Supplementary Material of Robust Negative Sampling for Network Embedding

Mohammadreza Armandpour,[1] Patrick Ding,[1] Jianhua Huang,[1] Xia Hu[2]
[1]Department of Statistics, Texas A&M University
[2]Department of Computer Science and Engineering, Texas A&M University
{armand, patrickding, jianhua}@stat.tamu.edu, hu@cse.tamu.edu

This article includes detailed proofs and some experimental results of the "Robust Negative Sampling for Network Embedding". The main paper has been accepted in AAAI 2019.

1 Proofs

1.1 Proof of Theorem 1:

Based on the definition of the vertex-level objective and equation related to the objective at edge-level for NS, the NS vertex level objective is:

$$\begin{aligned}
&\sum_{v_i \in N(u)} \log P^{NS}(v_i|u; f, f') \\
&= \sum_{v_i \in N(u)} [\log(\sigma(f'(v_i)^T \cdot f(u))) + \sum_{j=1}^{k} \mathbb{E}_{r_j \sim P(r)} \log(\sigma(-f'(r_j)^T \cdot f(u)))] \\
&= \sum_{v_i \in N(u)} [\log(\sigma(Z_{(v_i,u)})) + \sum_{j=1}^{k} \mathbb{E}_{r_j \sim P(r)} \log(\sigma(-Z_{(r_j,u)}))] \\
&= \sum_{v_i \in N(u)} [\log(\sigma(Z_{(v_i,u)})) + k\mathbb{E}_{r_j \sim P(r)} \log(\sigma(-Z_{(r_j,u)}))] \\
&= \sum_{v_i \in N(u)} [\log(\sigma(Z_{(v_i,u)})) + k \sum_{r_j \in V} P(r = r_j) \log(\sigma(-Z_{(r_j,u)}))] \\
&= \sum_{v_i \in N(u)} [\log(\sigma(Z_{(v_i,u)})) + k \sum_{r_j \in V} \frac{d_{r_j}^{\frac{3}{4}}}{C} \log(\sigma(-Z_{(r_j,u)}))],
\end{aligned} \tag{1}$$

where C is the term that is defined in the theorem. Notice that $\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}$, therefore we can write $\log(\sigma(Z)) = -\log(1 + e^Z) + Z$, and $\log(\sigma(-Z)) = -\log(1 + e^Z)$, using these two simplifications in equation 5 leads to following formula for the vertex level objective:

$$\begin{aligned}
&\sum_{v_i \in N(u)} [-\log(1 + e^{Z_{(v_i,u)}}) + Z_{(v_i,u)} + k \sum_{r_j \in V} -\frac{d_{r_j}^{\frac{3}{4}}}{C} \log(1 + e^{Z_{(r_j,u)}})] \\
&= \sum_{v_i \in N(u)} -\log(1 + e^{Z_{(v_i,u)}}) + \sum_{v_i \in N(u)} Z_{(v_i,u)} + kd_u \sum_{r_j \in V} -\frac{d_{r_j}^{\frac{3}{4}}}{C} \log(1 + e^{Z_{(r_j,u)}}) \\
&= -\sum_{r_j \in V} \left(\frac{d_{r_j}^{\frac{3}{4}} k d_u}{C} + \mathbb{1}_{r_j \in N(u)} \right) \log(1 + e^{Z_{(r_j,u)}}) + \sum_{v_i \in N(u)} Z_{(v_i,u)}
\end{aligned} \tag{2}$$

which is exactly equal to what we needed to prove for the vertex level objective of NS. And for the SGA objective:

$$\begin{aligned}\sum_{v_i \in N(u)} \log P^{SGA}(v_i|u; f, f') &= \sum_{v_i \in N(u)} \log(\frac{\exp(f'(v_i)^T \cdot f(u))}{\sum_{r \in V} \exp(f'(r)^T \cdot f(u))}) \\ &= \sum_{v_i \in N(u)} [Z_{(v_i,u)} - \log(\sum_{r \in V} e^{Z_{(r,u)}})] \\ &= -d_u \log(\sum_{r \in V} e^{Z_{(r,u)}}) + \sum_{v_i \in N(u)} Z_{(v_i,u)}.\end{aligned} \tag{3}$$

which is exactly the formula that we wanted to prove. For type I case the proof is similar, only f' is replaced with f.

