keep current

Graph Neural Networks

Liad Magen

~~March~~ October 2020

abacus.ac

keep current

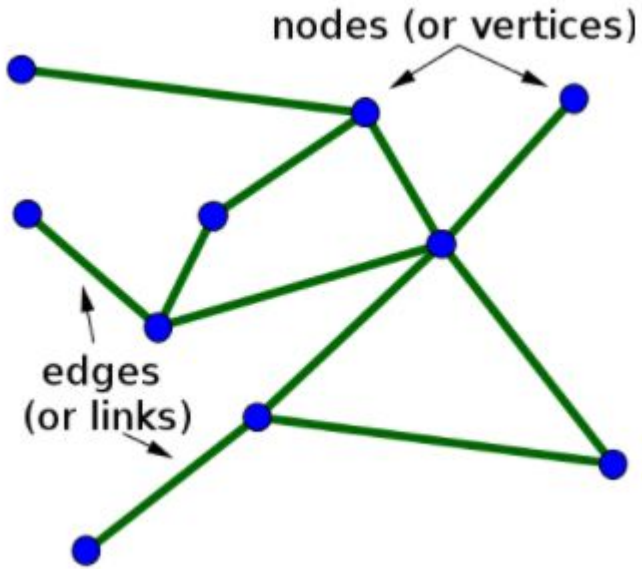
Hello, world!

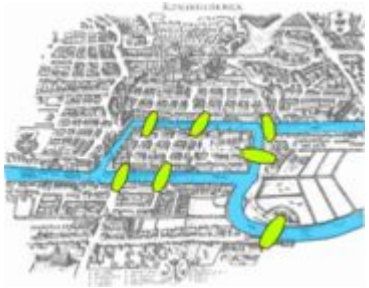
ASIGMO
turning data into value

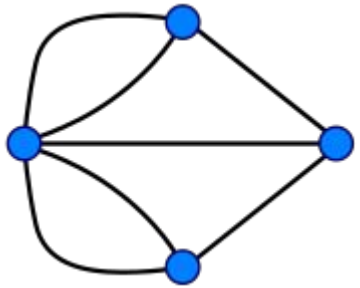
MeetPlans

Dr. Stefan Thurner

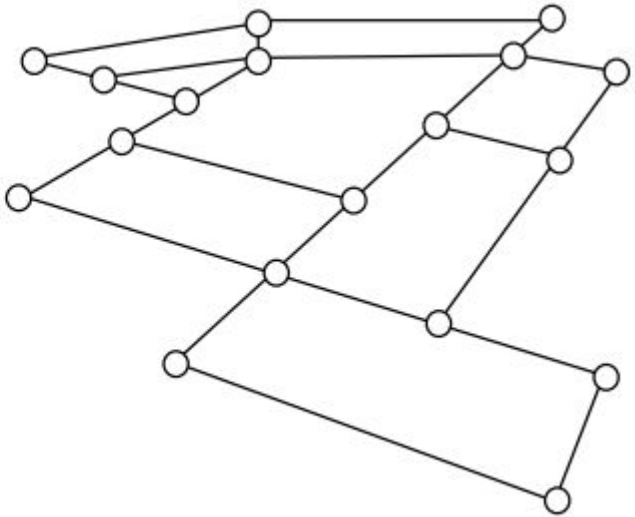
What are graphs?





