Team 12: Final Report

Automatic Summarization of News Articles about Hurricane Florence

CS 4984/5984
Instructor: Dr. Edward Fox

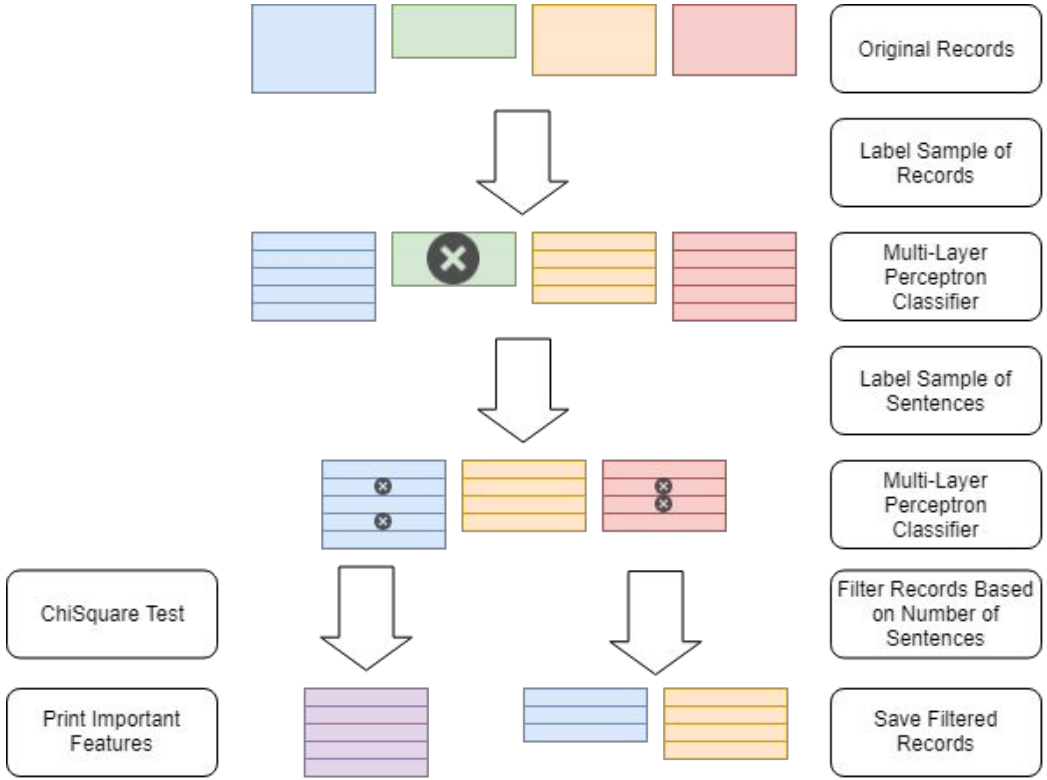
Frank Wanye
Samit Ganguli
Joy Zhang
Matt Tuckman

Department of Computer Science
Virginia Tech
Blacksburg, VA 24061
December 4, 2018

Introduction

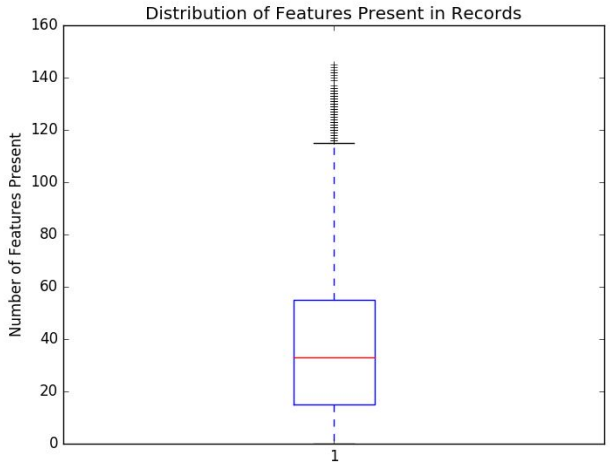
- Hurricane Florence dataset - 10948 Records
- Followed 10-step approach
- Included steps to familiarize ourselves with NLP
  - WARC to JSON conversion
  - Finding frequent words
  - Finding important words, collocations, named entities
  - LDA
  - Part-of-speech tagging
- Tools used
  - PySpark, TensorFlow, Python, Hadoop cluster, Anaconda, Gensim, NLTK[2], Solr, Screen

