Hybrid Summarization of Dakota Access Pipeline Protests (NoDAPL)

CS 5984/4984 Big Data Text Summarization Report

By

Xiaoyu Chen, Haitao Wang, Maanav Mehrotra, Naman Chhikara, Di Sun
{xiaoyuch, wanght, maanav, namanchhikara, sdi1995} *@vt.edu*

Instructor: Dr. Edward A. Fox

Dept. of Computer Science, Virginia Tech
Blacksburg, Virginia 24061

December 2018

Table of Contents

List of Tables

List of Figures

Abstract

Dakota Access Pipeline Protests (known with the hashtag #NoDAPL) are grassroots movements that began in April 2016 in reaction to the approved construction of Energy Transfer Partners' Dakota Access Pipeline in the northern United States. The NoDAPL movements produce many FaceBook messages, tweets, blogs, and news, which reflect different aspects of the NoDAPL events. The related information keeps increasing rapidly, which makes it difficult to understand the events in an efficient manner. Therefore, it is invaluable to automatically or at least semi-automatically generate short summaries based on the online available big data. Motivated by this automatic summarization need, the objective of this project is to propose a novel automatic summarization approach to efficiently and effectively summarize the topics hidden in the online big text data. Although automatic summarization has been investigated for more than 60 years since the publication of Luhn's 1958 seminal paper, several challenges exist in summarizing online big text sets, such as large proportion of noise texts, highly redundant information, multiple latent topics, *etc*. Therefore, we propose an automatic framework with minimal human efforts to summarize big online text sets (~11,000 documents on NoDAPL) according to latent topics with nonrelevant information removed. This framework provides a hybrid model to combine the advantages of latent Dirichlet allocation (LDA) based extractive and deep-learning based abstractive methods. Different from semi-automatic summarization approaches such as template-based summarization, the proposed method does not require a deep understanding of the events from the practitioners to create the template nor to fill in the template by using regular expressions. During the procedure, the only human effort needed is to manually label a few (say, 100) documents as relevant and irrelevant. We evaluate the quality of the generated automatic summary with both extrinsic and intrinsic measurement. In the extrinsic subjective evaluation, we design a set of guideline questions and conduct a task-based measurement. Results show that 91.3% of sentences are within the scope of the guideline, and 69.6% of the outlined questions can be answered by reading the generated summary. The intrinsic ROUGE measurements show our entity coverage is a total of 2.6% and ROUGE L and ROUGE SU4 scores are 0.148 and 0.065. Overall, the proposed hybrid model achieves decent performance on summarizing NoDAPL events. Future work includes testing of the approach with more textual datasets for interesting topics, and investigation of topic modeling-supervised classification approach to minimize human efforts in automatic summarization. Besides, we also would like to investigate a deep learning-based recommender system for better sentence re-ranking.

1. Introduction

1.1 Problem Statement

Dakota access pipeline protests (known with the hashtag #NoDAPL) are grassroots movements that began in April 2016 in reaction to the approved construction of Energy Transfer Partners' Dakota Access Pipeline in the northern United States. The NoDAPL movements produce many FaceBook messages, tweets, blogs, and news, which reflect different aspects of the NoDAPL events. And the related information keep increasing in exponential space, which makes it impossible to understand the events in an efficient manner. Therefore, it is invaluable to automatically or at least semi-automatically generate short summaries based on the online available big data. Motivated by this automatic summarization needs, the objective of this project is to propose a novel automatic summarization approach to efficiently and effectively summarize the topics hidden in the online big text data. Although automatic summarization has been investigated for more than 60 years since the publication of Luhn's seminal paper (Luhn, 1958), several challenges exist in summarizing online big text sets, such as large proportion of noise texts (*i.e.*, irrelevant documents/topics, noise sentences, etc.), multiple latent topics, *etc*.

In this project, we were tasked with developing methodology and workflow in generating a brief summary of a corpus of webpages about NoDAPL events. Such a summary should represent the important information about this event in different latent topics. It was expected to apply automatic text summarization techniques and a deep learning methodology in developing this project.

1.2 About CS5984/4984 Course

CS5984/4984 Big Data Text Summarization is a team project / problem-based learning course at Virginia Tech. It derived from CS 4984: Computational Linguistics, but involves deep learning as a potentially powerful tool to generate better summarization of a webpage collection. This course is driven by a single question:

> *How to automatically / semi-automatically generate a good summary from a corpus collected from webpages with noisy, highly redundant, and large-scale textual data?*

This is a fundamental research question in the area of automatic summarization, which has been an active research area for more than six decades (Mani & Maybury, 2001). In automatic summarization, researchers and practitioners developed various extractive and abstractive, supervised and unsupervised, methods for a single document and a collection of documents. However, there remain research challenges such as: large proportion of noise information, highly redundant information, multiple latent topics, large-scale text data sets, etc. Besides, most research in summarization deals with producing a short, paragraph-length summary (Nenkova & McKeown, 2011), which cannot be directly applied to this project. Therefore, a novel method needs to be proposed to facilitate effective and efficient summarization.

1.3 Proposed Framework for Hybrid Automatic Text Summarization

We proposed a hybrid automatic text summarization by adopting both deep learning based abstractive and latent Dirichlet allocation (LDA) based extractive summarization technologies. Figure 1 presents details of the proposed framework.

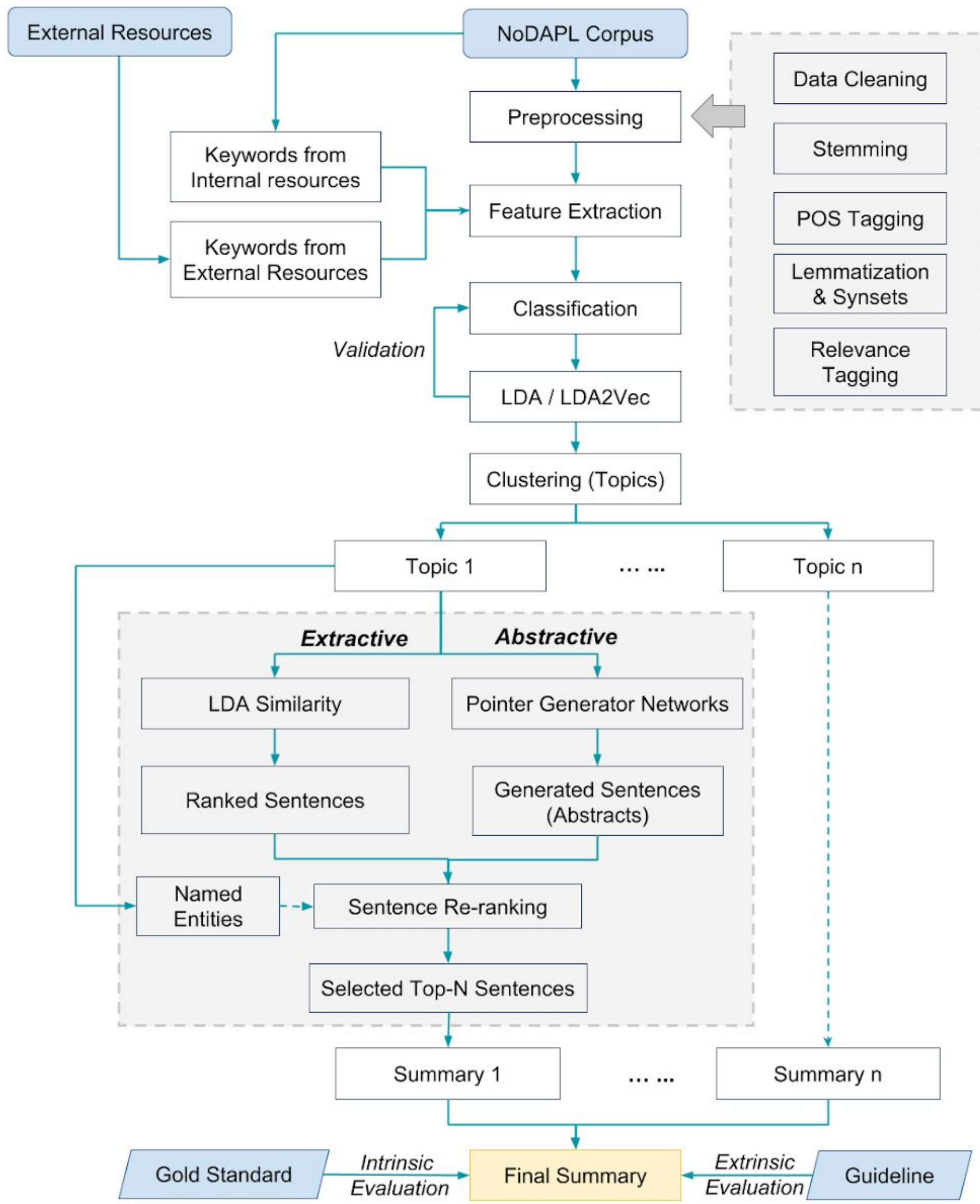

Figure 1. Outline of the proposed hybrid summarization approach.

2. Related Work

2.1 Categories of Text Summarization

The aim of text summarization is to reduce the source text into a compact version which will preserve contents and general meaning. It minimizes reading time and efforts to get the most important information. Manual creation of a summary can be a very time-consuming complicated task. Therefore the research community is developing new approaches for automatic text summarization.

The field of document summarization has moved forward in various aspects during recent years (Yao, Wan, & Xiao, 2017). Ani Nenkova and Kathleen McKeown reviewed the research in automatic text summarization, including the more traditional efforts in sentence extraction as well as the most novel recent approaches for determining important content, for the domain and genre-specific summarization and for evaluation of summarization (Nenkova & McKeown, 2011). There are several distinctions typically made in summarization, such as: a) summary from extracted or abstracted information; b) summary for a single document or multiple documents; c) generic or query-focused summarization, etc.

2.1.1 Based on the Processing Technique

Extractive summarization produces summaries by concatenating several sentences taken exactly as they appear in the original documents being summarized. On the other hand, *abstractive summarization* uses different words to describe the contents of the original documents rather than directly copying original sentences.

2.1.2 Based on Documents

Summarization of a *single document* produces a summary of one document, for example, a summary of a scientific research paper. By contrast, *multi-document summarization* was motivated by use cases on the Web. It is like the goal of this project which aims to produce one brief summary for thousands of news articles on the same event.

2.1.3 Based on Audiences

Summarization work can be based on the audience who is going to use the summary, *i.e.*, whether the summary is for a group of people, or it is for a specific user's query. *Generic summarization* makes few assumptions about the audience or the goal for generating the summary. In contrast, in *query-focused summarization*, the goal is to summarize only the information in the input document(s) that is relevant to a specific user query.

2.1.4 Based on Dependency of External Resources

The development of a summary for a target set of documents (internal resources) can rely on some external knowledge such as Wikipedia pages to gain insights about the document. A summarization work can be either *knowledge-poor* or *knowledge-rich* based on the availability of external resources relevant to the documents.

2.2 Automatic Text Summarization

Automatic summarization is the process of shortening a text document with software, in order to create a summary with the major points of the original document (Automatic summarization, 2018). Among a number of techniques, sequence to sequence (Seq2Seq) learning has recently been used for abstractive and extractive summarization. Khatri et al. proposed a novel document-context based Seq2Seq models using RNNs for abstractive and extractive summarizations. They trained the model on human-extracted golden summaries. Their models outperformed standard Seq2seq model. They also developed a semi-supervised technique for extracting approximate summaries and using it for training Seq2seq models at scale (Khatri, Singh, & Parikh, 2018). Narayan et al. worked on single document summarization and developed a new algorithm to train a neural summarization model on the CNN / DailyMail datasets. They conceptualized the extractive summarization as a sentence ranking task and proposed the training algorithm to globally optimize the ROUGE evaluation metric through a reinforcement learning objective. See et al. worked on multi-document summarization and proposed a novel architecture in abstractive text summarization that augments the standard Seq2Seq attentional model in two orthogonal ways. First, they used a hybrid pointer-generator network that can copy words from the source text via pointing, which aids accurate reproduction of information, while retaining the ability to produce novel words through the generator. Second, they used coverage to keep track of what has been summarized, which discourages repetition. Their model was applied to the CNN / Daily Mail summarization task, outperforming the current abstractive state-of-the-art by at least 2 ROUGE points (See, Liu, & Manning, 2017).

2.3 Deep Learning / Machine Learning

In general, a learning algorithm is a method used to process data to extract patterns to apply to a new situation (Goodfellow, Bengio, Courville, & Bengio, 2016). In more detail, we start by describing what the difference is between the challenge of fitting the training data and the challenge of finding patterns that generalize to new data, which is achieved by setting hyperparameters. In essence, machine learning is an application of high-level statistics, emphasizing on the use of the computer to estimate complex functions. Deep learning is a specific kind of machine learning. Most of the deep learning algorithms can be described as a combination of a specification of a dataset, a cost function, an optimization procedure, and a model (Goodfellow *et al.*, 2016).

One of the most important applications of deep learning is in human language. There is a large amount of information stored in some form of human language in the world, and in order to obtain and use the information, a kind of computational algorithm would be required. Natural language processing (NLP) deals with building these algorithms to automatically analyze and represent human language (Saravia, 2018). There are some well-known applications using NLP-based systems, such as aspects of the Google search engine, Apple's Siri, and Amazon's

Alexa. In addition, NLP is also useful to teach machines the ability to perform complex natural language-related tasks such as machine translation and dialogue generation (Saravia, 2018).

Since linguistic information was represented with sparse representations (high-dimensional features), word embeddings, character embeddings, and neural-based models are developed to lower the dimensionality. They have achieved excellent results. With word embeddings and character embeddings successfully applied to various NLP tasks, the words are modeled in terms of context and morphological information. Thus, we need a neural network to understand the concepts behind words. One of the neural networks used is Recurrent Neural Network (RNN). RNNs are specialized neural-based approaches that are effective at processing sequential information. The main strength of an RNN is the capacity to memorize the results of previous computations and use that information in the current computation. This makes RNN models suitable to model context dependencies in inputs of arbitrary length so as to create a proper composition of the input (Saravia, 2018).

3. Preprocessing

3.1 Data and Resources

Our corpus consisted of thousands of documents regarding Dakota Access Pipeline Protests (NoDAPL). Each document within the corpus had content from webpages which were crawled by the crawlers. All of this was stored in the WARC and CDX format and was provided to us. The Web ARChive (WARC) format specifies a method for combining multiple digital resources into an aggregate archive file together with related information. It consists of metadata, HTML tags, and a variety of data that is irrelevant for us. So we used a Scala script to remove / clean the irrelevant portion, such as HTML tags and Javascript so that we are left with mainly the relevant text content of a webpage. We then convert this to a JSON formatted file which was then used for processing.