1.2 Proof of Theorem 2:

Part 1: Showing the theorem for the SGA objective If we define the distribution Q_u as following:

$$Q_u(v) = \begin{cases} \frac{1}{d_u}, & \text{if } v \in N(u) \\ 0, & \text{otherwise,} \end{cases}$$

and the distribution P_u as

$$P_u(v) = \frac{e^{Z_{(v,u)}}}{\sum_{r \in V} e^{Z_{(r,u)}}}$$

Then the Kullback-Leibler divergence from P_u to Q_u is:

$$\begin{aligned}D_{KL}(Q_u||P_u) &= -\sum_{v \in V} Q_u(v) \log(\frac{P_u(v)}{Q_u(v)}) \\ &= -\sum_{v \in N(u)} \frac{1}{d_u} \log(\frac{P_u(v)}{\frac{1}{d_u}}) \\ &= -\log(d_u) - \frac{1}{d_u} \sum_{v \in N(u)} \log(P_u(v)).\end{aligned} \tag{4}$$

Therefore based on the above equation, minimizing the Kullback-Leibler distance between the two distributions is equivalent to maximizing the vertex level objective for the SGA objective. Hence to check whether the presented solution in Theorem 2 is the optimal solution for the objective we need to check whether it minimizes the distance between P_u, and Q_u. Notice, if $Z_{(r,u)} \to -\infty$, then $e^{Z_{(r,u)}} \to 0$, therefore if for non-neighbors of u, like r, $Z_{(r,u)} \to -\infty$, then P_u converges to distribution M_u, where:

$$M_u(v) = \begin{cases} \frac{e^{Z_{(v,u)}}}{\sum_{r \in N(u)} e^{Z_{(r,u)}}}, & \text{if } v \in N(u) \\ 0, & \text{otherwise,} \end{cases}$$

and if $Z_{(v,u)}$ for all $v \in N(u)$ are equal and they go to $+\infty$ (to be further dominant to the non-neighbor terms), then the of distribution M_u will be identical to that of Q_u, which means the distance between P_u and Q_u is zero. Therefore that set of values for Z minimize the KL distance and equivalently maximize the objective, completing the proof of part 1.

Part 2: Showing the theorem for the NS objective To derive the formula for optimal $Z_{(v,u)}$, in here we take the derivative of whole objective respect to the Z's and by making assumption of zero derivative, we find the optimal value. Based on the assumption that is made in the theorem, $\frac{\partial g(Z_{(v,u)})}{\partial Z_{(r,u)}} = 0$ for any function g if $r \neq v$. Therefore the derivative of whole objective of NS $O^{(NS)}$, respect to $Z_{(v,u)}$, based on the theorem 1 is:

$$\frac{\partial O^{(NS)}}{\partial Z_{(r,u)}} = \begin{cases} \frac{\partial \left[-(a_r+1)\log(1+e^{Z_{(r,u)}})+Z_{(r,u)} \right]}{\partial Z_{(r,u)}}, & \text{if } r \in N(u) \\ \frac{\partial \left[-a_r \log(1+e^{Z_{(r,u)}}) \right]}{\partial Z_{(r,u)}}, & \text{otherwise,} \end{cases}$$

where $a_r = \frac{d_r^{\frac{3}{4}} k d_u}{C}$. Hence, the derivative for $r \in N(u)$ is equal to:

$$-(a_r+1)\frac{e^{Z_{(r,u)}}}{1+e^{Z_{(r,u)}}} + 1 = -(a_r+1)\frac{1}{1+e^{-Z_{(r,u)}}} + 1$$

which should be equal to zero, therefore:

$$\begin{aligned} 1-(a_r+1)\frac{1}{1+e^{-Z_{(r,u)}}} = 0 &\iff 1 = (a_r+1)\frac{1}{1+e^{-Z_{(r,u)}}} \\ &\iff \frac{1}{1+a_r} = \frac{1}{1+e^{-Z_{(r,u)}}} \\ &\iff a_r = e^{-Z_{(r,u)}} \\ &\iff -\log(a_r) = Z_{(r,u)}, \end{aligned}$$

which is equal to the optimal value for $Z_{(r,u)}$ as mentioned in the theorem for $r \in N(u)$. For $r \notin N(u)$, the derivative is:

$$-a_r\frac{e^{Z_{(r,u)}}}{1+e^{Z_{(r,u)}}} = \frac{-a_r}{1+e^{-Z_{(r,u)}}}.$$

Notice if $Z_{(r,u)} \to -\infty$ then the derivative goes to zero which was what we needed to prove.

1.3 Proof of Theorem 3:

It easily can be seen that the derivative of $F(x,y)$ with respect to the first argument is:

$$\nabla_1 F(x,y) = \frac{y e^{x^T \cdot y}}{M + e^{x^T \cdot y}},$$

therefore $x - \eta\nabla_1 F(x,y) = x - by$, where b is a positive scalar, equal to $\frac{\eta e^{x^T \cdot y}}{M+e^{x^T \cdot y}}$. Now decompose y into two parts, y_1 and y_2 where $y_1 + y_2 = y$ and y_1 is the projected version of y on the vector x therefore y_2 is perpendicular to x and $y_1 = ax$ where a can be positive or negative.

