Global and Local Feature Learning for Ego-Network Analysis

Fatemeh Salehi Rizi
Michael Granitzer and Konstantin Ziegler

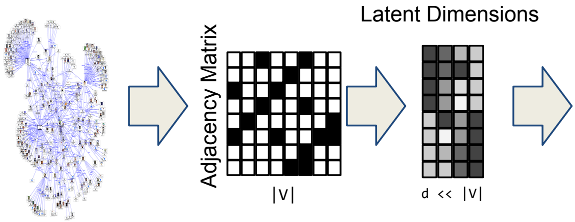
TIR Workshop

29 August 2017

Outline

Outline

Graph Embedding

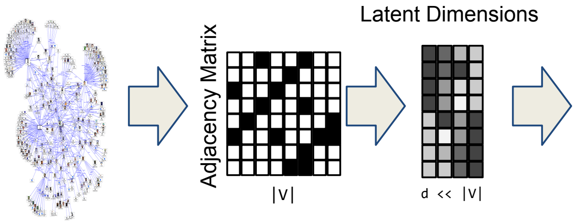


- Anomaly Detection
- Attribute Prediction
- Clustering
- Link Prediction
- ...

Given graph G with a set of nodes $V = \{v_1, \ldots, v_n\}$

$$f : v_i \mapsto y_i \in \mathbb{R}^d,\ d \ll |V|$$

- Methods based on eigen-decomposition of the Adjacency Matrix
- Methods inspired by NLP and Deep Learning



- Anomaly Detection
- Attribute Prediction
- Clustering
- Link Prediction
- ...

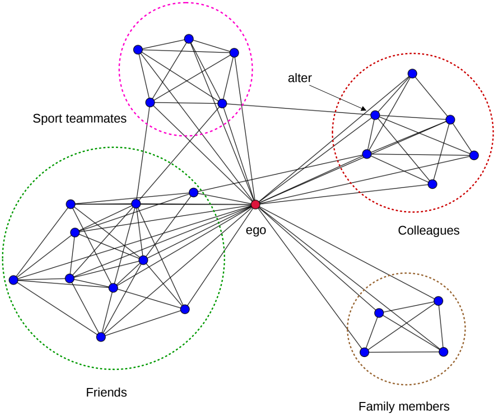
Given graph G with a set of nodes $V = \{v_1, \dots, v_n\}$

$$f : v_i \mapsto y_i \in \mathbb{R}^d,\ d \ll |V|$$

- Methods based on eigen-decomposition of the Adjacency Matrix
- **Methods inspired by NLP and Deep Learning**

What is an Ego Network?

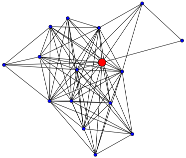
- Social graphs have been divided to several subgraphs (ego-networks) [1]
  - extracting features for nodes
  - detecting distinct neighborhood patterns
  - study social relationships
- Ego-network [1]
  - ego
  - alters
  - social circles



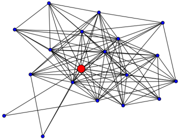
An ego-network with four social circles

Local Neighborhood Analysis

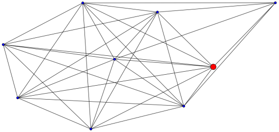
- Neighborhood around each ego has a different pattern [2]



