D3 Summary

Sentence Selection + Ordering Solution

Brandon Gahler
Mike Roylance
Thomas marsh

Architecture: Technologies

Python 2.7.9 for all coding tasks

NLTK for tokenization, chunking and sentence segmentation.

pyrouge for evaluation

textrazor for entity extraction

attensity for entity and semantic information extraction

svmlight for training our ranking classifier

Architecture: Implementation

Reader - Extracts data from topic-focused document clusters

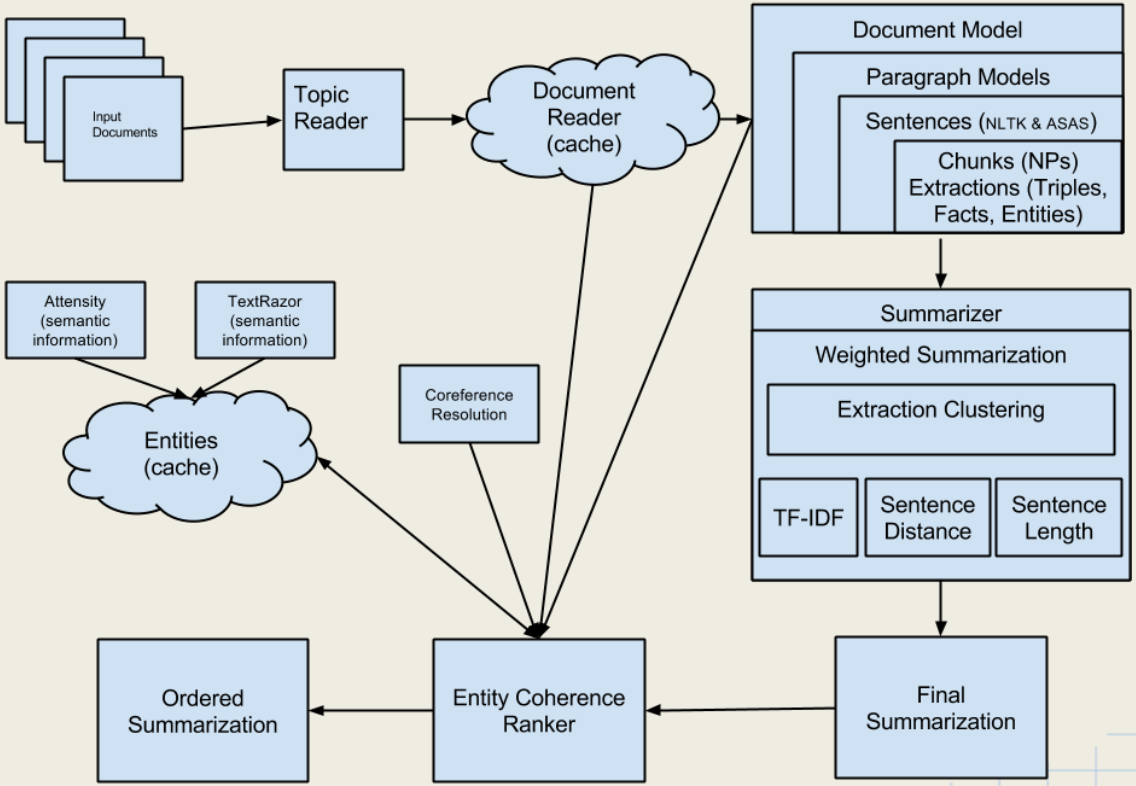
Document Model - Sentence Segmentation, Tokenization, NP Chunking

Summarizer - Creates summaries (using various techniques)

Evaluator - Uses pyrouge to call ROUGE-1.5.5.pl

Reorderer - Uses entity-coherence ranking to reorder

Architecture: Block Diagram



Summarizer

Employed Several Techniques:

Each Technique:

- Computes rank for all sentences normalized from 0 to 1
- Is given a weight from 0 to 1

Weighted sentence rank scores are added together
Overall best sentences are selected from the summary sum

Summary Techniques

- Extraction Clustering *(new)*
- Topic Similarity *(new)*
- Sentence Location
- Sentence Length
- tf-idf

Extraction Clustering

- Different extractions used for comparison
- **Entity** (Named Entity Recognition)
  - Semantic information
    - Text
    - Domain Role (person, location etc)
- **Triple**
  - Subject, Predicate Object
- **Fact**
  - Case frame building blocks
  - Element and mode

Trivial Techniques

- **Sentence Position Ranking** - Highest sentences get highest rank
- **Sentence Length Ranking** - Longest sentences get best rank
- **tf-idf** - All non-stop words get tf-idf computed and added. Sentences with the highest sum of tf-idf get best rank.
  - We use the gigaword corpus as a background corpus.

