us to extract meaningful sentences from a large document. After researching several options such as TF-IDF, RAKE, and TextRank, we decided to use the TextRank algorithm, which had an easy-to-use implementation in the Gensim library.

The TextRank algorithm is based on Google's PageRank algorithm that Google creator Larry Page devised for Google Search. Gensim's implementation of this algorithm essentially treats each sentence as a vertex in a graph, and creates links between sentences to weight the sentences. The algorithm does not require any training model to generate a summary [6].

On a first attempt, we tested what the Gensim implementation of TextRank would output if we passed it a concatenation of all documents in our small dataset. The results contained duplicate sentences and the sentences it chose tended to be very large which were left after cleaning.

To remedy these two issues, we used SciKit-Learn's (`sklearn`) `TFIDFVectorizer` to generate a matrix which we could use to remove similar/duplicate documents within a certain degree. We also filtered out sentences that consisted of more than 50 words, as these would not be very helpful for our summary, and as shown would influence the TextRank results.

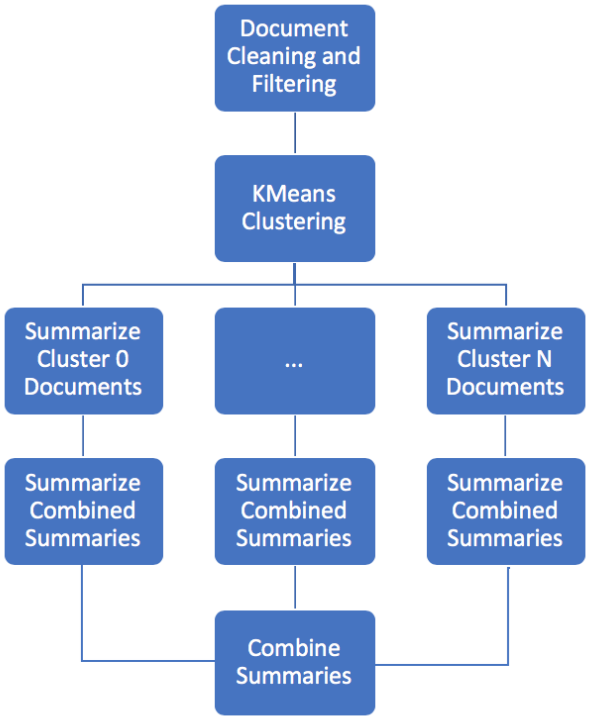

Figure 1: Unit 7 Final Pipeline

Five killed and 40 injured in Westminster terror attack. An assailant stabbed a policeman and was shot by police just outside Britain's parliament building in London on Wednesday (March 22) in what police described as a "terrorist incident". The ministry said minister Marco Minniti convened the Committee of Strategic Anti-terrorism Analyses following "the tragic facts in London," in which a vehicle mowed down pedestrians on a bridge and the attacker then stabbed a police officer outside the British Parliament. At least four people were killed and at least 20 injured in London on Wednesday after a car plowed into pedestrians and an attacker stabbed a policeman close to the British parliament, in what police called a terrorist incident.

While police believe Khalid Masood, a 52-year-old British man with a history of violent crimes, carried out the attack on Westminster Bridge and outside parliament alone, they are investigating what help he may have received and whether he had any accomplices. The attacker who killed three people near parliament in London before being shot dead was named on Thursday as British-born Khalid Masood, who was once investigated by MI5 intelligence officers over concerns about violent extremism. An attacker - now named as 52-year-old Khalid Masood - killed three pedestrians and injured around 40 other people as he mowed down members of the public with a car on Westminster Bridge at about 2:40pm, before crashing into the railings in front of Parliament. An attacker - now named as 52-year-old Khalid Masood - killed three pedestrians and injured around 40 other people as he mowed down members of the public with a car on Westminster Bridge at about 2:40pm, before crashing into the railings in front of Parliament.

Counter-terrorism detectives have been investigating Masood's life and associations across Britain in a race to discover what led him to run down pedestrians on Westminster Bridge, crash his vehicle outside parliament and attack two police officers. Security in the area has been noticeably more high profile since Khalid Masood killed four people by ramming them with his car on Westminster Bridge before stabbing to death Pc Keith Palmer just inside the grounds of Parliament on March 22. By the time of Wednesdays attack on Westminster Bridge, he had seamlessly gravitated to Birmingham, a city increasingly enveloped by sharia enclaves that, to varying degrees, have become no-go zones for non-Muslims and agents of the state, including police. Three civilians, and the Islamic terrorist, Khalid Masood, have died and at least 40 have been injured after an attacker drove a car along a pavement in Westminster, stabbed a policeman and was shot dead by police in the grounds of Parliament. Attacker Khalid Masood plowed through crowds of pedestrians on Westminster Bridge last Wednesday, fatally injuring three people and wounding dozens before stabbing veteran police officer Keith Palmer to death in the grounds of Parliament. After Trumps tweet, a spokesperson for Khan said: The mayor is busy working with the police, emergency services and the government to coordinate the response to this horrific and cowardly terrorist attack and provide leadership and reassurance to Londoners and visitors to our city. Travis Frain, 19, received abusive messages via social media from Islamist and far-right wing extremists who suggested he was making up his unwanted role in the attack by a British-born terrorist who killed five and injured 50 in London in March last year. Given that the terrorist who perpetrated the Westminster attack actually ran through the open gates which lead into Old Palace Yard, killing a police officer in doing so, may result in new security precautions being implemented there. "The perpetrator of the attacks yesterday in front of the British parliament in London is an Islamic State soldier and he carried out the operation in response to calls to target citizens of the coalition," the group's Amaq news agency said in a statement. The Islamic State has claimed responsibility for the terror attack that left four people dead, including the assailant and a police officer, outside of London's Parliament building on Wednesday, according to the Amaq media agency. Five people died and 50 were injured after an attacker named by the police as Khalid Masood born Adrian Russell Ajao drove a car across Westminster Bridge in London, mowing down pedestrians before crashing to the railings of the Houses of Parliament. Four people were killed and 40 more injured on Wednesday when lone-wolf attacker Khalid Masood drove his car through crowds on Westminster Bridge before crashing into the parliament building and fatally stabbing an unarmed officer.

What We Know About the Suspected Terrorist Attack Outside the U.K. Parliament A man has been arrested on suspicion of terrorism offenses after a car was driven into barriers near the UKs Houses of Parliament in Westminster, London on Tuesday morning, injuring three people.

Figure 2: Results of Unit 7 - TextRank on K-Means Clusters of Big Dataset

With these changes in place, we were able to extract sentences that were descriptive of our dataset, and limit the output's size by either a ratio of its original size or by a maximum word count. However, we wanted to break this large block-like summary into paragraphs. To accomplish this, we would need to group similar documents or sentences together using either clustering or topic modeling.

Because we were already using `sklearn` for TF-IDF, we decided to use its `KMeans` clustering algorithm as well, to cluster our documents into `N` clusters. We had the choice of clustering at different stages of summarization, for example we could cluster the documents before summarizing them, after summarizing them, or by clustering the sentences rather than the documents. After testing several methods, we chose to cluster the documents first, then summarize each document in the cluster, combine the summaries together, and summarize them again, combining the summaries for each cluster together as separate paragraphs. This process can be seen mapped out in Figure 1.

The final summary we obtained with our big dataset used 5 clusters, and used a limited set of our big dataset which was obtained by ranking every document by its inclusion of frequent words obtained in Unit 1, and selecting a percentage of the highest ranked documents. Ideally, we would have used a classifier to do this, however this basic method worked well for our purposes.

It is also important to note that the TextRank algorithm and clustering algorithm are not deterministic functions, and will not produce the same output for each run, even if the input is identical.

