APAN PS5430
Applied Text & Natural Language Analytics
Week 5: Information Extraction II

Javid Huseynov, Ph.D.
Thursday, February 20, 2020

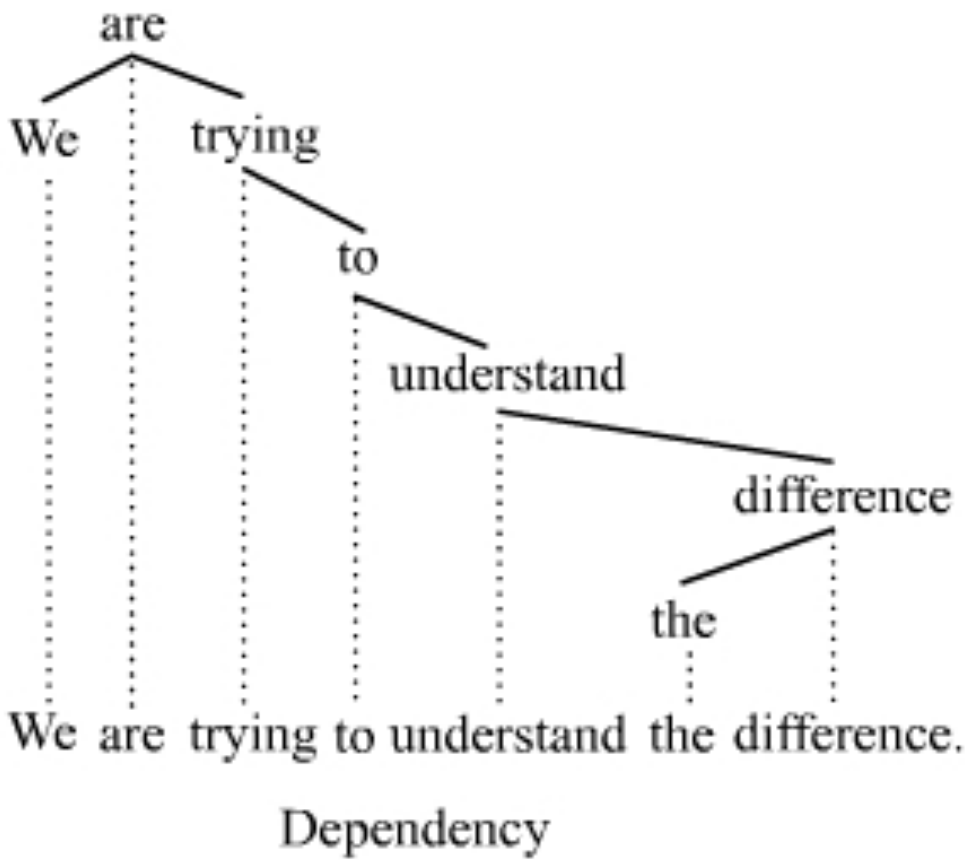
COLUMBIA UNIVERSITY

Week 5 Agenda

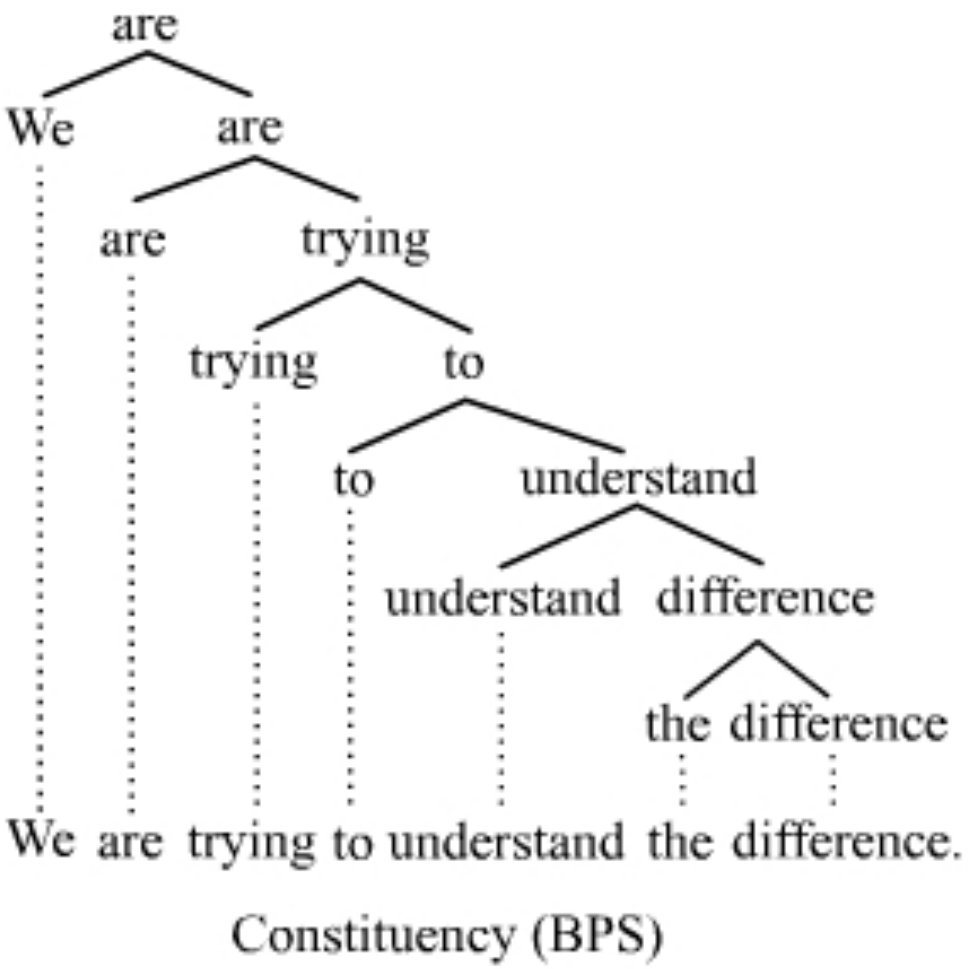
- Relationship Extraction
- Information Retrieval
- Graphs
- Keyword or Key Phrase Extraction
- Text Summarization
- Course Exercise