If α is the angle between x and y (range from $[-\pi, \pi]$), since $x^T \cdot y = ||x||||y||cos(\alpha)$, we have $x^T \cdot y > 0$, if and only if $|\alpha| < \frac{\pi}{2}$. Moreover, we know $y = y_1 + y_2$ and $x^T \cdot y_2 = 0$, therefore, $a > 0$ if and only if $|\alpha| < \frac{\pi}{2}$

Now we are calculating the norm of $x - \eta\nabla_1 F(x,y)$ and compare that with the norm of x:

$$\begin{aligned} ||x - \eta\nabla_1 F(x,y)||_2^2 &= (x - \eta\nabla_1 F(x,y))^T \cdot (x - \eta\nabla_1 F(x,y)) \\ &= (x - b(ax + y_2))^T \cdot (x - b(ax+y_2)) \\ &= ((1-ab)x - by_2)^T \cdot ((1-ab)x - by_2) \\ &= (1-ab)^2||x||_2^2 + b^2||y_2||_2^2, \end{aligned}$$

therefore, based on the last equation if $a \le 0$, the norm of $x - \eta\nabla_1 F(x,y)$ is not less than the norm of x, because $(1-ab)^2 \ge 1$ since b is positive.

However if $a > 0$, there is a b_0, where for any b such that $0 < b < b_0 < \frac{1}{a}$, the norm of $x - \eta\nabla_1 F(x,y)$ is less than the norm of x because:

$$\begin{aligned} (1-ab)^2||x||_2^2 + b^2||y_2||_2^2 < ||x||_2^2 &\iff b^2||y_2||_2^2 < ||x||_2^2(1-(1-ab)^2) \\ &\iff b^2||y_2||_2^2 < ||x||_2^2(a^2b^2 + 2ab) \\ &\iff \frac{b}{a^2b+2a} < \frac{||x||_2^2}{||y||_2^2}, \end{aligned}$$

since $\frac{b}{a^2b+2a}$ is an increasing function with respect to b for $b \geq 0$ and it is zero for $b = 0$, therefore there is a b_0 such that for any $b < b_0$ the last inequality is satisfied. This fact proves the theorem because having $b = \eta m$ where m is a scalar and positive a is equivalent to having $|\alpha| < \frac{\pi}{2}$, and we can let $x = f'_v$ and $y = f_u$.

1.4 Proof of Theorem 4:

To find the optimal value for the $Z_{(v,u)}$, we need to first write the whole objective of R-NS, then take the derivative respect to each $Z_{(v,u)}$ and set it equal to zero to find the optimal $Z_{(v,u)}$.

Similar to the calculation for deriving the vertex level objective of NS for proving Theorem 1, the vertex level objective of R-NS, excluding the penalty part, can be expressed as:

$$\sum_{v_i \in N(u)} [\log(\sigma(Z_{(v_i,u)})) + k \sum_{r_j \notin N(u)} \frac{d_{r_j}^{\frac{3}{4}}}{C'_u} \log(\sigma(-Z_{(r_j,u)}))] =$$

$$\sum_{v_i \in N(u)} \log(\sigma(Z_{(v_i,u)})) + kd_u \sum_{r_j \notin N(u)} \frac{d_{r_j}^{\frac{3}{4}}}{C'_u} \log(\sigma(-Z_{(r_j,u)})), \quad (5)$$

where $C'_u = \sum_{v_j \notin N(u)} d_{v_j}^{\frac{3}{4}}$. Therefore:

$$\frac{\partial O^{(R-NS)}}{\partial Z_{(r,u)}} = \begin{cases} \frac{\partial \log(\sigma(Z_{(r,u)}))}{\partial Z_{(r,u)}}, & \text{if } r \in N(u) \\ \frac{\partial b_r \log(\sigma(-Z_{(r,u)}))}{\partial Z_{(r,u)}}, & \text{otherwise,} \end{cases}$$

where $b_r = kd_u \frac{d_{r_j}^{\frac{3}{4}}}{C'_u}$. But we know $\frac{\partial \log(\sigma(Z_{(r,u)}))}{\partial Z_{(r,u)}} = \frac{1}{1+e^{Z_{(r,u)}}}$, hence if $Z_{(r,u)} \to +\infty$, the derivative goes to zero for the neighbors which is exactly what we needed to prove. Since, for the non-neighbors $\frac{\partial b_r \log(\sigma(-Z_{(r,u)}))}{\partial Z_{(r,u)}} = b_r \frac{-1}{1+e^{-Z_{(r,u)}}}$ therefore if $Z_{(r,u)} \to -\infty$,for the non-neighbors, the derivative goes to zero.