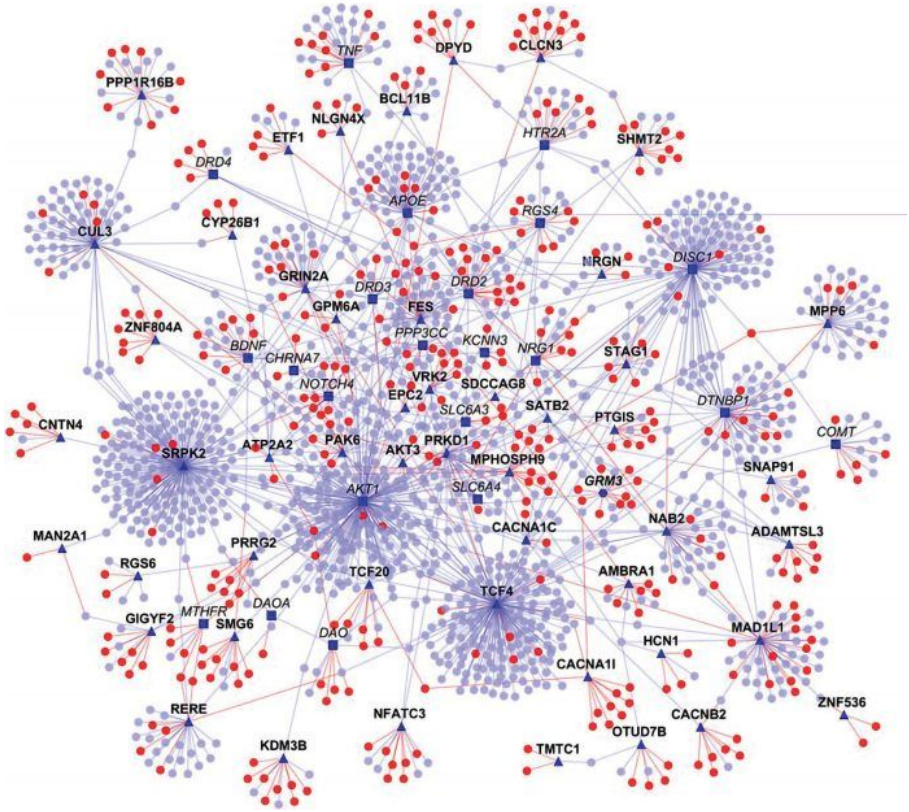


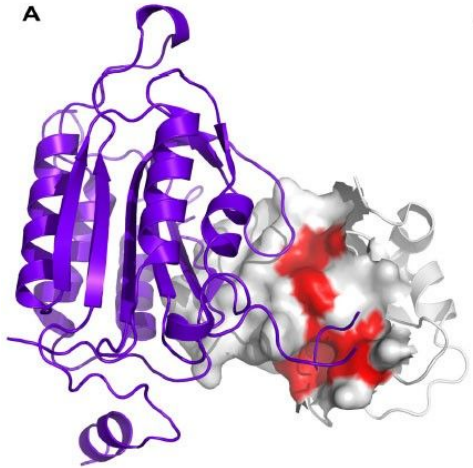
Seven Bridges of Königsberg

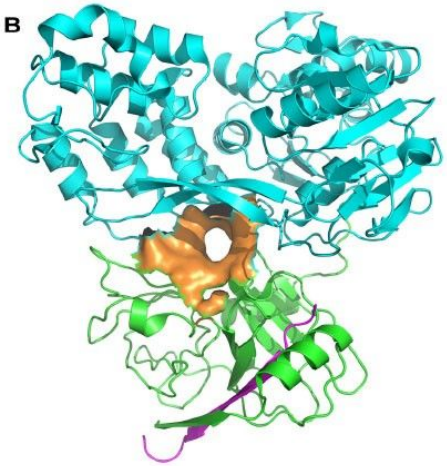


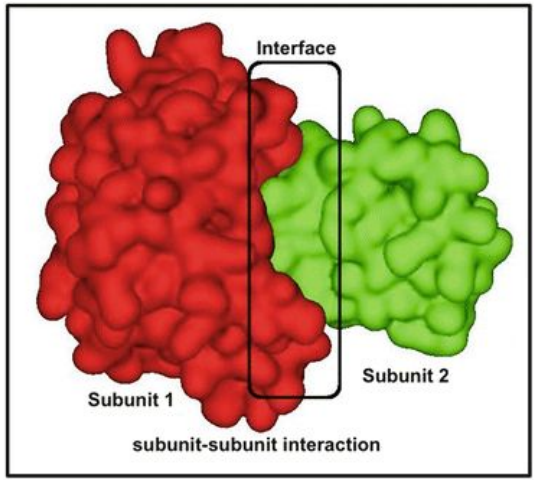
Location & Street representation

Social Network