There were two different collections given to us to work on. The smaller collection consisted of around 500 documents which we mainly used for code testing and idea implementation, while the larger one consisted of about 11,000 documents which we used eventually to generate our summary. We uploaded the dataset into Solr (see Figure 2). On seeing the data we found that each document consisted of a URL, timestamp, sentence, and ID. This was important as it allowed us to query the data conveniently. This also helped the other team for creating a gold standard for us.

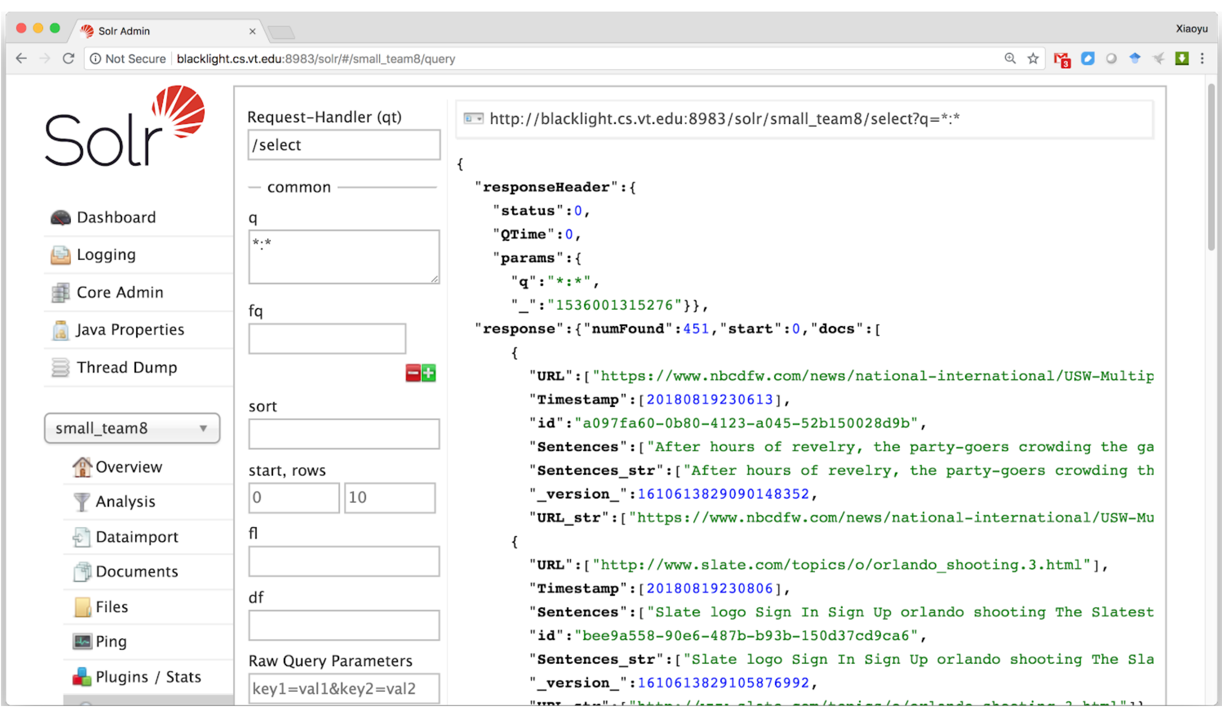

Figure 2. Solr index of large dataset for NoDAPL.

3.2 Data Cleaning

3.2.1 Webpages

By manually analyzing a few documents to get an idea of the quality of the data, we found that there were some 404 errors, repeating paragraphs and sentences within a document, repeating documents, and some completely irrelevant documents which talked about irrelevant things. This is what we considered noise. An example of noise is presented in Figure 3.


```
{
  "originalUrl": "https://fox8.com/2016/06/26/survivors-of
    -boston-marathon-bombing-visit-orlando-nightclub
    -massacre-survivors",
  "timestamp": "20180819230644",
  "title": "Survivors of Boston marathon bombing visit
    Orlando nightclub massacre survivors | fox8.com",
  "text": "GoSearch Fox 8 TV Schedule Autos Search Contact
    Us fox8.com Menu News Closings Seen on TV AM Show New
    Day Sports Traffic Contests Jobs Weather Cleveland 75°
    Low 65° High 79° Akron/Canton 76° Low 63° High 83° See
    complete forecast googletag.cmd.push(function() {
    googletag.display('acm-tag-120x60-atf'); }); googletag
    .cmd.push(function() { googletag.display('acm-tag-728x90
    -atf'); }); Survivors of Boston marathon bombing visit
    Orlando nightclub massacre survivors Posted 11:44 am,
    June 26, 2016, by CNNwire, Updated at 11:56AM, June 26,
    2016 Facebook Twitter Google Pinterest LinkedIn Email
    This is an archived article and the information in the
    article may be outdated. Please look at the time stamp
    on the story to see when it was last updated. ORLANDO,
    FL – Boston Strong. Orlando Strong. Scarred by acts of
    unimaginable terror, victims of the Boston Marathon
    bombing and the Orlando nightclub massacre are finding
    healing in unity. Ten Boston survivors, wounded by two
    pressure-cooker bombs planted at the finish line of the
    marathon in 2013, are in Orlando this weekend to show
    support for those wounded when a gunman armed with an
    assault rifle stormed the Pulse nightclub June 12. The
    bombing victims say they're in Orlando to show that they
```


Figure 3. An example of noise in the document (highlighted content).

3.2.2 Stopwords

High-frequency words like *to*, *is* and *also,* which come way too often in a document and have a very minimal lexical content, are considered as stop words. For any NLP problem, stopword removal is often the first step before processing. We began by tokenizing our text by word and then used the NLTK's stopwords corpus to remove the words which are used generally in all types of documents. We then created a frequency distribution of our corpus. The idea behind this approach is that frequently occurring words will likely give more insight into the data (see an example for the most frequent words in Table 1 and the count in Figure 4). For implementing this we had to ensure two things: removal of punctuation marks, special characters, and the case of the word (Trump and trump are different). On analyzing the results we found some irrelevant high-frequency words such as "the" or "this" which were not covered by NLTK's stopword corpus. To tackle this problem, we created another custom stopwords corpus which contained high frequency irrelevant words from our documents. We then used the concept of stemming and lemmatization. During this, we used both the NLTK and SpaCy library but chose to go for NLTK because it was relatively faster.

Table 1. Most frequent words for NoDAPL corpus.

Frequent Words										
The	Dakota	I	pipeline	Access	We	Standing	Rock	Oil	water	people

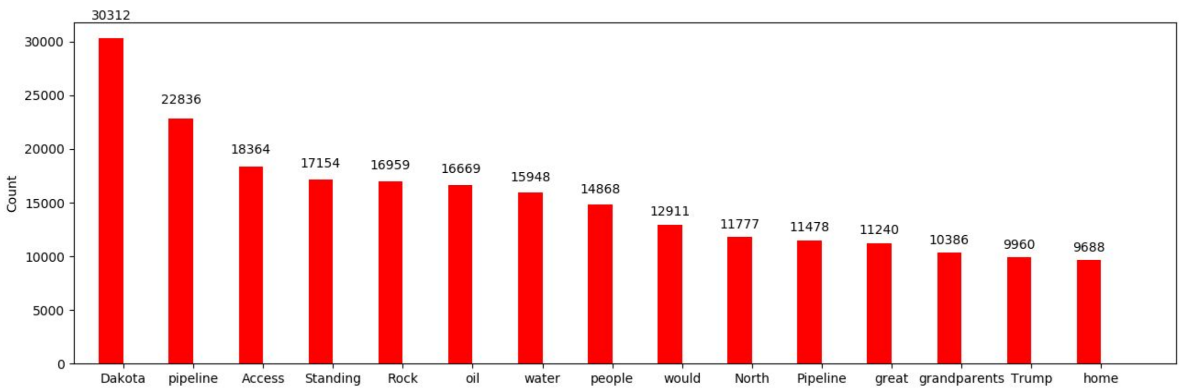

Figure 4. Frequency of top common words for NoDAPL.

3.3 Stemming

After removal of the stopwords, we tried stemming. Stemming algorithms work by cutting off the affixes, taking into account a list of common prefixes and suffixes that can be found in an inflected word (see example in Figure 5). This approach can be successful some of the time but not always. We tried two different stemmers (Porter and Snowball) but it didn't give as satisfying result as we wanted.

```
['need', 'to', 'get', 'someth', 'off', 'my', 'chest', 'that', 'i', 'wit', 'and', 'found', 'veri', 'disturb', 'in', 'm
y', 'brief', 'time', 'there', 'that', 'i', 'believ', 'mani', 'other', 'have', 'start', 'to', 'speak', 'up', 'about',
'as', 'well', '.', 'i', 'mean', 'that', 'serious', '.', 'plymouth', 'rock', 'serious', '.', 'they', 'are', 'come', 'i
n', ',', 'take', 'food', ',', 'cloth', 'and', 'occupi', 'space', 'without', 'ani', 'desir', 'to', 'particip', 'in', '
camp', 'mainten', 'and', 'without', 'respect', 'of', 'tribal', 'protocol', '.', 'offic', 'in', 'riot', 'gear', 'clash
', 'again', 'wednesday', 'with', 'protest', 'near', 'the', 'dakota', 'access', 'pipelin', ',', 'hit', 'dozen', 'with'
, 'pepper', 'spray', 'as', 'they', 'wade', 'through', 'waist-deep', 'water', 'in', 'an', 'attempt', 'to', 'reach', 'p
roperti', 'own', 'by', 'the', 'pipelin', "'s", 'develop', '.', 'offic', 'in', 'riot', 'gear', 'clash', 'again', 'wedn
esday', 'with', 'protest', 'near', 'the', 'dakota', 'access', 'pipelin', ',', 'hit', 'dozen', 'with', 'pepper', 'spra
y', 'as', 'they', 'wade', 'through', 'waist-deep', 'water', 'in', 'an', 'attempt', 'to', 'reach', 'properti', 'own',
'by', 'the', 'pipelin', "'s", 'develop', '.', 'offic', 'in', 'riot', 'gear', 'clash', 'again', 'wednesday', 'with', '
protest', 'near', 'the', 'dakota', 'access', 'pipelin', ',', 'hit', 'dozen', 'with', 'pepper', 'spray', 'as', 'they',
'wade', 'through', 'waist-deep', 'water', 'in', 'an', 'attempt', 'to', 'reach', 'properti', 'own', 'by', 'the', 'pipe
lin', "'s", 'develop', '.', 'offic', 'in', 'riot', 'gear', 'clash', 'again', 'wednesday', 'with', 'protest', 'near',
'the', 'dakota', 'access', 'pipelin', ',', 'hit', 'dozen', 'with', 'pepper', 'spray', 'as', 'they', 'wade', 'through'
, 'waist-deep', 'water', 'in', 'an', 'attempt', 'to', 'reach', 'properti', 'own', 'by', 'the', 'pipelin', "'s", 'deve
lop', '.', 'anoth', 'twitter', 'user', 'said', 'they', 'had', 'wit', 'a', 'protestor', 'turn', 'down', 'tap', 'water'
, 'to', 'spend', 'donat', 'on', "'fluorid", 'free', "'", 'water', '.', 'it', 'say', 'protestor', 'should', 'avoid', '
drug', 'and', 'alcohol', ',', 'engag', 'with', 'the', 'elder', ',', 'and', 'refrain', 'from', 'play', 'guitar', 'arou
nd', 'campfir', '.', 'the', 'north', 'dakota', 'pipelin', 'protest', 'wa', 'spark', 'by', 'plan', 'to', 'construct',
```

Figure 5. Stemmed words of the first document.

3.4 POS Tagging

Classifying words into their respective part of speech and labeling them accordingly is part of speech tagging. A Part-Of-Speech Tagger (POS Tagger) is a piece of software that reads a text and assigns parts of speech to each word/token, such as noun, verb, adjective, etc. We used the Punkt tokenizer (NLTK, 2006) on the raw text to split our documents into sentences and tagged the words with the nltk.pos_tag tagger (see example in Figure 6). Note: Do not remove stopwords before POS tagging. Words like *a, an, the* which are a part of the stopwords corpus play an important role to tag tokens correctly.

```
[('Need', 'NN'), ('to', 'TO'), ('get', 'VB'), ('something', 'NN'), ('off', 'IN'), ('my', 'PRP$'), ('chest', 'NN'), ('
that', 'IN'), ('I', 'PRP'), ('witnessed', 'VBD'), ('and', 'CC'), ('found', 'VBD'), ('very', 'RB'), ('disturbing', 'VB
G'), ('in', 'IN'), ('my', 'PRP$'), ('brief', 'JJ'), ('time', 'NN'), ('there', 'EX'), ('that', 'IN'), ('I', 'PRP'), ('
believe', 'VBP'), ('many', 'JJ'), ('others', 'NNS'), ('have', 'VBP'), ('started', 'VBN'), ('to', 'TO'), ('speak', 'VB
'), ('up', 'RP'), ('about', 'RB'), ('as', 'RB'), ('well', 'RB'), ('.', '.'), ('I', 'PRP'), ('mean', 'VBP'), ('that',
'DT'), ('seriously', 'RB'), ('.', '.'), ('Plymouth', 'NNP'), ('rock', 'NN'), ('seriously', 'RB'), ('.', '.'), ('They'
, 'PRP'), ('are', 'VBP'), ('coming', 'VBG'), ('in', 'IN'), (',', ','), ('taking', 'VBG'), ('food', 'NN'), (',', ','),
('clothing', 'NN'), ('and', 'CC'), ('occupying', 'VBG'), ('space', 'NN'), ('without', 'IN'), ('any', 'DT'), ('desire'
, 'NN'), ('to', 'TO'), ('participate', 'VB'), ('in', 'IN'), ('camp', 'JJ'), ('maintenance', 'NN'), ('and', 'CC'), ('w
ithout', 'IN'), ('respect', 'NN'), ('of', 'IN'), ('tribal', 'JJ'), ('protocols', 'NNS'), ('.', '.'), ('Officers', 'NN
S'), ('in', 'IN'), ('riot', 'NN'), ('gear', 'NN'), ('clashed', 'VBD'), ('again', 'RB'), ('Wednesday', 'NNP'), ('with'
, 'IN'), ('protesters', 'NNS'), ('near', 'IN'), ('the', 'DT'), ('Dakota', 'NNP'), ('Access', 'NNP'), ('pipeline', 'NN
'), (',', ','), ('hitting', 'VBG'), ('dozens', 'NNS'), ('with', 'IN'), ('pepper', 'NN'), ('spray', 'NN'), ('as', 'IN'
), ('they', 'PRP'), ('waded', 'VBD'), ('through', 'IN'), ('waist-deep', 'JJ'), ('water', 'NN'), ('in', 'IN'), ('an',
'DT'), ('attempt', 'NN'), ('to', 'TO'), ('reach', 'VB'), ('property', 'NN'), ('owned', 'VBN'), ('by', 'IN'), ('the',
'DT'), ('pipeline', 'NN'), ("'s", 'POS'), ('developer', 'NN'), ('.', '.'), ('Officers', 'NNS'), ('in', 'IN'), ('riot'
, 'NN'), ('gear', 'NN'), ('clashed', 'VBD'), ('again', 'RB'), ('Wednesday', 'NNP'), ('with', 'IN'), ('protesters', 'N
NS'), ('near', 'IN'), ('the', 'DT'), ('Dakota', 'NNP'), ('Access', 'NNP'), ('pipeline', 'NN'), (',', ','), ('hitting'
, 'VBG'), ('dozens', 'NNS'), ('with', 'IN'), ('pepper', 'NN'), ('spray', 'NN'), ('as', 'IN'), ('they', 'PRP'), ('wade
d', 'VBD'), ('through', 'IN'), ('waist-deep', 'JJ'), ('water', 'NN'), ('in', 'IN'), ('an', 'DT'), ('attempt', 'NN'),
('to', 'TO'), ('reach', 'VB'), ('property', 'NN'), ('owned', 'VBN'), ('by', 'IN'), ('the', 'DT'), ('pipeline', 'NN'),
("'s", 'POS'), ('developer', 'NN'), ('.', '.'), ('Officers', 'NNS'), ('in', 'IN'), ('riot', 'NN'), ('gear', 'NN'), ('
clashed', 'VBD'), ('again', 'RB'), ('Wednesday', 'NNP'), ('with', 'IN'), ('protesters', 'NNS'), ('near', 'IN'), ('the
', 'DT'), ('Dakota', 'NNP'), ('Access', 'NNP'), ('pipeline', 'NN'), (',', ','), ('hitting', 'VBG'), ('dozens', 'NNS')
, ('with', 'IN'), ('pepper', 'NN'), ('spray', 'NN'), ('as', 'IN'), ('they', 'PRP'), ('waded', 'VBD'), ('through', 'IN
'), ('waist-deep', 'JJ'), ('water', 'NN'), ('in', 'IN'), ('an', 'DT'), ('attempt', 'NN'), ('to', 'TO'), ('reach', 'VB
```

Figure 6. Tagging POS for words in the first document.