© 2019 Columbia University

Dependency vs Constituency



Dependency

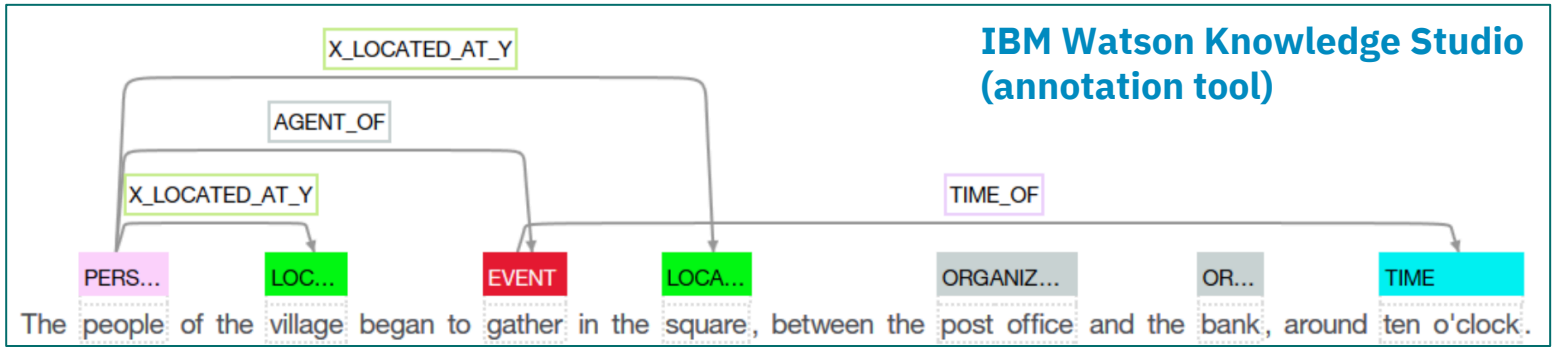


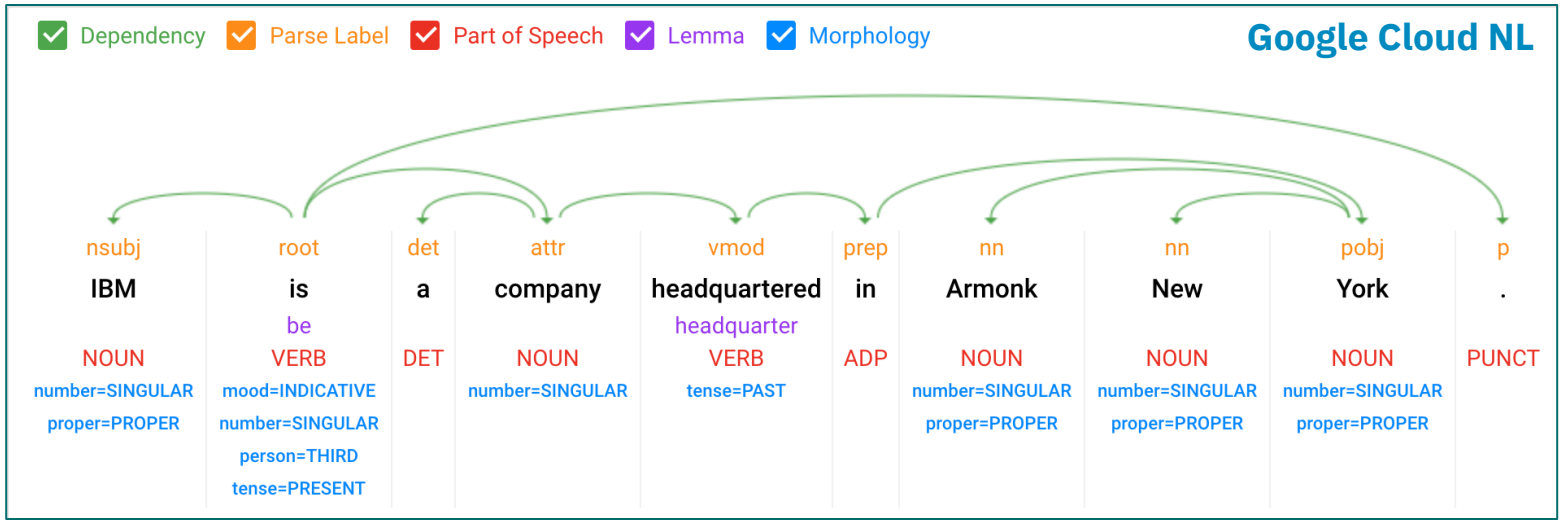
Constituency (BPS)

© 2019 Columbia University

Relationship Extraction

Task of *detecting* and *classifying* **semantic relations** between entities in text, to reconstruct dependencies and meaning





Methods

- Rule-based
- Bootstrapping (Pattern-based)
- Supervised Learning
- Unsupervised Learning
- Domain Ontologies

Applications

- Knowledge Graphs
- Document Summarization
- Concept Extraction

Annotation / Modeling Tools

- IBM Watson NLU
- IBM Watson Knowledge Studio
- Google Cloud NL
- MIT BRAT

© 2019 Columbia University

Relationship Extraction => Knowledge Discovery

Involvement of Tumor Necrosis Factor Receptor-associated Protein 1 (TRAP1) in Apoptosis Induced by β-Hydroxyisovalerylshikonin*