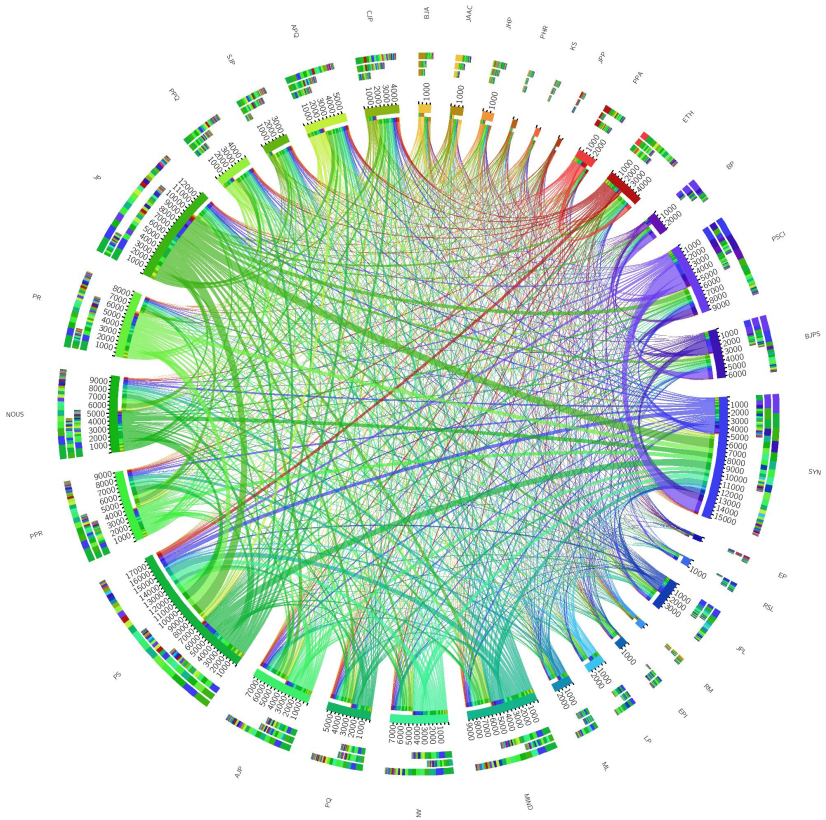




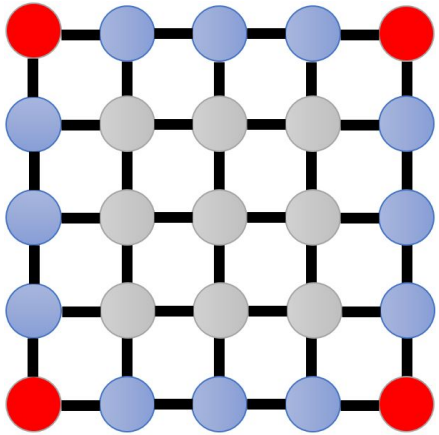


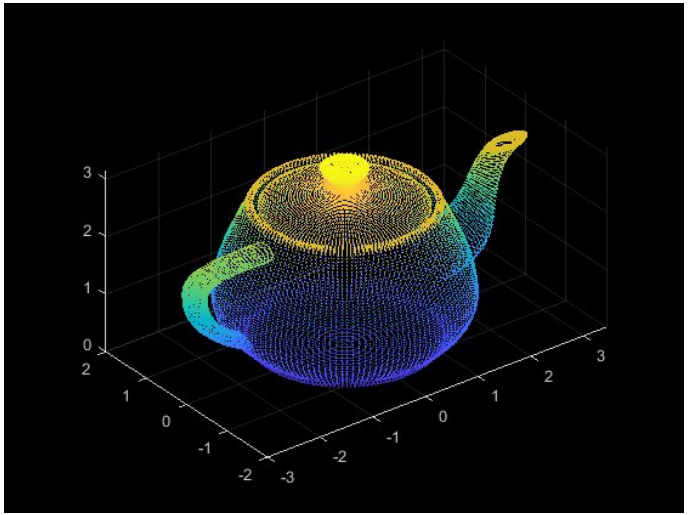


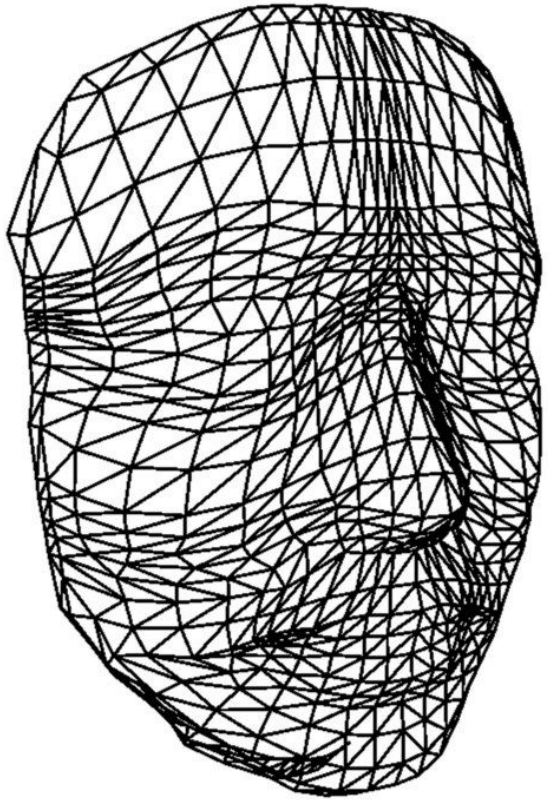
Protein-Protein interactions



Paper citations as a graph

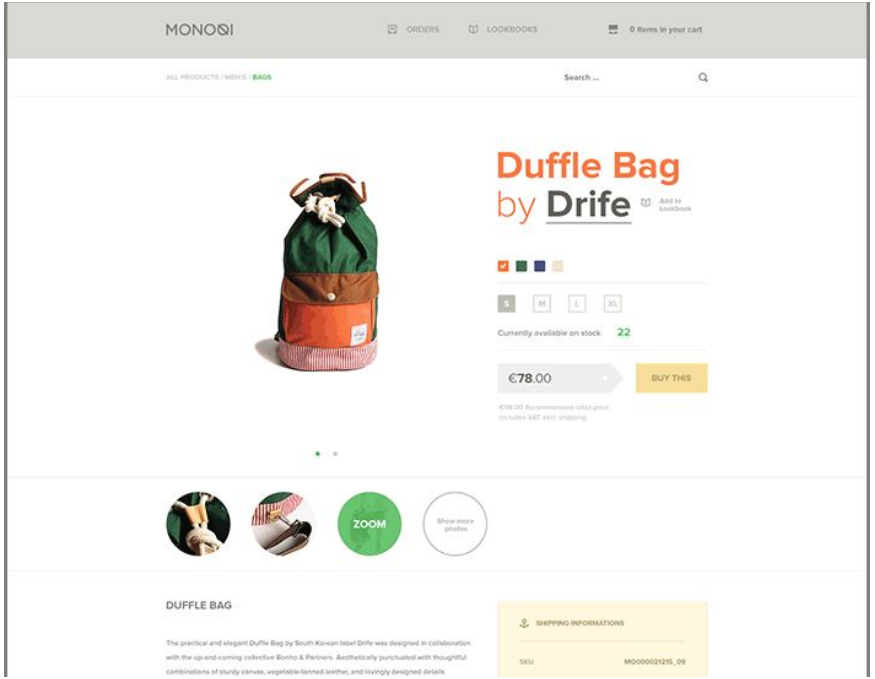