3.5 Lemmatization and Synsets

Lemmatization takes into consideration the morphological analysis of the words. To do so, it is necessary to have detailed dictionaries which the algorithm can look through to link the form back to its lemma. A lemma is the base form of all of its inflectional forms. For our data, we performed this after doing the POS tagging. We used the WordNet library of Python and the WordNet POS Tagger. WordNet is a large lexical database of English. Nouns, verbs, adjectives, and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept (see example in Figure 7). Synsets have an important aspect that allows us to identify synonymous words. We used synsets to make the feature vector which is given to the classifier.

```
['Need', 'to', 'get', 'something', 'off', 'my', 'chest', 'that', 'I', 'witness', 'and', 'find', 'very', 'disturb', 'i
n', 'my', 'brief', 'time', 'there', 'that', 'I', 'believe', 'many', 'others', 'have', 'start', 'to', 'speak', 'up', '
about', 'a', 'well', '.', 'I', 'mean', 'that', 'seriously', '.', 'Plymouth', 'rock', 'seriously', '.', 'They', 'be',
'come', 'in', ',', 'take', 'food', ',', 'clothing', 'and', 'occupy', 'space', 'without', 'any', 'desire', 'to', 'part
icipate', 'in', 'camp', 'maintenance', 'and', 'without', 'respect', 'of', 'tribal', 'protocol', '.', 'Officers', 'in'
, 'riot', 'gear', 'clash', 'again', 'Wednesday', 'with', 'protester', 'near', 'the', 'Dakota', 'Access', 'pipeline',
',', 'hit', 'dozen', 'with', 'pepper', 'spray', 'a', 'they', 'wad', 'through', 'waist-deep', 'water', 'in', 'an', 'at
tempt', 'to', 'reach', 'property', 'own', 'by', 'the', 'pipeline', "'s", 'developer', '.', 'Officers', 'in', 'riot',
'gear', 'clash', 'again', 'Wednesday', 'with', 'protester', 'near', 'the', 'Dakota', 'Access', 'pipeline', ',', 'hit'
, 'dozen', 'with', 'pepper', 'spray', 'a', 'they', 'wad', 'through', 'waist-deep', 'water', 'in', 'an', 'attempt', 't
o', 'reach', 'property', 'own', 'by', 'the', 'pipeline', "'s", 'developer', '.', 'Officers', 'in', 'riot', 'gear', 'c
lash', 'again', 'Wednesday', 'with', 'protester', 'near', 'the', 'Dakota', 'Access', 'pipeline', ',', 'hit', 'dozen',
'with', 'pepper', 'spray', 'a', 'they', 'wad', 'through', 'waist-deep', 'water', 'in', 'an', 'attempt', 'to', 'reach'
, 'property', 'own', 'by', 'the', 'pipeline', "'s", 'developer', '.', 'Officers', 'in', 'riot', 'gear', 'clash', 'aga
in', 'Wednesday', 'with', 'protester', 'near', 'the', 'Dakota', 'Access', 'pipeline', ',', 'hit', 'dozen', 'with', 'p
epper', 'spray', 'a', 'they', 'wad', 'through', 'waist-deep', 'water', 'in', 'an', 'attempt', 'to', 'reach', 'propert
y', 'own', 'by', 'the', 'pipeline', "'s", 'developer', '.', 'Another', 'Twitter', 'user', 'say', 'they', 'have', 'wit
ness', 'a', 'protestor', 'turn', 'down', 'tap', 'water', 'to', 'spend', 'donation', 'on', "'fluoride", 'free', "'", '
water', '.', 'It', 'say', 'protestors', 'should', 'avoid', 'drug', 'and', 'alcohol', ',', 'engage', 'with', 'the', 'e
lder', ',', 'and', 'refrain', 'from', 'play', 'guitar', 'around', 'campfire', '.', 'The', 'North', 'Dakota', 'Pipelin
```

Figure 7. Lemmatized words of the first document.

3.6 Document Relevance Tagging

A major part of our problem was to divide our documents into relevant and irrelevant. We found that a lot of noise was due to the irrelevant documents present in the corpus. To deal with this, we thought about building a classifier to classify our documents into the two. So while working in parallel on preprocessing we also started to label our documents. Labeling the document as relevant or irrelevant was a subjective decision. If upon reading the document we found the document to make sense about our topic, we labeled it as relevant, otherwise, it was labeled irrelevant. We began by labeling the first 50 documents but the results showed that our model was under-fitted. This prompted us to increase the count of labels to 100 (see example in Table 2, which is truncated to avoid redundancy).

Table 2. Relevance tagging and keywords identification for selected sample documents.

Document index ID	Relevance (0 or 1)	Keywords
63e65803-3546-4cab-a593-c78309e1356e	1	Indian Reservation, DAPL, Dakota Access pipeline, pipeline, Veterans, Sioux tribe, Standing Rock, Bakkan, fuel
208022ec-e5e4-40f1-8ac4-1c1c70807110	0	
37a24428-f7c5-4fb9-875d-985dd3e48916	1	Dakota Access pipeline, Oil, Donald Trump, U.S. Army Corps of Engineers
fe0fc10f-2e1a-4641-90de-6a278f1c992b	1	Dakota Access pipeline, Oil, drinking water
f3958d6f-41df-49a1-90b5-e7d727cb92f3	0	
be2be1cd-31ca-46ea-8afc-57b8c42a9f62	1	KLP, Dakota Access Pipeline, companies
4afcbd05-c48b-4c2f-bf4f-86b8c29ed449	1	Energy Transfer Partners, Drill, Missouri River, drinking water, Standing Rock Sioux Tribe

533214a9-7df2-4243-8bcc-71c1426906c3	1	Stop Trump, drinking water supplies, stop construction, Dakota Access pipeline
7347deed-dfc1-40dc-9f40-d60ea7db3b2d	0	
63be0e95-1c37-433f-b233-c985b9b6c298	1	Veterans, water protectors, Donation, Oil, Water source
0f44a3a3-f51b-4b10-9bdf-c9e0b6816dbc	0	
596ecaa4-3d80-4e45-ad90-16579975fd23	1	Dakota Access pipeline, Protecting water
98f7f5c1-bdab-4fd9-91f6-53fd09adfac7	0	
fb20dd11-e2b8-4c9f-9cd9-ca3762d4de25	1	Veteran, Mission, Patriots, Protestor, Obama, Campaign, Volunteer, Supplies, Duty, Dakota, Pipeline, Oil, Sioux, Tribe, Fossil fuel, Standing Rock
000e8895-ce59-4abd-b1cb-8fe59a1ef94e	0	Standing Rock, Dakota, Water protectors
4184ee1b-075b-43cd-81e4-477a49f1ffab	1	Standing Rock Sioux, Pipeline, Native American, Energy Transfer Partner, Dakota Access
55232830-681a-45ed-9d01-2817118dbeb8	0	Dakota,Oil, Trump
4d65b6e5-a5dc-4ad4-b45e-ec9868f4a43f	1	Nyemah, Dakota Access Pipeline, Shut down, Protests
67daff7b-2d54-4d00-8c4f-ce0c6ec410d9	0	
46f44087-9145-4fdd-9299-e1cbb1e72797	1	Standing Rock, Dakota Pipeline, Violence, Climate
d326a531-7da1-4bc0-a7ad-203a515290a5	1	Army Corp,Tribe, Police
6567dfec-0992-40f4-8b38-40d1a88a277e	1	U.S. Army Corps of Engineers, climate change, supplies, public comments
95e02eba-7f09-49fb-b520-8b79356fbe42	1	Standing Rock, Resistance, pipeline, water protectors, Morton County, Sioux nation, veterans, Dakota Access Pipeline, Missouri River, Donald Trump, Keystone XL, Indigenous, DAPL, Sioux, Protests
abc0277c-916e-4a55-98dc-b495afe803d8	0	Dakota Access pipeline, indigenous
53d0923f-d4df-475c-8c5c-35c866298608	1	Sacred Stone, Dakota Access Pipeline, NoDAPL movement, tribal, protesters

3.7 Wikipedia As an External Source

We replaced the idea of using manually extracted keywords, with words extracted from the Wikipedia article related to the Dakota Access Pipeline. This is because we believed it would have a larger set of words giving us better results on analysis. For using the Wikipedia article, we extracted all the text using the BeautifulSoup library of Python. We processed this text and then obtained the top frequent words. We also obtained the synsets of the top frequent words to give us a feature vector which we used to compare with other documents.

4. Classification and Topic Modeling

4.1 Feature Extraction

Feature engineering is the process to create features based on the attributes of the data to facilitate machine learning (Forman, 2003; Scott & Matwin, 1999; Sebastiani, 2002). In this project, the frequency of each keyword and its synsets within each document have been calculated as the feature for document classification. The procedure to generate the feature set is illustrated in Figure 8.

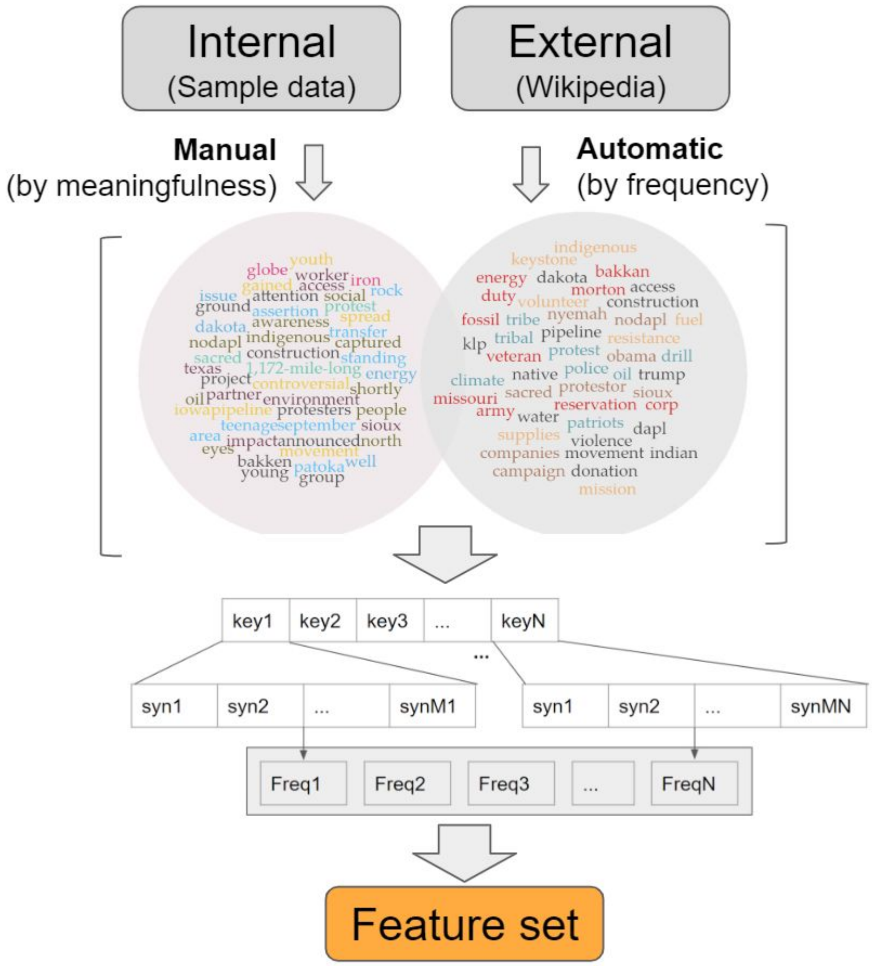

Figure 8. Procedure of feature extraction based on frequency.

First, the keywords are generated from two resources, respectively: internal and external. Specifically, "internal" means the manually selected keywords from sample documents, while "external" represents the automatically extracted frequent words from Wikipedia pages for NoDAPL. Thus, two data sets with NoDAPL related keywords are obtained, where one data set is summarized by meaningfulness and the other is automatically collected by frequency. The reason to generate keywords from two resources is to compare which resources can better classify the testing documents.

It is assumed that there are p keywords in each keyword dataset. As introduced in Section Preprocessing, the synsets for each keyword are generated through the NLTK library. For example, there are M1 synsets for key1 and MN synsets for keyN as shown in Figure 4. The purpose to consider synsets for each keyword is to add to the robustness and representativeness of the feature set.

To get the feature vector for a document, the frequency of each keyword and its synsets is calculated within the document. The frequencies serve as the features, which may be good indicators of the relevance of the corresponding document. The keywords automatically extracted from Wikipedia pages for NoDAPL are in Table 3.

Table 3. Keywords list extracted from Wikipedia on NoDAPL.