3.7 A Set of Values for Each Slot Matching Collection Semantics

The next form of summary that was attempted was to create a set of values that could be extracted from our text and could later be used to fill a generic template summary for our event type in Unit 9. To successfully accomplish this, we needed to determine:

1. The value types we intended to extract from our text
2. The methods for extracting each value (Note: Some values may be more susceptible to certain methods than others.)
3. The decision method for selecting which retrieved information best represents our data

First, our team brainstormed to determine the items which we would need to extract. Based on the "TAC 2011 Guided Summarization Task Guidelines" [27], for a criminal or terrorist attack, the main areas of focus should be:

- **What**: what happened
- **When**: date, time, other temporal placement markers
- **Where**: physical location
- **Perpetrators**: individuals or groups responsible for the attack
- **Why**: reasons for the attack
- **Who Was Affected**: casualties (death, injury), or individuals otherwise negatively affected by the attack

- **Damages**: damages caused by the attack
- **Countermeasures**: countermeasures, rescue efforts, prevention efforts, other reactions to the attack (e.g. police investigations)

We took these items into consideration, and came up with the following features to extract:

- **Date**: When the attack occurred
- **Location**: Where the attack occurred
- **Nearby Features**: Places near where the attack occurred
- **Attacker Name**: Who was involved in the attack
- **Type of Attack**: What happened, was it a shooting, bombing, etc.
- **Number Killed**: How many fatalities were there
- **Number Injured**: How many non-fatal injuries were there

Ideally, we had hoped to obtain other items such as the motivations of the attacker, countermeasures taken, and some more particulars on what happened, however these items were less measurable, and best represented as sentences rather than words or phrases like we were trying to extract.

As for the means of extracting these items, the simplest and greediest method to approach this is using regular expressions. In our initial attempt, we built regular expressions that could extract several of our intended items from our dataset by searching for keywords that we expected to surround it. Some examples of regular expressions we used can be seen in Table 8.

Table 8: Unit 8 - Regular expressions from initial attempt

Value	Regular Expression
Location	`(?:^|\s+)(?:in|In|at|At|on|On)\s+((?:[A-Z][A-Za-z]+,?\s+)*(?:[A-Z][A-Za-z]+))`
Killed	`(?:^|\s+)((?:\w+\s+){0,4}\w+)\s+(?:people have died|people died|people dead|died|victims|dead)`
Injured	`(?:^|\s+)(\w+)\s+(?:(?:\w+\s+){0,2}injured|(?:\w+\s+){0,2}hurt)`

Once we had obtained the values from the regular expression matching, it seemed simplest to select the most common value as our result, and generally that seemed to work well. However, we quickly found that building these regular expressions were becoming increasingly complicated, and when it came to extracting values such as the number killed, or the number injured, things became even more difficult. Numbers can be represented as digits (with or without commas separating them), as words, and as approximations. When it comes to news articles, the latter two become increasingly prevalent, and it would throw off our results to simply omit them. To solve this problem, we used the `word2number` and `num2words` libraries available in Python to convert between these, as well as creating a custom method to solve approximations represented by phrases such as "more than", "close to", "approximately", etc.

Even with these changes, the regular expressions were not giving us results that we felt confident in, and items such as the **Type of Attack** and **Nearby Features** were seemingly impossible to

obtain using this method. With that in mind, we began looking into the possibility of using part of speech tagging or named entities that we had obtained in Units 3 and 5, respectively, to aid our extraction.

We discovered that a library called spaCy had multiple trained models for named entity recognition, as well as a pattern based matcher that could be used to extract features. Not only that, but the pattern based matcher could match lematizations of words, named entities, and parts of speech to query for features, allowing us to make much more robust and focused queries. Results using this extraction method can be seen in Table 9.

Table 9: Unit 8 - Results Using SpaCy Pattern Matcher with Named Entities on Small Dataset

Query Name	**Extracted Value**	**Count**
Type of Attack	Terrorist Attack	119
Attacker	Khalid Masood	50
Location	London	358
People Killed	4	81
Date	Mar. 22, 2017	72
Nearby Feature	Westminster Bridge	242
People Injured	50	35

Looking at the results, only two items received questionable results: **People Killed** and **People Injured**. At the time when most reports about the Westminster attack were written, these numbers are perfect. However, one victim who was badly injured by the attack died later that month, increasing the number of people killed to 5, and decreasing the number injured to 49. Also, if you consider the attacker who was killed as well, the number of people killed could be 6, however, we were intending to find the number of civilians killed, so we were fine with this result. Perhaps if we were able to prioritize more recently published documents in our counts, we may have had more luck obtaining a correct death count.

3.8 A Readable Summary Explaining the Slots and Values

After extracting values out of the data set we created a general summary with slots about any kind of attacks. We tried to keep the template as general as possible so that it can be used for data sets about other attacks. For this reason the summary about the terrorist attack is not detailed in connection with the Westminster attack which was a very unique attack and not similar to e.g., a shooting.

Using the template in Figure 3 and the slot values extracted in the previous unit, we produce a good (although short) readable summary as shown in Figure 4.

3.9 An Abstractive Summary

One of the biggest advantages for the abstractive approach to generate a summary is that it can use words that were not in the original documents and make the summary more fluent and natural. Since extractive and abstractive are not conflicting ways, in this part, we will show what we tried

Template

On DATE a TYPEOFATTACK occurred in LOCATION near NEAR. The police investigated that the attacker was ATTACKER. During the TYPEOFATTACK the attacker killed KILLED people and injured INJURED.

Figure 3: Unit 9 - Template

Result

On Mar. 22, 2017 a terrorist attack occurred in London near Westminster Bridge. The police investigated that the attacker was Khalid Masood. During the terrorist attack the attacker killed 4 people and injured 50.

Figure 4: Unit 9 - Result

with a combined approach - Pointer Generator Network [25].

Basically, it combines the extractive and abstractive models by switching probability. The probablity is calculated for each decoder timestamp to determine suitability of generating words from the vocabulary or copying words from the source text. An overview of the model is shown in Figure 5.

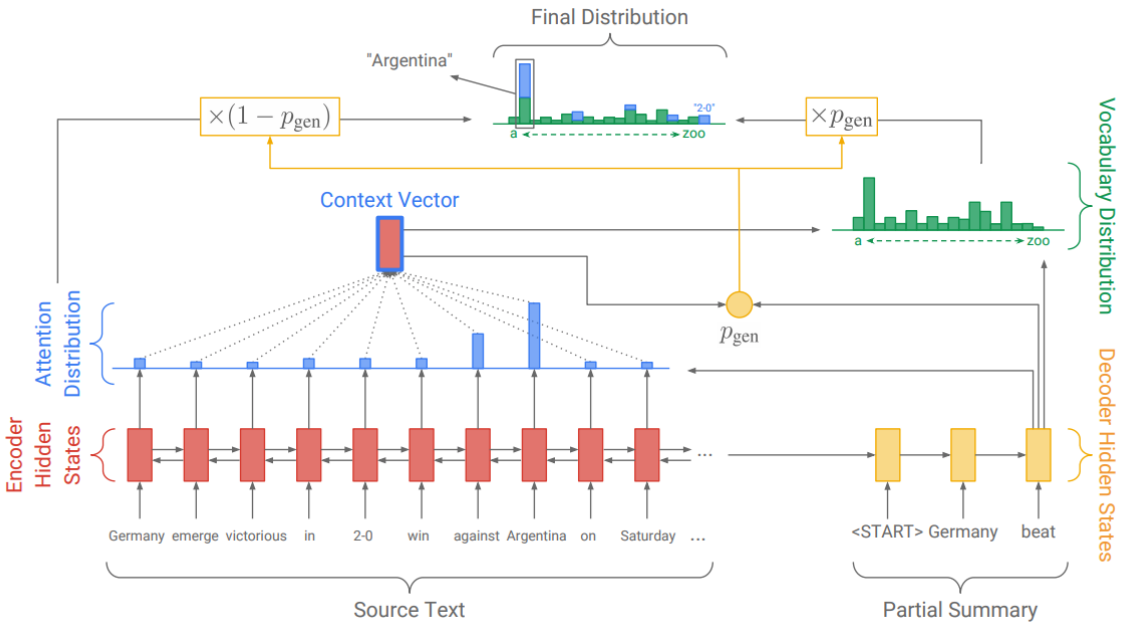

Figure 5: An overview of Pointer-generator Model [25].

Due to time limitations, we used a pre-trained model that was trained on the CNN/Daily Mail dataset. We test it in two ways; one is using the articles extracted from Unit 7, which is a limited set (about 999 articles) of our big dataset which was obtained by ranking every document by its

inclusion of frequent words obtained in Unit 1. Technically, this limited set contains about 25% of our big dataset (after jusText cleaning and removing duplicates). Another way is to classify the small dataset into different topics as described in Unit 6. In the preprocessing part, we select the most 3 important topics and concatenate the corresponding articles together, and then run Pointer-generator Network on each topic to generate a summary. Finally, we combined the generated 3 summaries together based on their logic.

