Wikipedia query Engine

NTDS Team 2 - Fall 2019

EL Amrani Ayyoub
Micheli Vincent
Myotte Frédéric
Sinnathamby Karthigan

Contents

- Introduction/Motivation
- Acquisition
- Exploration
- Exploitation
- Results & Limitations
- Data product

Introduction

Motivation

- Wikipedia
  - Variety of data
  - Completeness of Information
  - Rich Graph
- Graph ML:
  - Robust.
  - Leveraging hidden features and attributes
- Goal:
  - Recommend interesting pages related to the user's query desire.

Acquisition

Seealsology

https://densitydesign.github.io/strumentalia-seealsology/

Wikipedia Python API

https://pypi.org/project/wikipedia/

- Seealsology
  - Seed: Machine learning, Natural language processing, Artificial intelligence, Artificial neural network, Chatbot, Intelligent agent, Data visualization
- Graph:
  - Nodes: each page
  - Edges: from the “See also section”

- Keywords extraction through TF-IDF

Exploration

NetworkX

https://networkx.github.io/documentation/stable/

WordCloud

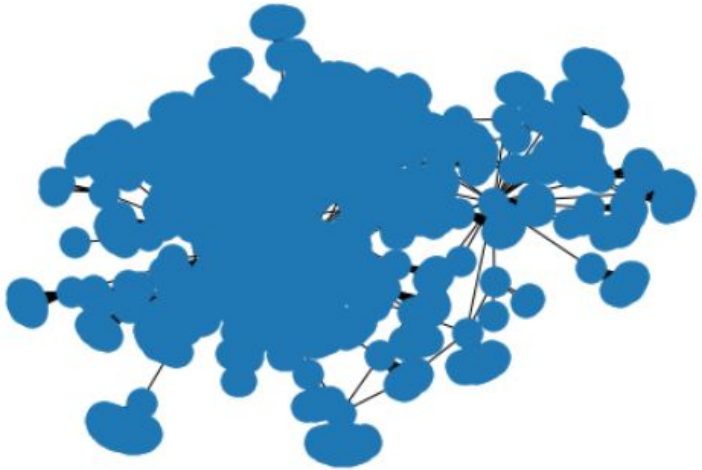
https://amueller.github.io/word_cloud/

- Graph Properties
  - Connected Components & Diameter
  - Sparsity & Degree distribution

Exploration – Graph Properties

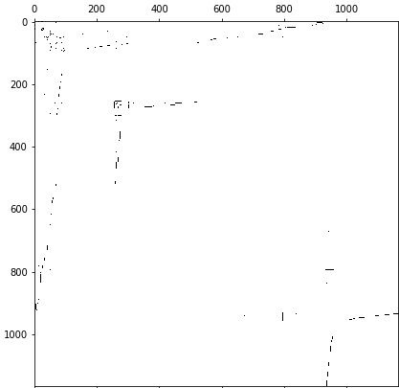
Key properties:

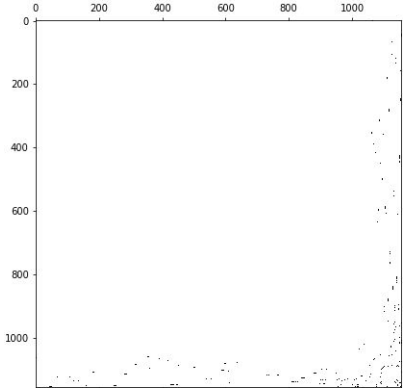
- Connected Graph
- 1166 nodes & 1439 edges
- Diameter of 9