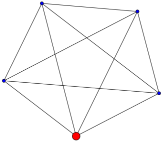
(a) Linked neighbors



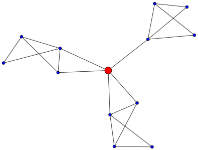
(b) Strongly linked



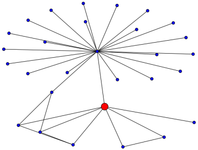
(c) Dense



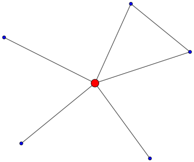
(d) Complete



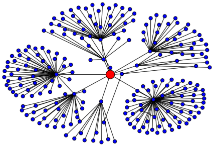
(e) Powerful ego node



(f) Strong ego neighbor



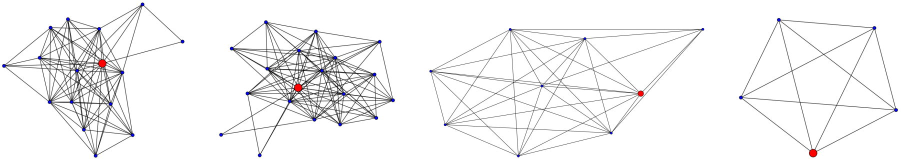
(g) Less cohesive star



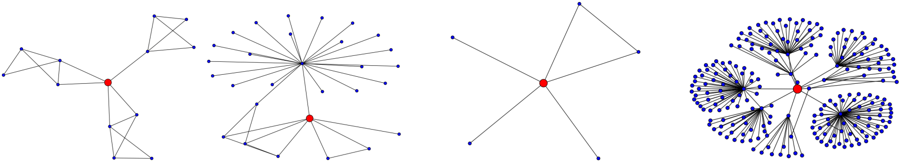
(h) Star

Local Neighborhood Analysis

- Neighborhood around each ego has a different pattern [2]



(a) Linked neighbors (b) Strongly linked (c) Dense (d) Complete

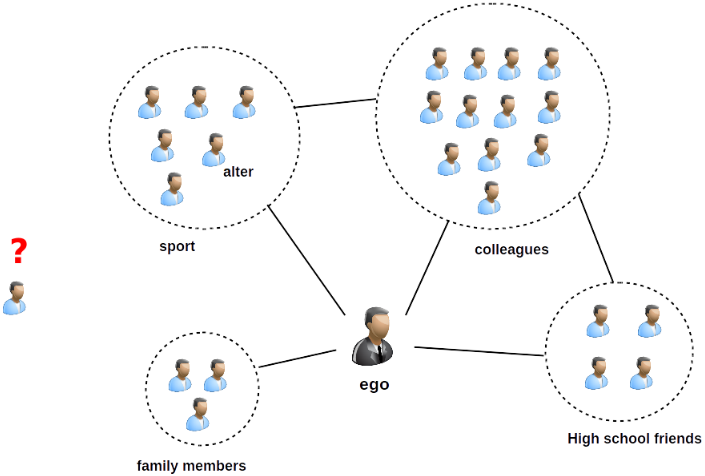


(e) Powerful ego node (f) Strong ego neighbor (g) Less cohesive star (h) Star

- Finding a vector representation for each ego-network
- Social circle detection and prediction

Social Circle Prediction

- Predicting the social circle for a new added alter to the ego-network



Outline

1. Introduction
   - Graph Embedding
   - Ego-Network Analysis
2. Contributions
3. State-of-the-art
4. Approach
5. Experimental Results
6. Future Work
7. References

Our Contributions

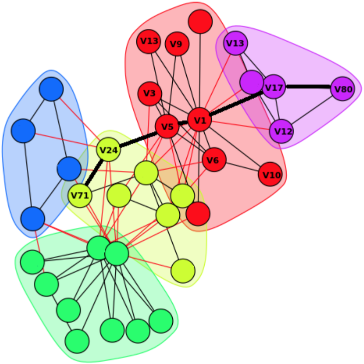
- We introduce local vector representations for egos to capture neighborhood structures
- We apply local vectors to the circle prediction problem
- We replace global representations by local to improve the performance

Outline

Vector Representation for Social Graphs

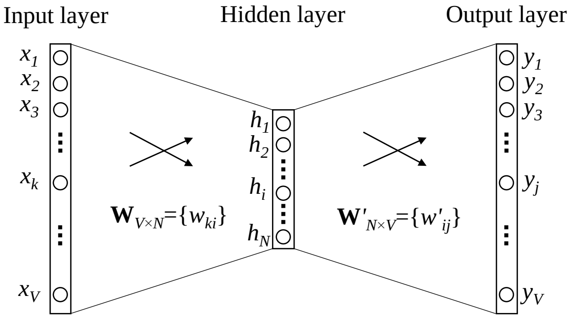
- DeepWalk [3]
  - walks globally over the graph and samples sequences of nodes
  - treats all these sequences as an artificial corpus
  - feeds the corpus to a Skip-Gram based Word2Vec [5]