There are a few hyperparameters we could tune:

- max_enc_steps (default is 400): The max time steps of the encoder (max source text tokens).
- max_dec_steps (default is 100): The maximum sequence length of the generated summary. If made too large, the summary starts to repeat itself. The longer the source article, the less repetition, as the value of max_dec_steps increases.
- min_dec_steps (default is 35): The minimum sequence length of the generated summary.
- beam_size (default is 4): Beam size for beam search decoding. The larger the beam size, the more detailed information is summarized, but processing takes longer.

Following are the hyperparameters we used, as suggested by Team 7:

- max_enc_steps = 1000
- max_dec_steps = 2000
- min_dec_steps = 850
- beam_size = 8

Before we feed our data into the pre-trained model, we need to convert our dataset into the expected format. Thanks to Chreston Miller from Team 7, we follow his guidelines at `https://github.com/chmille3/process_data_for_pointer_summrizer` to generate the binary file. The code for the Pointer-generator network is available at `https://github.com/abisee/pointer-generator`. But note, to run the code, we'd better follow the instructions from the README of the pre-trained model.

The command we used to generate an abstractive summary is as follows:

```
python run_summarization.py --mode=decode --data_path=test_000.bin
--vocab_path=vocab --log_root=logs/ --exp_name=pretrained_model_tf1.2.1
--max_enc_steps=1000 --max_dec_steps=2000
--min_dec_steps=850 --beam_size=8 --coverage=1 --single_pass=1
```

For the first methodology, the result is as follows:

the campaign group cage has expressed sympathy for todays victims. it also called on the uk government to enact policies that will bring an end to the global cycles of violence. reactionary policies in the uk have a global ramification and it is important that the government leads with policies that seek to end the global cycles of violence rather than further them. the united kingdoms threat level has been set at severe for some time and this will not change. acting deputy commissioner mark rowley said earlier that the officer who was killed was armed with a knife, then ran towards parliament, where he was confronted by the police officers who keep us and our democratic institutions safe. for those of us who were in parliament at the time, these events provide a particular reminder of the exceptional bravery of our police and security services who risk their lives to keep us safe. once again today, these exceptional men and women ran towards the danger even as they encouraged others to move the other way. together, we will remain strong and smart as we confront terrorists and defeat the threats to our nations. the united kingdom will always have a friend and partner in the american people. we will continue to work together with the uk and all our allies to show the world that echoes in some of the furthest corners of the globe. our friendship is based on shared values and history indeed, canadas parliament is a descendant of the chamber targeted this morning. the canadian parliament withstood a similar attack not so long ago by those who sought to instil fear and divide canadians against themselves. instead, canadians came together to celebrate the values of liberty, democracy and freedom of speech. the canadian prime minister 's statement on today 's attack i have just chaired a meeting of the governments emergency committee, cobra, following the sick and depraved terrorist attack on the streets of our capital this afternoon. the full details of exactly what happened are still emerging. i am confident the british people will do the same, and will emerge from their grief stronger and more united than ever before. on behalf of all canadians, i want to pay tribute to them and to all our emergency services for the work they have been doing to reassure the public and bring security back to the streets of our capital city. our thoughts and prayers are with the victims, and their family and friends who waved their loved ones off, but will not now be welcoming them home. i can confirm that this appalling incident began when a single attacker drove his vehicle into pedestrians walking across westminster bridge, killing two people and injuring many more, including three police officers. it has now been confirmed that he was not armed, scotland yard says. the special relationship between the united kingdom and the united states has been critical to securing peace around the world. the united kingdoms threat level has been set at the heart of our capital city, where people of all nationalities, religions and cultures come together to celebrate the values of freedom that echoes in some of the furthest corners of the globe. world leaders have condemned the attack : i was shocked and saddened to learn of the innocent people who were killed and injured as a result of this brutal terrorist attack in london. the canadian parliament is an attack on democracies around the world and the uk are the closest of friends and allies. acting deputy commissioner mark rowley will give a further operational update later this evening. the full cooperation and support of the united states are engrained with a spirit of freedom that require very little planning can not be thwarted by ever more securitisation and policing of communities. canadians stand united with the british people in the fight against terrorism. we stand ready to offer all possible assistance to the british government, to do what we can to bring to justice. together, we accept that the security services and the police play a crucial role in keeping the public safe, such attacks which require very little planning can not be thwarted by ever more remarkable. acting deputy commissioner rowley will give a call to the prime minister, justin trudeau, also released a statement : today, our thoughts and prayers go out to all who have been affected ; to the victims themselves and their family and friends who waved their own in todays attack only makes their calmness and professionalism under pressure all the more remarkable. on behalf of the whole country, i share in the outrage and horror at this brutal terrorist attack on the uk parliament is also shot.

Figure 6: Results of Unit 10 - Abstractive Summary Generated with Pointer-generator Network.

We can see that the generated summary seems to focus on the government's action and attitudes and omits some details like the attacker's name. The result for the second methodology is not good, mainly because we ran it on the small dataset instead of the big dataset. Thus the topic may not be completely representative. Improving this result is a reasonable next goal for any future work.

4 Evaluation

Table 10: ROUGE Results for Generated Summaries of Dataset

	ROUGE-1	**ROUGE-2**	**ROUGE-L**	**ROUGE-SU4**
Extractive Summary (Unit 7)	0.21429	0.03704	0.10714	0.05263
Slot-Value Summary (Unit 9)	0.18519	0.03846	0.18519	0.07534
Abstractive Summary (Unit 10)	0.07407	0.0	0.07407	0.0137

To evaluate the summaries our team generated, we used a script provided by the GTA for this course, Liuqing Li. The script performed 3 separate evaluations which compared our generated summary with the Golden Standard summary created by Team 3. Those evaluations are as follows:

1. ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4 evaluations on full summaries
2. Best sentence ROUGE-1 and ROUGE-2 scores
3. Named Entity coverage

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is a software tool for evaluating generated summaries compared to a human produced summary. ROUGE-N results (ROUGE-1 and ROUGE-2) are the overlap of N-grams between the generated and golden standard summaries, ROUGE-L results measure the overlap of longest common sub-sequences (LCS), and ROUGE-SU4 measures the overlap of skip-grams with a max length of 4 as well as unigrams [33].

These different metrics provide unique approximations of the generated summaries. For example, ROUGE-1 scores indicate relation of words used, while ROUGE-2 scores indicate similar words used in conjunction to each other. ROUGE-SU4 is similar to ROUGE-2, but allows more leniency for words between bigrams such as adjectives, punctuation, or stop-words [13]. The results of evaluating our summaries using ROUGE-1, ROUGE-2, ROUGE-L, and ROUGE-SU4 are shown in Table 10.

ROUGE-1 and 2 evaluations were also performed on each sentence in our generated summary, to determine the sentences with the highest ROUGE-N scores. These results, shown in Table 11, showed which sentences from our summaries were the most alike to sentences in the golden standard. If there are no strongly related sentences, that is an indicator that the summary most likely does not accurately portray any topic of the golden standard.

The final evaluation method utilized spaCy's named entity recognizer (similarly to the work done in Section 3.4) to determine the coverage of named entities from the golden standard in our generated summaries. The results for our generated summaries can be found in Table 12.

Overall, based on these evaluations in combination with personal evaluations by our team members, we believe that our Extractive Summary from Unit 7 (as described in Section 3.6) is the most representative summary that our team generated for our dataset. Overall, as seen in Tables 10, 11, and 12, the extractive summary yields higher scores on most ROUGE tests, as well as entity coverage. The slot-value summary (as described in Section 3.8) had some very good ROUGE scores, however it is clear to see from the Named Entity evaluation that its small size neglected to include

Table 11: ROUGE-1 and ROUGE-2 Sentences and Scores for Generated Summaries of Dataset

		ROUGE-1 Sentence	**ROUGE-1 Score**	**ROUGE-2 Sentence**	**ROUGE-2 Score**
Extractive Summary (Unit 7)	Predicted	An attacker - now named as 52-year-old Khalid Masood - killed three pedestrians and injured around 40 other people as he mowed down members of the public with a car on Westminster Bridge at about 2:40pm, before crashing into the railings in front of Parliament.	0.71429	While police believe Khalid Masood, a 52-year-old British man with a history of violent crimes, carried out the attack on Westminster Bridge and outside parliament alone, they are investigating what help he may have received and whether he had any accomplices.	0.33333
	Golden	The attacker was later identified as 52 year old Briton Khalid Masood.		The attacker was later identified as 52 year old Briton Khalid Masood.	
Slot-Value Summary (Unit 9)	Predicted	The police investigated that the attacker was Khalid Masood.	0.42857	The police investigated that the attacker was Khalid Masood.	0.20000
	Golden	The attacker was later identified as 52 year old Briton Khalid Masood.		He changed his name to Khalid Masood after he converted to Islam.	
Abstractive Summary (Unit 10)	Predicted	i can confirm that this appalling incident began when a single attacker drove his vehicle into pedestrians walking across westminster bridge, killing two people and injuring many more, including three police officers.	0.62500	i can confirm that this appalling incident began when a single attacker drove his vehicle into pedestrians walking across westminster bridge, killing two people and injuring many more, including three police officers.	0.42857
	Golden	Five died in the rampage, including a police officer and more than 50 people were injured.		Five died in the rampage, including a police officer and more than 50 people were injured.	