IE

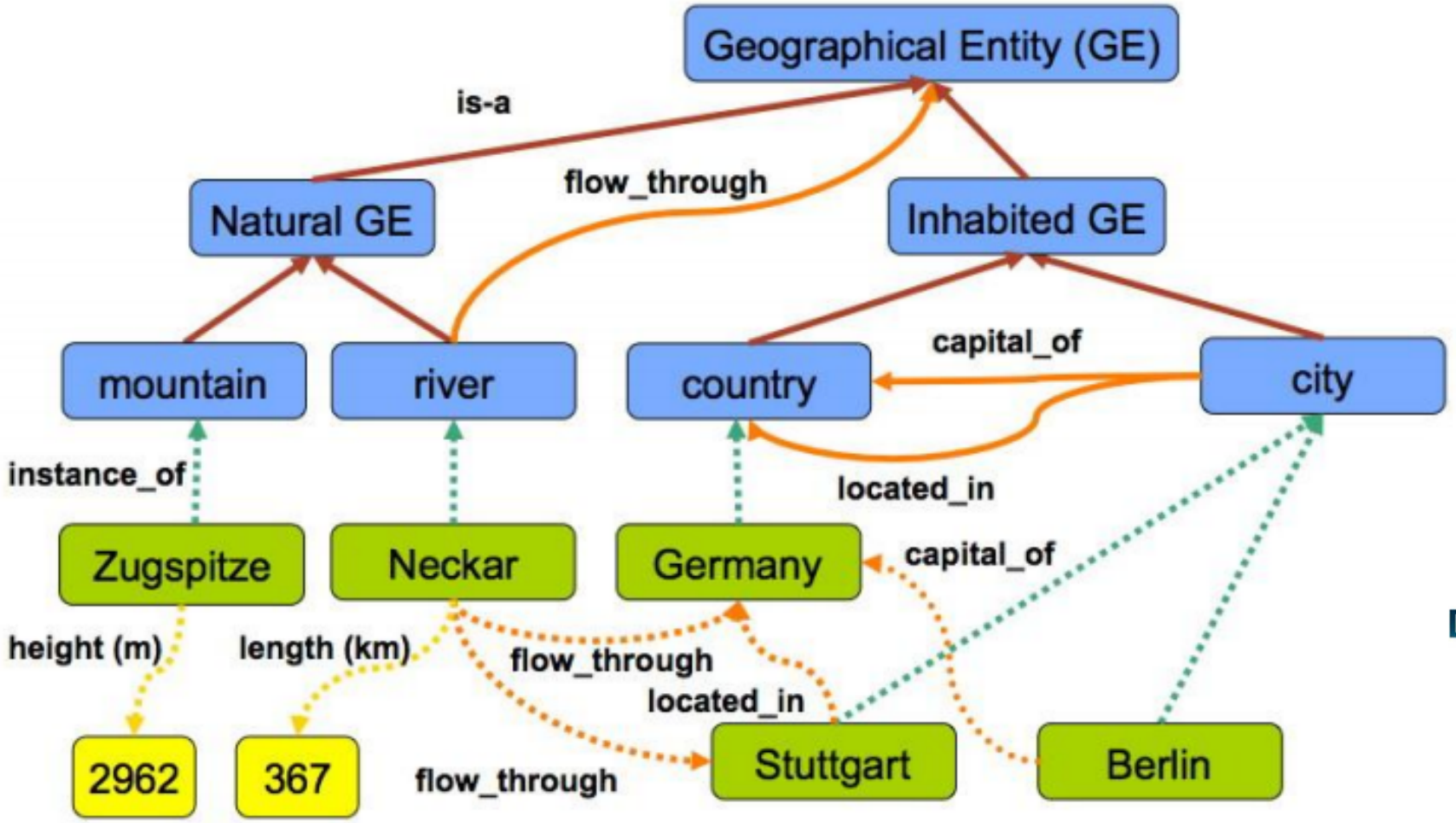
Subject	Relation	Object
p53	is_a	protein
Bax	is_a	protein
p53	has_function	apoptosis
Bax	has_function	induction
apoptosis	involved_in	cell_death
Bax	is_in	mitochondrial outer membrane
Bax	is_in	cytoplasm
apoptosis	related_to	caspase activation
...	...	...

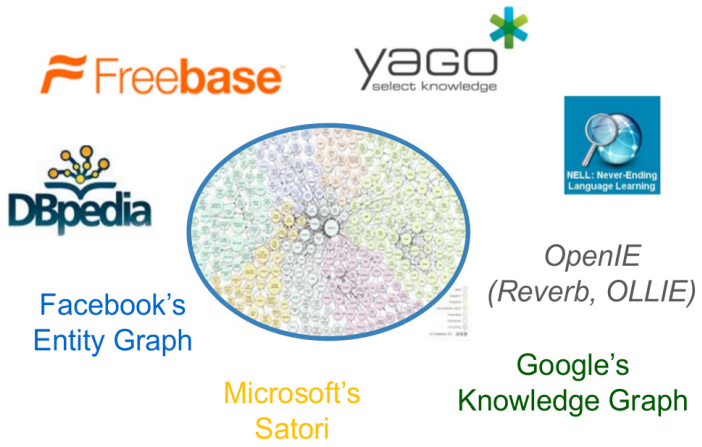
textual abstract: summary for human

structured knowledge extraction: summary for machine

© 2019 Columbia University

Knowledge Graphs





© 2019 Columbia University

Background: Information Retrieval and Graphs

Information Retrieval (IR)

- The task of obtaining information relevant to an input query or user interest

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

$$F = \frac{2 \cdot \text{precision} \cdot \text{recall}}{(\text{precision} + \text{recall})}$$

- Use cases
  - Search Engines
  - Question Answering / Chatbots
  - Keyword Extraction
  - Text Summarization

