LexRank

Graph-based Lexical Centrality as Salience in Text Summarization

Presented By
Eesh an Malhotra, Samkit Shah, Sanjeev Mk, Ankith Shetty

About the Paper

Authors

Günes Erkan

Dragomir R. Radev

University of Michigan, Ann Arbor

Published

2004, Journal of Artificial Intelligence Research

http://www.aaai.org/Papers/JAIR/Vol22/JAIR-2214.pdf

Citations

775

Introduction

- Text summarization is the process of automatically creating a compressed version of a given text that provides useful information for the user.

- Graph-based methods used earlier in NLP for word clustering, prepositional phrase attachment. Natural extrapolation to sentences

- Extractive summarization v/s Abstractive summarization

Introduction

- Topic oriented summarization v/s Generic summarization
- Single document summarization/s Multi document summarization
- This paper uses *multi-document extractive generic* text summarization
- Baseline for comparison: centroid based summarization using tf-idf clustering

Intuition

- In extractive summarization, goal is to find salient sentences of the document.
- They can be sentences which are related to large number of sentences in document.

Steps in the Algorithm

1. Find pairwise similarity between sentences

2. Construct (undirected) graph of sentences using similarity as edge weights. Use thresholding to drop irrelevant edges

3. Find salient sentences in the graph, using ‘prestige’-based methods

Calculating Sentence Similarity

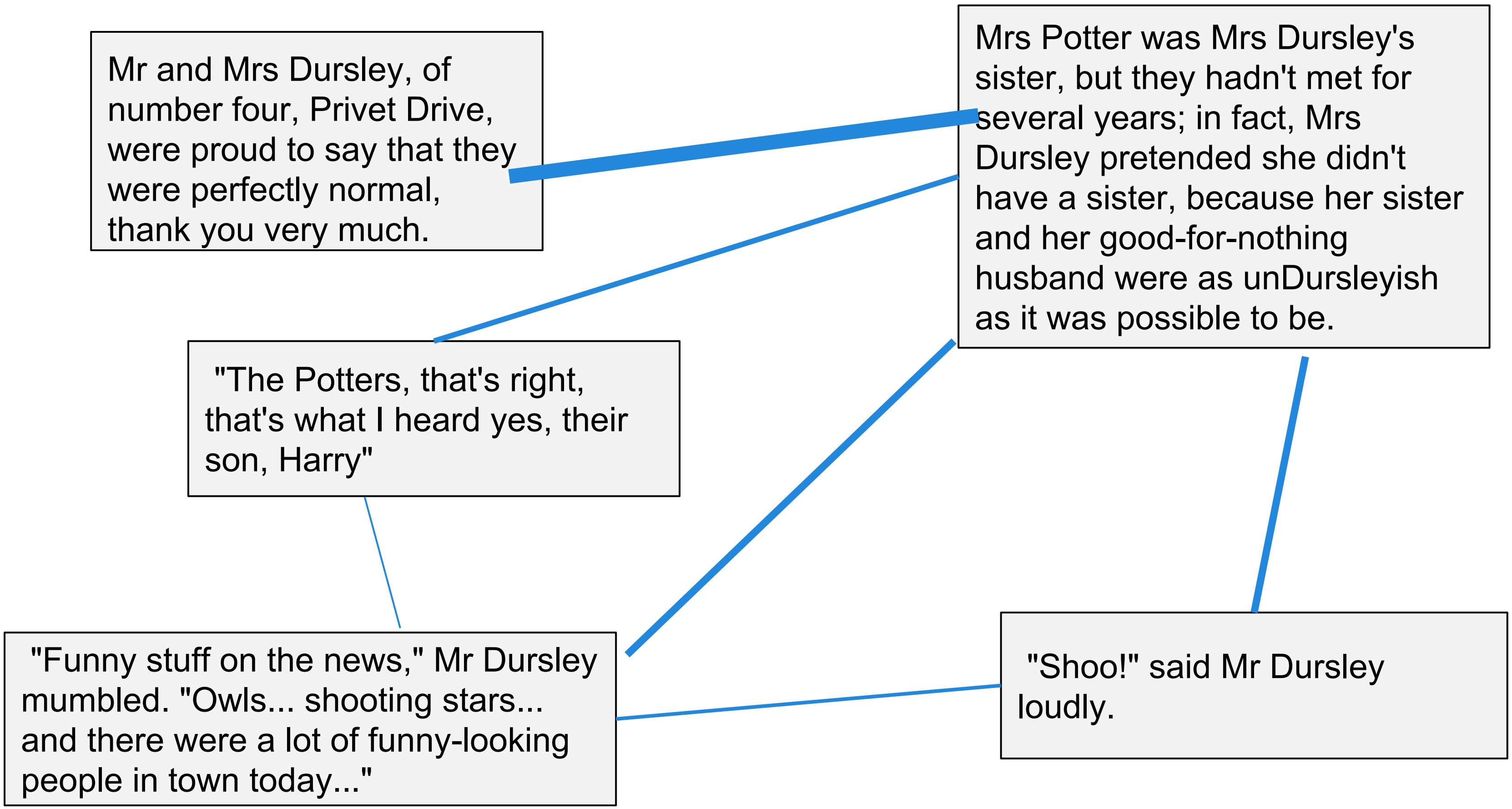
- The algorithms treats a sentence as a bag-of-words
- Each sentence is treated as a vector in word-space.
- The value in each dimension is calculated using tf-idf score on that word
- Then, sentence similarity is calculated using modified idf cosine similarity metric

