SEMI-SUPERVISED CLASSIFICATION WITH GRAPH CONVOLUTIONAL NETWORKS

NRU HSE

FCS AMI, 161

Maxim Kobelev

Plan:

- Semi-supervised on Graphs?
- What is it - Graph Embeddings
- Graph convolutions samples
- Approximate methods
- Model architecture & learning process
- Experiments & Results
- Conclusion

Semi-supervised??

- Classifying nodes (such as documents) in a graph (such as citation networks)
- Labels are only available for a small subset of nodes.
- Single model
- Weighted sum of two losses: supervised & unsupervised.

History for last 4 years:

- Heuirstics, defining environment for graph vertex
- Learning embeddings from local neighborhood prediction via random walk on graph. PageRank (?)
- Spectral methods
- Neural Networks
- (nowadays) Combine!

Old heuristics

- Number of common neighbors for vertexes

- It’s clear that as larger the intersection, as larger probability of edge between these two vertexes.

  In real life it’s happened like mostly new friends comes from common friends or friends’ friends.

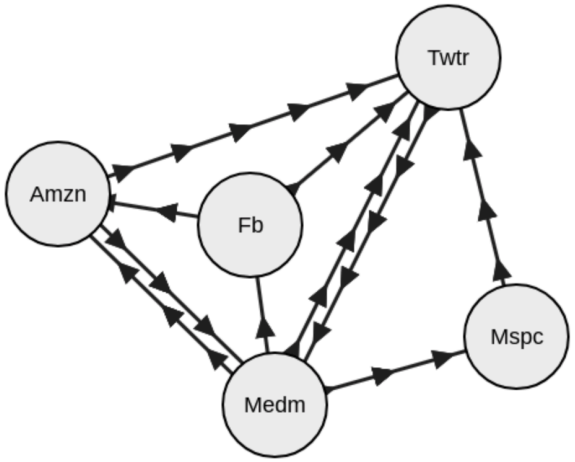
History for last 4 years:

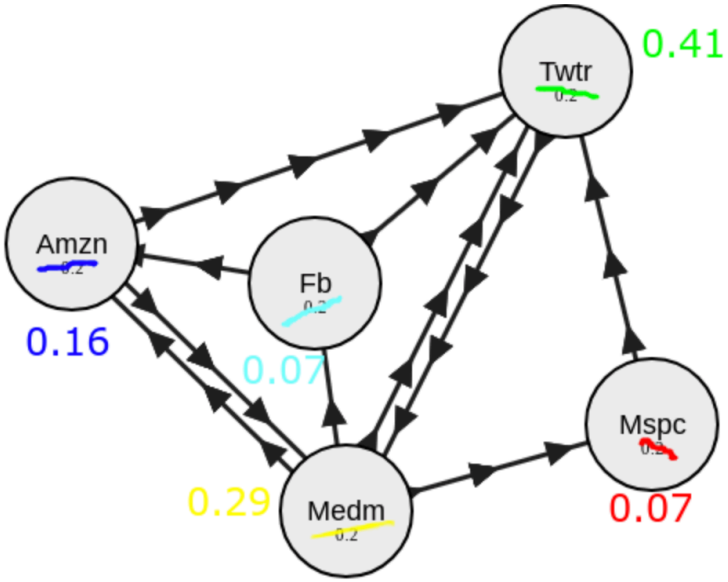
- Heuirstics, defining environment for graph vertex
- Learning embeddings from local neighborhood prediction via random walk on graph. PageRank (?)
- Spectral methods
- Neural Networks
- (nowadays) Combine!