['pipeline', 'dakota', 'nodapl', 'social', 'media', 'oil', 'facebook', 'rock', 'people', 'movement', 'standing', 'youth', 'access', 'issue', 'gained', 'project', 'hashtags', 'protest', 'ground', 'twitter', 'bakken', 'iowa', 'protesters', 'large', 'instagram', 'energy', 'announced', 'attention', 'workers', 'patoka', 'young', 'area', 'transfer', 'eyes', 'north', 'construction', 'awareness', 'iron', 'youtube', 'sacred', 'indigenous', 'shortly', 'sioux', 'well', 'globe', 'september', '1,172-mile-long', '3.78', 'hashtag', 'teenage']

Table 4 shows the significant keywords manually selected from sample data.

Table 4. Significant keywords from sample data for NoDAPL.

["access", "army", "bakkan", "campaign", "climate", "companies", "construction", "corp", "dakota", "dapl", "donation", "drill", "duty", "energy", "fossil", "fuel", "indian", "indigenous", "keystone", "klp", "mission", "missouri", "morton", "movement", "native", "nyemah", "obama", "oil", "nodapl", "patriots", "pipeline","police", "protestor", "protest", "reservation", "resistance", "sacred", "sioux", "supplies", "tribal", "tribe", "trump", "veteran", "violence", "volunteer", "water",'nodapl']

The snapshot of the extracted features from the big corpus is presented in Figure 9 as follows:

```
array([[ 0.00962567,  0.01283422,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       ...,
       [ 0.02777778,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.00962567,  0.01283422,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ]])
```

Figure 9. Feature matrix extracted from the big corpus by following the proposed procedure.

4.2 Regularized logistic regression-based classification

With the generated feature set in the previous steps, the unseen documents can be automatically classified into irrelevant and relevant categories. In this project, the regularized Logistic regression is applied and the model is as:

$$h(\boldsymbol{x}_i|\boldsymbol{\beta}) = \log(\frac{p(y_i = 1)}{1 - p(y_i = 1)}) = \beta_0 + \sum_{j=1}^{p} x_{i,j}\beta_j + \epsilon_i$$

$$\min_{\beta} -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(h(\boldsymbol{x}_i|\boldsymbol{\beta})) + (1 - y_i)\log(1 - h(\boldsymbol{x}_i|\boldsymbol{\beta}))] + \frac{\lambda}{n}\sum_{j=1}^{p}|\beta_j|$$

In the above regularized logistic regression, there are n documents, where x_i is the feature vector with length p for document i. Note that the feature-length p is equal to the number of keywords N in Figure 8. y_i is classification result for document i, where $y_i = 0$ represents the document is irrelevant and $y_i = 1$ is relevant. $\beta_0, \beta_1, ..., \beta_p$ are the logistic regression coefficients, which are obtained by minimizing the cost function. The regularization (i.e., second term) of the cost function can help select signification keywords to avoid overfitting, where λ is the tuning parameter. 5-fold cross-validation (CV) is used to select the tuning parameter as well as to evaluate the model performance.

The classification results are demonstrated below:
Classification Results of Regularized Logistic Regression using Keywords from External Source (LR1):

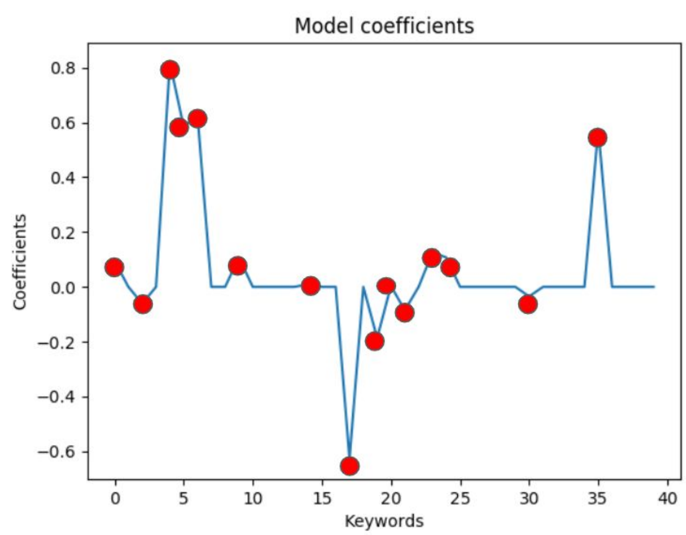

Figure 10. Model coefficients (red dots represent the non-zero coefficients) for LR1.

The significant keywords in LR1 are:
['pipeline', 'social', 'oil', 'rock', 'people', 'youth', 'protest', 'iowa', 'large', 'energy', 'announced', 'workers', 'young', 'awareness', 'sioux'].

Table 5. Confusion matrix of LR1.

		Predicted									
		Fold 1		Fold 2		Fold 3		Fold 4		Fold 5	
		0	1	0	1	0	1	0	1	0	1
True	0	13	1	10	3	11	2	5	2	11	2
	1	1	5	2	5	3	4	6	7	2	6

Table 6. Training and testing accuracy of LR1.

Fold No.	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Training Accuracy	0.827	0.840	0.951	0.889	0.875	0.876
Testing Accuracy	0.900	0.750	0.750	0.600	0.810	0.762

Take Fold 1 as an example. Among the 20 testing documents, there are 6 documents truly relevant to NoDAPL, where 5 documents are correctly predicted to be relevant and the other 1 is mistakenly predicted to irrelevant. On the contrary, there are 14 documents truly relevant to NoDAPL, where 13 documents are correctly predicted to be irrelevant and the other one mistakenly predicted to be relevant. In summary, the training accuracy is 0.876 and the testing accuracy is 0.762 for LR1.

Classification Results of Regularized Logistic Regression using Keywords from Internal Source (LR2):

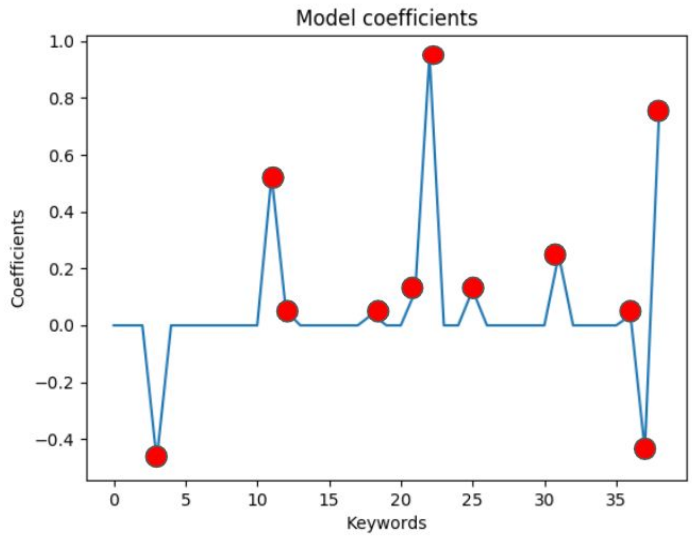

Figure 11. Model coefficients (red dots represent the non-zero coefficients) for LR2.

The significant keywords in LR2:
['climate', 'energy', 'fossil', 'missouri', 'native', 'oil', 'police', 'supplies', 'violence', 'volunteer', 'water'].

Table 7. Confusion matrix of LR2.

		Predicted									
		Fold 1		**Fold 2**		**Fold 3**		**Fold 4**		**Fold 5**	
		0	**1**	**0**	**1**	**0**	**1**	**0**	**1**	**0**	**1**
True	**0**	0	1	0	1	0	1	0	1	0	1
	1	9	3	9	4	7	2	10	4	9	3

Table 8. Training and testing accuracy of LR2.

Fold No.	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Average
Training Accuracy	0.876	0.988	0.876	0.876	0.962	0.916
Testing Accuracy	0.700	0.650	0.700	0.800	0.571	0.684

Taking Fold 1 as an example, among the 20 testing documents, there are eight documents truly relevant to NoDAPL, where 5 documents are correctly predicted to be relevant and the other three are mistakenly predicted to irrelevant. On the contrary, there are 12 documents truly irrelevant to NoDAPL, where nine documents are correctly predicted to be irrelevant and the other three mistakenly predicted to be relevant. In summary, the training accuracy is 0.916 and the testing accuracy is 0.684 for LR2. As a result, the trained classifier LR1 with the highest accuracy (say, Fold 1) is selected to classify the small and big corpus, which lead to 130 relevant documents for 480 documents in the small corpus and 3702 relevant documents for 11081 documents in the big corpus.

4.3 LDA-based Topic Clustering

Based on the documents classified to be relevant, the latent Dirichlet allocation (LDA) (Blei, Ng, & Jordan, 2003) and its deep learning variant LDA2Vec (Moody, 2016) are adapted to extract topics covered by the NoDAPL corpus. Specifically, LDA and LDA2Vec are tested on relevant documents classified from both the small corpus and the big corpus. Although the classifier has satisfactory accuracy and Type I and Type II errors, the testing performed on the corpus cannot be guaranteed due to unknown events/topics which fall outside of the scope of Wikipedia. Therefore, to further evaluate the performance of classification and to perform clustering on the corpus, LDA is investigated to automatically extract topics. The rationale to choose LDA for evaluation lies in the facts that 1) limited size of the mis-classified irrelevant documents will not affect the topic modeling performance, and will not have significant impact on the sentence ranking to get a good summary; and 2) significant size of the mis-classified irrelevant documents will be reflected as noise or as irrelevant topics in the LDA results.

4.3.1 Latent Dirichlet Allocation

LDA is a generative probabilistic model for collections of discrete data such as text corpora, which is a three-level hierarchical Bayesian model (Blei *et al*., 2003). In other words, LDA models each document within a collection as a finite mixture of topics and represents each topic as an infinite mixture distribution over an underlying set of topic probabilities. In this project, we use the LDA model provided by *Gensim* (https://radimrehurek.com/gensim/), which is a Python library for topic modeling. To visualize the topic discovery results, we further use *pyLDAvis* (https://github.com/bmabey/pyLDAvis), which is a Python library for interactive topic model visualization.

In order to evaluate the classification testing performance on the corpus, the LDA analysis is performed on both the small corpus before classification and after filtering out the irrelevant documents by classification. The results comparison is shown in Figure 11 and Figure 12 as follows.

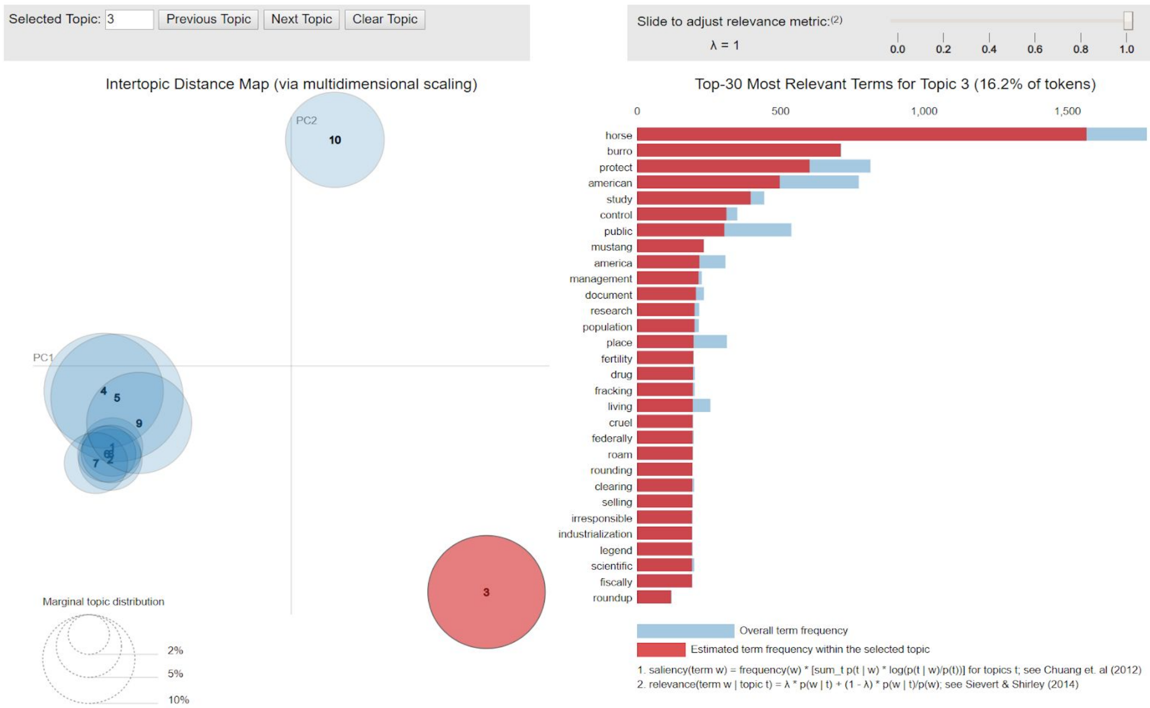

Figure 12. LDA analysis on original corpus without classification, the Topic 3 in red (*i.e.*, circle in red) is irrelevant according to the frequent word list on the right hand side.

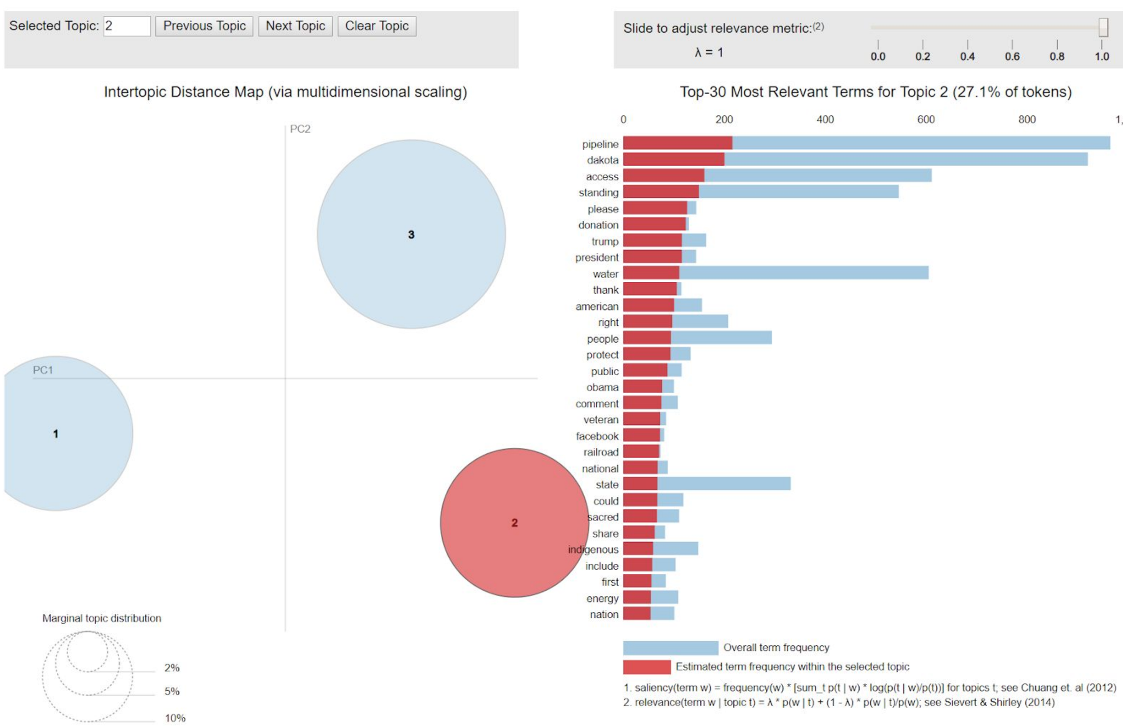

Figure 13. LDA analysis on the corpus after classification (relevant documents only). Three topics are found from the small corpus, each represents different relevant NoDAPL events/topics.