Table 12: Named Entity Coverage for Generated Summaries of Dataset

	Percent Coverage	**Common Entities**
Extractive Summary (Unit 7)	13.68%	'Khalid Masood', 'Keith Palmer', 'Adrian Russell', 'Westminster', 'Westminster Bridge', 'London', 'The Houses of Parliament', 'Islamic', '52-year-old', 'five'
Slot-Value Summary (Unit 9)	3.16%	'Westminster Bridge', 'Khalid Masood', 'London'
Abstractive Summary (Unit 10)	1.05%	'two'

many of the subjects that were covered in the golden standard. Finally the abstractive summary (as described in Section 3.9) had relatively poor results, which could have been improved based on other teams' work, we believe that a similar method to our proposed one could yield more positive results.

5 Gold Standard

5.1 Approach

We wrote a gold standard about the Earthquake New Zealand dataset. From prior knowledge we knew about the largest devastating earthquake in the last decade. Thus, we started to write a summary about Christchurch Earthquake 2011. Our first step was to become familiar with the event by reading the Wikipedia article and a historical government page of New Zealand. After writing a decent summary of the Christchurch earthquake, we checked the Solr index and noticed that only about 10% of the articles contain the words "201" and "Christchurch". To avoid doing unnecessary work, we checked the Solr index and found the articles in Solr were about the earthquake in 2016 in Kaikoura, 2011 in Christchurch, and smaller earthquakes without any damage. With this knowledge, we decided to write a gold standard about the two massive earthquakes. To write the gold standard, we focused on our own internet searches and webpages we found about these two events. Having all the information we created a draft of the gold standard. The sources used for generating the Earthquake New Zealand gold standard are shown in Table 13.

Table 13: Sources Used for Generating Gold Standard Summary

https://en.wikipedia.org/wiki/2011_Christchurch_earthquake	[31]
https://nzhistory.govt.nz/page/christchurch-earthquake-kills-185	[10]
https://teara.govt.nz/en/historic-earthquakes/page-13	[28]
https://www.cnn.com/2016/11/14/asia/new-zealand-earthquake/index.html	[7]
https://www.tvnz.co.nz/one-news/new-zealand/billions-needed-repair-kaikoura-roads-after-7-5-earthquake-says-pm	[8]
https://en.wikipedia.org/wiki/2016_Kaikoura_earthquake	[32]
https://www.theguardian.com/world/2016/nov/13/new-zealand-south-island-hit-by-74-magnitude-earthquake	[29]
https://www.stuff.co.nz/national/nz-earthquake/86654229/Timeline-of-the-7-8-quake-and-response-reveals-plenty-of-room-for-improvement%20last%20accessed%20March%202017	[30]
https://nzhistory.govt.nz/culture/february-2011-christchurch-earthquake	[12]
https://my.christchurchcitylibraries.com/canterbury-earthquake-2011-for-kids/	[2]
https://my.christchurchcitylibraries.com/canterbury-earthquake-2010-for-kids/	[3]
https://www.bbc.com/news/world-asia-pacific-12533291	[5]
https://www.britannica.com/event/Christchurch-earthquakes-of-2010-2011	[22]
https://teara.govt.nz/en/historic-earthquakes/page-15	[28]
https://www.sciencelearn.org.nz/resources/343-liquefaction	[16]
https://www.gns.cri.nz/Home/Our-Science/Natural-Hazards/Recent-Events/Canterbury-quake/Hidden-fault	[24]
https://www.odt.co.nz/news/national/11000-aftershocks-christchurch-quake	[1]

When we received feedback on our draft, we noticed that our first attempt for the gold standard was not as detailed as it was expected to be. We studied the comments carefully and reworked the draft.

Thank you to Areeb Asif, Michael Goldsworthy, Brendan Gregos, and Thoang Tran (Team 3) for creating the gold standard for our event. In the next two sections we will show both the gold standard we created for the New Zealand Earthquakes and the one created for us. The gold standard

for Attack Westminster was used to obtain ROUGE scores seen in the evaluation section.

5.2 Gold Standard Earthquakes - New Zealand

On Tuesday February 22, 2011 at 12:51 pm, an earthquake of magnitude 6.3 hit Christchurch, New Zealand, a South Island city of nearly 400,000 people. The country's deadliest natural disaster in 80 years killed 185 people from more than 20 countries and injured several thousand, 164 seriously. The earthquake epicenter was near Lyttelton, just 10 kilometers south-east of Christchurch's central business district at a depth of 5 kilometers. The earthquake, caused by a hidden fault, occurred more than five months after the 4 September 2010 earthquake, but is considered to be one of the more than 11,000 aftershocks of the earlier quake. In the ten minutes after the February 22 quake, there were 10 aftershocks of magnitude 4 or more. At 1:04 a magnitude 5.8, and at 2:50pm a magnitude 5.9, quake followed. The earthquakes stemmed from the deformation along regional plate boundaries where the Pacific and Indo-Australian tectonic plates push against one another. Although not as powerful as the magnitude 7.1 earthquake on 4 September 2010 which happened at night and was the first devastating earthquake of the current decade, causing some 3 billion pounds damage, this Canterbury region earthquake occurred on a shallow fault line that was close to the city, and there was very high ground acceleration, so the shaking was particularly destructive. It was lunchtime and many people were on the city streets. 115 people died in the Canterbury Television (CTV) building, 18 in the Pyne Gould Corporation (PGC) building, 36 in the central city (including 8 on buses crushed by crumbling walls), and 12 in the suburbs (including from falling rocks in the Redcliffs, Sumner, and Port Hills). The Chief Coroner determined that another four deaths were directly associated with the earthquake.

The earthquake brought down many buildings damaged in the previous September, especially older brick and mortar buildings. Up to 100,000 buildings were damaged and about 10,000 buildings were deemed unsalvageable, needing to be demolished. Heritage buildings suffered heavy damage, including the Provincial Council Chambers, Lyttelton's Timeball Station, the Anglican Christchurch Cathedral, and the Catholic Cathedral of the Blessed Sacrament. More than half of the buildings in the central business district have since been demolished, including the city's tallest building, the Hotel Grand Chancellor. Over a quarter of the buildings in the central business district were demolished.

Liquefaction was much more extensive than in the September 2010 earthquake, which is known as the Greendale, Darfield, Rolleston, September or Canterbury Earthquake. During the 2010 and 2011 Canterbury earthquakes, over 400,000 tons of silt came to the surface. Many roads, footpaths, schools and houses were flooded with silt. Properties and streets were buried in thick layers of silt, and water and sewage from broken pipes flooded streets. House foundations cracked and buckled, wrecking many homes. Despite the damage to homes, there were few serious injuries in residential houses in liquefaction areas. However, several thousand homes had to be demolished, and a large area of eastern Christchurch will probably never be reoccupied.

The government activated the National Crisis Management Centre and declared a national state of emergency the day after the quake. Authorities quickly cordoned off Christchurch's central business district. Christchurch's CBD remained cordoned off until June 2013. Power companies restored electricity to 75 percent of the city within three days, but re-establishing water supplies and sewerage systems took much longer. Rescue crews came to help from all over the world, especially Japan, the United States, the United Kingdom, Taiwan, and Australia.

In the weeks following the earthquake about 70,000 people were believed to have left the city due to uninhabitable homes, lack of basic services, and continuing aftershocks. Roads, bridges, power lines, cell phone towers, and ordinary phone lines were damaged. The water and sewage pipes were badly damaged. Many people needed to use portable or chemical toilets, and got their water from tankers for months after the quake. Timaru's population swelled by 20% and thousands of pupils registered at schools in other cities and towns. Many returned to Christchurch as conditions improved.