2 Choice of λ

Here we derive a formula for the optimal λ based on the fact that derivative of the objective of R-NS respect to any embedding vector like f_i is zero at the optimal solution. We make some intuitive assumptions to estimate terms to provide a formula for the optimal λ.

In the R-NS type I objective the term f_u, the embedding vector of node u, appears in the objective in the following way:

$$\sum_{v \in N(u)} \log(\sigma(f_v^T \cdot f_u)) + \sum_{v \in N(u)} \log(\sigma(f_u^T \cdot f_v))$$
$$+ \sum_{v' \notin N(u) \cup \{u\}} \log(\sigma(-f_{v'}^T \cdot f_u)) kd_u C_{v'} + \sum_{v' \notin N(u) \cup \{u\}} \log(\sigma(-f_u^T \cdot f_{v'})) kd_{v'} C_u$$
$$- \lambda d_u ||f_u||_2^2 - \frac{\lambda}{k+1} d_u ||f_u||_2^2 - \frac{\lambda}{k+1} kM_u 2(E - d_u) ||f_u||_2^2, \quad (6)$$

where E is the total number of pairs of node and its neighbor and d_u is number of neighbors of u and:

$$C_{v'} = \frac{d_{v'}^{\frac{3}{4}}}{\sum_{r \notin N(u) \cup \{u\}} d_r^{\frac{3}{4}}} \quad C_u = \frac{d_u^{\frac{3}{4}}}{\sum_{r \notin N(v') \cup \{v'\}} d_r^{\frac{3}{4}}}.$$

In fact $C_{v'}$ is the probability that u choses v' as its negative neighbor, and C_u is the probability that v' choses u as a negative sample. The M_u term is the weighted average of the probability that non-neighbors of u choose u as their negative neighbors, and can be formulated as:

$$M_u = \sum_{w \notin N(u) \cup \{u\}} \frac{d_u^{\frac{3}{4}}}{\sum_{r \notin N(w) \cup \{w\}} d_r^{\frac{3}{4}}} \frac{d_w}{\sum_{k \notin N(u) \cup \{u\}} d_k}.$$

Now we are ready to explain form of equation (6) term by term. A term like f_u appears in the objective because u has been considered as a center sample[1], positive sample, or negative sample. The right terms in all three lines (6) are related to the case where u is chosen as a center sample, while the left term in the first line and middle term in the last line are related to the case where u has been chosen as positive sample, and the rest of the terms are for the case where u is considered as a negative sample. In the derivation of the (6) we assume that the graph of the neighborhood is undirected for simplicity.

Here we provide an approximation to the above terms to make calculation of the optimal λ possible. We estimate each of the summation of the $\log(\sigma(...))$ terms by taking a typical value for those terms–for example the median–and multiplying by the number of terms within the summation. We differentiate this approximation to (6) with respect to f_u and get:

$$\begin{aligned} & d_u \frac{f_{v_1}}{1+e^{f_{v_1}^T \cdot f_u}} + d_u \frac{f_{v_2}}{1+e^{f_u^T \cdot f_{v_2}}} \\ & \qquad + (n-d_u-1)\frac{-f_{v_3}}{1+e^{-f_{v_3}^T \cdot f_u}} k d_u \frac{1}{n-d_u-1} + (n-d_u-1)\frac{-f_{v_4}}{1+e^{-f_u^T \cdot f_{v_4}}} kd \frac{1}{n-d-1} \\ & \qquad\qquad - \lambda 2 d_u f_u - \frac{\lambda}{k+1} d_u 2 f_u - \frac{\lambda}{k+1} k \frac{1}{n-d-1} 2\left(\frac{nd}{2} - d_u\right) 2 f_u. \end{aligned} \tag{7}$$

To be clear about how we did the estimation we kept the order of terms in (7) the same as in (6). d is the average node degree. v_1 and v_2 refer to a typical neighbor of u, while v_3 and v_4 refer to non-neighbors of u. The probability terms $C_u, C_{v'}$, and weighted averaged probability M_u are estimated by their mean.