By comparing Figure 12 and Figure 13, it can be concluded that the testing performance of the classifier is satisfactory since the topics are perfectly separated and they are meant for describing the NoDAPL events. Specifically, three topics include:

1. Topic 1 is about the activities from the community,
2. Topic 2 covers the government's activities and attitude towards the NoDAPL events,
3. Topic 3 discusses the protesting details.

These topics can only be treated as a subset of the topics in the big corpus since 130 relevant documents may not cover all the topics in the big corpus. Therefore, we applied the LDA on the big corpus, which resulted in five topics as follows (pyLDAvis visualization in Figure 14):

1. Topic 1 is about the donation petition to veterans of the United States Armed Forces for defending the water protectors from assault and intimidation at the hands of the militarized police force and DAPL security;
2. Topic 2 covers the government's activities and attitude towards the NoDAPL events;
3. Topic 3 is about protest preparations or meetings, including time, location and people;
4. Topic 4 is about the details of the protest in terms of time, location, argument, and so on;
5. Topic 5 talks the pipeline project status from the project owner and stockholder viewpoint.

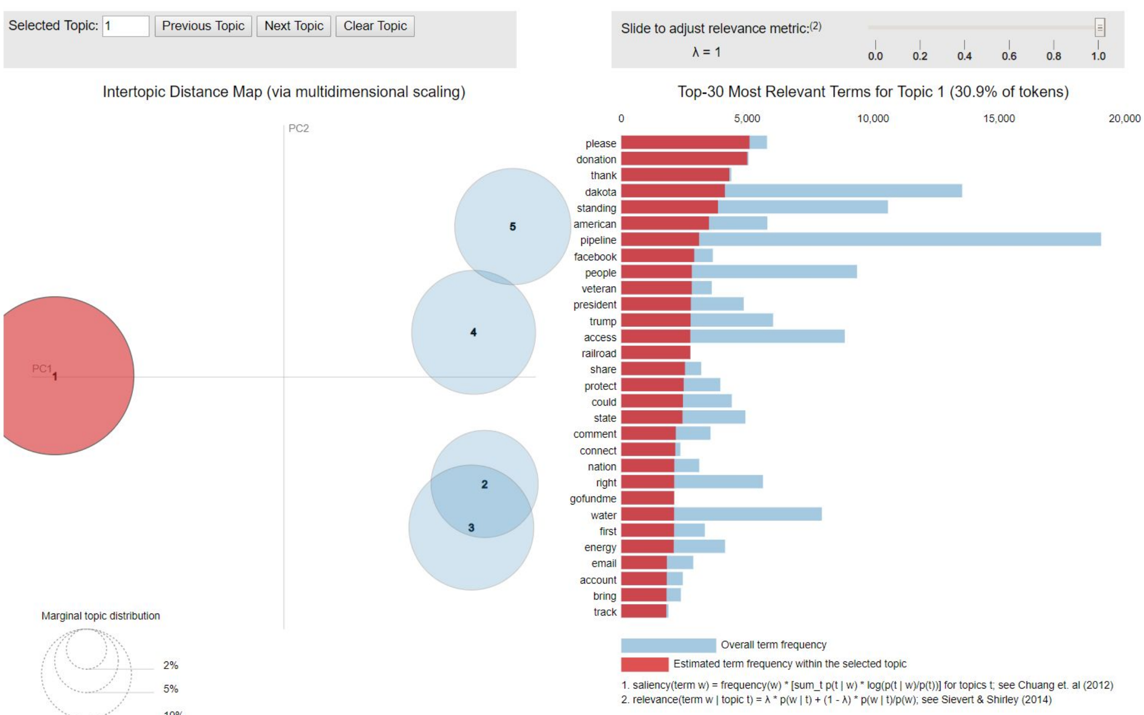

Figure 14. Topics extracted from the big corpus after performing classification (relevant documents only). Five topics are found, each represents different topics.

After obtaining the topics, we further use the trained LDA model to calculate the similarities between the documents and topics, to create document clusters in terms of topics. Specifically, the similarity between one document and five topics can be computed by using *similarity* *queries*. The similarity queries will also be used for sentence ranking in the extractive section. Then, the document will be assigned to the topic which has the highest similarity score (range: 0~1). As a result, five clusters are created from the relevant big corpus, and they contain 357, 700, 884, 924, and 837 documents, respectively. These clusters will be used for extractive and abstractive sentence generation in the following sections.

4.3.2 LDA2Vec Topic Modeling

LDA2Vec, which is a deep learning variant of LDA topic modeling, was recently proposed by Moody (Moody, 2016) to mix traditional LDA with word embedding methods, which are typically used to convert words to numerical vectors of a certain length as input to neural networks. The most famous word embedding method is called *word2vec (*Mikolov, Chen, Corrado, & Dean, 2013), which can efficiently compute the numerical representation of words in vector space for the large dataset. The LDA2Vec model mixes the best parts of word2vec and LDA into a single framework. To this end, we compared the performance of LDA2Vec and regular LDA on both the small corpus and the large corpus. The results are shown in terms of topics found:

LDA2Vec on SMALL corpus with relevant documents only:

1. Topic 0 keywords: access water stop rock people trump -PRON- construction tell president;
2. Topic 1 keywords: -PRON- donation thank please access people trump share rock land;
3. Topic 2 keywords: people standing rock -PRON- water say north access camp right.

LDA2Vec on BIG corpus with relevant documents only:

1. Topic 1 keywords: protester protest declare national site near rock north climate indigenous
2. Topic 2 keywords: and people go want place think story child feel the woman
3. Topic 3 keywords: climate stop access drink site construction tell indigenous deadline right
4. Topic 4 keywords: thank -PRON- please stand this right people president donation access
5. Topic 5 keywords: project bank shut access construction say activist protest etp report

According to the results, the traditional LDA outperforms the LDA2Vec due to the fact that the topics found by LDA are consistently better than the topics found by LDA2Vec, as evaluated by the team members. Therefore, we choose LDA to perform the topic modeling and clustering.

5. Extractive Method

Extractive methods in the process of automatic summarization involve selection of subsets of keywords, sentences, and phrases from the initial text document without any modification to create a short summary. The typical method to extract important sentence is sentence ranking. In this project, we investigated two sentence ranking methods, namely, TF-IDF based ranking, and LDA based ranking.

5.1 TF-IDF based Ranking

In creating an extractive summary, we need sentences which would rightly give the essence of the document and would make sense when stitched together. In other words, we need the most relevant sentences in a given document which would cover the entire breadth of discussion to summarise that text. The order of these sentences was equally important for the development of a coherent summary. To deal with this, we decided to extract and rank the sentences according to the TF-IDF technique.

5.1.1 TF-IDF Measures

The two most important components of TF-IDF technique are the Term Frequency (TF), which is a count of the number of times a word repeats itself in a given document, and the Inverse Document Frequency (IDF), which is the inverse of the number of documents a word occurs in, in the given corpus of documents. These two components are then used in calculating the weight of any given word. If a term occurs several times in a document, its term frequency increases and hence it has a positive effect on its weight. Now, words like *the* have a very good chance of a high term frequency but are of very low importance to us. Therefore, to adjust this fact, the inverse of the document frequency is taken into account which in turn lowers the weight of the words which are frequent in both the document and the corpus.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

We tried to expand this logic in sentences such that we considered a document as an individual tokenized sentence and the corpus was our set of sentences making our document. Doing this helped us in ranking our sentences in decreasing order of relevance, as we wanted, according to a query.

5.1.2 Corpus for Training

The primary requirement of a training corpus was to create a bag of words for the document. To do this, we played with several corpora hoping that different ones would give us different results. We began our experimentation by using the Brown Corpus, which contains roughly 500 samples of English language texts, consisting of around one million words compiled from words published in the United States. The intuition behind using this corpus was its small size and random distribution of its texts in different genres. The next we tried was the Wikipedia Corpus. This corpus contained around 4.4 million articles with roughly 1.9 billion words. The problem we found in this corpus was its huge size which was somewhat difficult to manage and unnecessary for our use case. Having realized this, we finally used our own corpus since it was enough to create a perfect count vector for the vocabulary.

5.1.3 Creating a Count Vector

To create a count vector (*a.k.a.* bag of words) we have used *sklearn*'s CountVectorizer (https://scikit-learn.org/stable/). The primary reason behind using CountVectorizer was its ability to tokenize and count the occurrence of terms in the same class. One has the flexibility to use any count function since more or less each gave the same sparse matrix.

5.1.4 Building the TF-IDF Matrix

A tf-idf matrix is a sparse matrix with weights assigned to each term in each document as shown in Figure 15. The CountVectorizer, as discussed, tokenizes and counts the occurrence of terms in the same class. We then pass this output into *sklearn's* TfidfTransformer which gives us the required tf-idf matrix. We can also use the TfidfVectorizer which would have performed the three tasks (tokenize, count, and create tf-idf matrix) in the same class. This, though, would have no effect on the results.

$$\begin{array}{cc} & \begin{array}{cccc} T_1 & T_2 & \cdots & T_t \end{array} \\ \begin{array}{c} D_1 \\ D_2 \\ \vdots \\ \vdots \\ D_n \end{array} & \begin{pmatrix} w_{11} & w_{21} & \cdots & w_{t1} \\ w_{12} & w_{22} & \cdots & w_{t2} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ w_{1n} & w_{2n} & \cdots & w_{tn} \end{pmatrix} \end{array}$$

Figure 15. An example of TF-IDF matrix.

5.1.5 Scoring Each Sentence

Scoring the sentences was the most important step in this process since it is now that we will assign ranks. One method could've been to take a summation of all the weights in a given sentence. This was one of the most basic methods which needed some advancements. Therefore, we used a few additional techniques which include:

1. We only considered and added the tf-idf values where the underlying token was a noun. This total was then divided by the summation of all the document tf-idf values.
2. We added an additional value to a given sentence if it had any word that was included in the title of the document. This value was equal to the count of all words in a sentence found in the title divided by the total number of words in the title. This "heading similarity score" was then multiplied by an arbitrary constant (0.1) and added to the tf-idf value.

3. We then applied a position weighting. Each sentence was ordered from 0 to 1 equally based on the sentence number in the document. This weighting was then multiplied by the value in point 2.

5.2 LDA-based Ranking

Given the fact that TF-IDF may not be efficient for large-scale dataset due to the generated large frequent words dictionary and large TF-IDF matrix (see example in Figure 15), we further propose to use a trained LDA model to rank the sentences within each document for all the topics. The rationale to use LDA similarity queries as ranking scores is: given a specific topic, extracted sentences should be highly relevant to this topic. As a result, a list of sentences and the associated similarity measures (range: 0~1) are obtained for each document. The Top-N sentences can be extracted by sorting the similarity scores.

6. Abstractive Summarization

The abstractive summarization aims to produce a generalized summary, conveying information in a concise way. It tries to understand the main concepts used in a document and represent them in basic natural language. Linguistic methods are involved in understanding and examining the text. After understanding of the text, it finds the new notions and terms to best clarify it, and then generates a new brief text that can represent the most significant information from the original text document using these notions and terms (Saziyabegum & Sajja, 2016).

6.1 Abstract from PGN (Point Generator Networks)

We made an attempt in applying a deep learning approach in abstractive summarization. To get brief abstracts for each webpage for our dataset, we used the hybrid pointer-generator networks (PGN) approach developed by See *et al*. (See, Liu, and Manning 2017). This hybrid network is able to copy words from the source via pointing while retaining the ability to generate words from a fixed vocabulary. Because there are similarities between our NoDAPL documents and See's dataset in terms of vocabulary and media source (i.e., they were both derived from news articles published by major news media like CNN), we decided to used their published model trained with the CNN/Daily Mail dataset, which contains online news articles paired with multi-sentence summaries.

We implement the PGN model using the same version of TensorFlow (1.2.1) in the process to match the pre-trained model. It is highly recommended to run the process in a Unix environment. Here are some lessons we had in the initial test run:

a) TensorFlow doesn't support Python 2.7 in the Windows operating system.
b) TensorFlow currently only supports up to Python 3.6 which means we have to uninstall the latest Python version 3.7.
c) Tensorflow-GPU supports Nvidia graphics card but not the AMD Radeon.

6.1.1 Data file preparation

We converted the NoDAPL dataset from JSON file to binary data file to feed the PGN model. This is the workflow to prepare the data file:

a) Create a unique hash from each document URL in the JSON file which contains "URL" and "Sentences" tags.
b) Create .story file containing the content of each article with that URL,
c) Export all URLs to a text file named "all_urls.txt" one per line,
d) Use PTBTokenizer in Stanford CoreNLP to tokenize the data,
e) Create binary data from the tokenized data for PGN with *make_datafiles_py3.py* script.

6.1.2 Beam search decoder

The framework we implemented replies on the encoder-decoder paradigm. The encoder encodes the input sequence of words, while the decoder, which uses a beam search algorithm, transforms the probabilities over each word in the vocabulary and produces a final sequence of words.

The hyperparameters we use for the beam search decoder are max_enc_steps=400, max_dec_steps=120. The coverage=1 eliminates repetition of same words. We used this command to generate the abstract text file from each input document:

```
python pointer-generator/run_summarization.py --mode = decode --data_path = finished_files/chunked/test_* --vocab_path = finished_files/vocab --log_root = log --exp_name = pretrained_model --max_enc_steps = 400 --max_dec_steps = 120 --coverage = 1 --single_pass = 1
```

Besides, this command generates attn_vis_data.js which can be used for visualization of attention:

```
python pointer-generator/run_summarization.py --mode = decode --data_path = finished_files/chunked/test_* --vocab_path = finished_files/vocab --log_root = log --exp_name = pretrained_model
```

Here is an example of generated abstracts (Figure 16). Besides, we modified the code so that the visualization will randomly display an abstract in each page reload.