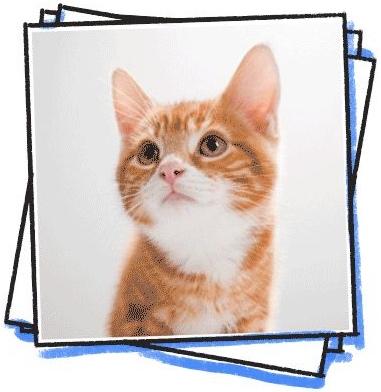


Mesh & Point cloud as a graph

Visually Rich Data

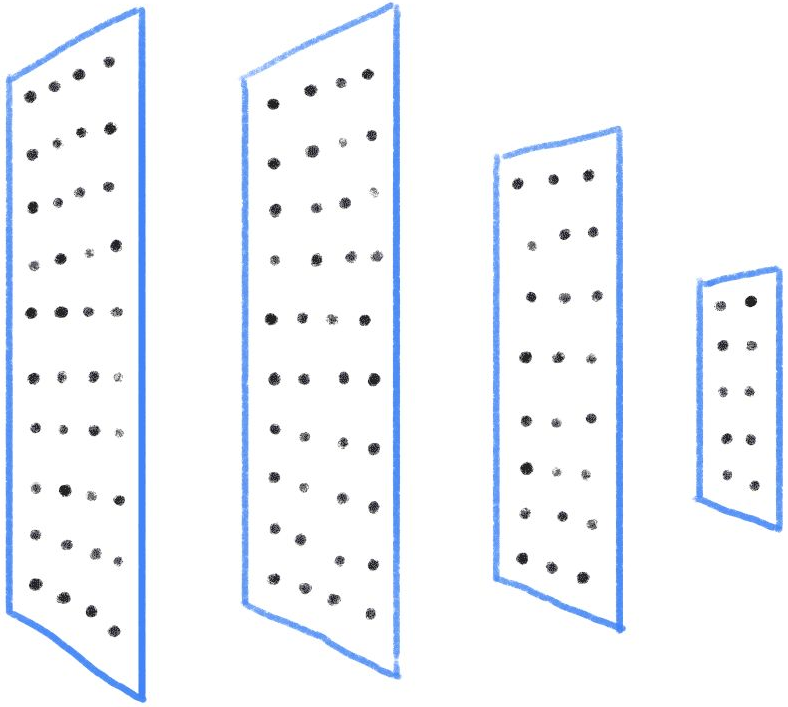


CAT



(LABELED PHOTOS)

DOG



OUTPUT

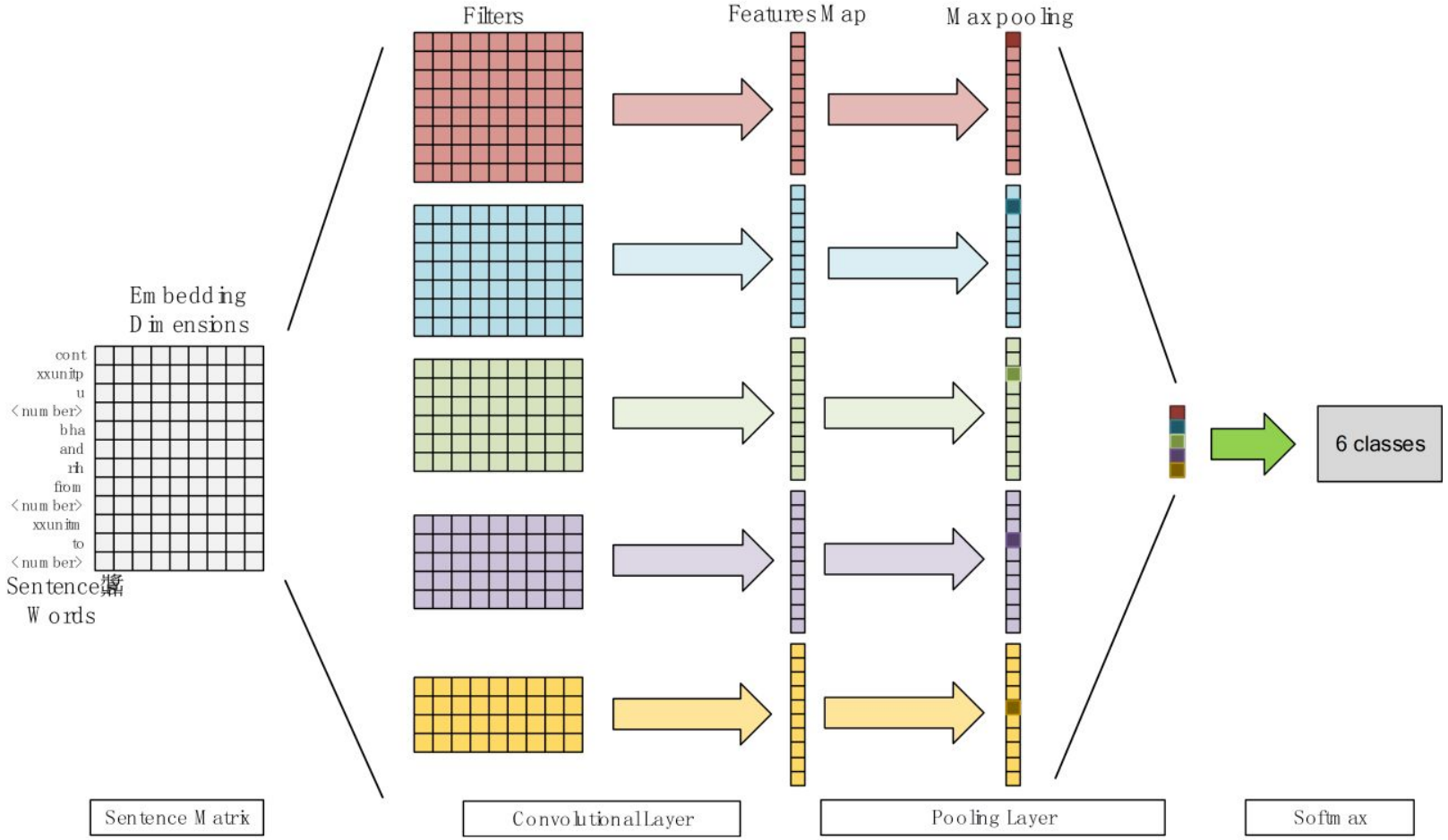
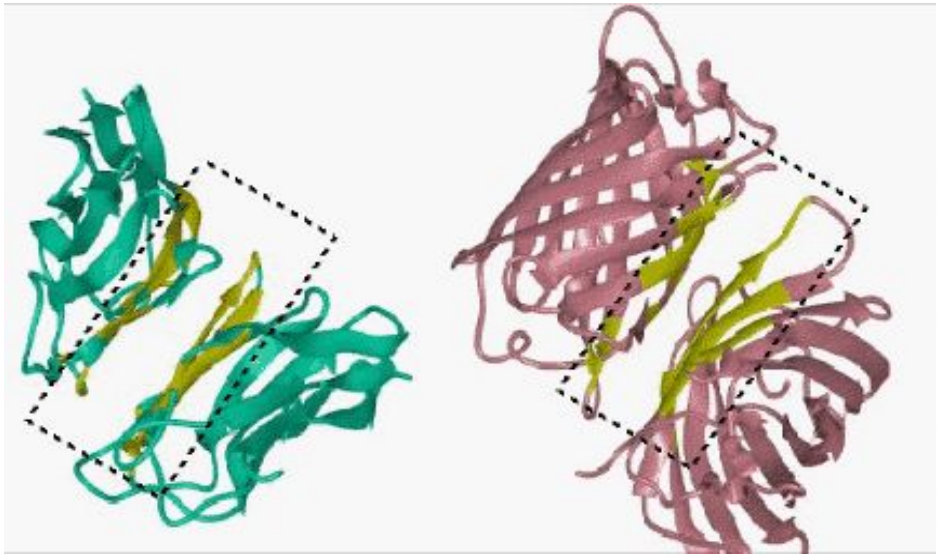