Identifying Discriminating Features



Sample Labeling

- Based on number of features present
- Use hard rules to label more irrelevant records



Multi-layer Perceptron Classifier

- Labeled sample split into train and test set with 7:3 ratio
- Train set imbalanced (more irrelevant records than relevant ones)
  - Undersampling
  - Add remainder of records to test set
- Final results:
  - 42,017 relevant sentences
  - 2,465 relevant records

Template Summary Simple Questions

- Questions based on Hurricane Metadata page
- First approach was to use exclusively regular expressions
  - 'Hurricane ([A-Z][a-z]+)'
  - Frequency count / numerical analysis
- Pros
  - Very accurate results
- Cons
  - Difficulty in regular expression generation
  - Too few results

- Looked to spaCy to help extract specific values.

Template Summary Remaining Holes

- Basic sentence filtering to a certain subject
  - Filter to all sentences with the word ‘landfall’
- spaCy tag all entities in the sentences
  - Now look at ‘DATE’, ‘QUANTITY’, and ‘GPE’ tags separately
- Filter to relevant entities
  - Only look at ‘DATE’ entities
- Numerical analysis, regex matching, and unit conversions
  - Find the most common year, then the most common month in that year, then the most common day in that month
- Repeat for other entities / other information
  - ‘GPE’ for what states the storm hit
  - ‘QUANTITY’ for measurements at time of landfall’
  - Regex for category of storm at landfall

Template Summary Results

Hurricane Florence was a huge storm occurring during the month of September 2018 which peaked as a Category 4 hurricane. The storm was first detected around September 9, 2018 when it was known as Tropical Storm Florence and the storm grew in size and ferocity until it become known as a hurricane. As the storm progressed, wind speeds were seen to be ranging from as low as 79 miles per hour to an astounding 139 miles per hour at the peak of the storm. Most areas that were affected by the storm had winds of 109 miles per hour on average when the storm hit them. For rainfall, some areas experienced as much as 32 inches of rain, however other areas on the fringes of the storm only got 5 inches. On average however, most areas affected by the storm had 18 inches of rainfall. During the storm's lifetime, it averaged a diameter of 400 miles across and at its peak reached over 470 miles.

In preparation for the storm's approach, evacuation was determined necessary in South Carolina, Virginia, and North Carolina. About 1.5 million people were evacuated in total over this area and have moved out of the storms primary path. Hurricane Florence made landfall as a Category 1 hurricane on September 12, 2018. The storm landed in both North Carolina and South Carolina. Throughout it's vicious path, Hurricane Florence killed 55 people and caused billions of dollars in damages.

Paragraphs				Sentences	
ROUGE-SU4	ROUGE-L	ROUGE-2	ROUGE-1	ROUGE-2	ROUGE-1
0.08537	0.2	0.03448	0.23333	0.42857	0.66667

Extractive Summary

- **PyTeaser**: Python port of TextTeaser
  - Needs URL to scan webpage
  - Scores sentence with regards to title, sentence length, position, and keyword frequency
  - Preprocessing/filtering issues
- **Summa**: Implementation of TextRank[1,6]
  - Does not need URLs
  - Faster than PyTeaser
- **Lexical Chains**: Attempted method following Dr. Hollingsworth of University of Georgia's patent[3–5]
  - Used Skimcast to test summarize randomly selected records that were concatenated together and generated a cohesive summary
  - Generated extractive summary seemed more personal

Extractive Summary - Implementation

- Clean the dataset
- From the JSONs iteratively extract the URLs and get the summary from text in the corresponding webpage
- Append the summaries in a list and then concatenate 20 such summaries to a single string that has 100 sentences
- Repeat until final summary is generated (~1 page/20 sentences)
- Manually clean summary

Extractive Summary - Results

Excerpts from generated summary:

- Hurricane Florence becomes first major storm of 2018 Atlantic season. The sixth named storm of the 2018 Atlantic hurricane season gathered strength Wednesday to become the first major hurricane of the year.
- A million told to flee as Hurricane Florence stalks US East Coast.
- Trump: "We are totally prepared" for Hurricane Florence. President Trump just had a meeting with the Department of Homeland Security and Federal Emergency Management Agency officials as Hurricane Florence continues to barrel toward the East Coast.

