April 26, 2018

Extractive Summarization

What are you saying, NEWS?
A case study in tl;dr systems

Group 4
Ayushi Aggarwal, Elijah Rippeth, Seraphina Goldfarb-Tarrant
LING573, Spring 2018

Walkthrough

1. Looking at Data
2. Related Reading
3. Our Approach
    a. Content Selection
    b. Content Ordering
    c. System architecture
    d. Stack
4. Evaluation
    a. Quantitative
    b. Qualitative
    c. Experiments
    d. D3 Goals
5. GroupWork

Looking at Data

NewsWire Data

- Topic-categorized dataset
- Multi-document set per topic
- Document ordering irrelevant
- Inverted pyramid-structured content per document

General Idea:

- All documents in docSet A as one document
- Find most-representative documents -> scoring documents

Related Reading

- Two surveys of Text Summarization
  - **Allahyari (2017), Nenkova (2012)**
- Focused reading:
  - **LexRank:** Erkan & Radev (2004)
  - Its predecessor **PageRank**: Brin & Page (1998)

Our Approach

Content Selection

Quick-and-dirty baselines:

- Random-choose
- First-things-first

Final baseline: **LexRank**[1]

[1] Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, *22*, 457-479.

LexRank

What?

- Sentence as graph nodes
- Sentence as BOW
- Tf-idf cosine similarity between sentences
- Centrality score for each sentence
    - Number of neighbors - how central is this node?
    - Weights of neighboring nodes

Why we like it?

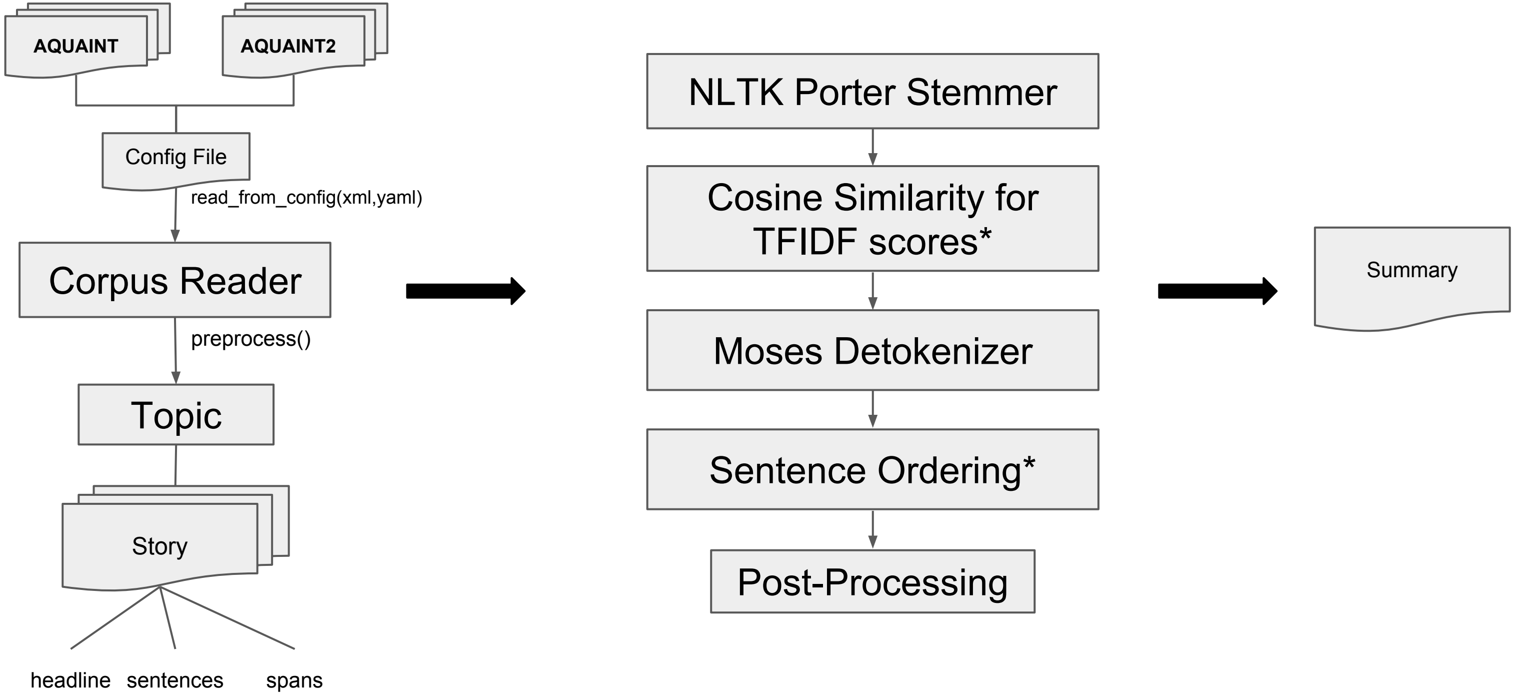
- Builds upon the general idea - most salient sentence is central to topic -> include in summary
- Tf-idf automatically downweights stopwords(language and domain specific)
- Language independence
- Ease of availability
- Configurable