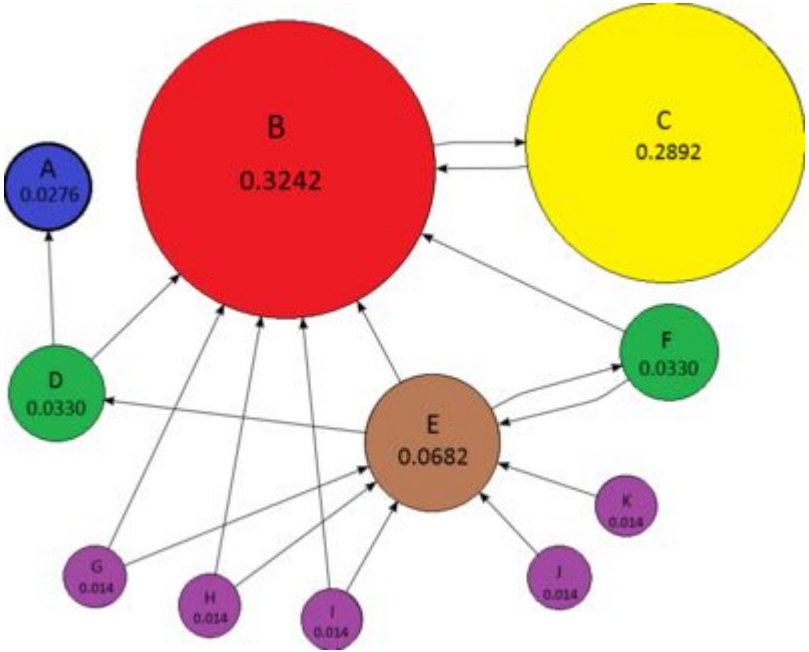
Figure 2. CNN-based text classification network.

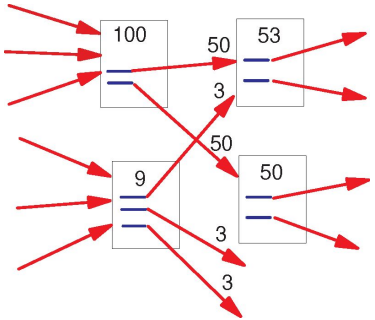
Process other (non-Euclidean) formats of data



Google PageRank

$$PR_i = \frac{1-d}{n} + d \sum_{j=1}^{n} \frac{PR_j}{c_j}$$





Inspired usages

$$PR_i = \frac{1-d}{n} + d \sum_{j=1}^{n} \frac{PR_j}{c_j}$$

PLoS One. 2015; 10(8): e0134794.
Published online 2015 Aug 19. doi:

OPEN ACCESS Freely available online

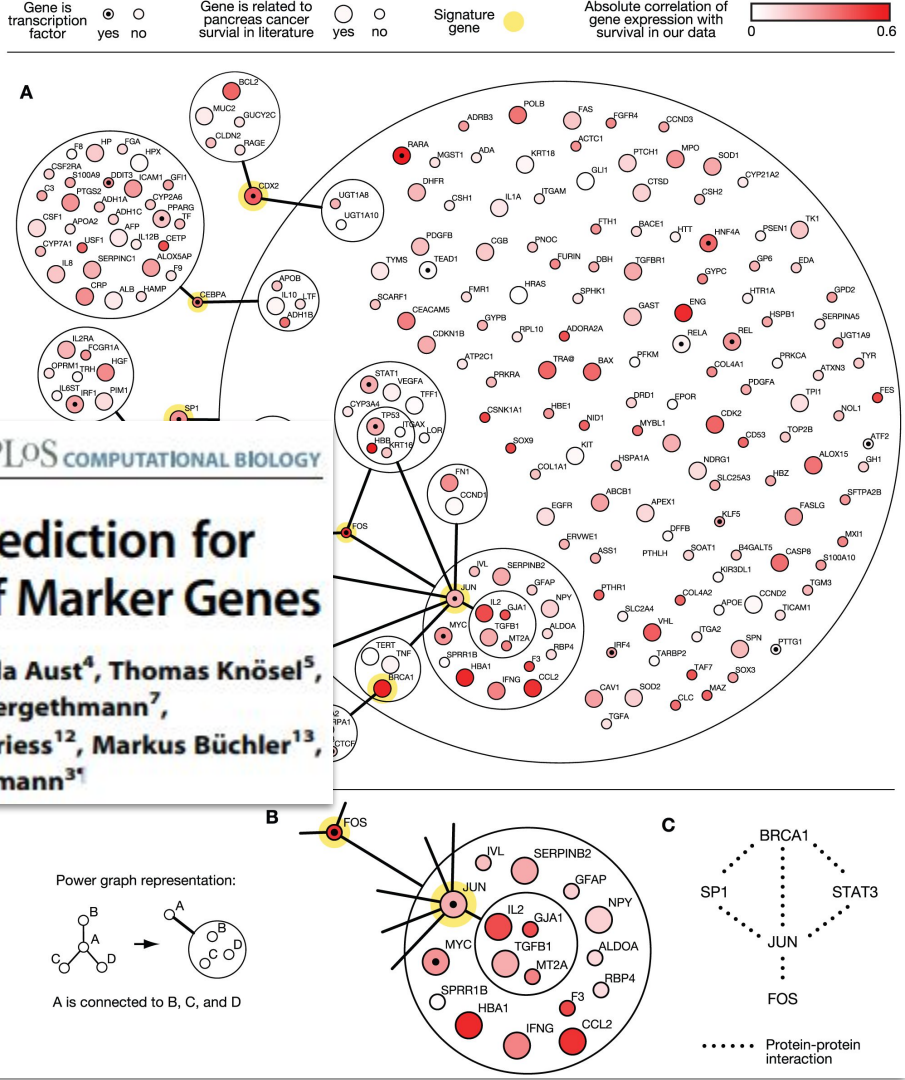
PLoS COMPUTATIONAL BIOLOGY

Google Goes Cancer: Improving Outcome Prediction for Cancer Patients by Network-Based Ranking of Marker Genes

Christof Winter[1*9], Glen Kristiansen[2,9], Stephan Kersting[3,9], Janine Roy[1], Daniela Aust[4], Thomas Knösel[5], Petra Rümmele[6], Beatrix Jahnke[3], Vera Hentrich[3], Felix Rückert[3], Marco Niedergethmann[7], Wilko Weichert[8], Marcus Bahra[9], Hans J. Schlitt[10], Utz Settmacher[11], Helmut Friess[12], Markus Büchler[13], Hans-Detlev Saeger[3], Michael Schroeder[1¶], Christian Pilarsky[3¶], Robert Grützmann[3¶]

Page

Ying Ding[1], Erjia Yan, Arthu

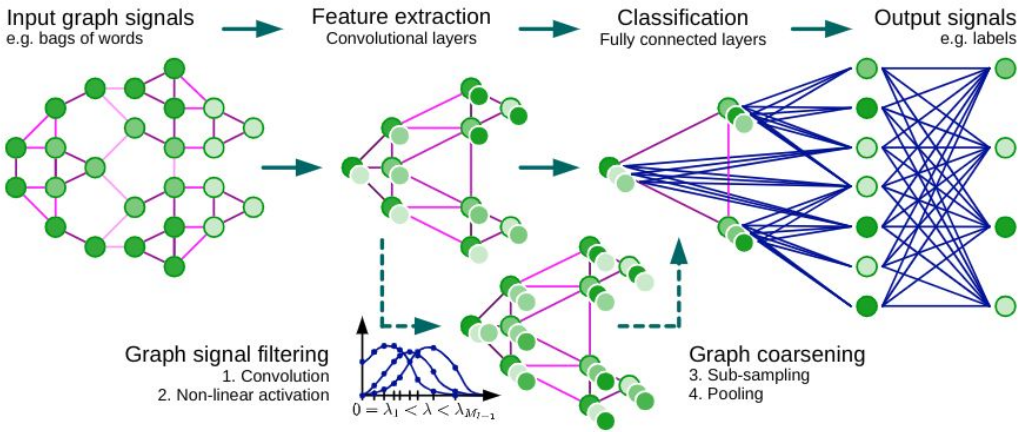


Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering

Michaël Defferrard **Xavier Bresson** **Pierre Vandergheynst**

EPFL, Lausanne, Switzerland

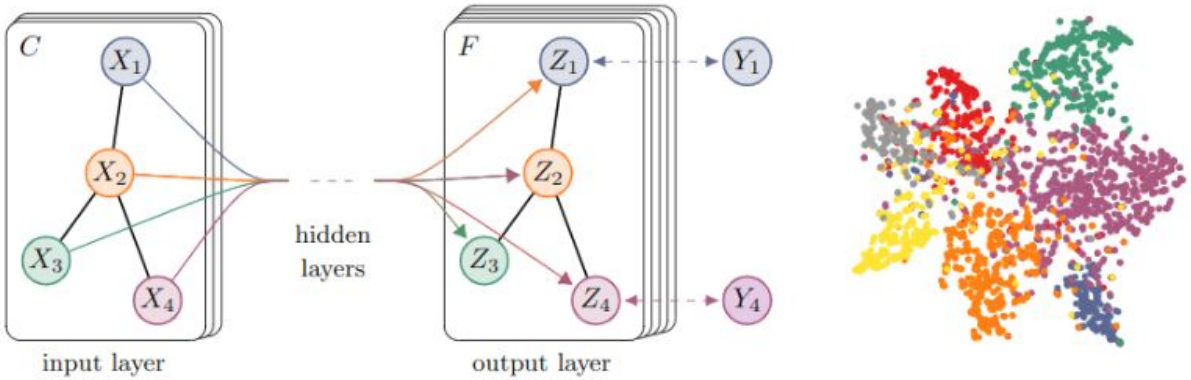
{michael.defferrard,xavier.bresson,pierre.vandergheynst}@epfl.ch



SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

Thomas N. Kipf
University of Amsterdam
T.N.Kipf@uva.nl

Max Welling
University of Amsterdam
Canadian Institute for Advanced Research (CIFAR)
M.Welling@uva.nl

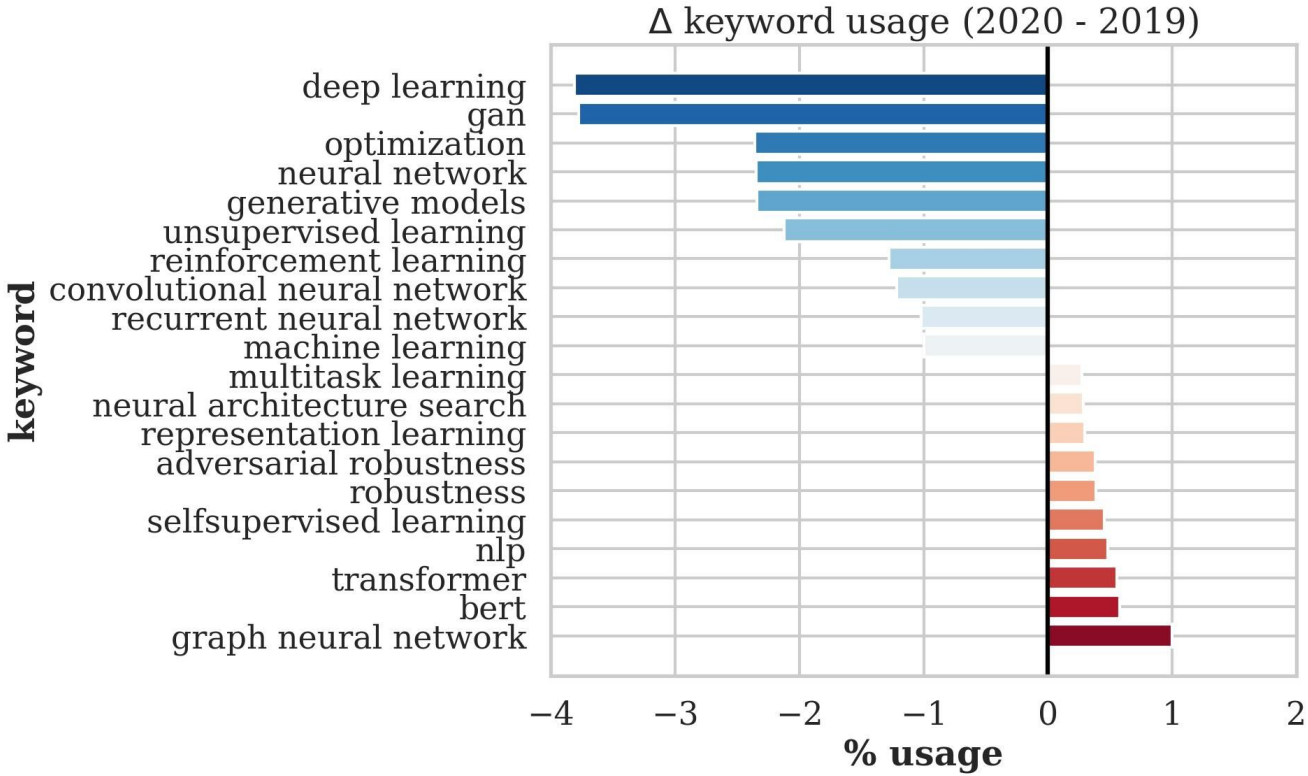


(a) Graph Convolutional Network

(b) Hidden layer activations

Figure 1: *Left*: Schematic depiction of multi-layer Graph Convolutional Network (GCN) for semi-

Graph Neural Networks



ICLR Submissions

Source:

Twitter - @prlz77

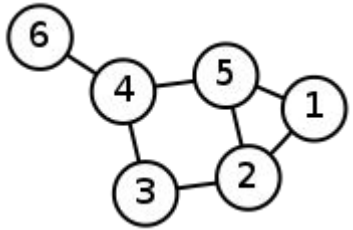
Back to basics

Graphs, Mathematically

G = (**V**etrices, **E**dges)

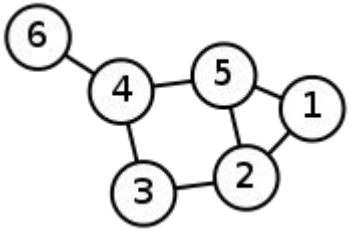
V = {v1 … vn)

E = {(v_i, v_j), … }



Graph - algebraic representation

G = (**V**etrices, **E**dges)