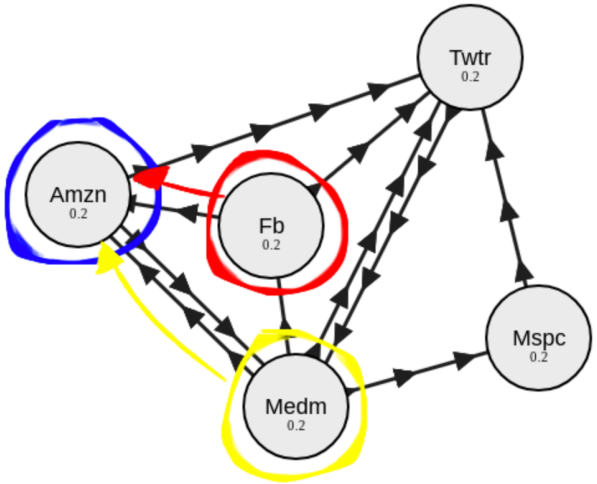
random walks on graph. PageRank

$$PR(p_i) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)}$$

M – input, L - output







NN + Spectral approaches

Graph-based semi-supervised learning

- Label information is smoothed over the graph

$$\mathcal{L} = \mathcal{L}_0 + \lambda \mathcal{L}_{\text{reg}} ,$$

$$\mathcal{L}_{\text{reg}} = \sum_{i,j} A_{ij} \| f(X_i) - f(X_j) \|^2 = f(X)^\top \Delta f(X) .$$

an adjacency matrix $A \in \mathbb{R}^{N \times N}$ (binary or weighted)

degree matrix $D_{ii} = \sum_j A_{ij}$

$\Delta = D - A$ (unnormalized graph Laplacian)

How can we learn it

h_v^t Vertex status at t

e_{vw}^t Edge status at t

$m_v^{t+1} = (\Sigma h_w^t, \Sigma e_{vw}^t)$ Message between v – w, (*,*) - concat

$H_t^{deg(v)}$ Parameters matrix for each t-step, for each deg(v)

$h_v^{t+1} = \sigma\left(H_t^{deg(v)} m_v^{t+1}\right)$ Refresh vertex status

$$R = \mathrm{f}\left(\sum_{v,t} softmax\left(W_t h_v^t\right)\right)$$

f – Neural Network, W – Output Matrix

Combined method:

- Let we define graph convolution as:

$$g_\theta \star x = U g_\theta U^\top x\,, \qquad \text{filter } g_\theta = \mathrm{diag}(\theta) \text{ parameterized by } \theta \in \mathbb{R}^N$$

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U \Lambda U^\top$$

$U^\top x$ being the graph Fourier transform

g_θ as a function of the eigenvalues of L, i.e. $g_\theta(\Lambda)$

$$g_{\theta'}(\Lambda) \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{\Lambda})\,, \quad \text{rescaled } \tilde{\Lambda} = \frac{2}{\lambda_{\max}}\Lambda - I_N.$$

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$$

$$T_0(x) = 1$$

$$T_1(x) = x$$

Layer—wise approach:

- Ok, now let's imagine that we limited the layer-wise convolution operation to K=1.
- Still recover a rich class of convolutional filter functions by stacking multiple such layers.
- This technique prevents overfitting.

$$g_{\theta'} \star x \approx \theta'_0 x + \theta'_1 \left(L - I_N\right) x = \theta'_0 x - \theta'_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x \,,$$

$$g_\theta \star x \approx \theta \left(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}}\right) x$$

$$I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$$ Renormalization trick

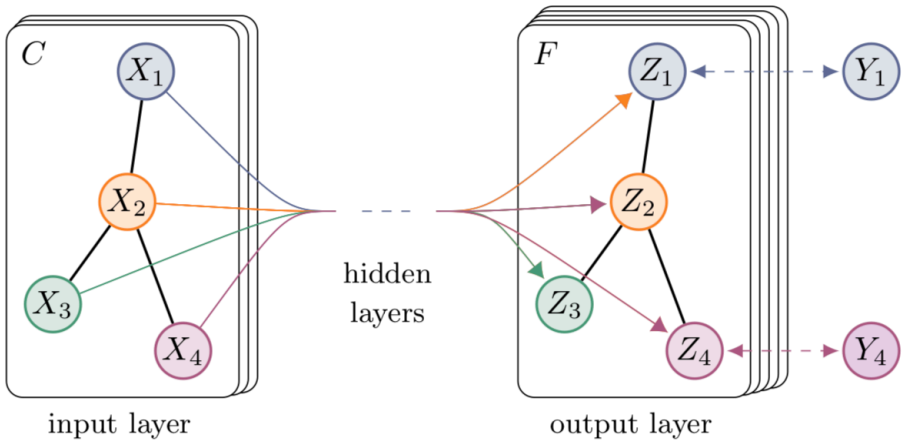
$$Z = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta$$