Notice the derivative (7) should be approximately equal to the zero vector in the embedding space at the learned embeddings point because the exact derivative is equal to zero. Therefore if we multiply (7) by the f_u^T vector, the result should be approximately equal to zero. Now, we provide an estimation for a typical value for the inner product of a vector and its neighbor, as $l^2 cos(\alpha_n) = f_u^T \cdot f_{v_{neighbor}}$, where l is the typical norm of the embedding vector and α_n is the typical angle between a node and its neighbor. We estimate $cos(\alpha_n)$ by:

$$\frac{ln(1+\frac{d}{2\bar{w}})}{ln(1+n)},$$

where $\bar{w}$ is the mean of the non-zero weights of the edges in the graph. In fact $\frac{d}{\bar{w}}$ is representative of the number of other nodes that a typical node is connected to in the network. Since in the full graph or a graph close to that for example each node is connected to approximately cn other nodes, where c is a constant, we expect the embedding vectors to be almost in the same direction for the connected nodes. Therefore the $cos(\alpha_n)$ should be approximately one, and the estimation also follows this rule. However, for the case that each node is only connected to a few others, we expect the embedding vectors to not be concentrated on a direction. Therefore the median angular distance between a node and its neighbors should be close to $\pi/2$, which results in a small value for $cos(\alpha_n)$. Again the presented formula satisfies this condition. It also has the desirable property that larger d leads to larger $cos(\alpha_n)$ and smaller angular distance.

We estimate the inner product of a node and its non-neighbor by $l^2 cos(\beta_n) = f_u^T f_{v_{nonneighbor}}$ where $\beta_n = \pi/2 + \alpha_n$. We did this estimation because if the typical absolute value of the angular distance for a node and its neighbor is α_n then the neighbor's embedding should have angular distance approximately in the range $[-2\alpha_n, 2\alpha_n]$ from the node of interest. Therefore the absolute value of angular distance of non-neighbors from the nodes of interest should approximately be in the interval $[2\alpha_n, \pi]$, therefore a typical angular distance is $\frac{\pi+2\alpha}{2}$ which is equal to β_n, as introduced above.

Based on the above approximation and multiplying (7) by f_u^T we have:

$$2d\frac{l^2 cos(\alpha_n)}{1+e^{l^2 cos(\alpha_n)}} + 2kd\frac{-l^2 cos(\beta_n)}{1+e^{-l^2 cos(\beta_n)}} - \lambda 2 d l^2 - \frac{\lambda}{k+1} d 2 l^2 - \frac{\lambda k d}{k+1}\frac{n-2}{n-d-1} 2 l^2 = 0. \tag{8}$$

Based on this equation we have the following formula for the optimal λ:

$$\lambda_{opt} := \frac{\frac{cos(\alpha_n)}{1+e^{l^2 cos(\alpha_n)}} + k\frac{-cos(\beta_n)}{1+e^{-l^2 cos(\beta_n)}}}{\left(\frac{k+2}{k+1} + \frac{k}{k+1}\frac{n-2}{n-d-1}\right)}. \tag{9}$$

[1]when we sample the edge containing that node and we are looking for negative sample for that node

By plugging in the formula that we had for β_n, we can write

$$\lambda_{opt} := \frac{\frac{m}{1+e^{l^2 m}} + k\frac{\sqrt{1-m^2}}{1+e^{l^2\sqrt{1-m^2}}}}{\left(\frac{k+2}{k+1} + \frac{k}{k+1}\frac{n-2}{n-d-1}\right)}. \tag{10}$$

where $m = \frac{ln(1+\frac{d}{2\bar{w}})}{ln(1+n)}$, $d = \frac{2E}{n}$ and l is related to the dimension of the embedding space; higher dimension requires larger norm to prevent learning tiny components. Based on several experiments with real world networks we suggest $l = \frac{\log_2(dim)}{6} + 2$.

For the type II objective case, a similar argument leads to the following approximation for the optimal λ:

$$\lambda_{opt}^{typeII} := \frac{\frac{m}{1+e^{l^2 m}} + k\frac{\sqrt{1-m^2}}{1+e^{l^2\sqrt{1-m^2}}}}{2,} \tag{11}$$

where m is as before but here we consider the l for the type II is $\frac{\log_2(dim)}{6} + 1.9$.

3 Scalability

The R-NS objective has a different negative sampling distribution for different nodes, so we need a method to draw negative samples from the desired distribution without changing the time complexity. We suggest two methods for adaptive sampling, depending on whether the network is densely connected or not. We use the term non-dense network when $O(E_N) << O(n^2)$, where E_N is the total number of pairs of nodes and their neighbors. By dense network we refer to the case where $O(E_N)$ is $O(n^2)$. Notice that non-dense or dense connectivity is based on how we defined the neighborhoods in the graph–the connectedness of the neighborhood can be different from that of the graph itself.

Non-dense network In this setup we make a one dimensional table for each node u, where each entry has a vertex ID, and the number of times a vertex ID v appears in the table of the node u is proportional to $d_v^{\frac{3}{4}}$ for neighbors of u and zero otherwise. Therefore, by randomly choosing one entry of the table for u, we will have a vertex ID which follows the P_u distribution. To make the tables we first make a "generative table". The generative table has a predetermined large size like L and contains each vertex ID in the graph approximately proportional to the corresponding node's degree to the power $3/4$. The table for node u is made by elimination of the vertex IDs related to its neighbors from the generative table as we go through the pairs of neighbors.