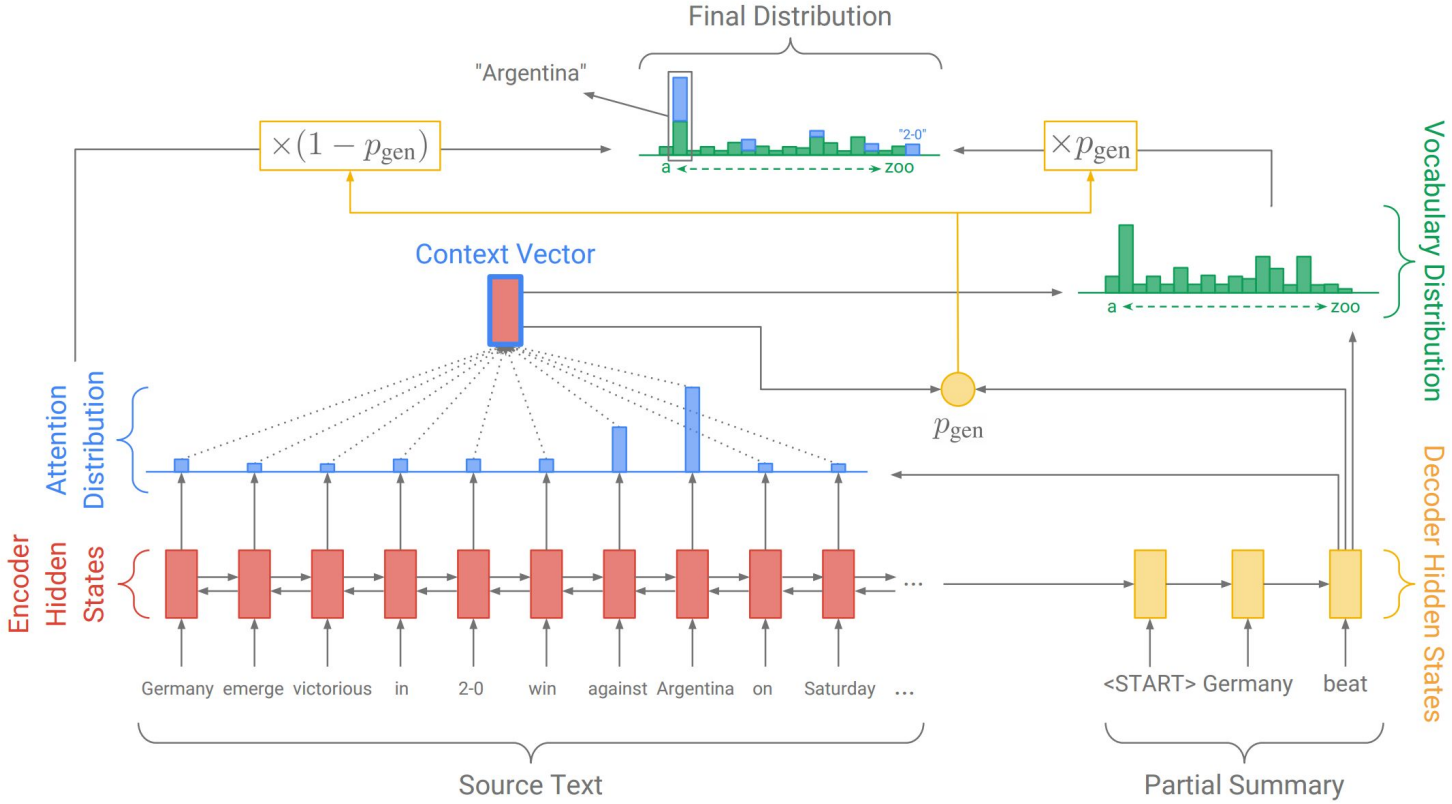
Paragraphs				**Sentences**	
ROUGE-SU4	ROUGE-L	ROUGE-2	ROUGE-1	ROUGE-2	ROUGE-1
0.06707	0.23333	0.0	0.26667	0.50	0.80