IR Models

- Set-Theoretical
  - Boolean
  - Fuzzy Retrieval (Boolean + Fuzzy Sets)
- Algebraic
  - Vector Space Model
  - Latent Semantic Analysis (LSA)
  - Singular Value Decomposition (SVD)
- Probabilistic
  - Bayesian Models
  - Latent Dirichlet Allocation LDA

© 2019 Columbia University

Standard Boolean Model

- Classical IR model based on Boolean (AND/OR) logic
- Given documents **$D=\{d_1,\ldots,d_n\}$** based on query **$q = \{q_1, q_2\}$**:
  OR – retrieved document **d_i** contains either **q_1** or **q_2**
  AND – retrieved document **d_i** contains both **q_1** and **q_2**
- Try Googling:
  "**Apple**" AND "**Fruit**" -> apple
  "**Apple**" OR "**Fruit**" -> iPhone

- Pros
  - Simple and Intuitive
  - Easy to implement
- Cons
  - All terms are equally weighted
  - No similarity measure
  - Difficult to scale

© 2019 Columbia University

Vector-Space Model

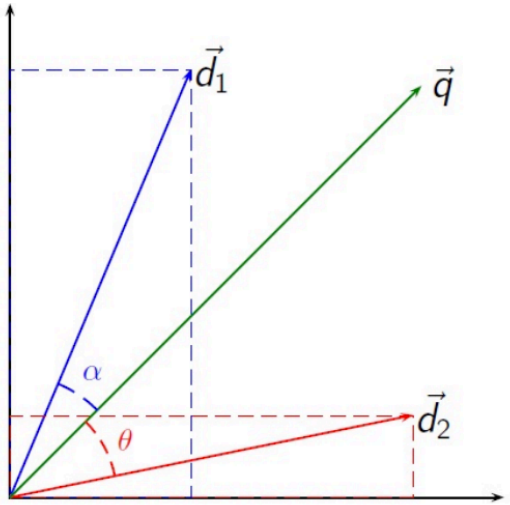
- An algebraic representation of text documents or queries as vectors

$$d_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$$
$$q = (w_{1,q}, w_{2,q}, \ldots, w_{n,q})$$

- Cosine similarity

$$\|\mathbf{q}\| = \sqrt{\sum_{i=1}^{n} q_i^2}$$

$$\cos\theta = \frac{\mathbf{d_2} \cdot \mathbf{q}}{\|\mathbf{d_2}\| \, \|\mathbf{q}\|}$$



$$\mathbf{sim}(d_j, q) = \frac{\mathbf{d_j} \cdot \mathbf{q}}{\|\mathbf{d_j}\| \, \|\mathbf{q}\|} = \frac{\sum_{i=1}^{N} w_{i,j} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,j}^2} \sqrt{\sum_{i=1}^{N} w_{i,q}^2}}$$

One-Hot Encoding (Word Embedding)

- Given "`Can I eat the pizza`" of `N=5`
  1. Convert to lower case
  2. Sort the words in alphabetical order
  3. Give numerical labels to each word:
     `can:0, i:2, eat:1, the:4, pizza:3`
  4. Transform to binary vectors

```
[[1. 0. 0. 0. 0.] #can
 [0. 0. 1. 0. 0.] #i
 [0. 1. 0. 0. 0.] #eat
 [0. 0. 0. 0. 1.] #the
 [0. 0. 0. 1. 0.]] #pizza
```


© 2019 Columbia University

Term-frequency inverse document frequency (TFIDF)

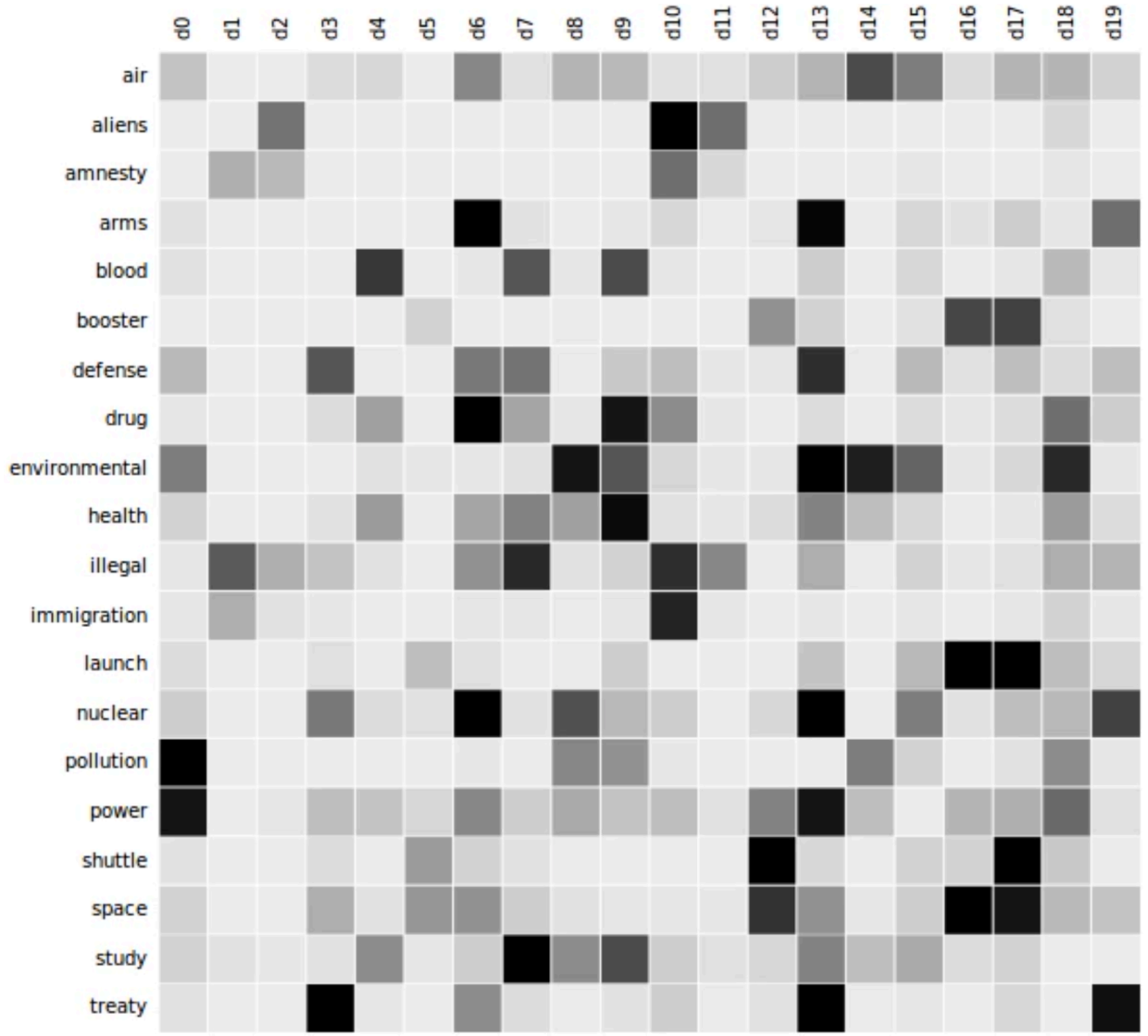
- Fundamental measure of importance of a given term ***t*** in a collection of documents ***D***
- ***tf*** – frequency of ***t*** in a given document ***d***