In this proposal we only need a pre-processing step to use R-NS instead of NS. Therefore the time complexity is $O(n) + O(E_N)$ which is less than the order of the rest of the algorithm, hence the time complexity does not change by substituting NS with R-NS.

Dense network In this regime we draw negative samples from an approximation to P_u, P'_u, which can also take negative samples from neighbors of u. Intuitively speaking the NS distribution P takes negative samples from all of the nodes in the graph while P_u is restricted to the set of non-neighbors. P'_u is a compromise between these two, which still allows taking samples from the neighbor nodes but is designed to focus more on the non-neighbors by giving them higher probability. This leads to a lower probability of selecting neighbors for the negative sample and consequently prevents the Popular Neighbor Problem.

To understand the algorithm suppose that a node u has w neighbors. If we draw samples from P but reject neighbors of u and only accept non- neighbors of u, this is essentially equivalent to taking samples from P_u. However to check whether the sample is a neighbor of u or not we need to search within the list of neighbors which leads to multiplying the time complexity by $\log(w)$. This is a problem in dense networks where w is large. To circumvent this problem, we draw negative samples from a randomly chosen subset of size l from the neighborhood of u and consider all other nodes as non-neighbors of u. After we have drawn a certain number of negative samples we choose a another random l sized neighborhood subset of u. This characterizes P', an estimate of P_u.

4 Other Methods' Use of Type I and II Objectives

DeepWalk, Node2Vec and Struc2vec all use the type II objective because their implementations use word2vec, which is designed with a type II objective. LINE uses both type II and Type I objectives through the second order and first order proximity objectives introduced in [2]. First order proximity is expressed a little differently than the type I SGA objective, but because LINE never used that objective directly and instead estimated it with NS, it essentially used a type I objective.

5 Karate Network

The karate network [3] consists of 34 nodes and 78 edges, where each node represents a member in the club and each edge represents a friendship between two members.

We consider this network because it's a small, real network which permits direct visualizations of embeddings. Furthermore the network has two perceivable communities which enables us to judge the quality of embeddings. It also has the characteristics of large-scale-free networks since it contains a large number of low degree nodes and a few high degree nodes.

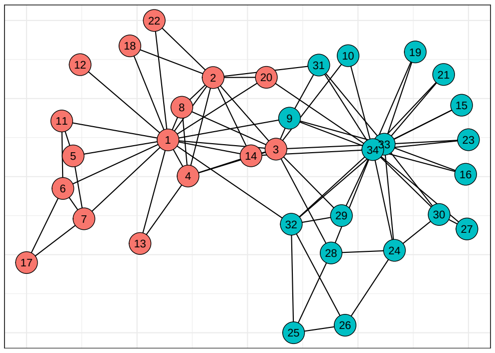

Figure 1: Zachary's karate club.

In Figure 2 we see the result of NS with the number of negative samples $k = 5$, the default setting of LINE, and $k = 20$, the suggested value for small networks in [1]. For the Type I objective we see that NS provides unsatisfactory embeddings for the karate network. In the $k = 5$ case the embeddings of the two groups are not separable and concentrated around 0, while in the $k = 20$ case all the embeddings are shrunk too far toward the origin and nodes within the same community are indistinguishable.

In the Type II objective setting, for NS with both $k = 5$ and $k = 20$, the context and embedding vectors of the nodes are not aligned. In fact there is a large angular distance between the embedding of a node and the context vector of its neighbors, which is counter to the goals of embedding. Moreover the norms of the embedding vectors of NS have a large range and nodes with high degree are concentrated near zero. This is problematic for important tasks like classification based on the embeddings because this behavior causes high degree nodes to be indistinguishable and the large range of norms further increases the difficulty of classification. Also, based on the SGA model, if a node's embedding has large norm, it should have a larger inner product with vectors with which it is close in terms of angular distance. This leads to higher probability of having edges to other nodes. Therefore we expect nodes with high degree to have large norms, contrary to the NS estimate of the SGA model.

The above disadvantages are the manifestations of the popular neighbor problem.

R-NS solves these problems for both Type I and Type II objectives. In Figure 2 we see that the norms are in a reasonable range and high degree nodes have large norms compared to other nodes. Also the direction of the embedding vectors is better determined because there is a reasonable separation between nodes related to different clusters. The angular distance between context vectors and embedding vectors under the Type II objective is much smaller. The context vectors of neighbors are more aligned.