$$X \in \mathbb{R}^{N \times C} \quad \Theta \in \mathbb{R}^{C \times F} \quad Z \in \mathbb{R}^{N \times F}$$ Convolved signal matrix

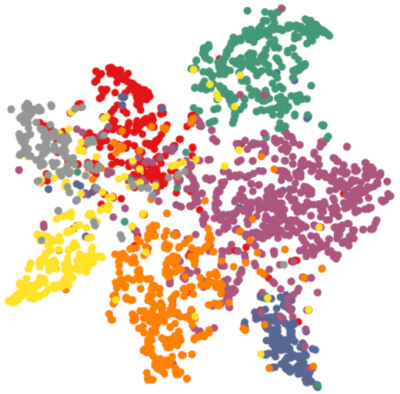
Model & Learning:

$$\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$$

$$Z = f(X, A) = \mathrm{softmax}\Big(\hat{A}\ \mathrm{ReLU}\Big(\hat{A}XW^{(0)}\Big)\,W^{(1)}\Big)\ .$$



(a) Graph Convolutional Network



(b) Hidden layer activations

- (a) – GCN schematic, with C input channels and F feature maps in output layer
- (b) – t-SNE visualization of hidden layer activations of two-layer GCN trained on Cora dataset using 5% of labels. Colors denote document class.

Results:

Table 1: Dataset statistics, as reported in Yang et al. (2016).

Dataset	Type	Nodes	Edges	Classes	Features	Label rate
Citeseer	Citation network	3,327	4,732	6	3,703	0.036
Cora	Citation network	2,708	5,429	7	1,433	0.052
Pubmed	Citation network	19,717	44,338	3	500	0.003

Contains sparce bag-of-words feature vectors for each document and a list of citation links between documents. Citation link is an edge, document is a vertex.

Results:

Table 2: Summary of results in terms of classification accuracy (in percent).

Method	Citeseer	Cora	Pubmed
ManiReg [3]	60.1	59.5	70.7
SemiEmb [28]	59.6	59.0	71.1
LP [32]	45.3	68.0	63.0
DeepWalk [22]	43.2	67.2	65.3
ICA [18]	69.1	75.1	73.9
Planetoid* [29]	64.7 (26s)	75.7 (13s)	77.2 (25s)
GCN (this paper)	**70.3** (7s)	**81.5** (4s)	**79.0** (38s)
GCN (rand. splits)	67.9 ± 0.5	80.1 ± 0.5	78.9 ± 0.7

- Manifold Regularization (Belkin et al., 2006)
- Semi-Supervised embedding (Weston et al., 2012)
- Label Propagation (Yang et al., 2016)
- Iterative Classification Algorithm (Lu & Gethor, 2003)
- Planetoid (Yang et al., 2016)

Conclusions:

- При работе с графовыми данными – очень часто применяется semi-supervised learning
- Существует два продвинутых способа представления и работы с графовыми данными: трансформация эмбеддингами и регуляризация лапласиана.
- Вместо одного слоя с K-order polynomials Лапласиана без потерь качества используем стакнутые слои полиномов степени 1, тем самым настраивая свой фильтр.
- Архитектура прекрасно справляется с классификацией графов, основанных на текстовых данных (напр. научных работ, соц.сетей).