$$f_{t,d} \,/\, \sum_{t' \in d} f_{t',d}$$

- ***idf*** – inverse of the number of documents in which ***t*** occurs, i.e. importance of the term ***t***

$$\operatorname{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

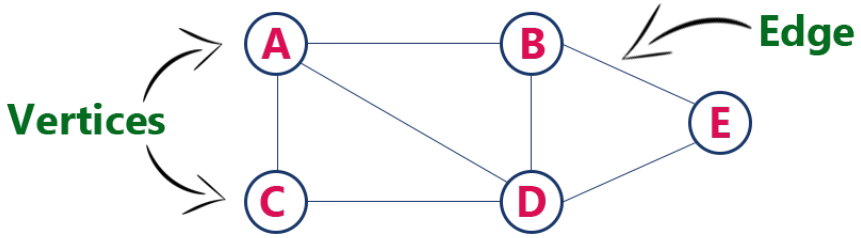
$$\operatorname{tfidf}(t, d, D) = \operatorname{tf}(t, d) \cdot \operatorname{idf}(t, D)$$



© 2019 Columbia University

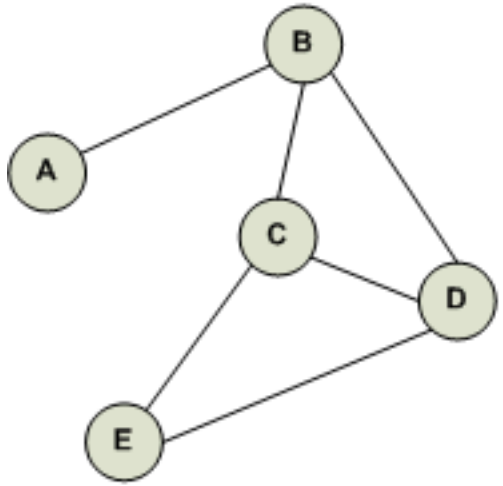
What is a Graph?

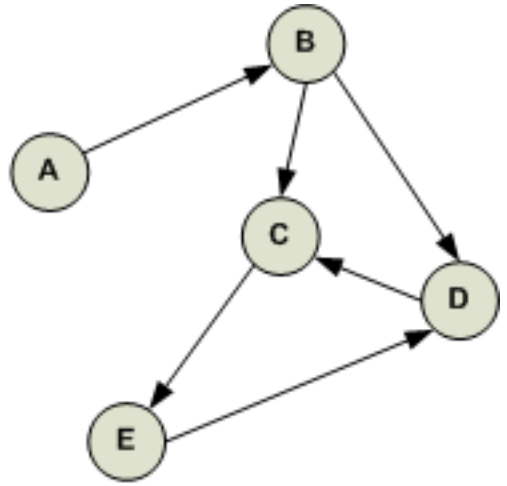
- Graph G(V, E) is a data structure:
  - V – set of vertices
  - E – set of edges connecting vertices



- Graph edges may have weights and directions
- Directed graphs may have cycles

- Directed vs Undirected Graph





© 2019 Columbia University

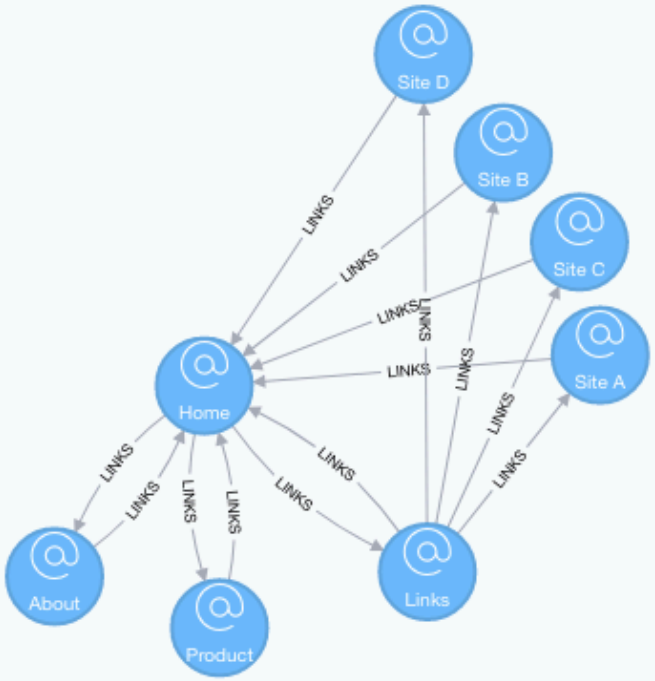
PageRank

- PageRank (PR) – an algorithm used to rank websites retrieved by Google Search
- **Intuition**: Number of links to a given Page A determine the importance of Page A

$$PR(A) = (1-d) + d\ (PR(T_1)/C(T_1) + \ldots + PR(T_n)/C(T_n))$$

- Page **A** has pages T_1 through T_N pointing to it
- **d** is a damping factor between 0 and 1, typically 0.85
- C(A) the number of links going out page A