Calculating Sentence Similarity

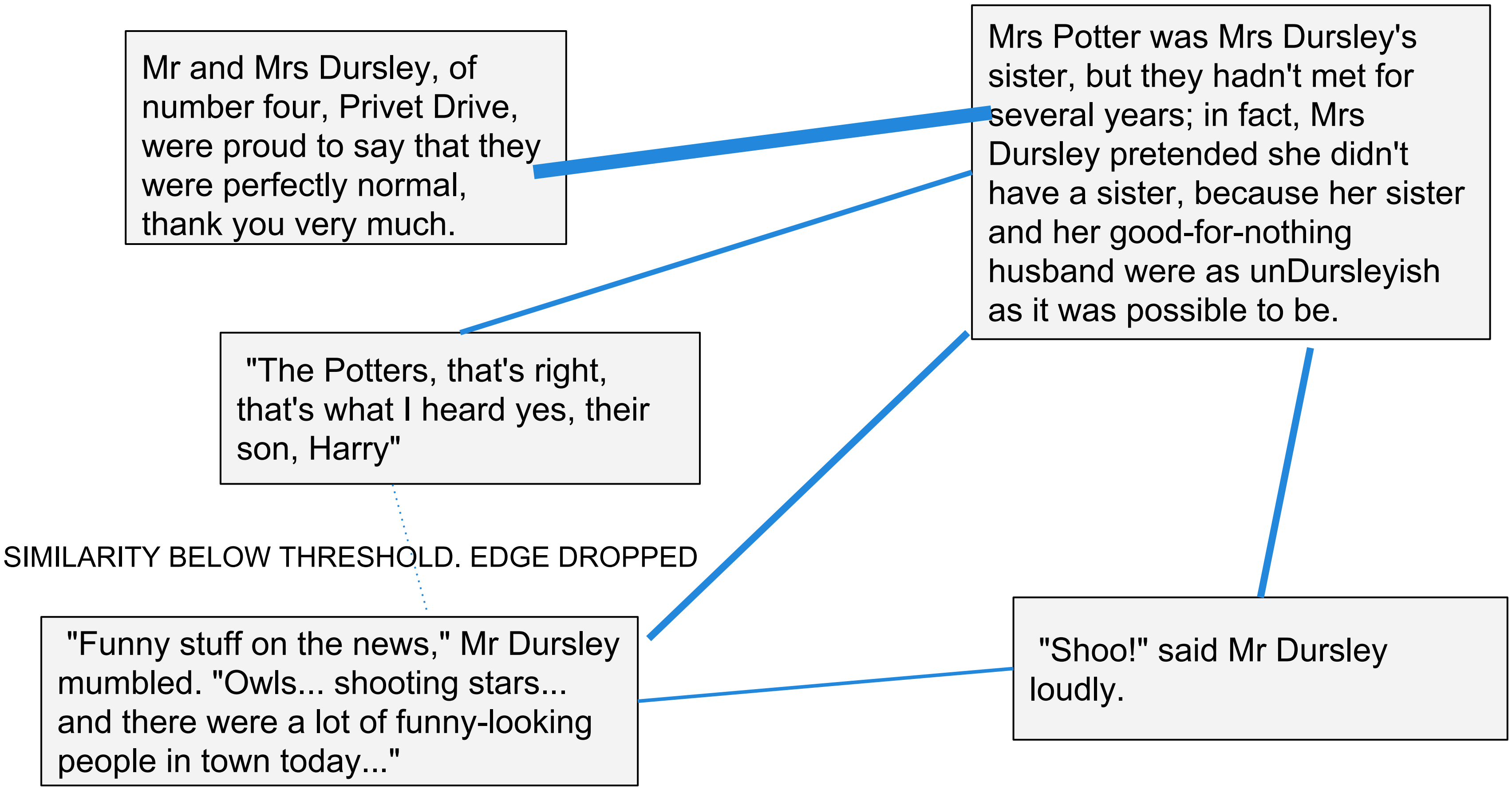
	...	the	bat	cat	...
"The cat took the bat"		0.7	0.7	0.5	
"The cat is sleeping"		0.5	0	0.6	