Technique Weighting

It is difficult to tell how important each technique is in contributing to the overall score. Because of this, we established a **weight generator** which did the following:

for each technique:

- compute unweighted sentence ranks.

- Iterate weights of each technique from 0 to 1 at intervals of 0.1
  - for each weight set:
    - rank sentences based on new weights
    - generate rouge scores

At the end, the best set of weights is the one with the optimal score!

Optimal Weights

Optimal Technique Weights:

Technique	Weight
Extraction Clustering	1.0
Topic Similarity	0.0
tf-idf	0.0
Sentence Position	0.0
Sentence Length	0.0

Extraction Clustering Results

Average ROUGE scores for the Extraction Clustering solution:

ROUGE Technique	Recall	Precision	F-Score
ROUGE1	0.23582	0.24725	0.24071
ROUGE2	0.06255	0.06447	0.06337
ROUGE3	0.01995	0.02048	0.02018
ROUGE4	0.00659	0.00672	0.00664

Extraction Clustering Results

Change in Average ROUGE scores From D2 to D3:

ROUGE Technique	Recall		Precision		F-Score	
ROUGE1	0.23582	+0.82%	0.24725	-13.41%	0.24071	-5.69%
ROUGE2	0.06255	+9.05%	0.06447	-8.59%	0.06337	+1.04%
ROUGE3	0.01995	+23.76%	0.02048	+4.01%	0.02018	+14.79%
ROUGE4	0.00659	+23.64%	0.00672	+2.28%	0.00664	+13.70%

Sentence Ordering Introduction

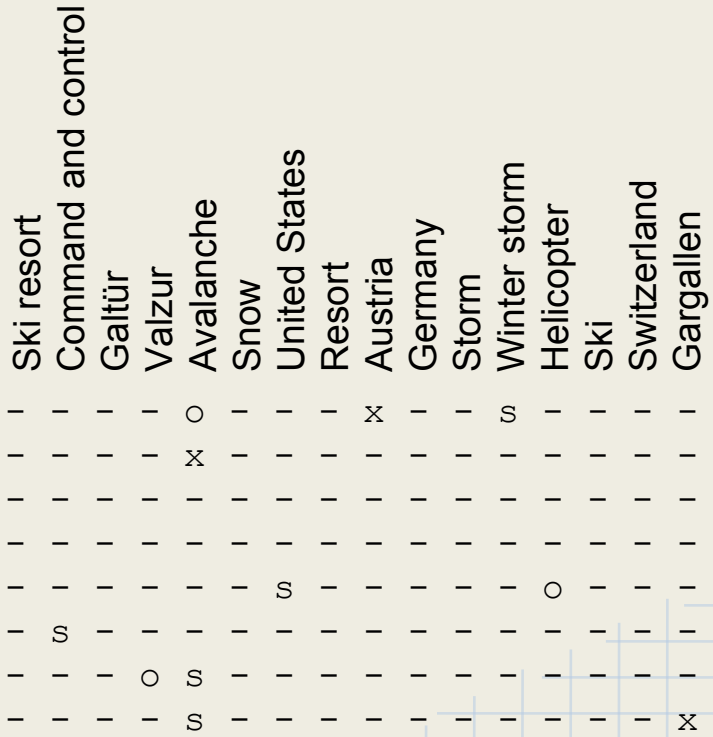
Entity-Based Coherence solution similar to Barzilay and Lapata (2005).

- **NER**: We used a named entity recognizer to extract entities to use in the transition grids.
  - Entities were originally extracted via TextRazor https://www.textrazor.com/
  - Later Converted to using Attensity (ASAS) parser http://attensity.com/
- **Coreference**: Though we have a coreference resolver in our system, we did not have time to connect it up to our entity grid generator.