Page	Rank
Home	3.232
Product	1.059
Links	1.059
About	1.059
Site A	0.328
Site B	0.328



© 2019 Columbia University

Keyword Extraction & Text Summarization

Automated Key Phrase or Keyword Extraction

- **Keywords** – a sequence of one or more words that provide a compact representation of document's content
- Methodology
  - Corpus- or External Reference-based, e.g. WordNet
  - Document-based
- Steps
  - Candidate Identification
  - Task Reformulation: Classification or Ranking
- Salience vs Relevance

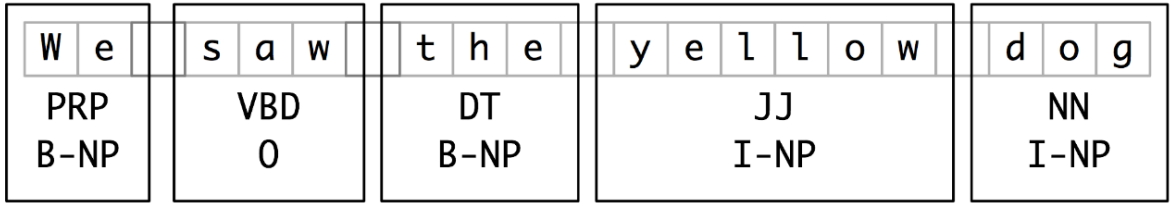
Approaches

- Rule-based
- Supervised
  - Binary classification
- Unsupervised
  - Graph-based ranking
  - Topic-based clustering

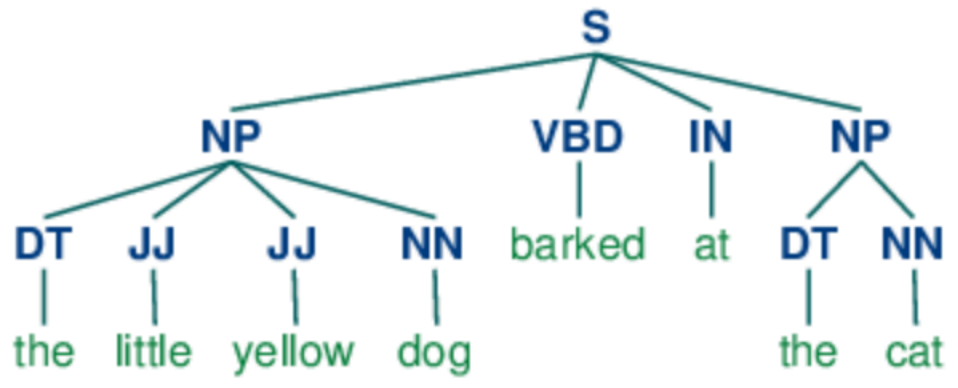
© 2019 Columbia University

Candidate Identification

- Brute-force: Consider all words/phrases as candidates
- Basic Heuristics:
  - Remove stop words and punctuation
  - Filter words for POS tags and patterns
  - Use external knowledge bases, e.g. WordNet



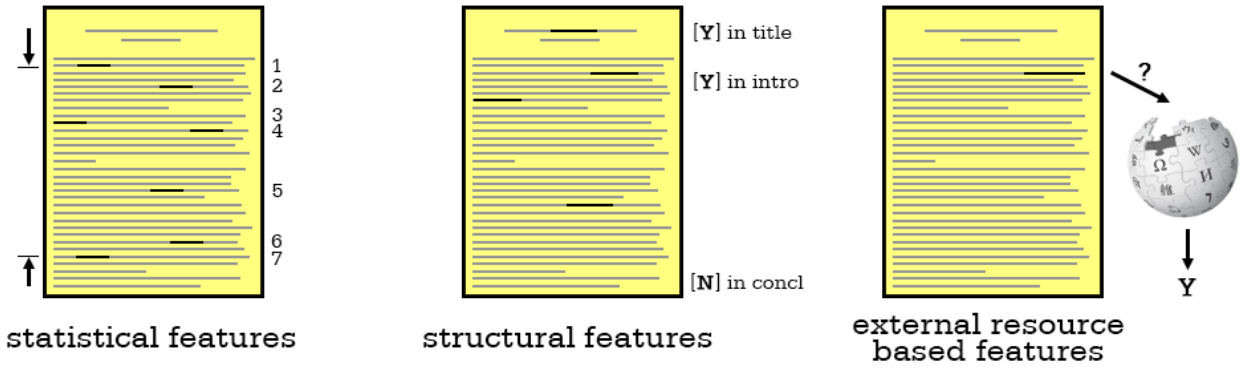

```
>>> sentence = [("the", "DT"), ("little", "JJ"), ("yellow", "JJ"), ❶
... ("dog", "NN"), ("barked", "VBD"), ("at", "IN"),  ("the", "DT"), ("cat", "NN")]

>>> grammar = "NP: {<DT>?<JJ>*<NN>}" ❷

>>> cp = nltk.RegexpParser(grammar) ❸
>>> result = cp.parse(sentence) ❹
>>> print(result) ❺
(S
  (NP the/DT little/JJ yellow/JJ dog/NN)
  barked/VBD
  at/IN
  (NP the/DT cat/NN))
>>> result.draw() ❻
```