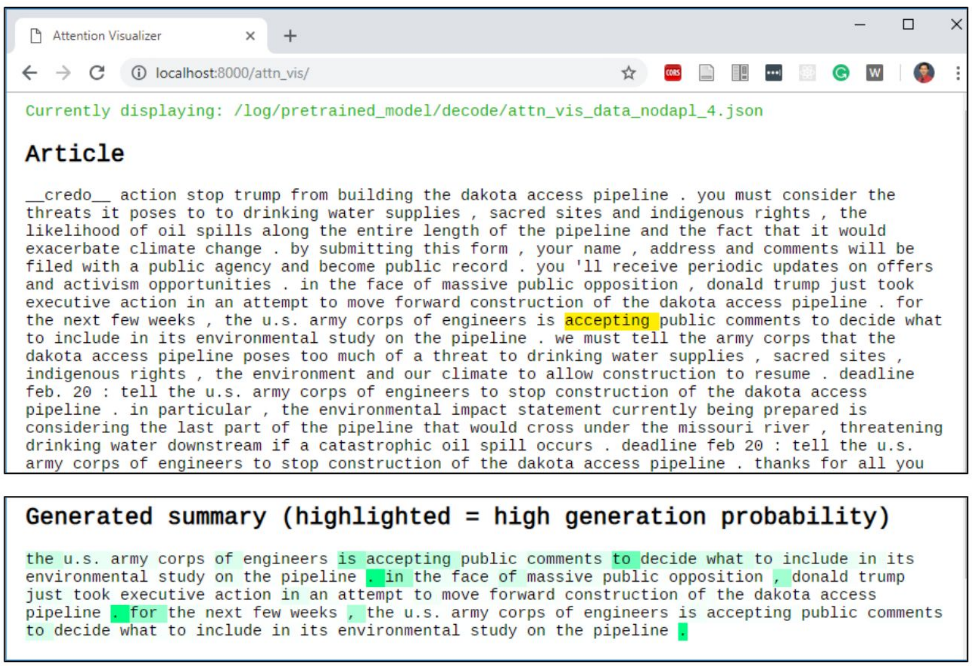

Figure 16. Example of a PGN-generated abstract (in attention visualization).

7. Hybrid Summarization by Sentence Re-ranking

We proposed a hybrid automatic summarization approach to further improve the summarization performance in two considerations: 1) the extractive summarization (*i.e.*, LDA-based ranking) only leads to sentences which are highly relevant to the corresponding topic but ignores the important named entities; 2) the abstractive summarization (*i.e.*, PGN) produces sentences which are high-level summarization of the document including the named entities but may not have high relevance with the topics. Therefore, we propose to re-rank these sentences to generate better summarization according to: 1) the topics and 2) the named entities.

A flowchart of the proposed method is shown in Figure 17. In each topic, the LDA-based similarities and the NERs-based frequency are calculated for each of the abstractive sentences and each of the extractive sentences, where the NERs-based frequency is a sum of frequencies of all named entities in one sentence. These measures are organized in vectors (*i.e.*, V1~V4). Afterward, the V1 and V3, V2 and V4 are concatenated, respectively. The concatenated LDA-based similarity and concatenated NERs-based similarity are added as a convex combination with adjustable weight. The higher the weight is, the more relevant the re-ranked sentences are to the topic. The lower the weight is, the more important the named entities are. These re-ranked sentences will be joined into a paragraph as a summary for the documents under the Topic. The cleaned paragraph will be obtained after further removing the duplicate sentences from the paragraph. During the duplicate sentence removal, we observed sentences in German. Therefore, we manually translated sentences into English without any other manipulations. Finally, the cleaned paragraphs from all topics are combined as the summarization of the whole corpus.

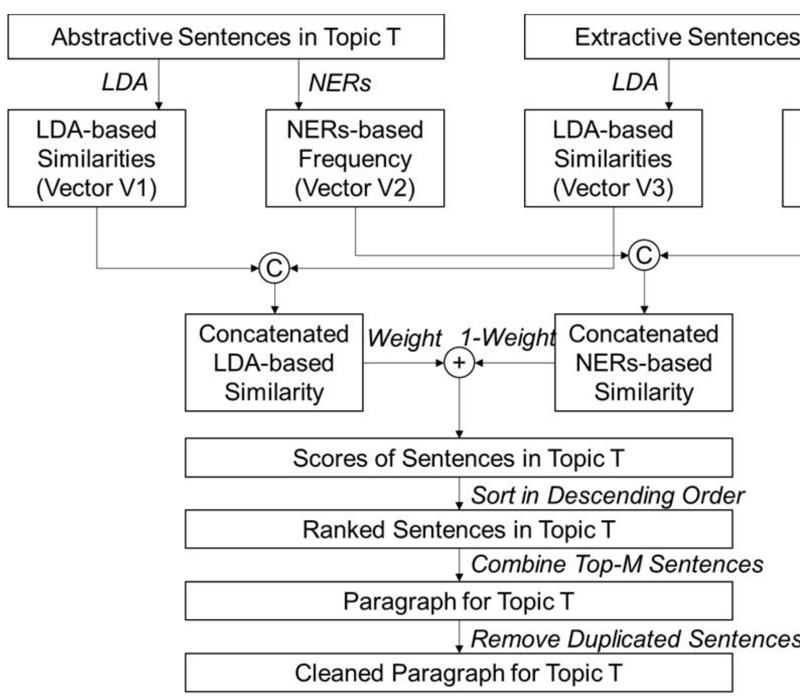

Figure 17. Flowchart of the proposed hybrid summarization method.

7.1 Named Entity Recognition

A named entity is a "real-world object" that is assigned a name – for example, a person, a country, a product, or a book title. Named-entity recognition (NER) is a subtask of information extraction that seeks to locate and classify named entity mentions in unstructured text into predefined categories such as the person names, organizations, locations, etc. We sought to use spaCy's NER. spaCy features an extremely fast statistical entity recognition system, that assigns labels to contiguous spans of tokens. The default model identifies a variety of named and numeric entities, including companies, locations, organizations, and products and also gives us the ability to add arbitrary classes to the entity recognition system and update the model with new examples. Named entities are extracted from each topic clusters. As a result, five sets of named entities are obtained. Table 9 shows an example of the extracted named entities from the NoDAPL dataset.

Table 9. An example list of identified named entities.

```
GPE Need
GPE Plymouth
ORGANIZATION Dakota Access
ORGANIZATION Dakota Access
ORGANIZATION Dakota Access
ORGANIZATION Dakota Access
LOCATION North Dakota Pipeline
PERSON Energy Transfer Partners
GPE Rock
PERSON Rock Sioux Tribe
ORGANIZATION Army Corps
GPE US
ORGANIZATION US Army Corps
ORGANIZATION Engineers
PERSON Katy Perry
PERSON Orlando Bloom
PERSON Lupita Nyong'o
PERSON Whoopi Goldberg
PERSON Mark Ruffalo
```

Table 10 Descriptions of types of named entity.

Type	DESCRIPTION
PERSON	People, including fictional
ORGANIZATION	Companies, agencies, institutions etc
GPE	Countries, cities, states
LOCATION	Non GPE locations, bodies of water etc
EVENT	Named hurricanes, sport events etc
DATE	Absolute or relative dates or periods
TIME	Times smaller than a day
ORDINAL	First, second etc
CARDINAL	Numerals that do not fall under another type

7.2 Compiled summary

In this project, we set weight as 0.8 (range: 0~1) to put more importance on the relevance between the sentences and the corresponding topics. The generated summary of five topics (*i.e.*, extracted by LDA) are as follows:

Automatic Summary for NoDAPL

President obama says the dakota access pipeline will be delayed. Thank you for temporarily halting the dakota access. Thank you for temporarily halting the dakota access pipeline. We know that president elect trump has a serious conflict of interest by owning large investments in dapl and other fossil fuel assets; and his energy team includes harold hamm, billionaire founder of continental resources oil company, and someone mr. Trump might name as his secretary of energy. After the failed keystone pipeline, a new and virtually regulation free dakota access pipeline was approved. The dakota access pipeline will be built on top of several burial grounds and sacred sites. Veterans of the united states armed forces, including the u.s. Army, u.s. Air force and u.s. Coast guard and we are calling for our fellow veterans to assemble as a peaceful, unarmed militia at the standing rock indian reservation on dec 4-7.

Members of the new orleans community stood with water protectors at standing rock to protest dakota access pipeline. Arlea ashcroft has been to the standing rock camps in north dakota three times. The wplc provided legal support on the front lines of the dakota access pipeline resistance at standing rock is far from over. This year, all were in proud support of the courageous and determined struggle of the native people in standing rock, north dakota, against the dakota access pipeline. Everyone keeps asking how they can help fight against the dakota access pipeline. By now, we all know how that turned out : trump steamrolled efforts to block the dakota access pipeline. Sioux in the dakota borderlands and thousands of their supporters are camped near the dakota access pipeline construction site north dakota. Cambridge is standing in solidarity with the water protectors and indigenous nations at standing rock. Oceti sakowin camp is one of eight stanley mission residents walking to standing rock sioux tribe reservation in support of the protests towards the dakota access pipeline. The first amendment is under attack along the dakota access pipeline route in north dakota. The standing rock nation film & music festival has been created to support the water protector movement that began at standing rock. Since the beginning of this movement, thousands have come to standing rock to serve as true allies and guests of the standing rock while dapl construction is paused.

Native american food has spread around the world. Sioux-stammen standing rock demonstrerar. No one at standing rock was going anywhere. John floberg of standing rock. Sarah shomin was one of the first friends i made at standing rock. They are capitalizing on online interest in standing rock, and native american culture in general, to make money. Buzzfeed news identified more than 60 facebook pages with more than 6 million fans that are generating money either by selling counterfeit native american merchandise, or by driving traffic to ad-filled websites that in some cases have little or nothing to do with native american issues.

Activists and tribes protesting plans to run the dakota access oil pipeline under a lake near the standing rock sioux reservation in north dakota. Lake oahe is the water source for the standing rock sioux tribe. The standing rock sioux tribe and cheyenne river sioux tribe have filed court actions to stop the dakota access pipeline. The dakota access pipeline began moving north dakota oil to illinois on june 1. Months of intense protests and clashes with north dakota law enforcement occurred at the standing rock sioux reservation in north dakota. The pipeline is owned, in part, by energy transfer partners, the same company behind the dakota access pipeline. The leader of the standing rock sioux tribe called on barack obama to stop the dakota access pipeline. Military veterans gathered to protest the dakota access pipeline in north dakota earlier this month. As of this week, bakken oil is expected to flow through the dakota access pipeline under lake oahe near the standing rock sioux reservation. If completed, the dakota access pipeline would route crude oil from north dakota to illinois. The conflicts along 1 172 miles of the dakota access pipeline, the site of months of clashes near the standing rock sioux reservation in north dakota. North dakota police claim rubber bullets not used on standing rock protesters. Reuters oil is expected to start flowing through the dakota access pipeline near the standing rock sioux reservation soon. Camps close to where the pipeline is being constructed under lake oahe in north dakota.

In particular, the environmental impact statement currently being prepared is considering the last part of the pipeline that would cross under the missouri river, threatening drinking water downstream if a catastrophic oil spill occurs. We must tell the army corps that the dakota access pipeline poses too much of a threat to drinking water supplies, sacred sites, indigenous rights, the environment and our climate to allow construction to resume. The dakota access pipeline would result in oil leaks in the missouri river watershed. Oil pipeline threatens standing rock sioux land, drinking water holy places. For the next few weeks, the u.s. Army corps of engineers to stop construction of the dakota access pipeline.

8. Evaluation

Summarization evaluation methods can be classified into two categories (Jones & Galliers, 1995). In extrinsic evaluation, the summary quality is judged on the basis of how helpful summaries are for a given task, but intrinsic evaluation is directly based on analysis of the summary. Within the scope of these two categories, we applied two measures to assess the quality of our automatic summary: a *subjective evaluation* which evaluates document quality and coverage from human perception and tasks-taking, and an automatic ROUGE (Recall-Oriented Understudy for Gisting Evaluation) evaluation which gives a precise and quantitative results from statistics.

Both measurements involve a comparison with a "gold standard summary" to measure how many main ideas of the source document are covered by our automatically generated summary. Such a "gold standard" varies in the two measurement. In subjective evaluation, we used a self-developed summary guideline (or questionnaire) which covers the most important aspects of the NoDAPL event, while in the ROUGE evaluation we used a summary developed and refined by another team in our class.

8.1 Extrinsic Subjective Evaluation

Developing a standard summary for evaluation is a very challenge task. To have a better idea about which important information should be covered in the summary, we established a guideline for this event based on both internal and external resources, including the first 100 articles from our large collections, Wikipedia content, and various news reports. These resources helped us to gain a whole picture about the NoDAPL event in several aspects like event cause, people, actions and attitude, and so on. Based on these aspects, we further developed the guideline in detail as a list of questions covering different folds like the following:

- About the timeline
- About the facts: NoDAPL hashtag, pipeline project
- What happened to the opposition party: Native Americans or activists
- What happened to the supporting party: Energy Transfer Partners
- What were governments' attitudes: Obama and Trump administration, North Dakota state government, etc.

Here is the full list of detailed guidelines with related questions:

Summarization Guideline for NoDAPL

T1. The timeline

a. When and where did the pipeline start construct?
b. When and where did the campaign start?
c. When and where did the campaign end?
d. When did the pipeline begin to service?

T2. The pipeline project fact

a. What was the purpose of building the pipeline?
b. Who are the pipeline owners and constructors?(*e.g.*, Energy Transfer Partners)
c. What is the magnitude or dimension of the project?
d. Is there any details about the construction?
(*e.g.*, project cost, stakeholders, *etc*.)
e. What are the status changes of the project? (*e.g.*, suspended, in service)

T3. The hashtag fact (optional)

a. What are the meaning of the hashtag? (*e.g.*, NoDAPL, DAPAL, DAPL)
b. What are the slogans of the movement?
c. Who and how it started in social media?

T4. The opposition party

a. Who (or which organizations) opposed the project?
(*e.g.*, Native Americans tribes, social groups or activists, *etc.*)
b. How many people involved in the campaign?
c. What are the reasons they against the project?
(*e.g.*, environment, water, sacred land, public health)
d. What are the affected area or states from this project?
(*e.g.*, Standing Rock Sioux Reservation, *etc*.)
e. What kinds of the campaign or protest? (*e.g.*, violence or non-violence)
f. Are there any major conflicts?
g. Are there any people were injured or arrested in the protest?
h. Are there any lawsuits / legal actions involved? (*e.g.* at Federal or state court)
i. Are there any positive progress or results for the party from the campaign?

T5. The supporting party

a. Who are the people supporting the construction?
b. What are the reasons they support the construction?

T6. Government attitudes

a. Did the administrations react to the event?
(*e.g.*, Obama or Trump administration, North Dakota state governors)
b. Are there any government agencies get involved / what they did?
(*e.g.*, U.S. Army Corps of Engineers, State police, Department of Justice, Department of the Interior, and Environmental Protection Agency, *etc*.)