Entity Coherence

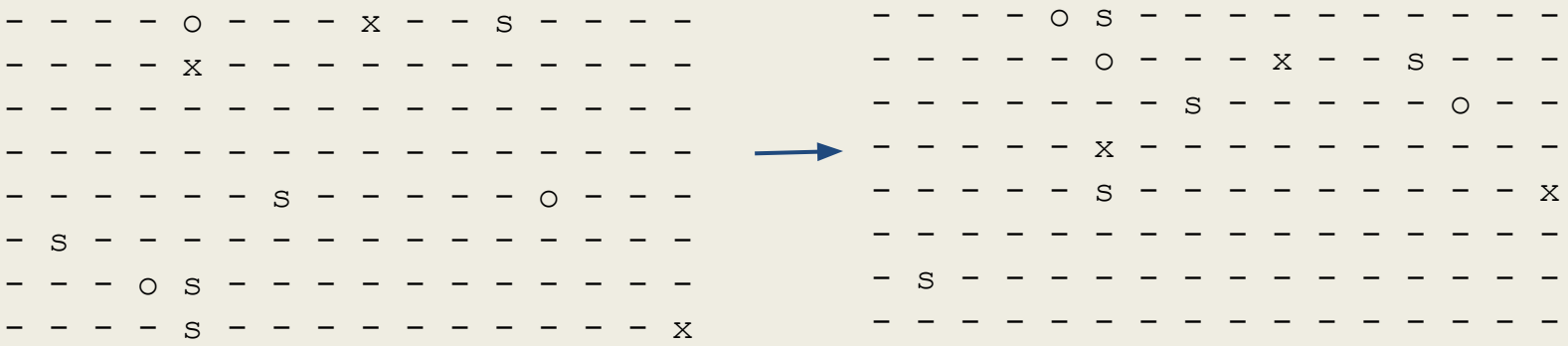
Step1: Build the entity graphs:

1. Read source documents
2. extract the entities using TextRazor or Attensity
3. Assign Subject, Object, Oblique, etc. from dependency parser (textrazor) or semantic frame (attensity)
4. Make sure that smaller entities don't get added additionally.
5. Use freebase ids for entity unique identification



Entity Coherence

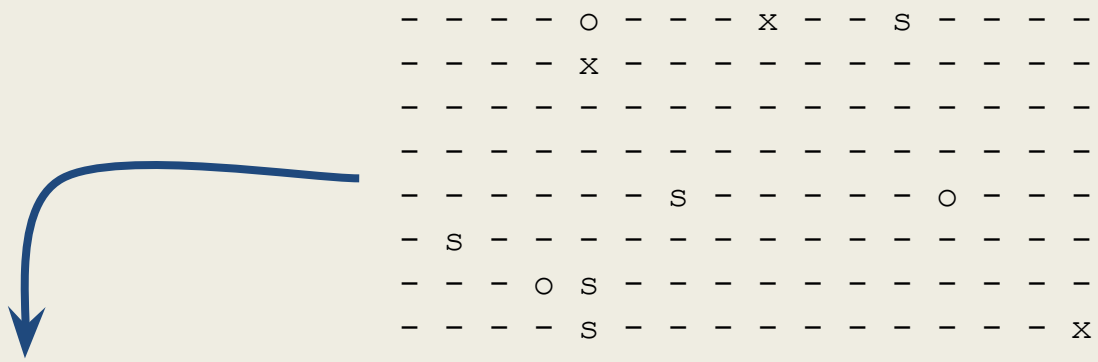
Step 2: Reorder sentences and create the new grid



Entity Coherence

Entity Coherence

Step 3: Extract Feature Vectors



--:0.87 -x:0.01 -o:0.02 -s:0.03 x-:0.02 xx:0.0 xo:0.0 xs:0.0 o-:0.02 ox:0.01 oo:0.0 os:0.0 s-:0.03 sx:0.0 so:0.0 ss:0.01

1:0.87 2:0.01 3:0.02 4:0.03 5:0.02 6:0.0 7:0.0 8:0.0 9:0.02 10:0.01 11:0.0 12:0.0 13:0.03 14:0.0 15:0.0 16:0.01

Entity Coherence

Step 4: Do ranking training on feature vectors to build a model with svmlight:

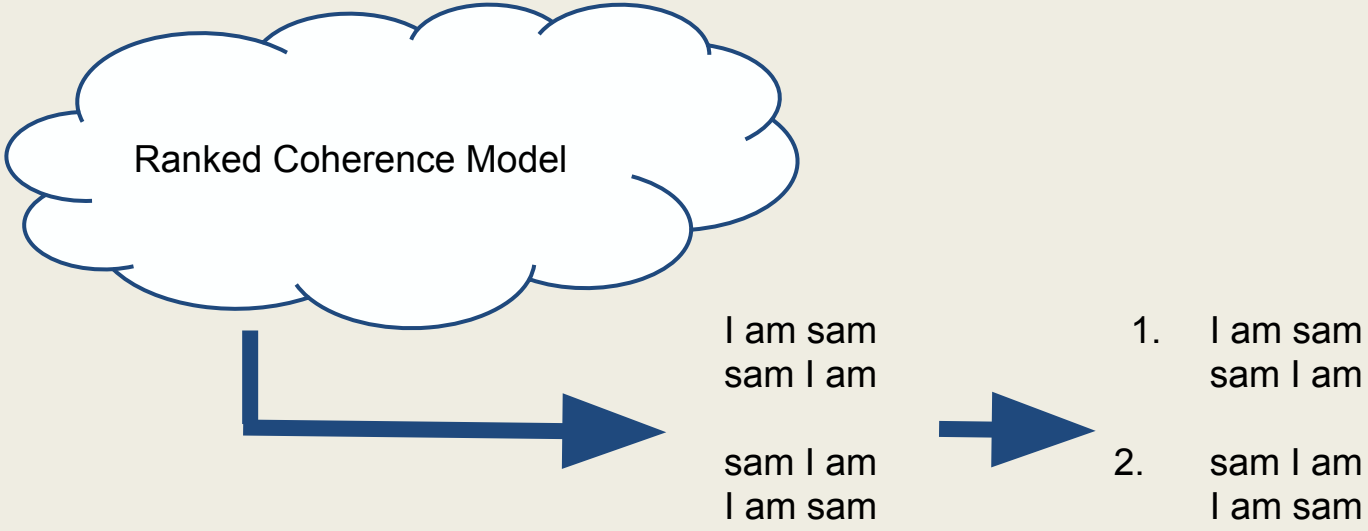
```
1:0.87 2:0.01 3:0.02 4:0.03 5:0.02 6:0.0 7:0.0 8:0.0 9:0.02 10:0.01 11:0.0 12:0.0 13:0.03 14:0.0 15:0.0 16:0.01

                                >

1:0.84 2:0.03 3:0.01 4:0.03 5:0.02 6:0.0 7:0.0 8:0.01 9:0.03 10:0.0 11:0.0 12:0.0 13:0.04 14:0.0 15:0.01 16:0.0
```

Entity Coherence

Step 5: Reorder based on rank assessment from SVM classifier of all permutations of sentences



Entity Coherence

Step 6: Evaluate.

At the end of the day, does it give us a better ROUGE score?

Entity Coherence

Step 6: Evaluate.

At the end of the day, does it give us a better ROUGE score?

Entity Coherence Results

Change in Average ROUGE scores From Unordered to Ordered:

ROUGE Technique	Recall		Precision		F-Score	
ROUGE1	0.22494	-4.61%	0.27984	+13.18%	0.24734	+2.75%
ROUGE2	0.05357	-14.36%	0.06688	+3.74%	0.05893	-7.01%
ROUGE3	0.01532	-23.21%	0.01904	-7.03%	0.01683	-16.60%
ROUGE4	0.00491	-25.49%	0.00614	-8.63%	0.00541	-18.52%

Entity Coherence Challenges

Problem #1: what do we train on?

- Barzilay and Lapata (2005) trained on different things depending on what they were reordering:
  a. **Summaries:** human-generated summaries with coherence scores were used as input to the classifier for ranking
  b. **Documents:** documents and reordered documents were used as input for ranking

- We don't have human-generated summaries with coherence scores, so we tried two different approaches:
  1. **Train on documents**, summarize summaries.
  2. **Train on gold-standard summaries**. Rerank the gold-standard summaries using random reordering as in the document case

Entity Coherence Challenges

Problem #2: Summaries are too short.

There is not a lot of data to go on in the summaries. Reranking an average of three sentences based on transition scores of only length 2 provides very little data by which to understand the transitions for decoding with the ranking classifier.

Possibly, this factor will be mitigated once we have some culling in our sentence realization solution for D4. Once this happens, we will have more sentences in our summary, and may be able to generate a more meaningful feature vector for SVM decoding.

Entity Coherence Challenges

Things to try next:

1. **Tuning:** It would be instructive to try some tuning with Barzilay and Lapata's two "tuning" parameters:
   a. **Transition Length** - try transition lengths from 2 to 4 (we used length 2 always)
   b. **Salience Pruning** - prune entites by "salience" defined by frequency. (we didn't do this at all)

2. **Better Training Data:** We'd love to train on some human-generated summaries scored for coherence. Is there a good corpus around?

3. **Coreference Resolution:** This is next! We have coref resolution in the project, we just haven't hooked it up to the Entity Coherence feature.

References

Heinzerling, B and Johannsen, A (2014). pyrouge (Version 0.1.2) [Software]. Available from https://github.com/noutenki/pyrouge

Lin, C (2004). ROUGE (Version 1.5.5) [Software]. Available from http://www.berouge.com/Pages/default.aspx

Roylance, M (2015). Attensity ASAS (Version 0.1) [Software]. Available from http://www.attensity.com

Crayston, T (2015). TextRazor (Version 1.0) [Software]. Available from https://www.textrazor.com/

Jaochims, T (2002a). SVMLight (Version 6.02) [Software]. Available from http://svmlight.joachims.org/

Barzilay, R., & Lapata, M. (2008). Modeling local coherence: An entity-based approach. Computational Linguistics, 34(1), 1-34.

Jurafsky, D., & Martin, J. H. (2009). Speech & language processing. Pearson Education India.