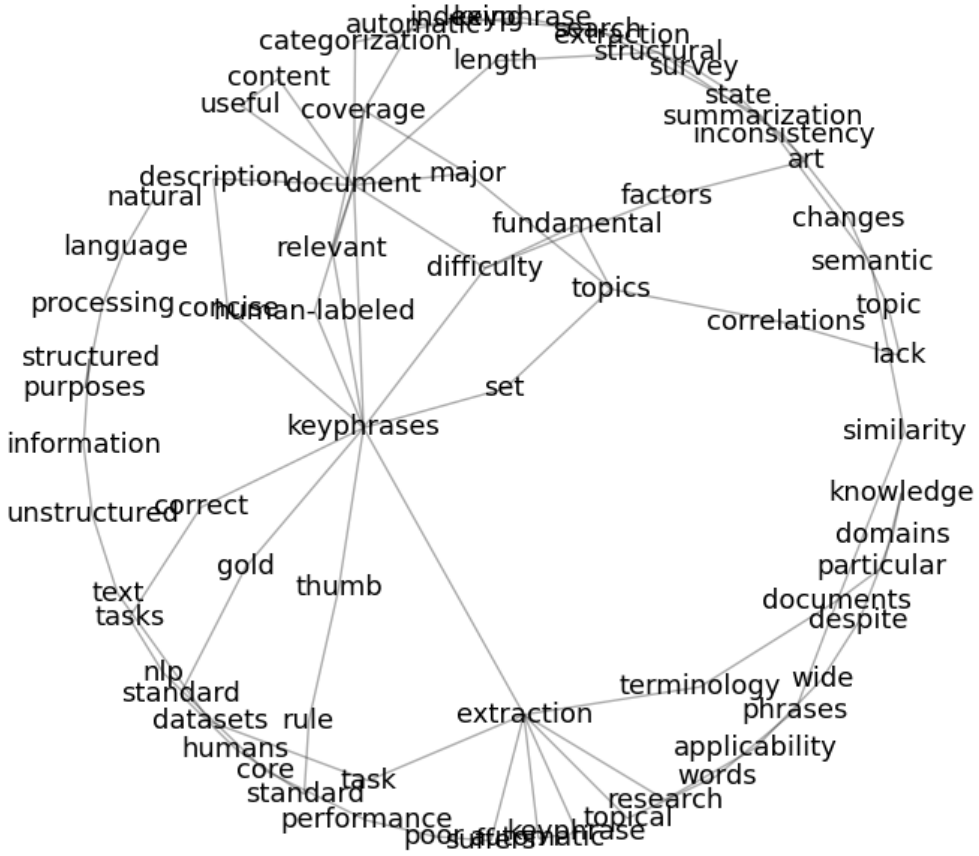
© 2019 Columbia University

Keyword Extraction

- Supervised (Task Reformulation)
  - Classification and Ranking



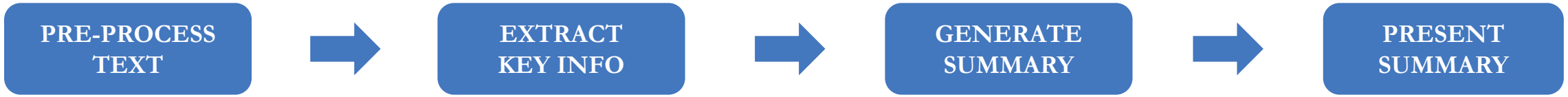
- Unsupervised
  - Graph-based Ranking or Clustering



© 2019 Columbia University

Text Summarization

- Extracting the most important context from a body of text and expressing it in a condensed form consistent with user or application needs
- **Steps**



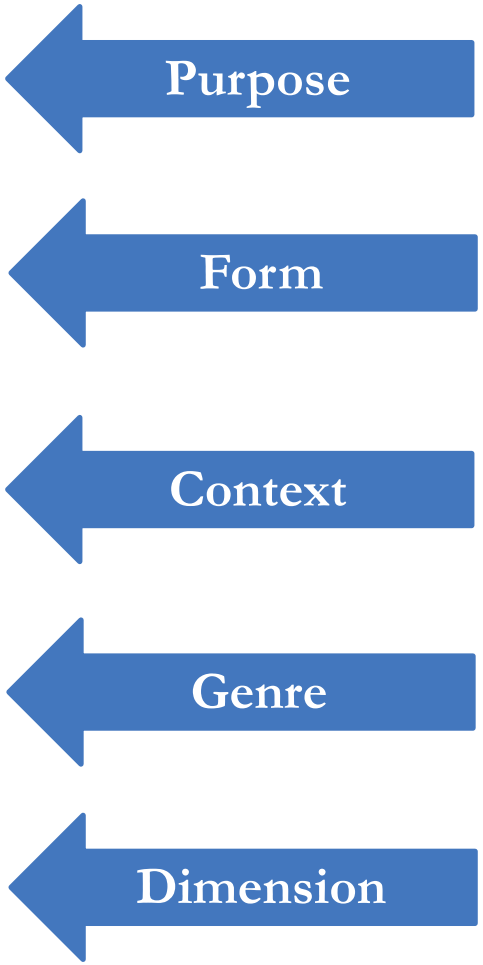
- **Categories**

Extractive Summarization	Abstractive Summarization
• Produce just a short summary	• Produce the summary of what is important
• Lexicon, word frequencies, similarity	• Dependencies, relations, graphs
• Statistical	• Conceptual
• Information Retrieval-based	• Information Extraction-based
• Relevance	• Salience

© 2019 Columbia University

Types of Summaries

- Indicative *vs.* informative
  ...used for quick categorization vs. content processing
- Extract *vs.* abstract
  ...lists fragments of text vs. re-phrases content coherently.
- Generic *vs.* query-oriented
  ...provides author's view vs. reflects user's interest.
- Background *vs.* just-the-news
  ...assumes reader's prior knowledge is poor vs. up-to-date.
- Single-document *vs.* multi-document source
  ...based on one text vs. fuses together many texts.