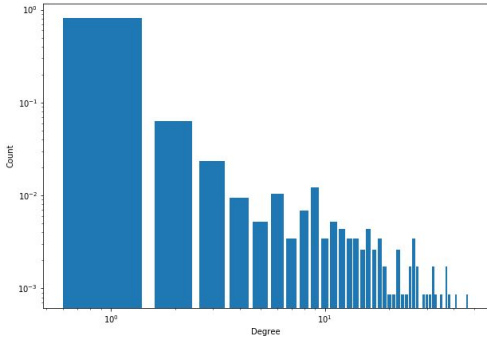
Exploration - Graph Properties

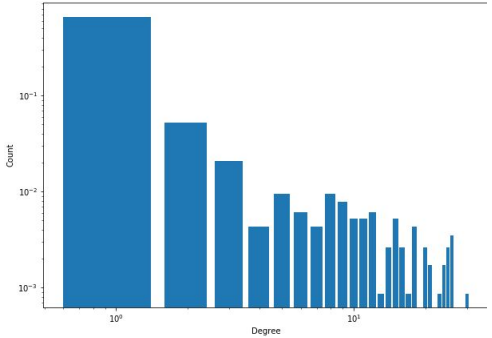
Sparsity of the original graph (left) and its pruned version (right)





Degree distributions (log-log scale) of the original graph (left) and its pruned version (right)





Exploration

NetworkX

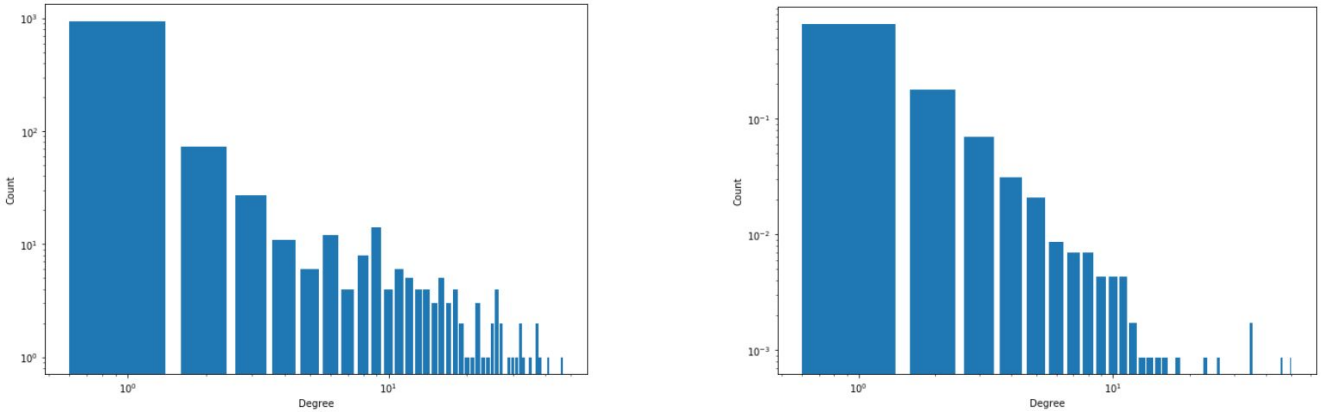
https://networkx.github.io/documentation/stable/

WordCloud

https://amueller.github.io/word_cloud/

- Graph Properties
  - Connected Components & Diameter
  - Sparsity & Degree distribution
- Type of Graph Identification

Exploration - Type of Graph Identification



Degree distributions (log-log scale) of the original network (left) and its best fit approximation that follows a powerlaw (right)

Exploration

NetworkX

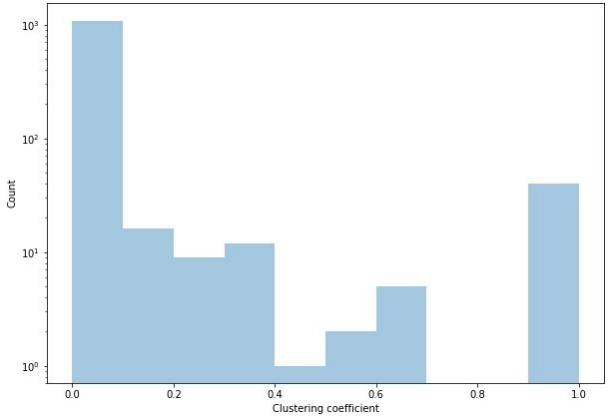
https://networkx.github.io/documentation/stable/