Abstractive Summary: Clustering

- Can't combine articles together - articles that are too long are harder to summarize
- Used clustering to identify topics
  - Different methods:
    - LDA
    - K-Means
    - Bisecting K-Means (didn't work)
  - Different inputs:
    - Word counts
    - TF-IDF word scores
  - Different number of topics
    - 5 - 10 - 15 - 20 - 25
- Hard to compare b/n LDA and K-Means, since the measures were different
  - Resorted to comparing how 'balanced' the clusters were
- Picked LDA with TF-IDF word scores and 10 topics

Abstractive Summarization - Pointer-Generator

- Used a pre-trained Pointer-Generator model[7]
- Selected 'best' article from each cluster
- Combined resulting summaries
- Manually edited the resulting summary
  - Fixed punctuation
  - Fixed capitalization
  - Deleted duplicate sentences
  - Deleted unrelated sentences



The Pointer-Generator Model[7]

Abstractive Summary - Results

Excerpts from generated summary:

- His wife Angie Wood said their home was also flooded by the nearby little river after Matthew, but not nearly to the same levels.
- Florence will likely hit land as a category 4 hurricane late thursday evening before weakening again.
- The storm was moving west at 13 mph (20 kmh) and expected to accelerate over the next 36 hours.
- Flood waters from the cresting rivers inundated the area after the passing of Hurricane Florence.

Paragraphs				**Sentences**	
ROUGE-SU4	ROUGE-L	ROUGE-2	ROUGE-1	ROUGE-2	ROUGE-1
0.08537	0.26667	0.06897	0.26667	0.25000	0.50000

Lessons Learned / Challenges

- Extractive and abstractive summaries contained too much irrelevant information
  - Need better ways to clean our data
- Extractive and abstractive summaries had poor flow/cohesion
  - Could sort articles by date
- Summaries contained duplicate information
  - Find a way to remove duplicates
- Hard to analyze clustering results
  - Need a common metric between LDA and K-Means
- Lots of time spent debugging PySpark errors
  - Should have gone through some PySpark tutorials

Acknowledgements & Sponsor

- Team 7
  - For their help with the pointer-generator network
- Team 9
  - For generating a Golden Standard Summary
- Michael Horning
  - For his help with the Golden Standard Summary
- James Xie, Lei Xu
  - For making their sequence-to-sequence model available on GitHub
- Abigail See
  - For making the Pointer-Generator model available on GitHub
- Xiao Hu
  - For adapting TextTeaser into a Python library
- Jolo Balbin
  - For releasing the original TextTeaser API
- Liuqing Li
  - The GTA for the course
- Dr. Edward Fox
  - The instructor for CS 4984/5984
- The SummaNLP organization
  - For making the Summa library available on GitHub
- Funding
  - NSF Grant No. IIS-1619028.

References

[1] Federico Barrios, Federico López, Luis Argerich, and Rosita Wachenchauzer. 2015. Variations of the Similarity Function of TextRank for Automated Summarization. In *Proceedings of Argentine Symposium on Artificial Intelligence*, 65–72. Retrieved December 3, 2018 from http://sedici.unlp.edu.ar/handle/10915/52082

[2] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media. Retrieved October 7, 2018 from http://www.nltk.org/book_1ed/

[3] W. A. Hollingsworth. 2008. Using lexical chains to characterise scientific text. University of Cambridge. Retrieved November 28, 2018 from https://ethos.bl.uk/OrderDetails.do;jsessionid=0D6EA59350E32781968819AF3DDB5D5F?uin=uk.bl.ethos.604174

[4] William Hollingsworth. Accessible Learning - With Artificial Intelligence. *Accessible Learning*. Retrieved November 28, 2018 from http://accessible.blog/

[5] William Hollingsworth. 2007. US8676567B2 - Automatic text skimming using lexical chains - Google Patents. Retrieved November 28, 2018 from https://patents.google.com/patent/US8676567

[6] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of EMNLP*, 404–411. Retrieved December 3, 2018 from http://aclweb.org/anthology/W04-3252

[7] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Association for Computational Linguistics*, 1073–1083. DOI:https://doi.org/10.18653/v1/P17-1099