$$v_{71} \rightarrow v_{24} \rightarrow v_{5} \rightarrow v_{1} \rightarrow v_{17} \rightarrow v_{80} \rightarrow$$
$$v_{92} \rightarrow v_{2} \rightarrow v_{3} \rightarrow v_{1} \rightarrow v_{12} \rightarrow v_{73} \rightarrow$$
$$v_{37} \rightarrow v_{34} \rightarrow v_{9} \rightarrow v_{1} \rightarrow v_{10} \rightarrow v_{94} \rightarrow$$
$$v_{73} \rightarrow v_{64} \rightarrow v_{5} \rightarrow v_{1} \rightarrow v_{12} \rightarrow v_{1} \rightarrow$$
$$v_{75} \rightarrow v_{14} \rightarrow v_{6} \rightarrow v_{1} \rightarrow v_{13} \rightarrow v_{61} \rightarrow$$



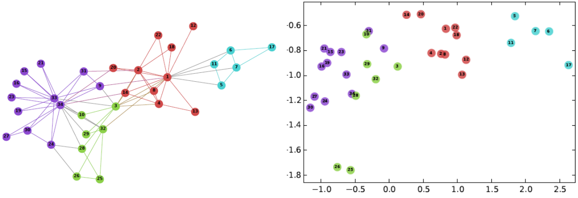
Vector Representation for Social Graphs

- DeepWalk [3]
  - walks globally over the graph and samples sequences of nodes
  - treats all these sequences as an artificial corpus
  - feeds the corpus to a Skip-Gram based Word2Vec [5]
  - Word2Vec: Having the sequence of words $\{w_1, w_2, \ldots, w_{t-1}, w_t, w_{t+1} \ldots, w_n\}$, language models aims to maximize $P(w_t | w_1, \ldots, w_{t-1})$.



Vector Representation for Social Graphs

- DeepWalk [3]
  - walks globally over the graph and samples sequences of nodes
  - treats all these sequences as an artificial corpus
  - feeds the corpus to a Skip-Gram based Word2Vec [5]
  - Word2Vec: Having the sequence of words $\{w_1, w_2, \ldots, w_{t-1}, w_t, w_{t+1} \ldots, w_n\}$, language models aims to maximize $P(w_t|w_1, \ldots, w_{t-1})$.
  - given sequence of nodes $\{v_1, v_2, \ldots, v_{t-1}, v_t, v_{t+1}, \ldots, v_n\}$ it maximizes: $\sum_{t=1}^{n} \log Pr(v_t|v_{t+c}, \ldots, v_{t_1}, v_{t+1}, \ldots, v_{t-c})$
  - $\mathrm{glo}\colon V \rightarrow \mathbb{R}^d$

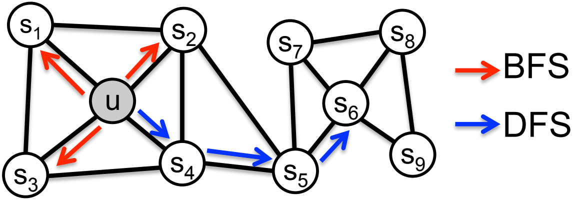


Input Output

Zachary's karate club embedding [2]

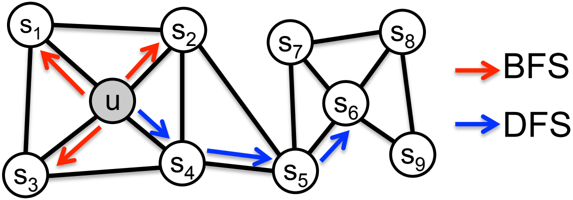
Vector Representation for Social Graphs

- node2vec [4]
  - similar to DeepWalk with two additional parameters
  - hyper-parameters $p \in \mathbb{R}^+$ and $q \in \mathbb{R}^+$ control random walks
  - $q > 1$ and $p < \min(q, 1)$ walk locally (BFS)
  - $p > 1$ and $q < \min(q, 1))$ walk explorative (DFS)