Figure 2 also shows the result for the case where we only require negative samples to be from non-neighbors of a node and the node itself, which is equivalent to setting the R-NS $\lambda = 0$. Compared to NS, this sampling approach leads to context and embedding vectors that are more aligned and avoids overshrinking embeddings of high degree nodes. However it does not provide a clear separation of clusters in embedding space, which is a sign of poor estimation of angular distance. Also the range of vector norms is too large. This example demonstrates the importance of the penalty term λ.

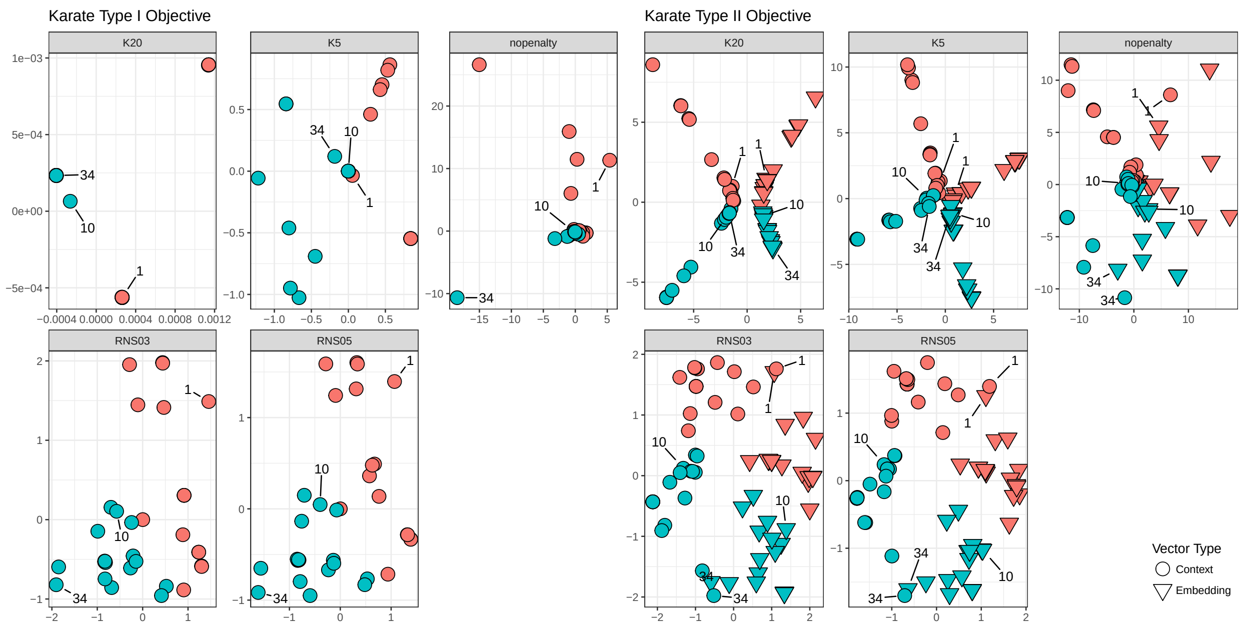

Figure 2: Zachary's karate network embedding negative sampling algorithm comparison.

6 LINE can emulate others

The way LINE defines neighborhoods (the set of immediate neighbors of the the nodes in the network) allows it to emulate other embedding algorithms like Struc2vec, Node2vec, DeepWalk. This can be shown in the following way. For any given network $G = \{V, E\}$ and embedding algorithm A with definition of a neighborhood of u, $N(u)$, we can create a new graph $G' = \{V, E_A\}$ such that E_A consists of directed edges like (u, n_i), where $n_i \in N(u)$. If n_i appears r times in $N(u)$ we weight the directed edge from u to n_i by r. Notice G and G' differ only in their edges; E is the original set of edges while E_A is the set of weighted directed edges constructed by the embedding algorithm A. Since LINE only considers immediate neighbors of u as $N(u)$, we can emulate algorithms like node2vec, DeepWalk, struc2vec, or others by passing their corresponding G' to LINE.

7 Parameter Settings

We set parameters according to the default settings of LINE. The number of negative samples K is 5 for all methods we considered, the mini-batch size for is 1 for all the methods, we use the learning rate starting value $\rho_0 = 0.025$ and $\rho_t = \rho_0(1 - t/T)$, where T is the total number of mini-batches or edge samples. The number of edge samples for all the methods and networks is 800 million, which took in almost all cases less than 30 minutes to run on a single machine with an Intel Core i7 processor. We set the dimension of the embeddings to 100 instead of 128 as in previous methods. This change in dimension does not have a major effect on results.

8 Experiment Details

The following tables 1,2 show the results for multi-lable node classification task with standard deviation inside the parentheses. The ROC curve of the baseline methods for 6 different labels in wikipedia data-set are shown in the figure **??**, which shows superiority of R-NS in most cases.