Content Ordering

- First-sentences-first and Random-choose
    - First sentences in combined doc within word-limit

- LexRank
    - Non-chronological as date information sparse
    - Top sentences by lexRank score that are just within the word limit

Baseline System Architecture



AQUAINT

AQUAINT2

Config File

read_from_config(xml,yaml)

Corpus Reader

preprocess()

Topic

Story

headline

sentences

spans

NLTK Porter Stemmer

Cosine Similarity for TFIDF scores*

Moses Detokenizer

Sentence Ordering*

Post-Processing

Summary

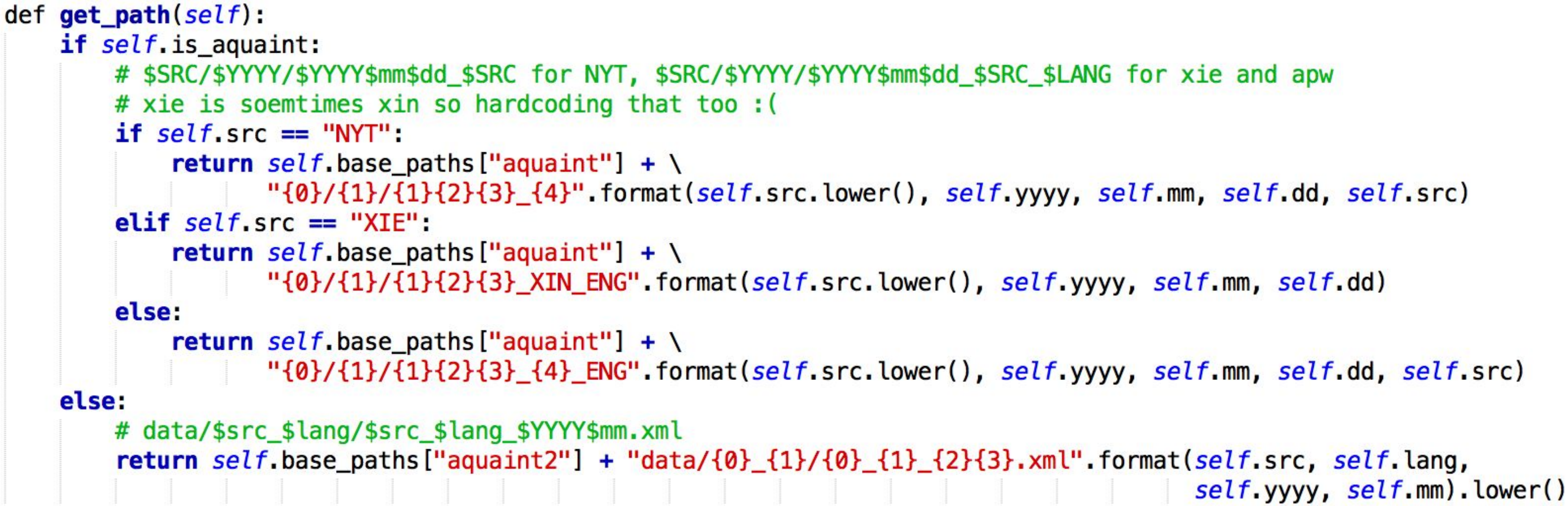
*Drawn heavily from Sumy's LexRank implementation