WordCloud

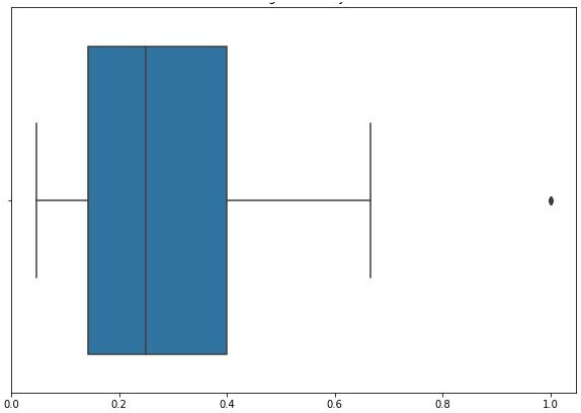
https://amueller.github.io/word_cloud/

- Graph Properties
  - Connected Components & Diameter
  - Sparsity & Degree distribution
- Type of Graph Identification
- Nodes Properties
  - Clustering Coefficients
  - Centrality

Exploration -Nodes Properties



Distribution of the clustering coefficient of nodes



Distribution of the average centrality among communities

Exploration

NetworkX

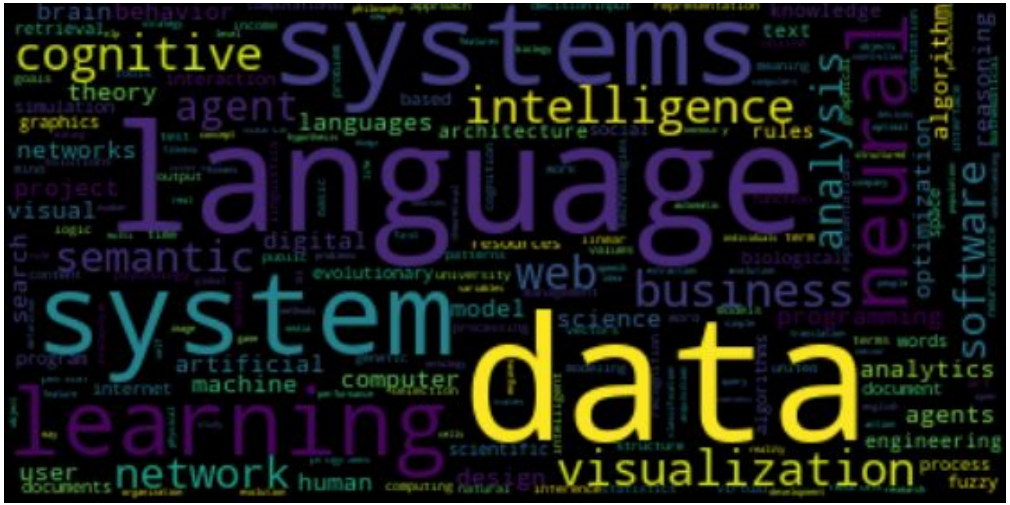
https://networkx.github.io/documentation/stable/

WordCloud

https://amueller.github.io/word_cloud/

- Graph Properties
  - Connected Components & Diameter
  - Sparsity & degree distribution
- Type of Graph Identification
- Nodes Properties
  - Clustering Coefficients
  - Centrality
- Analysis Of Nodes Attributes

Exploration – Nodes Attributes Analysis



WordCloud performed on nodes attributes
(keywords extracted from pages summaries)

Exploitation

fastText

https://arxiv.org/abs/1607.04606
https://fasttext.cc/docs/en/english-vectors.html

Node2Vec

https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

https://github.com/eliorc/node2vec

Models - Node Embeddings

- Spectral Clustering
- Node2Vec
- FastText walks

Exploitation

fastText

https://arxiv.org/abs/1607.04606
https://fasttext.cc/docs/en/english-vectors.html

Node2Vec

https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

https://github.com/eliorc/node2vec

Spectral Clustering

- We compute the eigenvectors u1, u2, ..., uk associated with the k smallest eigenvalues of the normalized laplacian of our graph.
- We set U_k = [u1|u2|...|uk] ∈ RN×k
- Embed the i-th node to the normalized i-th entry of U_k