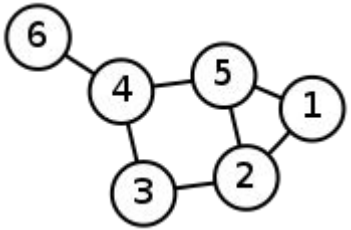
Adjacency matrix

A_{ij}
- 1 - if $\{v_i, v_j\} \in \mathbf{E}$ && i != j
- 0 - Otherwise

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Graph - algebraic representation

G = (**V**etrices, **E**dges)



Laplacian matrix *

L = D-A

$$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$$

Degree — Adjacency — **Laplacian**

* For undirected graphs

GCN - Algorithm

GCN

GCN Layer:

$$Y = \mathrm{ReLU}(AXW)$$

...but with message passing

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$$

Message Passing

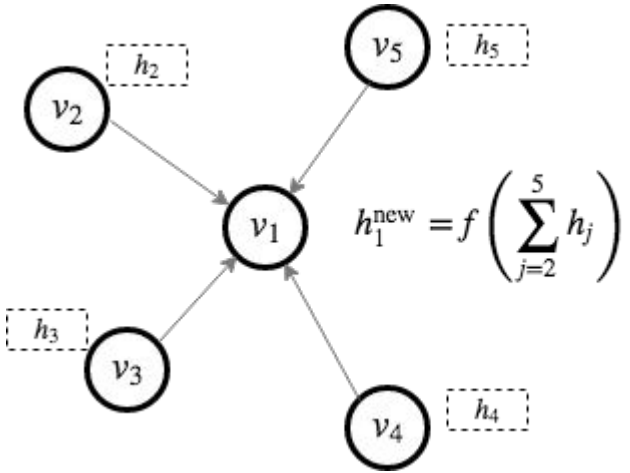
For each node u:

1. Aggregate neighbor nodes h_v into an intermediate representation $\hat{h}_u$
2. Transform the aggregated representation $\hat{h}_u$ with a linear projection followed by a non-linearity (ReLU) $h_u = f(W_u \hat{h}_u)$

Mathematically:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$$

H →Network Layer
W→Network Weights
A →Adjacency Matrix
D →Degree Matrix



Model types

- Graph classification
  - Chemical properties of a molecule
  - Comparing user preferences / activities
- Node classification - node label prediction
  - Malicious users in a social network
  - Visually inferred Named Entity Recognition (NER)
  - Node clustering
- Edge prediction
  - Recommendation system
  - Protein-protein interaction
  - “Friend” suggestion

Frameworks

NetworkX

- Store and mutate Graphs
- Graph algorithms (Shortest path - Dijkstra, TreeWidth, clustering, centrality)
- Network analysis
- Node / edge data
- Visualization tools

DGL

DeepGraphLibrary

- Building blocks
- Great tutorials
- Generative graphs
- Great for research and complicated tasks

PyTorch Geometric

PyTorch geometric

- An extension library for pyTorch
- Officially part of the pyTorch ecosystem
- Easily extensible
- Papers are implemented directly in it
- Looooooooooooooong list of ready-to-use methods and algorithms:
  - TransformerConf (2020)
  - GCN2Conv (2020)
  - DeeperGCN (2020)
  - Top-K Pooling
  - PairNorm

Model types

- Graph classification
  - Chemical properties of a molecule
  - Comparing user preferences / activities
- Node classification - node label prediction
  - Malicious users in a social network
  - Visually inferred Named Entity Recognition (NER)
  - Node clustering
- Edge prediction
  - Recommendation system
  - Protein-protein interaction
  - “Friend” suggestion

Practical advices

PyTorch Geometric Example

- DataLoader
- GCN Layers

Potential Pitfalls when going ‘deep’

- Vanishing Gradient
- Overfitting
- **Over-smoothing**
  Node-vectors become too similar
- **Bottleneck (Over-Squashing)**
  A single node vector contains data of too many nodes

Model	2-Layer	4-Layer	8-Layer	16-Layer	32-Layer	64-Layer
GCN-res	88.18±1.59	86.50±1.87	84.83±1.93	78.60±4.28	59.82±7.74	39.71±5.15
PairNorm	79.98±3.80	82.32±2.79	81.52±3.66	82.29±2.62	81.91±2.45	81.72±2.82
NodeNorm	**89.53**±1.29	88.60±1.36	88.02±1.67	88.41±1.25	88.30±1.30	87.40±2.06

https://arxiv.org/abs/2006.07107 - Effective Training Strategies for Deep Graph Neural Networks

Scaling

- GraphSAGE
  https://arxiv.org/abs/1706.02216
- ClusterGCN
  https://arxiv.org/abs/1905.07953
- Graph-BERT
  https://arxiv.org/abs/2001.05140
  https://github.com/jwzhanggy/Graph-Bert

Additional reading:
https://towardsdatascience.com/simple-scalable-graph-neural-networks-7eb04f366d07

Model Evaluation

OPEN GRAPH BENCHMARK

https://ogb.stanford.edu/

- Datasets + Data Loaders (DGL / pyTorch Geometric)
- Node pred
- Edge pred
- Graph pred

Please feel free to ask questions ;)

Additional Resources

https://towardsdatascience.com/how-to-do-deep-learning-on-graphs-with-graph-convolutional-networks-7d2250723780

https://tkipf.github.io/graph-convolutional-networks/

https://github.com/deepmind/graph_nets

https://www.scihive.org/paper/1912.12693#section-13

https://link.springer.com/article/10.1186/s40649-019-0069-y

https://www.youtube.com/watch?v=cWIeTMklzNg

https://www.youtube.com/watch?v=YrhBZUtgG4E