Stack

- XML parsing
  - bs4
  - lxml
- Linguistics
  - nltk
  - sumy
    - forked to make it more extensible
- Number crunching
  - numpy
- Config
  - PyYAML

Config

The unglamorous side of data science



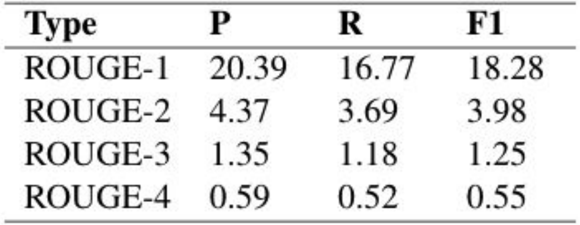
```
def get_path(self):
    if self.is_aquaint:
        # $SRC/$YYYY/$YYYY$mm$dd_$SRC for NYT, $SRC/$YYYY/$YYYY$mm$dd_$SRC_$LANG for xie and apw
        # xie is soemtimes xin so hardcoding that too :(
        if self.src == "NYT":
            return self.base_paths["aquaint"] + \
                   "{0}/{1}/{1}{2}{3}_{4}".format(self.src.lower(), self.yyyy, self.mm, self.dd, self.src)
        elif self.src == "XIE":
            return self.base_paths["aquaint"] + \
                   "{0}/{1}/{1}{2}{3}_XIN_ENG".format(self.src.lower(), self.yyyy, self.mm, self.dd)
        else:
            return self.base_paths["aquaint"] + \
                   "{0}/{1}/{1}{2}{3}_{4}_ENG".format(self.src.lower(), self.yyyy, self.mm, self.dd, self.src)
    else:
        # data/$src_$lang/$src_$lang_$YYYY$mm.xml
        return self.base_paths["aquaint2"] + "data/{0}_{1}/{0}_{1}_{2}{3}.xml".format(self.src, self.lang,
                                                                                      self.yyyy, self.mm).lower()
```

Evaluation Results

First...wins?

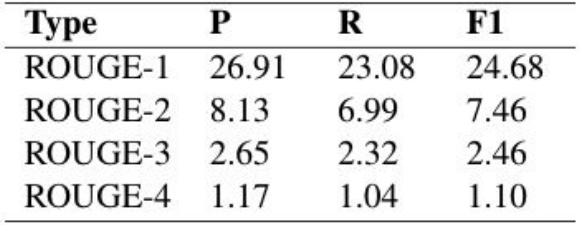
Thoughts:

- Higher recall numbers may have been driven by redundancy, which qualitatively we should remove
- Some of the precious 100 words was wasted with meaningless ‘tag’ words (location, etc) that are integrated within sentences



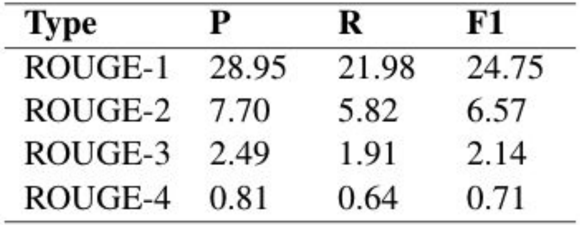
Type	P	R	F1
ROUGE-1	20.39	16.77	18.28
ROUGE-2	4.37	3.69	3.98
ROUGE-3	1.35	1.18	1.25
ROUGE-4	0.59	0.52	0.55