Several earthquakes happened during the following years but the next devastating and geological complex earthquake occurred in 2016 near the tourist town Kaikoura.

On November 14, 2016 at 12:02 am, a 7.8 magnitude earthquake, the second strongest quake since European settlement, struck 15 kilometers northeast of Culverden and 60 kilometers southwest of Kaikoura, lasting for nearly 2 minutes. There were 2 casualties with another 57 people treated for injuries. One of the casualties was a man who had been crushed by his collapsing home outside of Kaikoura. Two of his family members were rescued from the rubble. The other casualty was a woman who had a head injury during the earthquake. The earthquake also significantly damaged many roads, buildings, and bridges. Prime Minister John Key estimated that costs to fix all the infrastructure was in the billions.

Most significantly, the earthquake, which brought down between 80,000 and 100,000 landslides, effectively cut off all roads from Kaikoura to the rest of the country. By November 16, 200 people had been airlifted out of Kaikoura, while another 1,000 waited for evacuation. Many of the people who were stranded in Kaikoura were tourists, which elicited other nations to help in the evacuation efforts. The US volunteered the use of two navy helicopters. China sent helicopters to evacuate Chinese tourists.

Restoration of roads between Kaikoura and the rest of the country took a couple of weeks. On November 30, the designated Kaikoura Emergency Access Road was reopened to civilian drivers with permits. By the end of December 2016, the Kaikoura Emergency Access Road and State Highway 1 south of Kaikoura were completely reopened. The Main Rail North line was also severely damaged by the earthquake. It was partially restored by September 2017, but passenger service will not resume until December 2018. Large areas of the South Island's northeast coast shifted upward and northward; about 20 kilometers of the Marlborough coast was raised between 0.5 and 2 meters.

The tsunami following the 2016 earthquake had a maximum height of 7 meters at Goose Bay. After the earthquake, the tide level dropped 2.5 meters over 25 minutes. The tide level then rose 4 meters over the next 15 minutes. The tsunami caused some additional destruction, but the impact was minimized because the tsunami occurred at mid to low tide. One building was heavily damaged. The tsunami also scattered marine plants and animals across the Oaro River flood plain. The tsunami caused no additional casualties or injuries.

5.3 Gold Standard - Attack Westminster

On Wednesday March 22nd 2017, an attack took place at the Houses of Parliament in London. A car was driven at high speed across Westminster Bridge - hitting victims as it went. The horror

began at the south London end of Westminster bridge when a driver in a Hyundai Tucson jumped onto the pavement at around 2.30pm, mowing down pedestrians. He continued along the pavement, plowing through at least a dozen victims in total, before coming to a stop at the Houses of Parliament where he crashed his vehicle into railings. The driver then, armed with a knife, tried to make his way into the Parliamentary estate. The assailant killed an unarmed police officer before being shot dead by firearms officers just inside the gates. The attacker was later identified as 52 year old Briton Khalid Masood. It is strongly believed that Masood was inspired by Islamic extremism.

Five died in the rampage, including a police officer and more than 50 people were injured. The five victims who lost their lives were PC Palmer, mother-of-two Aysha Frade, US tourist Kurt Cochran, Romanian visitor Andreea Cristea, and pensioner Leslie Rhodes from south London. Aysha Frade, who worked as a Spanish teacher, was making her way across Westminster Bridge en route to collect her children from school when she was mowed down by the terrorist. Frade had only just left her place of work at the nearby DLD College London and was on her way to pick up her two children when fate intervened. Kurt Cochran was an American on European tour with his wife. The man, in his 50s, is believed to have been killed while walking on Westminster Bridge. Cochran was visiting London from Utah to celebrate his 25th anniversary with his wife Melissa, who was among the injured. Romanian Andreea Cristea, 31, who was visiting London with her boyfriend Andrei Burnaz, was knocked into the river Thames after being struck by the car driven by Khalid Masood as she strolled along Westminster Bridge with her boyfriend. She later died from her injuries. Leslie Rhodes, 75, a retired window cleaner, was on the way to a hospital appointment at the time. The police officer killed was PC Keith Palmer, 48, an unarmed police officer who was on duty with the Parliamentary and Diplomatic command. The 52-year-old attacker was shot by armed police and was the first casualty to arrive at St Mary's Hospital in Paddington, where he was pronounced dead. Among those injured were people from Ireland, America, South Korea, Romania, France, Poland and Germany. They were taken to St. Thomas' Hospital, which faces the Palace of Westminster across the Thames and also to King's College Hospital and the Royal London Hospital.

Born in Kent, Masood was brought up in Rye, East Sussex and later attended secondary school in Tunbridge Wells in Kent. Most recently he was living in the West Midlands. When he was 16, he dropped out of school and by 18 he was described as a heavy drug user. In 2000, he was sentenced to two years in prison for stabbing and slashing the face of a cafe owner at Northiam in Sussex. In 2003, he was sentenced to a further six months in prison for possession of an offensive weapon following another knife attack in Eastborne in Sussex. As well as these two prison terms, Masood had convictions for public order offenses going back to 1983. Masood was not known to be a religious person and frequently went to pubs.

Masood reportedly converted to Islam in 2005. He was described by police as a criminal with a twenty year record of offending. He was born Adrian Russell Elms, but later changed the name to Adrian Russell Ajao when he took the name of his stepfather. He changed his name to Khalid Masood after he converted to Islam. The 52-year-old had recently split from his partner, Jane Harvey, following bitter rows after she refused to allow her daughter to move from Kent. The couple met at a Tunbridge Wells pub in 1991 before separating nine years later when he was jailed. The couple had two daughters - now aged 19 and 24. The younger of the two still lives with her mother, who is a director of a chemical company in Kent. The friend said the tumultuous relationship was punctuated by Masood's violence towards others, who was then known by his birth name. Masood masked his life of crime with a CV which claimed he was an experienced English teacher who had worked across the world. The document lists Masood as a university-educated English teacher with

experience working in places such as Saudi Arabia and Luton.

The Metropolitan Police initially believed the attack was inspired by international terrorism. On March 23rd, The Amaq News Agency, affiliated with ISIS, took credit for the killings and announced that the attacker was "a soldier of the Islamic State, executing the operation in response to calls to target citizens of coalition nations", but has been unable to prove officials were in any contact with Masood. Describing him as a terrorist, the Metropolitan Police confirmed that he acted alone. Later on March 27th, Neil Basu, Deputy Assistant Commissioner of the Metropolitan Police and Senior National Coordinator for UK Counter-Terrorism Policing, announced that Masood clearly had an interest in Jihad, but they have found no evidence he was linked with any terrorist Organization. Six property raids and eleven arrests were made, but no charges were filed and all suspects were released without charges.

6 Lessons Learned

In this section, we will introduce how we get through the whole process with a detailed timeline. Also, we demonstrate what challenges we faced during this course as well as how we solved it.

6.1 Timeline

Progress 1: Get familiar with Hadoop cluster, Web archives (WARC file and CDX file), Solr index. Set up project management, also get some prelimary results on the Unit 1 and 2 for the Hurricane Sandy dataset.

Progress 2: Switch to the Attack Westminster dataset. Rebuilt the Solr index and migrated some scripts to PySpark for the adaption of the big dataset. Also ran jusText to clean our dataset.

Progress 3: Get the result for the Unit 1, 2, and 3 as well as bigrams and trigrams. Also figured out how to run the PySpark on Zepplin and the Hadoop Cluster.

Progress 4: Continue working on the gold standard for team 5. Did some research on the upcoming units like methodlogy and models we shold look into.

Progress 5: Finished the gold standard draft for team 5. Tried textrank and grasshopper on Unit 7 and used regular expression to get some initial results on Unit 8.

Progress 6: Ran LSA on Unit 6, used multiprocessing on Unit 7, and swith into spaCy to get more accurate results for Unit 8.

Progress 7: Used spaCy's named entity recognition to finish Unit 5, got a decent result on Unit 7 with k-means clustering. Tried some test dataset on the Pointer-generator Network with a pre-trained model.

6.2 Challenges Faced

The greatest challenge our team faced was unfamiliarity, be it with Natural Language Processing, the tools we used such as Spark, Zeppelin, Multiprocessing, etc., or with the unique course setup

this class had. Our team worked through this challenge through hard work, communication within our team and others, and hands-on learning. Outside of that, there were many specific challenges our team faced.