Vector Representation for Social Graphs

- node2vec [4]
  - similar to DeepWalk with two additional parameters
  - hyper-parameters $p \in \mathbb{R}^+$ and $q \in \mathbb{R}^+$ control random walks
  - $q > 1$ and $p < \min(q, 1)$ walk locally (BFS)
  - $p > 1$ and $q < \min(q, 1))$ walk explorative (DFS)

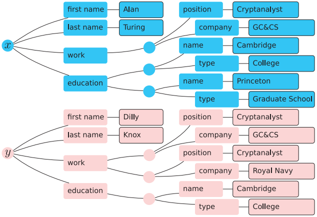


- Even the local walk can exceed the ego-network

Social Circle Prediction by McAuley et al. [1]

ALGORITHM 2: Update memberships node x and circle k.

Data: node x whose membership to circle C_k is to be updated
Result: updated membership for node x
initialize $\ell_x^k(0) := 0$, $\ell_x^k(1) := 0$;
construct a dummy node x_0 with the communities and features of x but with $x \notin C_k$;
construct a dummy node x_1 with the communities and features of x but with $x \in C_k$;
for $(c, f) \in \mathtt{dom}(types)$ **do**
    // c = community string, f = feature string
    $n := types(c, f)$;
    // n = number of nodes of this type
    if $S(x) = c \wedge Q(x) = f$ **then**
        // avoid including a self-loop on x
        $n := n - 1$;
    end
    construct a dummy node y with community memberships c and features f;
    // first compute probabilities assuming all pairs (x, y) are non-edges
    $\ell_x^k(0) := \ell_x^k(0) + n \log p((x_0, y) \notin E)$;
    $\ell_x^k(1) := \ell_x^k(1) + n \log p((x_1, y) \notin E)$;
end
for $(x, y) \in E$ **do**
    // correct for edges incident on x
    $\ell_x^k(0) := \ell_x^k(0) - \log p((x_0, y) \notin E) + \log p((x_0, y) \in E)$;
    $\ell_x^k(1) := \ell_x^k(1) - \log p((x_1, y) \notin E) + \log p((x_1, y) \in E)$;
end
// update membership to circle k
$types(S(x), Q(x)) := types(S(x), Q(x)) - 1$;
$z \leftarrow \mathcal{U}(0, 1)$;
if $z < \exp\{T(\ell_x^k(1) - \ell_x^k(0))\}$ **then**
    $S(x)[k] := 1$
else
    $S(x)[k] := 0$
end
$types(S(x), Q(x)) := types(S(x), Q(x)) + 1$;



$$\mathbf{1} - \sigma_{x,y} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \begin{array}{l} \textit{first name : Dilly} \\ \textit{last name : Knox} \\ \textit{first name : Alan} \\ \textit{last name : Turing} \\ \textit{work : position : Cryptanalyst} \\ \textit{work : location : GC\&CS} \\ \textit{work : location : Royal Navy} \\ \textit{education : name : Cambridge} \\ \textit{education : type : College} \\ \textit{education : name : Princeton} \\ \textit{education : type : Graduate School} \end{array}$$

$$\mathbf{1} - \sigma'_{x,y} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{array}{l} \textit{first name} \\ \textit{last name} \\ \textit{work : position} \\ \textit{work : location} \\ \textit{education : name} \\ \textit{education : type} \end{array}$$

- A Probabilistic Classifier
- Time Complexity $O(n^3)$

Outline

Local Representations using Paragraph Vector

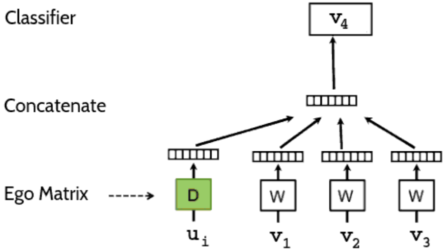
- walking locally over an ego-network to generate sequence of nodes
- treating this sequence as an artificial paragraph
- applying Paragraph Vector [6] to learn vector representation