Table 1: Random



Type	P	R	F1
ROUGE-1	26.91	23.08	24.68
ROUGE-2	8.13	6.99	7.46
ROUGE-3	2.65	2.32	2.46
ROUGE-4	1.17	1.04	1.10

Table 2: First

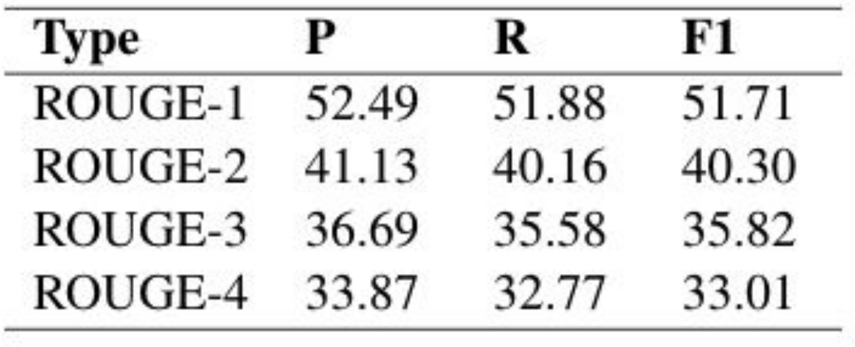


Type	P	R	F1
ROUGE-1	28.95	21.98	24.75
ROUGE-2	7.70	5.82	6.57
ROUGE-3	2.49	1.91	2.14
ROUGE-4	0.81	0.64	0.71

Table 3: LexRank

First vs. LexRank

- The most connected sentence often is the first sentence
- But LexRank is qualitatively better...



Type	P	R	F1
ROUGE-1	52.49	51.88	51.71
ROUGE-2	41.13	40.16	40.30
ROUGE-3	36.69	35.58	35.82
ROUGE-4	33.87	32.77	33.01

Table 4: Lexrank vs. First

Results Analysis: Redundancy

First-Sentences

ARVADA, Colo.

LITTLETON, Colo.

LITTLETON, Colo.

LITTLETON, Colo.

LITTLETON, Colo.

LITTLETON, Colo.

LITTLETON, Colo….

LexRank

littleton, colo. (ap) -- the day that columbine high school students are to return to class has been delayed because so many have been attending funerals for students killed in the april 20 massacre, an administrator said tuesday.

columbine.

littleton, colo. (ap) -- students returned to classes thursday at chatfield high school, but the bloodbath at rival columbine high haunted the halls.

Qualitative Analysis - Successes

- Intrinsically avoids detailed opening sentences by valuing connectedness
  - *Ex: When Emily Martin was hospitalized for emergency gallbladder surgery last summer, her doctors found that she had also had acid reflux, causing erosion of her esophagus.*
- Best-Rank ordering is (for now) good enough
  - ROUGE is order independent beyond 4-words, but ordering changes didn’t help qualitatively either.
- Redundant, but less so than baseline

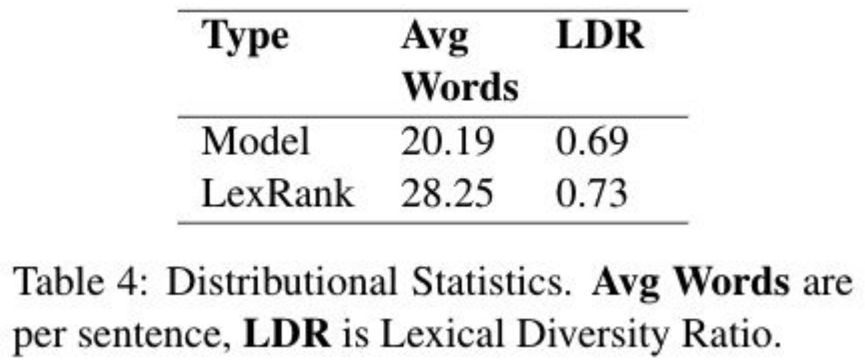
Qualitative Analysis - Failures

- Sentences as independent BOW:
  - No discourse coherence
  - No anaphora resolution
- Bias towards long sentences and rare words
  - From cosine similarity and tf-idf, respectively
  - Notably different from the bias shown by the model summarisers
- Sometimes of X important events (which all the models cover) it goes into detail about only one of those events
  - Can be a source of incoherence - missing causal links
  - Could be a results of different documents being of different length or verbosity, and thus one document being artificially overweighted
  - Could fix with some penalty for Event/Concept recall
- Some issues with sentence segmentation:
  - Punkt, may be how it was trained

Experiments - I

- ROUGE sensitivity
  - Experimented with how sensitive it is and tried running gold standard model files through tokenization and detokenization, and controlling for case.
  - Both experiments showed negligible difference (no more than randomness)

- Statistical Significance
  - Mann-Whitney U Test
  - model (seven humans) vs. LexRank
  - p=0.01
  - passes *easily*