Exploitation

fastText

https://arxiv.org/abs/1607.04606
https://fasttext.cc/docs/en/english-vectors.html

Node2Vec

https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

https://github.com/eliorc/node2vec

Node2Vec

- Random Walks
- Word2Vec

Exploitation

fastText

https://arxiv.org/abs/1607.04606
https://fasttext.cc/docs/en/english-vectors.html

Node2Vec

https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

https://github.com/eliorc/node2vec

fastText walks

Input:
An embedding map M. A node n_{source}.
A list of walks L starting from n_{source}.
Output: A new embedding for n_{source}.
buffer = empty_list()

for *walk in L* **do**
  source_contribution = source_weight $\times M(n_{source})$;
  walk_contribution = $\frac{1-source\_weight)}{length(walk)-1} \times \sum_{i=1}^{length(walk)} M(walk[i])$;
  new_embedding = source_contribution + walk_contribution;
  buffer.append(new_embedding);
end
return buffer.mean()

Algorithm 1: Walk averaged node embedding

Exploitation

fastText

https://arxiv.org/abs/1607.04606
https://fasttext.cc/docs/en/english-vectors.html

Node2Vec

https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

https://github.com/eliorc/node2vec

Recommendation algorithm

- Query embedding
- Recommendation

Results

- Quantitative
  - ~~Link Prediction~~
- Qualitative
  - Queries
  - Clustering

Result – Queries

Model	Result: cosine score
Node2Vec	machine learning: 0.99 machine learning in bioinformatics: 0.83 explanation-based learning: 0.77 one-shot learning: 0.75 model selection: 0.69 quantum machine learning: 0.69 quantum image: 0.68 hyperparameter optimization: 0.67 quantum annealing: 0.66 automated machine learning: 0.65
Spectral	machine learning: 0.99 machine learning in bioinformatics: 0.91 explanation-based learning: 0.83 one-shot learning: 0.55 automated machine learning: 0.42 gene expression programming: 0.37 quantum machine learning: 0.35 weak ai: 0.33 parallel distributed processing: 0.31 hyperparameter optimization: 0.31
FastText	automated machine learning: 0.86 machine learning: 0.86 quantum machine learning: 0.85 rule-based machine learning: 0.85 applications of machine learning: 0.84 machine learning in bioinformatics: 0.84 computer-assisted language learning: 0.82 never-ending language learning: 0.81 representation learning: 0.81 virtual world language learning: 0.81

Model	Result: cosine score
Node2Vec	natural language processing: 0.77 artificial intelligence: 0.74 1 the road: 0.67 spoken dialogue system: 0.65 truecasing: 0.65 philosophy of artificial intelligence: 0.64 printing press check: 0.63 computer-assisted reviewing: 0.62 foreign language writing aid: 0.62 natural language user interface: 0.62
Spectral	natural language processing: 0.72 artificial intelligence: 0.69 1 the road: 0.53 automated essay scoring: 0.53 biomedical text mining: 0.53 language and communication technologies: 0.53 language technology: 0.53 spoken dialogue system: 0.53 transformer (machine learning model): 0.53 truecasing: 0.53
FastText	natural language processing: 0.92 natural-language processing: 0.90 philosophy of artificial intelligence: 0.89 marketing and artificial intelligence: 0.88 natural computation: 0.88 artificial development: 0.87 computational models of language acquisition: 0.87 existential risk from artificial general intelligence: 0.87 personality computing: 0.87

