Topics in Pattern Recognition

All2Vec

Instructor

Prof. M Narsimha Murthy

Candidate

Lokesh N
Aayush Moroney

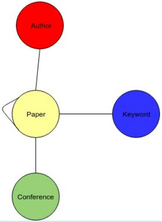
April 13, 2018

What is Embedding???

- Embedding refers to low dimensional representation of an entity, may it be word,node in a graph etc.
- Network Embedding is a method of learning low dimensional representation of individual nodes of a given network using the network structure and the available domain knowledge about the network.
- The basic aim in learning network embedding is to ensure that embedding of nodes in close proximity should be similar.

Heterogeneous Information Network

- An Information network defined as a directed graph G(V,E) with an object type mapping function $\tau : V \rightarrow O$ and a link type mapping function $\phi : E \rightarrow R$, where each object $v \in V$ belongs to one particular object type $\tau(v) \in O$, each link $e \in E$ belongs to a particular relation $\phi(e) \in R$. When the type of objects $|O| > 1$ or the type of relations $|R| > 1$, then the network is called Heterogeneous Information Network.



Some Embedding learning methods for Homogeneous Network

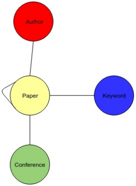
- **Adjacency Matrix** - Here dimensions are equal to number of nodes.
- **DeepWalk** -Make use of a random walk over graph to find neighbours of a node and then uses word2vec to learn embedding.
- **Node2Vec** - Make use of biased Random walk which oscillates between BFS and DFS to capture network neighbourhood, and finally uses word2vec to learn embeddings.
- **LINE** - Calculates 1^{st} and 2^{nd} order proximity of a node to learn network embedding.

Task Guided and Path Augmented Heterogeneous Network Embedding for Author Identification
T Chen and Y Sun

Paper Introduction

- This paper discusses about identifying author of a anonymous paper in a double blind review setting.
- Similarity between nodes in the graph is defined by the cosine similarity between their embeddings.
- Loss function is defined in such a way that,if the embedding of all the nodes are correctly captured respecting all the network structure and semantics, the cosine similarity would reveal the true authors.

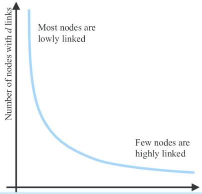
Task Guided Embedding learning



- Paper embedding is weighted average of average of all the neighbours.
- Other embeddings are as learnt by the algorithm.
- $V_p = \sum_t w_t V_p^{(t)}$
- $V_p^{(t)} = \sum_{n \in X_p^{(t)}} \frac{u_n}{|X_p^t|}$

Negative Sampling

- It is a method of obtaining sample of nodes from a graph for optimization purposes because considering all nodes in a graph for each optimization iteration may cost huge amount of time.
- Nodes are sampled based on Power Law Degree Distribution.
- Power Law states that fraction P(k) of nodes having degree k goes for large value of k as, P(k) $\sim$ k(-γ) , where γ is a parameter whose value lies in the range 2 to 3.



Task Guided Embedding

- Task is to learn embeddings such that real authors of paper should have highest similarity with the paper.
- Sample - (p,a,$a^{'}$), p is sampled paper, a is true author of p and $a^{'}$ is negative sampled author.
- Here objective is to make p more similar to a than $a^{'}$.
- Natural choice of loss function is Hinge Loss.
- Loss $= max(0, p.a^{'} - p.a + \zeta)$

Path Augmentation

- Along with task specific loss, they have also considered a general purpose loss function which tries to predict the link between two nodes.
- Given that Graph is Heterogeneous, they have utilized the rich semantics by using different meta-paths to define network neighbourhood. Meta-Paths are selected based on the task at hand.
- Exploring all the metapaths is a combinatorially difficult problem.
- Assuming some fixed meta-paths through some prior domain knowledge the way they learn the embeddings is by defining a convex combination of task-specific and network general loss functions.
- Loss $= \omega_1 * L_{task-guided} + \omega_2 * L_{path-augmented}$.

Moving to our Approach

Problems with path Augmentation Approach

- Till date there are no specific approaches to select optimal meta-path for a task.
- No set of predefined meta-paths would generalize to all tasks/datasets.
- Thatswhy we decided not to use meta-path based approach.
- Removing meta-path based component from baseline paper would not deliver promising results.

Our Approach

- Our's is a network general objective.
- We try to learn the importance of each relation along with the embedding.
- Every relation has some significance and it is very unlikely to be uniform.
- Hence we have to define mechanisms to learn the importance adaptively according to network structure, data sparsity and other relevant attributes.
- Therefore it is intuitively appealing to incorporate this importance somehow in the loss function so that gradient flow takes care of this by itself.

Our loss function is so big that it deserves a single slide of attention.