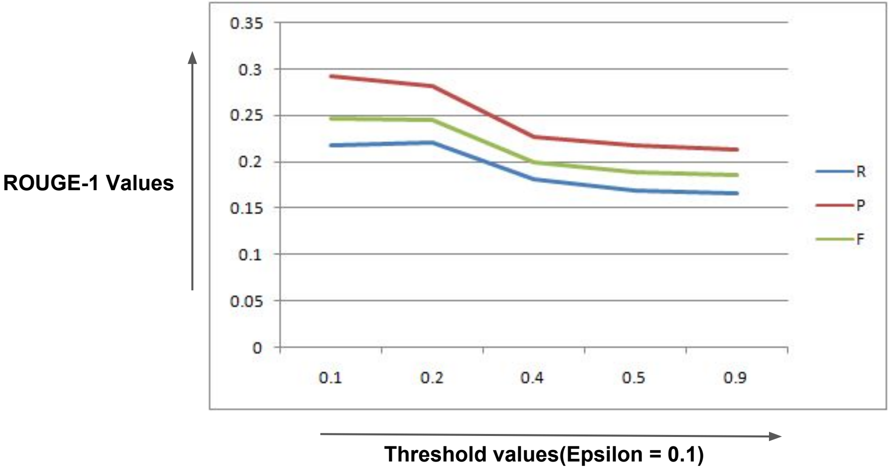


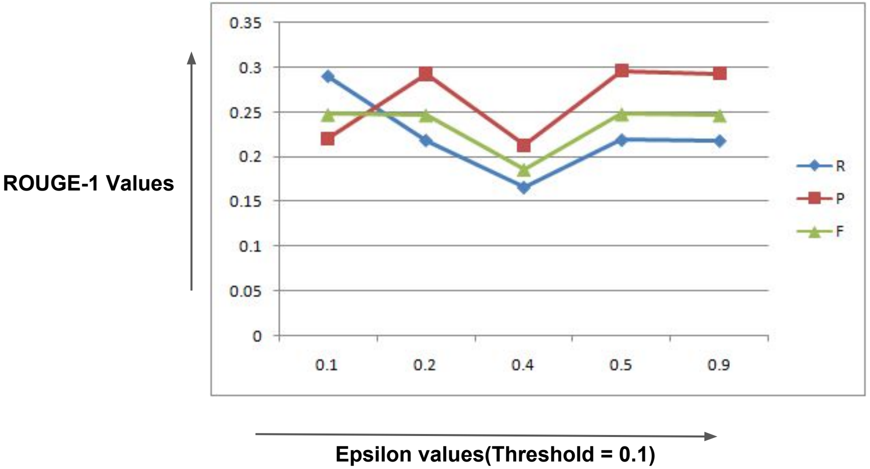
Type	Avg Words	LDR
Model	20.19	0.69
LexRank	28.25	0.73

Table 4: Distributional Statistics. **Avg Words** are per sentence, **LDR** is Lexical Diversity Ratio.

Experiments - II: LexRank Hyperparameter Tuning

- **Threshold**
  - For edge weight, controls graph connectedness
  - Best ROUGE values for 0.1
- **Epsilon** as a convergence measure to stop iteration for Power Method





D3 Goals

- Penalise redundancy (both large sentence fragment overlap and conceptual redundancy) in selected sentences or remove in post-processing
  - Artificially boosts summarization scores without adding new saliency
  - Could substitute tf-idf vectors for sentence or sub-sentence embeddings
- Detokenization with Moses NLTK module - only a heuristic and *not* language agnostic
- Take sentence word ordering into account - Discourse Coherence
- Experiment with training our own Punkt tokenization
- Anaphora resolution
- Incorporate Concept/Event Coverage
  - Topic title into consideration

Communication & Collaboration Methods

- Tools: Slack, Github, Google docs
- Tried pair programming
  - Good for quick development
  - pair_programming.txt(on Github)
- Issues:
  - Insufficiently clear ownership of tasks
- Work on - a more Agile development/work style:
  - Check for possibly redundant functionality before task-allocation and development
  - Better division of total work into clearly scoped small tasks
  - Code reviews, for catching bugs earlier, and so we all understand more aspects of each others work