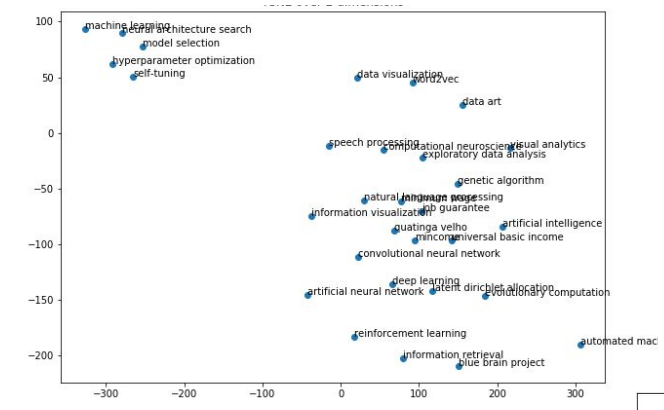
“artificial intelligence, natural language processing” (Top)

“Machine Learning” (Left)

Model	Result: cosine score
Node2Vec	None (not in the corpus)
Spectral	None (not in the corpus)
FastText	cash transfers: 0.74 redistribution of income and wealth: 0.67 revenue shortfall: 0.66 negative income tax: 0.65 universal credit: 0.65 fairtax: 0.64 post-scarcity economy: 0.64 consumer demand tests (animals): 0.63 guaranteed minimum income: 0.63 working time: 0.63

“Money money money”

Results – Clusters

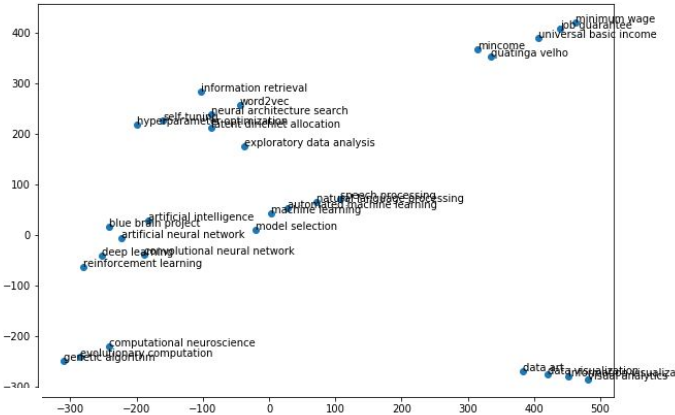


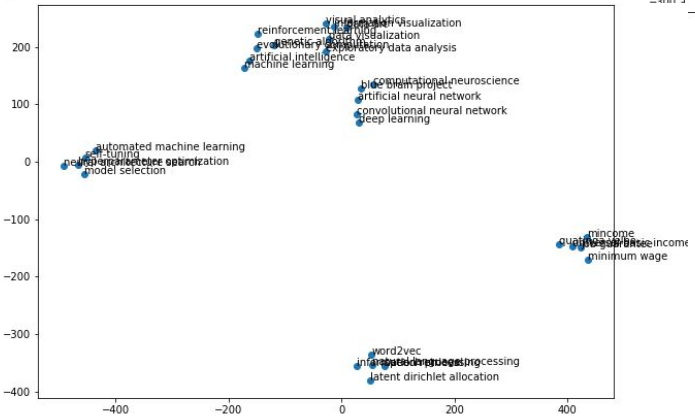
T-SNE Clusters

Spectral Clustering (Left)

FastText (Right)

Node2Vec (Bottom)





Limitations

- Graph acquisition
  - Insightful graph
- User-Friendliness
  - Out-of-Corpus
- Scalability
  - Laplacian Computation

Data Product

Interactive Data visualisation implemented with Dash by Plotly

Link: ec2-18-219-204-33.us-east-2.compute.amazonaws.com

Conclusion

- Powerful in extracting meaningful recommendations
  - Strength of the embeddings
- Exploitation of the graph structure
  - All methods
- Exploitation of the semantic content
  - FastText
- Improvable
  - more data to construct the attributes

Ressources

Seealsology

https://densitydesign.github.io/strumentalia-seealsology/

Wikipedia Python API

https://pypi.org/project/wikipedia/

fastText

https://arxiv.org/abs/1607.04606

https://fasttext.cc/docs/en/english-vectors.html

Node2Vec

https://cs.stanford.edu/~jure/pubs/node2vec-kdd16.pdf

https://github.com/eliorc/node2vec

Dash by Plotly

https://plot.ly/dash/