L$=\alpha_{PA} max(0, P_T A_T - P_T A_{NP}) + \alpha_{PC} max(0, P_T C_T - P_T C_{NP}) + \alpha_{PK} max(0, P_T K_T - P_T K_{NP}) + \alpha_{AP} max(0, A_T P_T - A_T P_{NA}) + \alpha_{KP} max(0, K_T P_T - K_T P_{NK}) + \alpha_{CP} max(0, C_T P_T - C_T P_{NC}) + \alpha_{PR} max(0, P_T R_T - P_T R_{NP})$

- $\underset{\alpha_{ij}, U_n}{min}$ L.
- Our Tuple for this loss function will be $(P_t, A_t, A_{np}, C_t, K_t, K_{np}, R_t, R_{np}, P_{na}, P_{nc}, P_{nk})$.

Gradient Calculations

$$\frac{\partial L}{\partial \alpha_{pa}} = max(0, P_T A_T - P_T A_{NP})$$

$$\frac{\partial L}{\partial P_T} = \alpha_{pa} * (A_T - A_{NP}) + \alpha_{pc} * (C_T - C_{NP}) + \alpha_{pk} * (K_T - K_{NP})$$
$$+\alpha_{pr} * (R_T - R_{NP}) + \alpha_{ap} * A_T + \alpha_{kp} * K_T + \alpha_{cp} * C_T$$

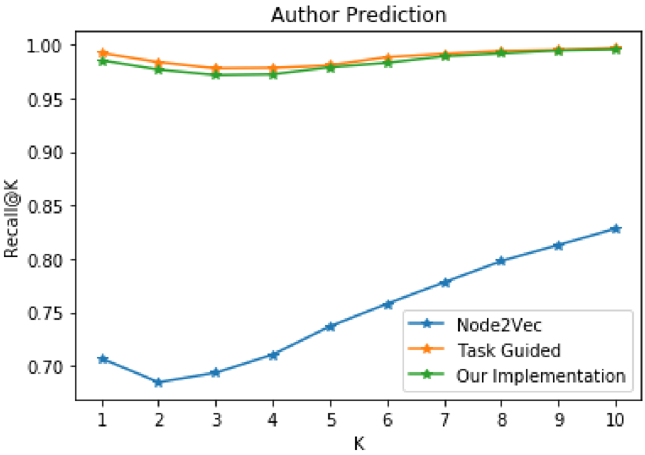
New Results after Mid Term

1. Added a regulariizer in the loss function ||term||2

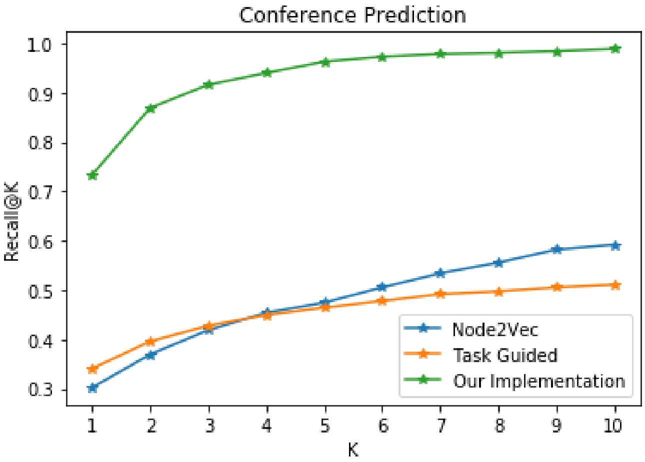
2. Added a constraint max(0, 1-alpha_ij) to ensure that alphas are always positive

3. Got the results for dblp dataset and compared the results against node2vec and the results are positive.

Results

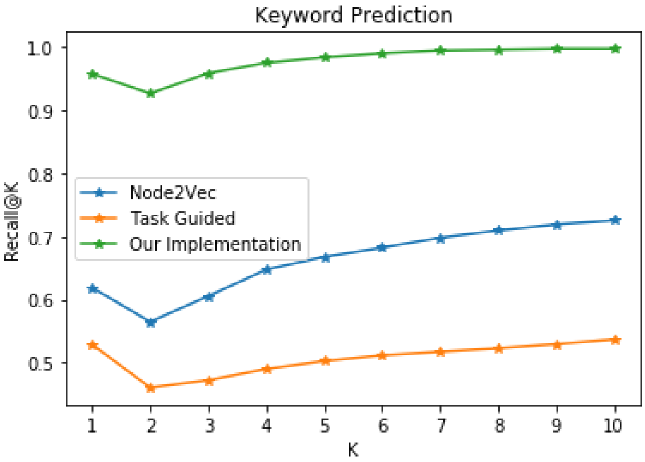

Author Prediction
Recall@K
K
1.00
0.95
0.90
0.85
0.80
0.75
0.70
1
2
3
4
5
6
7
8
9
10
Node2Vec
Task Guided
Our Implementation

Results



Conference Prediction
Recall@K
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
1
2
3
4
5
6
7
8
9
10
K
Node2Vec
Task Guided
Our Implementation

Results



Keyword Prediction
Recall@K
K
Node2Vec
Task Guided
Our Implementation
1.0
0.9
0.8
0.7
0.6
0.5
1
2
3
4
5
6
7
8
9
10