$$\text{idf-modified-cosine}(x, y) = \frac{\sum_{w \in x,y} \text{tf}_{w,x} \text{tf}_{w,y} (\text{idf}_w)^2}{\sqrt{\sum_{x_i \in x} (\text{tf}_{x_i,x} \text{idf}_{x_i})^2} \times \sqrt{\sum_{y_i \in y} (\text{tf}_{y_i,y} \text{idf}_{y_i})^2}}$$

Graph Construction



Graph Construction



Finding Salient Sentences

- The problem is now to discover more salient sentences in the graph.

  - Naive Method: Centrality measures, such as degree centrality.
    Factors *how many neighbours a vertex has*

  - LexRank: Use prestige based method based on Markov processes
    Also factors *how important are the neighbours of a vertex*

Finding Salient Sentences

- This motivates a Page-Rank like algorithm.
- The importance of a vertex depends on the number and importance of its neighbours, which, in turn, depend on *their* neighbours
- This creates a self-referential loop
- However, an iterative algorithm leads to convergence
- Alternatively, the problem can be modeled as a markov chain, which allows use of eigenvector based solutions

Salient Sentences: Markov Chain

- The state vector is a vector P_1, P_2, ..., P_n of the probabilities of each sentence of occurring in the summary
- This is initialized to (1/n) for each sentence
- In each iteration, this value is updated by summing the probabilities of its neighbours, weighted by the similarities

$$p(u) = \sum_{v \in adj[u]} \frac{p(v)}{deg(v)}$$

Salient Sentences: Markov Chain

$$p(u) = \sum_{v \in adj[u]} \frac{p(v)}{deg(v)}$$

This can be modified to account for damping as:

$$p(u) = \frac{d}{N} + (1-d) \sum_{v \in adj[u]} \frac{\text{idf-modified-cosine}(u,v)}{\sum_{z \in adj[v]} \text{idf-modified-cosine}(z,v)} p(v)$$

Demo

Results

Datasets for comparison are used from *DUC 2003, 2004* with hand-annotated summaries

(Document Understanding Conference; task: Text Summarization)

Comparison metric:

Recall-Oriented Understudy for Gisting Evaluation,

Or *ROUGE*.

Two common variants are

- ROUGE-N: N-gram based co-occurrence statistics.
- ROUGE-L: Longest Common Subsequence (LCS) based statistics

Results

ROUGE-1 scores for different MEAD policies on 17% noisy DUC 2003 and 2004 data.

	min	max	average
Random			0.3315
Centroid	0.3706	0.3898	0.3761
Degree (t=0.1)	0.3874	0.3943	0.3906
LexRank (t=0.1)	0.3883	0.3992	0.3928

Conclusions

- Using a similarity graph provides a better view of important sentences compared to the centroid approach

- This approach of lexrank is domain-independent and easily portable, allowing for quick and efficient summarization

- The graph-based representation has allows processing with many other applications in NLP as well

Thank You