8.1.1 Task-based Evaluation

We used a task-based method to assess the coverage of an automatic summary. It finds out how many questions in our guideline can be answered in the summary. Just like the way of doing a reading comprehension exercise, a reader should be able to find all answers from the paragraphs without a priori knowledge about the topic. Similarly, we tried to link each individual sentence in the summary to the guideline question, and then counted the numbers of matched sentences and unanswered questions.

For example: The first sentence in the first paragraph "*President obama says the dakota access pipeline will be delayed*" answers question T6-a "*Did the administrations react to the* *event?*", so we marked it as a match to indicate that one of the two aspects in "government attitudes" is covered in the summary. Table 11 shows the matching list from this task.

Table 11. Question - sentence matching table for subjective evaluation (Blank cell indicate sentence is irrelevant to NoDAPL).

Sentence ID	Sentences	Question ID
I-1	President obama says the dakota access...	T6-a
I-2	Thank you for temporarily halting the...	T4-i
I-3	Thank you for temporarily halting the...	T4-i
I-4	We know that president elect trump has...	T6-a
I-5	Trump might name as his secretary of...	
I-6	After the failed keystone pipeline, a ...	T2-e
I-7	The dakota access pipeline will be built...	T4-c
I-8	Veterans of the united states armed force...	T4-a
II-1	Members of the new orleans community ...	T4-a
II-2	Arlea ashcroft has been to the standing ...	T4-e
II-3	The wplc provided legal support on the...	T4-f
II-4	This year, all were in proud support of ...	T4-a
II-5	Everyone keeps asking how they can help ...	T4-b
II-6	By now, we all know how that turned out ...	T4-i
II-7	Sioux in the dakota borderlands and...	T4-e
II-8	Cambridge is standing in solidarity with...	T4-a
II-9	Oceti sakowin camp is one of eight stanl...	T4-a
II-10	The first amendment is under attack along...	T4-i
II-11	The standing rock nation film & music...	T4-e
II-12	Since the beginning of this movement...	T4-i
III-1	Native american food has spread around ...	
III-2	Sioux-stammen standing rock demonstrerar...	T4-a
III-3	No one at standing rock was going...	T4-e
III-4	John floberg of standing rock....	T4-a

III-5	Sarah shomin was one of the first friend...	
III-6	They are capitalizing on online interest...	
III-7	Buzzfeed news identified more than 60...	
IV-1	Activists and tribes protesting plans to...	T4-c
IV-2	Lake oahe is the water source for the st...	T4-c
IV-3	The standing rock sioux tribe and...	T4-h
IV-4	The dakota access pipeline began moving ...	T2-e
IV-5	Months of intense protests and clashes w...	T4-f
IV-6	The pipeline is owned, in part, by ...	T2-b, T5-a
IV-7	The leader of the standing rock sioux ...	T4-a
IV-8	Military veterans gathered to protest the...	T1-b
IV-9	As of this week, bakken oil is expected ...	T1-d
IV-10	If completed, the dakota access pipeline...	T4-d
IV-11	The conflicts along 1 172 miles of the ...	T2-c, T4-d
IV-12	North dakota police claim rubber bullets...	T6-b
IV-13	Reuters oil is expected to start flowing...	T1-d
IV-14	Camps close to where the pipeline is being ...	T4-e
V-1	In particular, the environmental impact ...	T4-c
V-2	We must tell the army corps that the ...	T4-c
V-3	The dakota access pipeline would result ...	T4-c
V-4	Oil pipeline threatens standing rock ...	T4-c
V-5	For the next few weeks, the u.s. Army ...	T4-i

According to Table 11, we evaluate the summary quality in terms of content relevancy and content coverage:

(1) *Content relevancy*: among the 46 sentences from 5 paragraphs (topics), there are only 4 sentences that are irrelevant to NoDAPL, showing a very high content relevancy (91.3%) of the automatic summary to the NoDAPL event.

(2) *Content coverage*: We found 16 out of 23 questions (69.6%, except for optional T3) were answered by all the sentences. (Questions covered per category: T1: 2 out of 4, T2: 3 out of 5, T3: 0 out of 3, T4: 8 out of 9, T5: 1 out of 2, T6: 2 out of 2)

These question-sentence matching results indicate a decent quality of the automatically generated summary.

8.2 Intrinsic Evaluation with ROUGE

Recall-Oriented Understudy for Gisting Evaluation (ROUGE), is a set of metrics and a software package used for evaluating automatic summarization and machine translation software in natural language processing. It is used to compare the automatically generated summary with the summary used as reference. It has the following five evaluation metrics:

- ROUGE-N: Overlap of N-grams between the system and reference summaries.
  - ROUGE-1 refers to the overlap of *1-gram (each word)* between the system and reference summaries.
  - ROUGE-2 refers to the overlap of *bigrams* between the system and reference summaries.
- ROUGE-L: Longest Common Subsequence (LCS) based statistics. Longest common subsequence problem takes into account sentence level structure similarity naturally and identifies the longest co-occurring in sequence n-grams automatically.
- ROUGE-W: Weighted LCS-based statistics that favors consecutive LCSes.
- ROUGE-S: Skip-bigram based co-occurrence statistics. Skip-bigram is any pair of words in their sentence order.
- ROUGE-SU: Skip-bigram plus unigram-based co-occurrence statistic

The following are different ways to interpret ROUGE scores:

- ROUGE-n recall=30% means that 30% of the n-grams in the reference summary are also present in the generated summary.
- ROUGE-n precision=30% means that 30% of the n-grams in the generated summary are also present in the reference summary.
- ROUGE-n F1-score=40% is more difficult to interpret, like any F1-score.

Table 12 shows our ROUGE recall scores comparing our generated summary to the golden standard made by team 7 (reference summary).

Table 12 ROUGE Scores

ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-SU4
0.2075	0.06112	0.14821	0.06493

On evaluation of ROUGE score by sentence list we get the following results:

Max ROUGE-1 score among sentences: 0.50000
Predicted Sentence: President obama says the dakota access pipeline will be delayed.
Golden Sentence: Furthermore, based on original Dakota Access Pipeline design.

Max ROUGE-2 score among sentences: 0.40000
Predicted Sentence: President obama says the dakota access pipeline will be delayed.
Golden Sentence: Furthermore, based on original Dakota Access Pipeline design.

Table 13 and 14 show entities in the generated summary and reference summary. The entity coverage of our summary is 2.60%.

Table 13 Entities from the generated summary

['Cambridge', 'North', 'first', 'american', 'Buzzfeed', '&', 'John', 'Sioux', 'june 1.', 'u.s', 'one', 'Air force', 'Arlea', 'three', 'Oceti', '4-7', 'This year', 'the beginning of this movement', 'the next few weeks', 'earlier this month', 'Sarah', '1 172 miles', 'eight', 'Months', 'Veterans', 'more than 6 million', 'this week', 'more than 60', 'Army', 'thousands']

Table 14 Entities from the golden standard

['Shailene Woodley', 'the Corps of Engineers', 'Sioux', 'Missouri', 'Energy Transfer Partners', '$3.8 Billion', 'the National Environmental Policy Act', 'Oct. 26, 2016', 'The United Nations Permanent Forum on Indigenous Issues', 'Sept. 26 2016', 'NHPA', 'Energy Transfer', 'Illinois', 'December 2016', 'DAPL', 'July 27, 2016', 'the National Historic Preservation Act', 'January 24, 2017', '2010', 'Dakota Access', 'Sept. 9, 2016', 'Jan. 14, 2017', 'Obama', 'Nov. 1, 2016', 'US', "Standing Rock's Historic Preservation Officer", 'February 23, 2017', 'Washington', 'NEPA', 'Oct. 9, 2016', 'South Dakota', 'Donald Trump', 'Dakota Access Pipeline', 'IUB', 'April 2017', 'Facebook', 'Bakken', 'Dakota', 'April', 'Sept. 3, 2016', 'the Army Corps of Engineers', '2018', 'Sacred Stone Camp', 'Iowa Utilities Board', 'Fort Laramie', 'Trump', 'first', 'Lake Oahe', 'the Dakota Access Pipeline', 'April 1, 2016', 'Barack Obama', 'Shailene', 'Kelcy Warren', 'October 2014', 'July 2014', 'USACE', 'https://www.nodaplarchive.com/. Indigenous', 'the North Dakota Public Service Commision on Jan. 25, 2016', 'more than 3300', 'Iowa', 'Oct. 10, 2016', 'North Dakota', 'the U.S. Army Corps of Engineers', 'National Guard', 'Twitter', 'Aug. 31, 2016', 'the Missouri River', 'Warren', '1,886', 'NoDAPL', 'Federal', 'The Dakota Access Pipeline', 'Dec. 4, 2016', '450,000 barrels', 'May 14, 2017', 'the Standing Rock Sioux Tribe', '2014']

From Table 13 we observe that 20.75% of the 1-*grams* in the reference summary are also present in the generated summary. A ROUGE score of 0.35 is considered to be among the best so; 0.2075 is good result.

Although the ROUGE score is good to use, it is still not the perfect method for evaluating a computer-generated summary. This model depends heavily upon external factors like the quality of the golden standard. A golden standard created from sources different than the ones used for creating an automatic summary might show huge disparities. This is because the entities covered in the golden standard are extracted by reading through a very limited number of external sources because of human limitations. On the other hand, an automatic summary is generated by passing through a very large corpus of articles about the same topic. This might result in a huge mismatch between the entities present in the golden standard and the automatic summary. Furthermore, ROUGE is also incapable of determining if the results are coherent or the flow of sentences is sensible. Hence, it can be concluded that a manual, subjective evaluation is a better parameter to evaluate an automatic summary.

9. Gold Standard Summary

9.1 Summary for Hurricane Irma

Below shows a manually created gold standard summary for Team 9 on Hurricane Irma event (Table 15). We initially studied a bit about Hurricane Irma and identified important categories of data. After identification of important categories we generated different questions for each category and found answers for those questions from resources like sample documents from team 9, Wikipedia pages, and news reports. We then combined these together to create the following golden summary.

Table 15. Gold standard summary for Hurricane Irma (team 9).

Hurricane Irma was a Category 5 Atlantic hurricane, the most powerful in history, with winds of 185 mph for 37 hours, longer than any storm recorded. Tropical force winds extended 185 miles from the center. Irma held 7 trillion watts of energy. Storm surges brought waves 20 feet higher than normal. It hit the Caribbean and the United States, causing damage of over $64 billion, making it the fifth most costly storm. Hurricane Irma took 129 lives and left hundreds of people injured. The storm began as a weak wave of low pressure off the west African coast on 27 August. It formed just west of the Cape Verde Islands on 30 August. Fueled by above average (in the mid-80s F) sea surface temperatures, Irma increased in intensity to a Category 3 by late August 31, with 115 mph sustained winds, and by September 5 to Category 5, continuing that for 3 days. At near peak strength as it approached the Leeward Islands, it made landfall along the northern coast of Barbuda, damaging 90% of the buildings, but with less severe damage in Antigua, nevertheless causing over $150 million losses on those islands, and devastating wildlife. Irma's pressure bottomed out at 914 Mb. Maintaining its intensity, it made landfalls September 6 on Saint Martin (where over 90 percent of the structures were damaged, coupled with ripping out trees, ruin of marinas, and over $4 billion in losses), Sint Maarten (with an estimated $1.5 billion losses), and Virgin Gorda in the British Virgin Islands (where it caused massive defoliation that was apparent even from space), also affecting Saint Barthelemy with flooding. It passed north of Puerto Rico on September 6 and Hispaniola on September 7. On September 8 it passed south of the Turks and Caicos Islands, and weakened to Category 4, but devastated parts of the southern Bahamas with 155 mph winds, and intensified to Category 5 with 160 mph winds before landfall in Cuba, after which it weakened to Category 2 on September 9. It strengthened to Category 4 on September 10, making landfall in Cudjoe Key, Florida, with 130 mph winds. It weakened to Category 3, making its final landfall in Marco Island, Florida, with 115 mph winds. And then it down to a Category 2 storm with sustained winds of 110mph by late September 10.

Irma caused a large amount of rainfall. The maximum recorded precipitation was 23.90 inches in Topes De Collantes, and 21.66 inches in St. Lucie County in the United States. The average rainfall in the affected areas was around 10-15 inches.

Power outages were widespread, in Anguilla, the Lesser Antilles, U.S. Virgin Islands, Puerto Rico (for over a million residents), the Bahamas, and USA (with more than 9 million outages). It left Puerto Rico without power. Saint Martin, It disconnected the public transportation and bridges, and damaged properties of more than 70 million people's. Barbuda was left with no water or communications after the storm. Coming to Florida, there were widespread power outages and flooding. It brought 21 storms in Florida and caused much more damage.

Before hurricane Irma hit, officials had a minimum of 3-4 days for warnings and evacuations. Most of the Caribbeans (i.e., Puerto Rico, Antigua and Barbuda, Guadeloupe, Turks and Caicos, Haiti, Bahamas, Dominican Republic, and Cuba) and Southern states of US (i.e., Florida, Georgia, and parts of South Carolina) were issued with warnings or evacuated. In the Caribbean islands, Puerto Rico declared a state of emergency on September 4. In the Dominican Republic, 112,000 people were evacuated from vulnerable areas. Another 7400 tourists were moved to Santo Domingo. In the Bahamas, 1,609 people were evacuated by air from the southern islands, including 365 from Bimini. The government began preparation a week ago by securing national sports facilities to use as shelters. About 1200 were housed at the Atlantis Paradise Island. In the United States, Florida was the first state to declare a state of emergency on September 4. The Florida Keys, Jupiter Inlet to Bonita Beach were the first places to issue watches and warnings on September 7. There were around 700 emergency shelters set up for 191,764 people. Georgia started issuing warnings from September 9. Governor Nathan Deal declared a state of emergency on September 8. Around 540,000 people living near the coastline and east of I-95 were mandatorily evacuated. North Carolina and South Carolina also declared a state of emergency on September 6 to mobilize resources.

After the hurricane, teams of experts from the Federal Emergency Management Agency and other federal agencies formed the Interagency Recovery Coordination group. The US Department of Children and Families provided $1 billion in food aid to Florida. Electric power for 95% of customers was restored in Florida by September 18. The International Rescue Committee and the Red Cross helped out in helping rebuild and distributing relief items.

9.2 Summary for NoDAPL

Table 16 is a manually created gold standard summary for our Team by Team 7 on Dakota Access Pipeline Protests.

Table 16. Gold standard summary for NoDAPL (team 7).

Introduction

The Dakota Access Pipeline, also known as DAPL, is a $3.8 Billion construction program that began in July 2014. It's a 1,172-mile-long (1,886 km) underground oil pipeline project that starts in the Bakken shale oil fields in northwest North Dakota, passes through South Dakota and Iowa, and ends in Illinois at the oil tank farm near Patoka. The ambition for this project is to transport 450,000 barrels of crude oil from Dakota to Illinois. The DAPL project is the result of an extensive process that involved review and approval by the U.S. Army Corps of Engineers and regulators in North Dakota, South Dakota, Iowa, and Illinois. #NoDAPL is a Twitter hashtag from 2014 referring to the Dakota Access Pipeline protests. Some of the discussion is archived at https://www.nodaplarchive.com/. Indigenous activities led the movement, which continued into 2018 since activists were still doing time for their actions to stop the pipeline.