PySpark on the Hadoop Cluster

Getting our PySpark scripts to run on the Hadoop cluster was not an easy task for our team at the beginning of the semester. There was no way to export code blocks from Zeppelin to a script that could be used on the cluster, there were certain environment variables that were set on Zeppelin that weren't necessarily defined on the cluster, and external libraries as well as NLTK Data needed to be packaged and pushed to the server properly to be used. Debugging issues was also very difficult, as the error messages tended to be hidden in a flood of other logging messages outputted by the `spark2-submit` executable.

Multiprocessing Gensim's TextRank Implementation

While Gensim's TextRank implementation was instrumental to our work in Unit 7, for a period of time our team was roadblocked because the library's summarize method would silently crash any time we tried to concurrently compute summaries. We attempted to use both Python's `multiprocessing` library, as well as Spark's `parallelize` method on both Zeppelin and the cluster, however no method seemed to work. We were unable to come up with a proper solution to this problem, however our final pipeline for Unit 7 proved to create text samples small enough that the library was able to run efficiently across our document selection, and allowed us to form the final summary.

6.3 Solutions Developed

PySpark on the Hadoop Cluster

We were able to solve many of the issues faced with executing PySpark on the Hadoop cluster by asking questions on Piazza, following guides referenced on the course's Canvas page, and researching problems on StackOverflow and other forums. We've outlined a lot of the problems and solutions we faced in the following note: `https://slack-files.com/TCGQWEZ9A-FCX420QPM-b7889c6782`

While we worked hard to push through these issues, by the end of the semester, we switched to using Python's built-in `multiprocessing` library to pool processes instead. This allowed us to write scripts once, use them on both the small and big datasets, and efficiently execute them.

7 Conclusion

In this course, we are focusing on generating various summaries of web archives about the 2017 Westminster Attack. We applied two different methods, extractive and abstractive, to summarize the most important information from the web archives. The extractive summary we generated is quite relevant and includes key info like what, when, where, perpetrators, why, who was affected, countermeasures. While due to time limit, the abstractive summary is more natural yet focus on governments' attitude and reactions instead of the event itself. Both results could be improved by adjusting parameters like the number of clusters or classifying documents as relevant or irrelevant.

Based on the results and evaluation of our generated summaries, we conclude several useful tips in the process of text summarization from big data. The first is that data cleaning is difficult

but very important. It's not that simple to clean all the noises and remain the most important information. And keep in mind the data cleaning should be done in the early stage. Another tip is to apply a classifier to filter irrelevant documents and further select documents for both extractive and abstractive summary. Our experience told us that splitting the big dataset into smaller clusters saved much time (especially on TextRank algorithm). The last tip is to get familiar with third-party algorithms and calculate the time complexity before utilizing it.

8 User's Manual

8.1 Introduction

Our code is available at `https://github.com/whyisyoung/AttackWestminster`. For brevity, we only explain part of the code here. To use our Python scripts, a user must have Python 2.7 (we use 2.7.14 to be consistent with the Hadoop cluster) installed on his machine. Also, the user should have NLTK dependencies installed as well as corpora which can be obtained by nltk.download(). Another important package is spaCy [26] , which is helpful for named entity extraction and rule-based matching. Other required packages and corresponding version could be found via the file – "requirements.txt". To install all the required packages, the user could easily run this command on his shell:

```
pip install -r requirements.txt
```

Our goal in this project is shown in Fig. 7. Thus the code for the different units are constrained in their own folders.

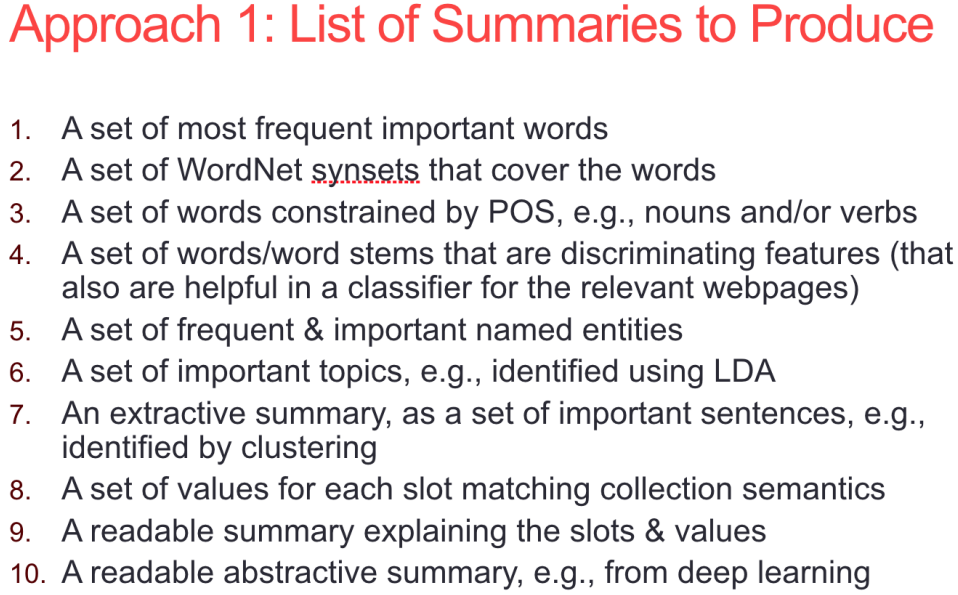

Figure 7: A list of summaries to produce in this project.

8.2 How to Run Scripts

Most of files with extension ".py" could be executed with the default Python syntax:

```
python script.py
```

If the format of a script file is pyspark, then you should follow the guidelines at `https://slack-files.com/TCGQWEZ9A-FCX420QPM-b7889c6782` and run the code with:

```
--conf spark.executorEnv.NLTK_DATA=./nltk_data/ script.py
```

The JSON files under the folder "ZepplinScripts" need to be run in the Zepplin environment. To set up the environment, the fastest way is to follow the instructions written by Xinyue Wang (`https://github.com/xw0078/VT_Fall18_CS4984-CS5984`). If more information is needed on how to run specific files, we have included that information as a comment at the beginning of the file.

8.3 Main functionality

Data Cleaning

We use jusText [14] to remove boiler template of the web archives. The corresponding code was located in the "cleaning" folder.

Most Frequent Important Words

We use NLTK's built-in function "word_tokenize" and corpora "stop_words" to find the most frequent words. To make these words meaningful (important), we also add punctuation and some customized words to filter more stopwords. The corresponding code was located in the "unit_1_westminster" folder.

A set of WordNet synsets that cover the words in Unit 1

We use NLTK's built-in corpus "wordnet", "brown", "words", and "state_union" to help find a better and more meaningful list of important words. The corresponding code was located in the "unit_2_westminster" folder.

A Set of Words Constrained by POS

We use NLTK's built-in function "pos_tag" to obtain the most frequent nouns and verbs. The corresponding code was located in the "unit_3_westminster" folder.

A Set of Frequent and Important Named Entities

For this task, we use spaCy's named entity recognizer to help determine the named entities. We use the `en_core_web_md` model from spaCy. Since the result of this task would be applied in Unit 8. We didn't write specific code for it. The corresponding code was located in "unit_8_westminster/Unit8_NESpacy.py".

A Set of Important Topics

We use Gensim's LSA model to help identify important topics. The result is not intuitive and only contains keywords to describe the topic. The corresponding code was located in "unit_6" folder.

An Extractive Summary, as a Set of Important Sentences

We apply sklearn `TFIDFVectorizer` to remove similar/duplicate documents and filtered out sentences that consist of more than 50 words. Then we use `KMeans` clustering algorithm to cluster our web archives into 5 clusters. Finally, we utilize Gensim's TextRank algorithm to summarize each document from each cluster and summarize the combined summaries within that cluster. The final summary was obtained by combining summaries from 5 clusters. The corresponding code was located in "unit_7_westminster" folder.

A Set of Values for Each Slot Matching Collection Semantics

Another form of summary was to create a set of values that could be extracted from documents and later be used to fill a generic template summary. We follow the guidelines listed on "TAC 2011 Guided Summarization Task Guidelines" and utilize spaCy's pattern based matcher to extract the most important information for the event. The corresponding code was located in "unit_8_westminster" folder.

9 Developer's Manual

Apart from following the guidelines listed on User's Manual, there are some preprocessing steps that a developer should know.