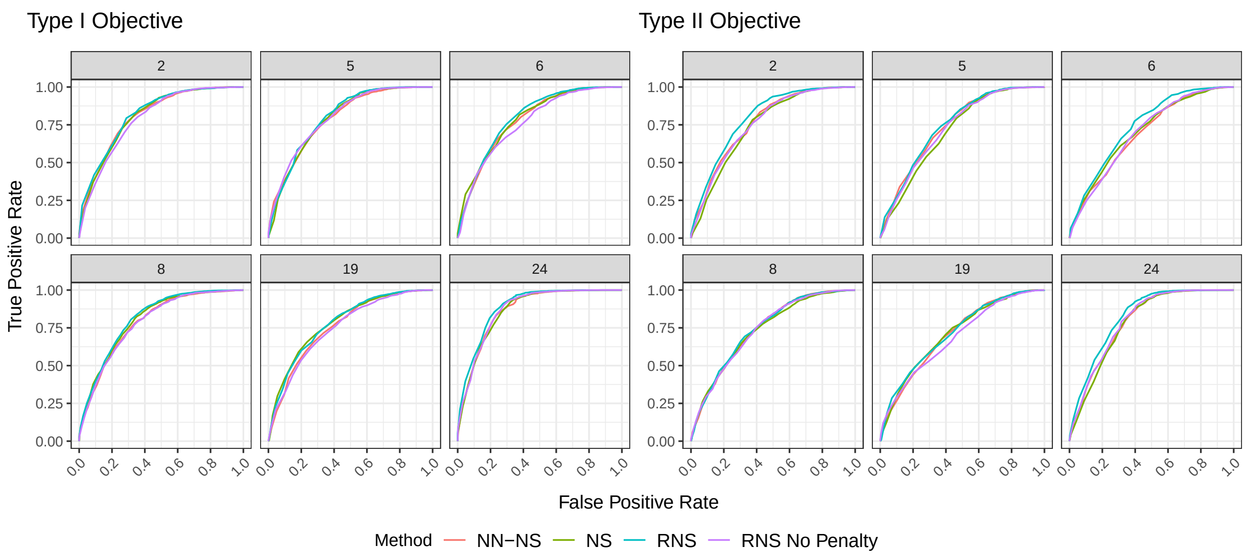

Figure 3: ROC curve for 6 different node labels of the wikipedia data-set, the base-line algorithms are shown in different colors.

Table 1: Algorithm comparison for 50% data labeled, Type I objective. The standard deviation of the numbers presented inside the parentheses

Score (Std. Error)	Micro-F_1			Macro-F_1		
Algorithm	Blog Catalog	Flickr	Wikipedia	Blog Catalog	Flickr	Wikipedia
NS	36 (0.11)	54 (0.50)	47 (0.23)	25 (0.20)	43 (0.48)	15 (0.44)
NN-NS	32 (0.39)	58 (0.48)	46 (0.14)	22 (0.27)	46 (0.52)	14 (0.20)
R-NS ($\lambda = 0$)	34 (0.34)	55 (0.48)	42 (0.42)	23 (0.20)	43 (0.50)	11 (0.14)
R-NS	**38 (0.26)**	**60 (0.26)**	**48 (0.45)**	**27 (0.28)**	**49 (0.25)**	**15 (0.21)**
R-NS λ	0.046 / 0.059	0.043 / 0.056	0.053 / 0.067	0.046 / 0.059	0.043 / 0.056	0.053 / 0.067

Table 2: Algorithm comparison for 50% data labeled, Type II objective.

Score (Std. Error)	Micro-F_1			Macro-F_1		
Algorithm	Blog Catalog	Flickr	Wikipedia	Blog Catalog	Flickr	Wikipedia
NS	12 (0.30)	46 (0.29)	47 (0.38)	7 (0.13)	35 (0.36)	15 (0.21)
NN-NS	15 (0.49)	41 (0.48)	45 (0.33)	9 (0.28)	31 (0.27)	11 (0.14)
R-NS ($\lambda = 0$)	15 (0.26)	40 (0.56)	44 (0.74)	9 (0.16)	28 (0.47)	11 (0.28)
R-NS	**25 (0.24)**	**57 (0.75)**	**54 (0.28)**	**17 (0.08)**	**47 (0.72)**	**20 (0.28)**
R-NS λ	0.046 / 0.059	0.043 / 0.056	0.053 / 0.067	0.046 / 0.059	0.043 / 0.056	0.053 / 0.067

References

[1] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[2] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, 2015.

[3] W. Zachary. An information flow model for conflict and fission in small groups. *J. of Anthropological Research*, 33:452–473, 1977.