Local Representations using Paragraph Vector

- walking locally over an ego-network to generate sequence of nodes
- treating this sequence as an artificial paragraph
- applying Paragraph Vector [6] to learn vector representation
- given an artificial paragraph $v_1, v_2, v_3, \ldots, v_t, \ldots, v_l$ for ego u_i, it maximizes the average log probability:

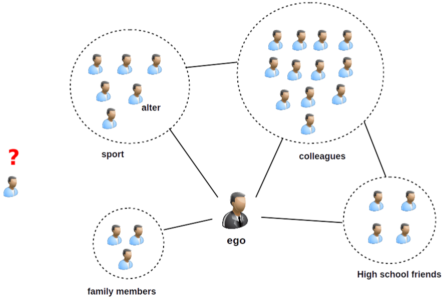
$$\sum_{t=1}^{l} \log Pr(v_t | u_i, v_{t+c}, \ldots, v_{t-c})$$

- $\text{loc}\colon U \rightarrow \mathbb{R}^d$



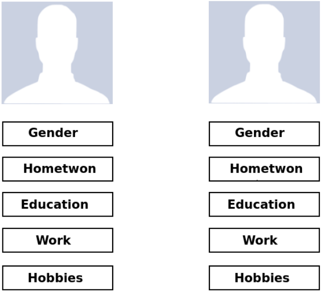
Social Circle Prediction

- Setting
  - social network $G = (V, E)$ with egos U and alters $V \setminus U$
  - profile information $(v.\,\mathrm{feat}_1, \ldots, v.\,\mathrm{feat}_f)$ for every $v \in V$
- Input/Output
  - predict social circles $c \colon V \setminus U \to \{C_1, \ldots, C_k\}^*$ given several samples
- Approach
  - Feature selection (users' profile information, graph embeddings)
  - A Neural Network Classifier



Incorporating Profile Information

- Similarity of ego's and alter's profile as a feature

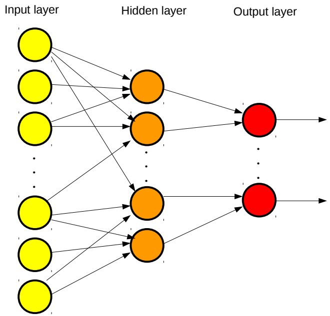


  - ego's profile feature: $u.\text{feat}_1, \ldots, u.\text{feat}_f$
  - alter's profile feature: $v.\text{feat}_1, \ldots, v.\text{feat}_f$
  - $\text{sim}(u, v) = (b_1, \ldots, b_f)$, where

$$b_i = \begin{cases} 1 & \text{if } u.\text{feat}_i = v.\text{feat}_i, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Social Circle Prediction

- A feed-forward neural network classifier
- Predicting social circle for alter v which belongs to ego-network of ego u
- **Input layer:**
  - **locglo:** $\mathrm{loc}(u) \oplus \mathrm{glo}(v)$
  - **gloglo:** $\mathrm{glo}(u) \oplus \mathrm{glo}(v)$
  - **locgloglo:** $\mathrm{loc}(u) \oplus \mathrm{glo}(u) \oplus glo(v)$
  - **locglosim:** $\mathrm{loc}(u) \oplus \mathrm{glo}(v \oplus \mathrm{sim}(u, v)$
  - **gloglosim:** $\mathrm{glo}(u) \oplus \mathrm{glo}(v) \oplus \mathrm{sim}(u, v)$
  - **locgloglosim:** $\mathrm{loc}(u) \oplus \mathrm{glo}(u) \oplus \mathrm{glo}(v) \oplus \mathrm{sim}(u, v)$



- **Hidden layer:** a single dense layer with ReLU activation units
- **Output layer:** softmax units (same number as circles)
- Ground-truth
  - alter's circle label (family, colleagues, etc)

Outline

Dataset

Table 1: Statistics of Social Network Datasets [7]

		Facebook	**Twitter**	**Google+**
nodes	$\lvert V \rvert$	4,039	81,306	107,614
edges	$\lvert E \rvert$	88,234	1,768,149	13,673,453
egos	$\lvert U \rvert$	10	973	132
circles	$\lvert \mathcal{C} \rvert$	46	100	468
features	f	576	2,271	4,122

Experimental Results

Table 2: Performance (F_1-score) of different embeddings for circle prediction on three dataset. Standard deviation is less than 0.02 for all experiments.