9.1 Solr Index

The dataset provided by the instructor is a compressed WARC file (Attack_Westminster.warc.gz) as well as its corresponding index file (Attack_Westminster.cdx). To get a better glance of the dataset and to facilitate team 5 to create a gold standard for us, we imported the dataset to the Solr server (http://blacklight.cs.vt.edu:8983/solr/), which enabled us and other teams to perform various queries on the dataset. The following instructions on creating the index from the WARC file are provided by our teaching assistant, Liuqing Li.

Process WARC file and CDX file

Step 1. Copy the event to the Hadoop Distributed File System (HDFS):

```
hadoop fs -put /home/public/cs4984_cs5984_f18/unlabeled/data/4_Attack_Westminster
/user/cs4984cs5984f18_team4/4_Attack_Westminster
```

Step 2. Use the ArchiveSpark script to generate a json file that contains all the sentences.

- Modify the Scala file provided by TA to fit the directory of our event.
- $ export JAVA_HOME=/usr/java/jdk1.8.0_171/
- $ spark2-shell -i ArchiveSpark_sentence_extraction_with_openNLP.scala
  –files /home/public/cs4984_cs5984_f18/unlabeled/lib/en-sent.bin
  –jars /home/public/cs4984_cs5984_f18/unlabeled/lib/archivespark-assembly-2.7.6.jar,
  /home/public/cs4984_cs5984_f18/unlabeled/lib/archivespark-assembly-2.7.6-deps.jar,
  /home/public/cs4984_cs5984_f18/unlabeled/lib/stanford-corenlp-3.5.1.jar,
  /home/public/cs4984_cs5984_f18/unlabeled/lib/opennlp-tools-1.9.0.jar

Step 3. Copy the generated JSON file from HDFS to our home folder.

```
hadoop fs -get /user/cs4984cs5984f18_team4/4_Attack_Westminster/sentences .
```

Create Solr Index

Step 1. Copy the generated JSON file from previous subsection to the Solr server (blacklight.cs.vt.edu)
Step 2. Copy the json_formatter.py from /home/fox to our home folder on the Solr server.
Step 3. Run the Python script to convert original JSON file to the expected format.

```
python json_formatter.py [JSON_FILE]
```

Step 4. Add or delete the index.

- ADD:

```
curl 'http://localhost:8983/solr/[CORE]/update/json?commit=true'
--data-binary @/path/to/JSON/file -H 'Content-type:application/json'
```

- DELETE:

```
curl 'http://localhost:8983/solr/[CORE]/update?commit=true'
-H "Content-Type: text/xml"
--data-binary '<delete><query>*:*</query></delete>'
```

9.2 Filtering Noisy Data

Originally, we didn't realize how much noise was in our dataset until after hearing the first progress reports from other teams. After these progress reports, we manually read part of the original dataset and tried to find what may be noisy. Thanks to a recommendation from team 14, we utilized jusText [14] to remove boilerplate content and empty webpages from the original web archives. To this end, we first modified the ArchiveSpark script to keep the original HTML body instead of fetching sentences. Then we applied the jusText function to keep content that is not regarded as boiler template. Thanks to jusText, we could also remove webpages that return a 404 not found error. Finally, we could reduce the small dataset from 462 articles to 299 articles. For the big dataset, the size of the dataset shrank around 40%. In Fig. 8, we can see that jusText is powerful when handling and removing irrelevant contents.

9.3 Files for Developers

Here we will explain the code structure and list files and folders for our code.

- Preprocess
  - `ArchiveSpark_sentence_extraction_with_jusText.scala`: Get the html body for each document for later filtering with jusText.
  - `ArchiveSpark_sentence_extraction_with_openNLP.scala`: Obtain all the sentences from the original documents.
- cleaning
  - justTextClean.py: Clean the html body generated from the preprocessing step with jusText.
  - JusText_Clean_Zeppelin.json: The Zeppelin version of the script that performs the same functionality.

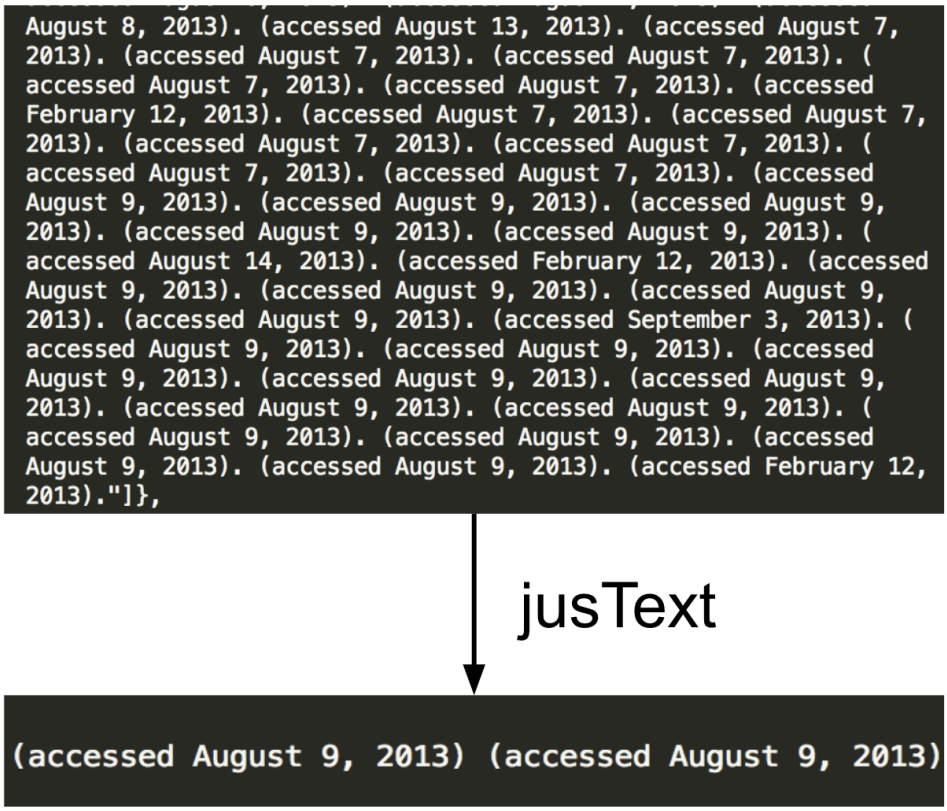

Figure 8: An example of applying jusText on a webpage. The original webpage can be found via `https://www.heritage.org/homeland-security/report/after-hurricane-sandy-time-learn-and-implement-the-lessons-preparedness`. We can see that in the bottom of this page, there are a lot of references and they are irrelavant for our summarization. JusText reduced multiple references to only two.

- data
  - Attack_Westminster_small.cdx: The index file for the web archives on the small dataset.
  - Attack_Westminster_small.warc.gz: The compressed data file for the web archives on the small dataset.
- python_evaluation
  - eval.py: Written by the GTA, Liuqing Li. There are 3 modes to evaluate our generated summary compared to the gold standards. The first mode is to get the ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-SU4 score. The second is for retrieving the sentences that achieves the best ROUGE-1 and ROUGE-2 score. The last one is to calculate the named entity coverage. More instructions are avaiable at `https://github.com/tagucci/pythonrouge`.
- tokenize
- unit_1_westminster

  - Unit1-Pyspark.py: The python script that collects the most frequent words of our dataset excluding stop words as described in Section 3.1.
  - Unit1-Zeppelin.json: The Zeppelin version of the Unit1 python script.

- unit_2_westminster
  - Unit2-Pyspark.py: The python script that collects the most frequent words of our dataset by using lemma synsets as described in 3.2.
  - Unit2-Zeppelin.json: The Zeppelin version of the Unit2 python script.

- unit_3_westminster
  - Unit3-Pyspark.py: The python script that collects the most frequent nouns and verbs of our dataset as described in Section 3.3.
  - Unit3-Zeppelin.json: The Zeppelin version of the Unit3 python script.

- unit_6
  - Unit6.py: The python script that produces the top 15 topics given our input of cleaned articles as described in Section 3.5.

- unit_7_westminster
  - Unit7.py: The python script that produces our extractive summary as described in Section 3.6.
  - Unit7-Zeppelin.json: The Zeppelin version of the Unit7 python script.

- unit_8_westminster
  - Unit8_NESpacy.py: The python script that selects our desired values for Unit 8 as described in Section 3.7.
  - Unit8-Zeppelin.json: The Zeppelin version of the Unit8 python script.

- zeppelin_scripts: A folder containing copies of all the zeppelin scripts located in other folders.
- requirements.txt: Required Python libraries and corresponding versions. Could be installed via pip install -r requirements.txt.