Detail on Affected Area

The affected area included North Dakota in the early stage of planning. The pipeline passed through disputed Sioux land, which was promised to the tribe in the 1851 Fort Laramie treaty but was later taken away. The project also was designed to cross the Missouri river, which is the main source of drinking water for the Sioux people. Large-scale protests at Lake Oahe were held against possible water pollution and destruction of sacred tribal sites. According to a report from The Pipeline and Hazardous Materials Safety Administration (PHMSA), since 2010 there are more than 3300 incidents reported about oil and gas pipeline leaks and ruptures. In fact, even a tiny spill could damage and destroy the local environment and water supply. Furthermore, based on original Dakota Access Pipeline design.

Timeline

The project was announced and approved on July 2014 by Energy Transfer Partners.

There was an information session held in October 2014 with South Dakota and Illinois landowners.

The project was submitted to the IUB (Iowa Utilities Board). Dakota Access announced that they were approved by the North Dakota Public Service Commision on Jan. 25, 2016.

Attorneys for the Standing Rock Sioux Tribe filed a lawsuit on July 27, 2016 and soon after the lawsuit, many demonstrations and protests begin.

Aug. 31, 2016 The United Nations Permanent Forum on Indigenous Issues offered support to the tribe.

Sep. 3, 2016, the protests become more aggressive. The security hired by Dakota Access started to use pepper spray.

Sep. 9, 2016 a Federal judge denied the tribe’s request for injunction.

Sep. 26 2016, bow to society pressure, President Obama weighed in and kept in touch with USACE, trying to illustrate the possibility of rerouting the pipeline to avoid paa through North Dakota.

Oct. 9, 2016 the federal appeals court denied the appeal of the tribe.

Oct. 10, 2016, actress Shailene was arrested during the protest.

Oct. 26, 2016, authorities started to clear the protesters and removed a roadblock set by the protesters that was blocking a state highway.

Nov. 1, 2016 President Obama announced that he will be monitoring the situation closely.

Dec. 4, 2016, USACE said that they will not grant easement for drilling the pipeline under Lake Oahe.

Jan. 14, 2017, President Trump signed an executive order to advance the construction of the pipeline. The construction was completed by April 2017.

Opponent Response

The opponents against the Dakota Access Pipeline projects include local aboriginal Sioux, environment protectors, and veterans. At the beginning of the protest, the scale of opponents was quite small. With the development proceeding, many celebrities and public figures, like actress Shailene Woodley, traveled to North Dakota and supported the protest in public. Much more people started focusing and following this event from social media such as Facebook and Twitter. To provide support for the opponents, on April 1, 2016, Standing Rock's Historic Preservation Officer founded and built a Sacred Stone Camp as a spiritual resistance to the Dakota Access pipeline. Protesters set up teepees and tent camps and threatened to block the highway to slow construction progress. Meanwhile, protesters also sued the Army Corps of Engineers, claiming that they violated the National Historic Preservation Act (NHPA) and the National Environmental Policy Act (NEPA). They requested the government reconsider the cultural significance of federally-permitted sites and implications for the waterways.

Federal Government/President Response

The federal government has asked the company to "voluntarily suspend" construction near the area until further research is done on the area. Kelcy Warren, chairman and CEO of Energy Transfer partners, responded to a request from the federal government, calling concerns about pipeline impacts on water supplies "unfounded". State historic preservation offices couldn't find sacred items on the route. Warren wrote that the company would meet officials in Washington "to understand their positions and reaffirm our commitment to getting Dakota access to the pipeline into operation." In December 2016, under President Barack Obama’s administration, the Corps of Engineers refused to provide an easement to build a pipeline under the Missouri River. On January 24, 2017, US President Donald Trump signed an executive order that overturned Obama's legislation and advanced the pipeline under "terms and conditions that remain to be negotiated," speeding up environmental reviews of what Trump has called an "extremely burdensome, lengthy, and horrible approval process." The number of protesters decreased after Trump approved the construction of the pipeline. National Guard and law enforcement officers evicted the remaining people on February 23, 2017. The pipeline was completed in April and the first oil was delivered on May 14, 2017.

10. Developer Manual

This project's code repository is at https://github.com/Xiaoddd/cs4984cs5984f18_team8

10.1 About the Files

- Use `pip install -r requirements.txt` to install dependencies.
- `classification.py` is to classify documents into relevant and irrelevant
- `preproc_lda.py` is to preprocess the corpus and conduct LDA analysis, then visualize the results
- `preproc_lda2vec.py` is to preprocess the corpus and conduct lda2vec analysis, then visualize the results
- `make_datafiles_py3.py` is a Python 3 script to process data for pointer generator; refer to https://github.com/chmille3/process_data_for_pointer_summrizer
- `pointer-generator/run_summarization.py`: use pre-trained model and vocabulary to run decode on our dataset.
- `Python_Evaluation/eval.py`: uses pyRouge to run evaluations and compare the generated summary and the referenced summary.

10.2 Pre-trained Model

Please download and unzip the pre-trained model for TensorFlow 1.2.1 from the source git at: https://github.com/abisee/pointer-generator. Create a "log" folder in the root, and then extract the content into it. Note: only keep "eval" and "train" folders, remove the decoding result folder "decode_test_400maxenc_4beam_35mindec_120maxdec_ckpt-238410"

10.3 Vocabulary

Please download and unzip finished_files.zip, and then extract the vocab file to "finished_files" folder

10.4 Run beam search decoding

```
python pointer-generator/run_summarization.py --mode = decode --data_path = finished_files/chunked/test_* --vocab_path = finished_files/vocab --log_root = log --exp_name = pretrained_model
```

or

```
python pointer-generator/run_summarization.py --mode = decode --data_path = finished_files/chunked/test_* --vocab_path = finished_files/vocab --log_root = log --exp_name = pretrained_model --max_enc_steps = 400 --max_dec_steps = 120 --coverage = 1 --single_pass = 1
```

10.5 Run Attention Visualizer (Attn-vis)

To run the visualizer, navigate to your root folder of the project (where "attn_vis" and "log" folders are located), then

- in Python 2: `run python -m SimpleHTTPServer`
- in Python 3: `run python3 -m http.server`

and then open http://localhost:8000/attn_vis/ to view. It random displays an article in each page refresh. Note: The `SimpleHTTPServer` module has been merged into `http.server` in Python 3.0.

10.6 Requested orders to run

The developers are suggested to execute the Python scripts according to the following order:

- Run `python feature_extraction.py` to extract features from the raw corpus for each document.
- Run `python classification.py` to train the regularized logistic regression classifier and then classify the documents into relevant (1) and irrelevant (0).
- Run `python preproc_lda.py` to train the LDA topic model and store the intermediate results.
- Run `python topic_Clustering.py` to group the documents into different topics by using the trained LDA model.
- Run `python LDA_sents_ranking.py` to rank the sentences within each topic and store the top-N sentences.
- Run `python summary_initial.py` to rank the sentences within each topic and store the top-N sentences.
- Run `python PGN_reorganizing.py` to reorganize the sentences for each document within each topic generated by pointer-generator.
- Run `python sents_reranking.py` to rerank the extractive and abstractive sentences from the last two steps and generate paragraphs for each topic.
- Copy referenced summary to `Python_Evaluation/golden/test.1.txt` and generated summary to `Python_Evaluation/predict/test.txt`
- Run `Python_Evaluation/eval.py` to evaluate and compare the generated summary and the referenced summary.

11. User Manual

Code repository for this project is at https://github.com/Xiaoddd/cs4984cs5984f18_team8 .

11.1 Requirements

Most of the code is written in Python, and thus requires a Python environment to execute. The following are a few of the libraries used:

- sklearn
- numpy
- matplotlib
- spacy==1.9.0 # python -m spacy download en
- nltk
- json
- Gensim
- pyLDAvis
- chainer
- pandas
- pickle
- bs4

There is also code written in PySpark for conversion of WARC and CDX files to JSON formatted files. For the deep learning model we have used a specific version of TensorFlow (1.2.1).

11.2 Steps to follow

The users are suggested to execute the Python scripts according to the following order:

- Run `python feature_extraction.py` to extract features from the raw corpus for each document.
- Run `python classification.py` to train the regularized logistic regression classifier and then classify the documents into relevant (1) and irrelevant (0).
- Run `python preproc_lda.py` to train the LDA topic model and store the intermediate results.
- Run `python topic_Clustering.py` to group the documents into different topics by using the trained LDA model.
- Run `python LDA_sents_ranking.py` to rank the sentences within each topic and store the top-N sentences.
- Run `python summary_initial.py` to rank the sentences within each topic and store the top-N sentences.
- Run `python PGN_reorganizing.py` to reorganize the sentences for each document within each topic generated by pointer-generator.
- Run `python sents_reranking.py` to rerank the extractive and abstractive sentences from the last two steps and generate paragraphs for each topic.

12. Conclusion and Future Work

Large-scale automatic summarization is faced with multiple challenges, such as large proportion of noise information, high redundancy, and multiple latent topics. To this end, extractive and abstractive automatic summarization methods have been proposed. However, most of the existing methods deal with a single document and generate short paragraphs to automatically summarize the document. Therefore, this project proposes a hybrid method-based automatic summarization to summarize a big online textual data into different topics with limited human efforts. In this project, a NoDAPL dataset with over 10,000 documents is investigated to test the performance of the proposed hybrid automatic summarization method. To evaluate the generated summary, we designed a summarization guideline according to our thorough understanding of NoDAPL events/topics. This summarization guideline was used to evaluate the generated summary by adopting a task-based evaluation method. Results show that 91.3% sentences are within the scope of the guideline, and 69.6% of the outlined questions can be answered by reading the generated summary. Furthermore, the ROUGE scores show that we have unigrams and bigrams overlapping around 0.2 and 0.06 for type 1 evaluation and 0.5 and 0.4 for type 2, respectively. Our entity coverage was a total of 2.6% and ROUGE L and ROUGE SU4 scores were 0.148 and 0.065. Therefore, the proposed hybrid automatic summarization method has satisfactory performance in content relevancy and coverage with limited human effort, which is only needed in creating class labels for relevancy classification.

In the future, more textual data for interesting topics will be tested by using the proposed hybrid text summarization approach. Topic modeling-supervised classification approach can be investigated to minimize human efforts in automatic summarization. Besides, deep learning-based recommender system can be investigated for better sentence re-ranking.

Acknowledgment

This project is funded by NSF IIS-1619028. We would like to thank the following individuals for all of the assistance in this project. They provided guidance and assistance to allow us to complete this project.

Edward A. Fox (fox@vt.edu)

Liuqing Li (liuqing@vt.edu)

Classmates at CS4984 / 5984 Big Data Text Summarization 2018 Fall

We also thank Anuj Arora, Jixiang Fan, Yi Han, Shuai Liu, Chreston Miller for generating gold standard summary for NoDAPL event.

References

Alexander Crosson. (2016). Summarize Documents using Tf-Idf. Retrieved from https://medium.com/@acrosson/summarize-documents-using-tf-idf-bdee8f60b71

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). Text Summarization Techniques: A Brief Survey. *ArXiv:1707.02268 [Cs]*. Retrieved from http://arxiv.org/abs/1707.02268

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research*, *3* (Jan), 993–1022.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, *3*(Mar), 1289–1305.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.

Jones, K. S., & Galliers, J. R. (1995). *Evaluating natural language processing systems: An analysis and review* (Vol. 1083). Springer Science & Business Media.

Khatri, C., Singh, G., & Parikh, N. (2018). Abstractive and Extractive Text Summarization using Document Context Vector and Recurrent Neural Networks. *ArXiv:1807.08000 [Cs]*. Retrieved from http://arxiv.org/abs/1807.08000

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, *2*(2), 159–165.

Mani, I., & Maybury, M. T. (2001). Automatic summarization. John Benjamins Publishing Co.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *ArXiv Preprint ArXiv:1301.3781*.

Moody, C. E. (2016). Mixing Dirichlet topic models and word embeddings to make lda2vec. *ArXiv Preprint ArXiv:1605.02019*.

Nallapati, R., Zhou, B., santos, C. N. dos, Gulcehre, C., & Xiang, B. (2016). Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. *ArXiv:1602.06023 [Cs]*. Retrieved from http://arxiv.org/abs/1602.06023

Narayan, S., Cohen, S. B., & Lapata, M. (2018). Ranking Sentences for Extractive Summarization with Reinforcement Learning. *ArXiv:1802.08636 [Cs]*. Retrieved from http://arxiv.org/abs/1802.08636

Nenkova, A., & McKeown, K. (2011). Automatic Summarization. *Foundations and Trends in Information Retrieval*, *5*(2), 103–233. https://doi.org/10.1561/1500000015

Paulus, R., Xiong, C., & Socher, R. (2017). A Deep Reinforced Model for Abstractive Summarization. *ArXiv:1705.04304 [Cs]*. Retrieved from http://arxiv.org/abs/1705.04304

Saravia, E. (2018, August 23). Deep Learning for NLP: An Overview of Recent Trends. Retrieved November 27, 2018, from
https://medium.com/dair-ai/deep-learning-for-nlp-an-overview-of-recent-trends-d0d8f40a776d

Saziyabegum, S., & Sajja, P. (2016). Literature Review on Extractive Text Summarization Approaches. *International Journal of Computer Applications*, *156*(12), 28–36. https://doi.org/10.5120/ijca2016912574

Scott, S., & Matwin, S. (1999). Feature engineering for text classification. In *ICML* (Vol. 99, pp.

379–388). Citeseer.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR)*, *34*(1), 1–47.

See, A., Liu, P. J., & Manning, C. D. (2017). Get To The Point: Summarization with Pointer-Generator Networks. *ArXiv:1704.04368 [Cs]*. Retrieved from http://arxiv.org/abs/1704.04368

Steinberger, J., & Ježek, K. (2012). Evaluation measures for text summarization. *Computing and Informatics*, *28*(2), 251–275.

Wikipedia: Automatic summarization. Retrieved from https://en.wikipedia.org/wiki/Automatic_summarization. Accessed November 26 2018.

Wikipedia: Dakota Access Pipeline protests. Retrieved from https://en.wikipedia.org/wiki/Dakota_Access_Pipeline_protests. Accessed November 26 2018.

Yao, J., Wan, X., & Xiao, J. (2017). Recent advances in document summarization. *Knowledge and Information Systems*, *53*(2), 297–336.

Yohei Seki. (2001). Sentence Extraction by tf/idf and Position Weighting from Newspaper Articles. Retrieved from http://research.nii.ac.jp/ntcir/workshop/OnlineProceedings3/NTCIR3-TSC-SekiY.pdf