Approach	**Facebook**	**Twitter**	**Google+**
gloglo	0.37	0.46	0.49
locglo	**0.42**	**0.50**	**0.52**
locgloglo	0.37	0.44	0.48
gloglosim	0.40	0.49	0.51
locglosim	**0.45**	**0.53**	**0.55**
locgloglosim	0.38	0.46	0.47
Φ^1, McAuley & Leskovec [1]	0.38	0.54	0.59

Outline

Future Work

- Using embeddings to approximate more complex measures (e.g. shortest-path distance)
- Using embedding to find similar egos
- Learning embedding for directed graphs

Outline

References

McAuley, Julian J., and Jure Leskovec. "Learning to Discover Social Circles in Ego Networks." In NIPS, vol. 2012, pp. 548-56. 2012.

Muhammad, Syed Agha, and Kristof Van Laerhoven. "DUKE: A Solution for Discovering Neighborhood Patterns in Ego Networks." In ICWSM, pp. 268-277. 2015.

Perozzi, Bryan, Rami Al-Rfou, and Steven Skiena. "DeepWalk: Online learning of social representations." In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 701-710. ACM, 2014.

Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855-864. ACM, 2016.

T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In NIPS, 2013

Le, Quoc V., and Tomas Mikolov. "Distributed Representations of Sentences and Documents." In ICML, vol. 14, pp. 1188-1196. 2014.

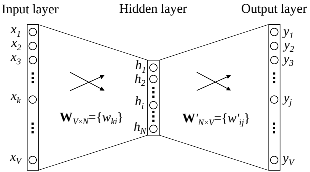
https://snap.stanford.edu/data/

Word2vec [5]

- Language Modeling
    - Distributional Hypothesis in the natural languages: semantically similar words dispose to appear in similar word neighborhoods
- Having the sequence of words $\{w_1, w_2, ..., w_{t-1}, w_t, ..., w_n\}$, language models aims to maximize $P(w_t|w_1, ..., w_{t-1})$.
- In word2vec [2], they defined a fix context length surrounding each word
- with length context c and the sequence of words $\{w_1, w_2, ..., w_{t-1}, w_t, ..., w_n\}$ the goal is to word2vec is to maximize: $\sum_{t=1}^{n} \log P(w_t|w_{t+c}, ..., w_{t-c})$
- The neural network which learns word representations:
    - One hidden layer
    - The number of input layer entries is equal to the vocabulary size of the text
    - The number of units in the hidden layer determines dimensionality of the vectors

Word2vec [5]

- Having a sequence of words $\{w_1, w_2, ..., w_{t-1}, w_t, ..., w_n\}$ and context window with length one
- Predict one target word, given one context word $P(w_t|w_{t-1})$.



- Each input is a one-hot encoding vector
- The probability is computed using the softmax function:
  $h = W^T \times x, \quad u = W'^T \times h, \quad P(w_t|w_{t-1}) = y_t = \frac{e^{u_t}}{\sum_{i=1}^{V} e^{u_j}}$
- After several iterations matrix W will not change

Paragraph Vector

- Given a sequence of paragraphs $p_1, p_2, ..., p_q$ and training words $w_1, w_2, w_3, \ldots, w_t, \ldots w_n$, the idea of Paragraph Vector [6] is to maximize $p(w_t|p_j, w_{t-c}..., w_{t+c}))$
- For example consider 2 paragraphs and window size of 3
  - P_1 : $\boxed{\text{The cat sat}}$ on the mat
  - P_2 : I ate potato crisps for evening snack
  - $p(on|P_1, The, cat, sat)$