© 2019 Columbia University

Extractive Summarization

Word frequency or TFIDF-based

- High-frequency words are related to the document topic
- Sentence importance depends on the number of occurrences of significant words
- Rank sentences by importance and pick the top *k*

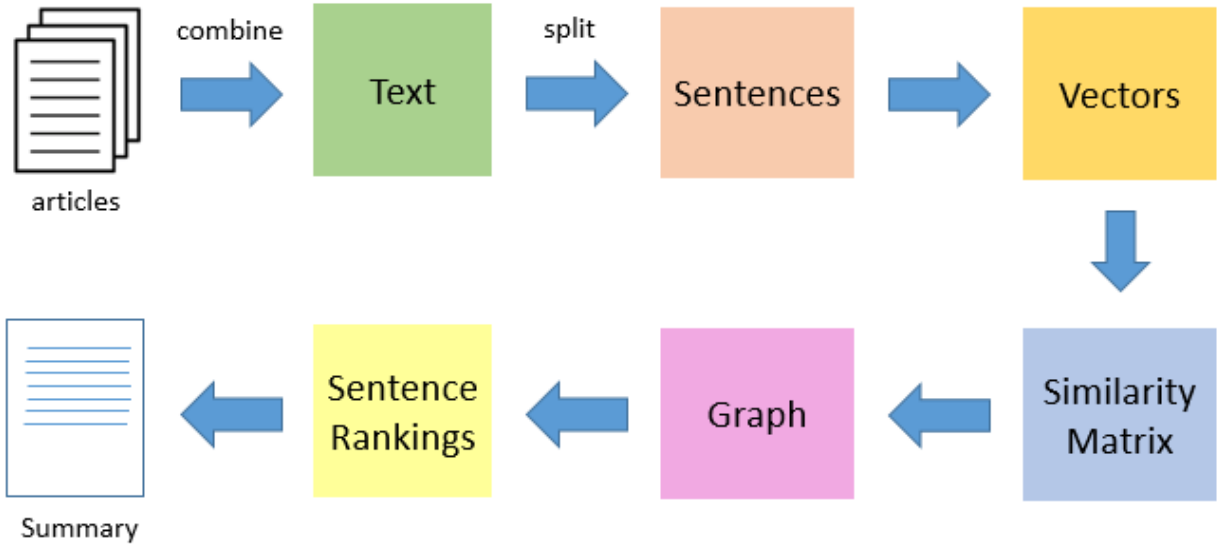
Graph or Centrality-based

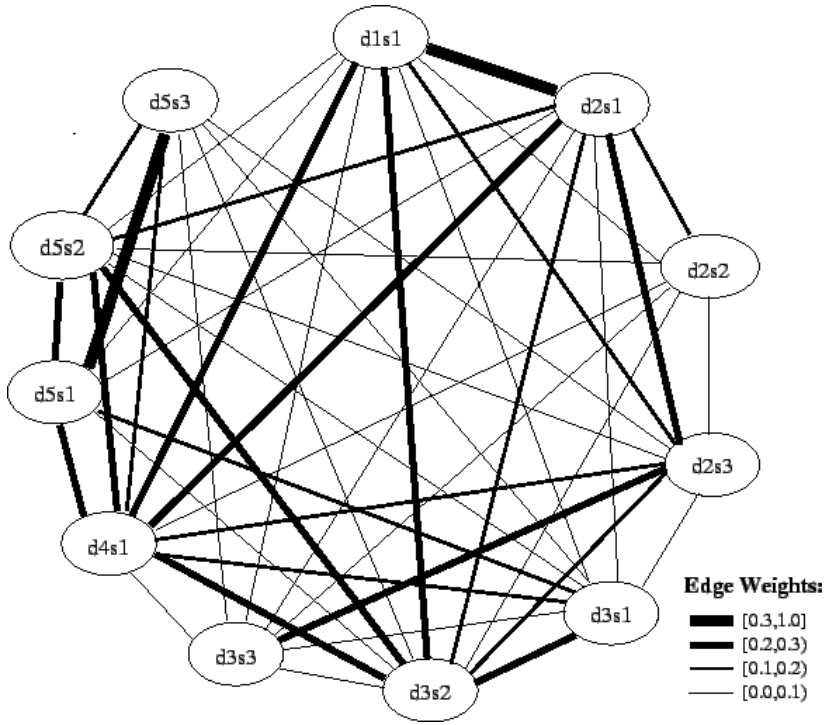
- Summarizing sentences are well connected and similar to other sentences
- Rank sentences by their centrality degree
- Pick top *k* as summarizing sentences
- **Example**: LexRank & TextRank

© 2019 Columbia University

TextRank & LexRank (a la PageRank)

1. Concatenate all texts contained in the articles
2. Split the text into individual sentences
3. Find vector representation (word embeddings) for every sentence
4. Calculate similarities between sentence vectors are store in a matrix
5. Convert the similarity matrix into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation
6. A certain number of top-ranked sentences form the final summary





© 2019 Columbia University

Abstractive Summarization

Rhetoric-based Summarization

- Rhetoric Relation
  - Between two non-overlapping texts
  - **Nucleus**: Core idea
  - **Satellite**: Arguments in support of core idea
- Method
  - Generate Rhetoric Structure Trees
  - Pick the best tree based on clustering or tree-shape
  - Pick top K nodes (sentences) closest to the root of the tree

© 2019 Columbia University