10 Acknowledgements

This project is supported by the National Science Foundation under Grant No. IIS-1619028. We would like to thank our instructor Dr. Edward Fox (fox@vt.edu) as well as GTA Liuqing Li (liuqing@vt.edu) for all of the effort they put into this class. Both of them helped us a lot by sharing their knowledge about that topic.

Additional thanks to Srijith Rajamohan (srijithr@vt.edu), Xuan Zhang (xuancs@vt.edu), and Michael Horning, Department of Communication, for their support by holding lectures about their specialized research areas.

Finally, we would like to thank our classmates for providing feedback and support. At this point we would like to give special thanks to Teams 1, 7 and 14 for sharing ideas, approaches, and code. We also want to thank Team 3 for creating the gold standard for us.

Bibliography

[1] *11,000 aftershocks since Christchurch quake*. Otago Daily Times Online News. Sept. 2012. URL: https://www.odt.co.nz/news/national/11000-aftershocks-christchurch-quake (visited on 10/23/2018).

[2] *22 February 2011 Canterbury earthquake for kids*. Christchurch City Libraries. URL: https://my.christchurchcitylibraries.com/canterbury-earthquake-2011-for-kids/ (visited on 10/23/2018).

[3] *4 September 2010 Canterbury earthquake for kids*. Christchurch City Libraries. URL: https://my.christchurchcitylibraries.com/canterbury-earthquake-2010-for-kids/ (visited on 10/23/2018).

[4] Madhi Ahmed Ali, Ali A Al-Dahoud, and Bilal H Hawashin. "Enhanced Feature-Based Automatic Text Summarization SystemUsingSupervised Technique". In: *INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY* 15.5 (2016), pp. 6757–6767.

[5] Jonathan Amos. *New Zealand earthquake: 65 dead in Christchurch*. BBC News. Feb. 2011. URL: https://www.bbc.com/news/world-asia-pacific-12533291 (visited on 10/22/2018).

[6] Federico Barrios et al. "Variations of the Similarity Function of TextRank for Automated Summarization". In: *CoRR* abs/1602.03606 (2016). arXiv: 1602.03606. URL: http://arxiv.org/abs/1602.03606.

[7] Joshua Berlinger. *Thousands stranded in New Zealand after earthquakes*. CNN. Nov. 2016. URL: https://www.cnn.com/2016/11/14/asia/new-zealand-earthquake/index.html (visited on 10/22/2018).

[8] Katie Bradford. *'Billions' needed to repair Kaikoura roads after 7.5 earthquake, says PM*. 1 NEWS NOW. Nov. 2016. URL: https://www.tvnz.co.nz/one-news/new-zealand/billions-needed-repair-kaikoura-roads-after-7-5-earthquake-says-pm (visited on 10/22/2018).

[9] Yen-Chun Chen and Mohit Bansal. "Fast Abstractive Summarization with Reinforce-Selected Sentence Rewriting". In: *arXiv preprint arXiv:1805.11080* (2018).

[10] *Christchurch earthquake kills 185*. RSS. Feb. 2011. URL: https://nzhistory.govt.nz/page/christchurch-earthquake-kills-185 (visited on 10/23/2018).

[11] Greg Corrado. *Computer, respond to this email*. ai.googleblog.com. Nov. 2015. URL: https://ai.googleblog.com/2015/11/computer-respond-to-this-email.html (visited on 11/23/2018).

[12] *February 2011 Christchurch earthquake*. RSS. May 2016. URL: https://nzhistory.govt.nz/culture/february-2011-christchurch-earthquake (visited on 10/23/2018).

[13] Kavita Ganesan. *What is ROUGE and how it works for evaluation of summaries?* Jan. 2017. URL: http://kavita-ganesan.com/what-is-rouge-and-how-it-works-for-evaluation-of-summaries/#.W5LhLJNKidt/ (visited on 12/05/2018).

[14] *Justext - Heuristic based boilerplate removal tool.* URL: http://corpus.tools/wiki/Justext (visited on 12/13/2018).

[15] Takahiro Kubo. *Awesome-Text-Summarization.* 2017. URL: https://github.com/icoxfog417/awesome-text-summarization (visited on 11/23/2018).

[16] *Liquefaction.* Science Learning Hub. Nov. 2012. URL: https://www.sciencelearn.org.nz/resources/343-liquefaction (visited on 10/20/2018).

[17] Rada Mihalcea. "Graph-based ranking algorithms for sentence extraction, applied to text summarization". In: *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions.* Association for Computational Linguistics. 2004, p. 20.

[18] Rada Mihalcea and Paul Tarau. "Textrank: Bringing order into text". In: *Proceedings of the 2004 conference on empirical methods in natural language processing.* 2004.

[19] Ani Nenkova, Kathleen McKeown, et al. "Automatic summarization". In: *Foundations and Trends® in Information Retrieval* 5.2–3 (2011), pp. 103–233.

[20] Valerie Niechai. *Beginner's Guide to Google PageRank: How It Works & Why It Still Matters in 2018.* Feb. 2018. URL: https://www.link-assistant.com/news/page-rank-2018.html (visited on 11/23/2018).

[21] Makbule Gulcin Ozsoy, Ferda Nur Alpaslan, and Ilyas Cicekli. "Text summarization using latent semantic analysis". In: *Journal of Information Science* 37.4 (2011), pp. 405–417.

[22] John P. Rafferty and Lorraine Murray. *Christchurch earthquakes of 2010–11.* Encyclopædia Britannica. Oct. 2017. URL: https://www.britannica.com/event/Christchurch-earthquakes-of-2010-2011 (visited on 10/23/2018).

[23] Radim Rehurek and Petr Sojka. "Software framework for topic modelling with large corpora". In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks.* Citeseer. 2010.

[24] GNS Science. *The hidden fault that caused the February 2011 Christchurch earthquake.* Auckland Volcanic Field Geology / Volcano Geology and Hazards / New Zealand Volcanoes / Volcanoes / Science Topics / Learning / Home - GNS Science. URL: https://www.gns.cri.nz/Home/Our-Science/Natural-Hazards/Recent-Events/Canterbury-quake/Hidden-fault (visited on 10/22/2018).

[25] Abigail See, Peter J Liu, and Christopher D Manning. "Get to the point: Summarization with pointer-generator networks". In: *arXiv preprint arXiv:1704.04368* (2017).

[26] *spaCy - Industrial-Strength Natural Language Processing IN Python.* URL: https://spacy.io/ (visited on 12/13/2018).

[27] National Institute of Standards and Technology. "TAC 2011 Guided Summarization Task Guidelines". In: (2011). URL: https://tac.nist.gov//2011/Summarization/Guided-Summ.2011.guidelines.html.

[28] Heritage Te Manatu Taonga. *Story: Historic earthquakes.* Māori clothing and adornment – kākahu Māori – Te Ara Encyclopedia of New Zealand. Nov. 2017. URL: https://teara.govt.nz/en/historic-earthquakes/page-13 (visited on 10/23/2018).

[29] Matthew Taylor. *Tsunami hits New Zealand east coast after 7.5-magnitude earthquake*. The Guardian. Nov. 2016. URL: https://www.theguardian.com/world/2016/nov/13/new-zealand-south-island-hit-by-74-magnitude-earthquake (visited on 10/22/2018).

[30] *Timeline of the 7.8 quake and response reveals plenty of room for improvement*. Nov. 2016. URL: https://www.stuff.co.nz/national/nz-earthquake/86654229/Timeline-of-the-7-8-quake-and-response-reveals-plenty-of-room-for-improvement%20last%20accessed%20March%202017 (visited on 10/20/2018).

[31] Wikipedia contributors. *2011 Christchurch earthquake — Wikipedia, The Free Encyclopedia*. 2018. URL: https://en.wikipedia.org/w/index.php?title=2011_Christchurch_earthquake&oldid=870241590 (visited on 11/27/2018).

[32] Wikipedia contributors. *2016 Kaikoura earthquake — Wikipedia, The Free Encyclopedia*. 2018. URL: https://en.wikipedia.org/w/index.php?title=2016_Kaikoura_earthquake&oldid=868716680 (visited on 11/27/2018).

[33] Wikipedia contributors. *ROUGE (metric) — Wikipedia, The Free Encyclopedia*. 2018. URL: https://en.wikipedia.org/w/index.php?title=ROUGE_(metric)&oldid=845513445 (visited on 12/05/2018).