THE IMPORTANCE OF NORM REGULARIZATION IN LINEAR GRAPH EMBEDDING: THEORETICAL ANALYSIS AND EMPIRICAL DEMONSTRATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning distributed representations for nodes in graphs is a crucial primitive in network analysis with a wide spectrum of applications. Linear graph embedding methods learn such representations by optimizing the likelihood of both positive and negative edges while constraining the dimension of the embedding vectors. We argue that the generalization performance of these methods is not due to the dimensionality constraint as commonly believed, but rather the small norm of embedding vectors. Both theoretical and empirical evidence are provided to support this argument: (a) we prove that the generalization error of these methods can be bounded by limiting the norm of vectors, regardless of the embedding dimension; (b) we show that the generalization performance of linear graph embedding methods is correlated with the norm of embedding vectors, which is small due to the early stopping of SGD and the vanishing gradients. We performed extensive experiments to validate our analysis and showcased the importance of proper norm regularization in practice.

1 INTRODUCTION

Graphs have long been considered as one of the most fundamental structures that can naturally represent interactions between numerous real-life objects (*e.g.*, the Web, social networks, protein-protein interaction networks). Graph embedding, whose goal is to learn distributed representations for nodes while preserving the structure of the given graph, is a fundamental problem in network analysis that underpins many applications. A handful of graph embedding techniques have been proposed in recent years (Perozzi et al., 2014; Tang et al., 2015b; Grover & Leskovec, 2016), along with impressive results in applications like link prediction, text classification (Tang et al., 2015a), and gene function prediction (Wang et al., 2015).

Linear graph embedding methods preserve graph structures by converting the inner products of the node embeddings into probability distributions with a softmax function (Perozzi et al., 2014; Tang et al., 2015b; Grover & Leskovec, 2016). Since the exact softmax objective is computationally expensive to optimize, the negative sampling technique (Mikolov et al., 2013) is often used in these methods: instead of optimizing the softmax objective function, we try to maximize the probability of positive instances while minimizing the probability of some randomly sampled negative instances. It has been shown that by using this negative sampling technique, these graph embedding methods are essentially computing a factorization of the adjacency (or proximity) matrix of graph (Levy & Goldberg, 2014). Hence, it is commonly believed that the key to the generalization performance of these methods is the dimensionality constraint.

However, in this paper we argue that *the key factor to the good generalization of these embedding methods is not the dimensionality constraint, but rather the small norm of embedding vectors*. We provide both theoretical and empirical evidence to support this argument:

- Theoretically, we analyze the generalization error of two linear graph embedding hypothesis spaces (restricting embedding dimension/norm), and show that only the norm-restricted hypothesis class can theoretically guarantee good generalization in typical parameter settings.
- Empirically, we show that the success of existing linear graph embedding methods (Perozzi et al., 2014; Tang et al., 2015b; Grover & Leskovec, 2016) are due to the early stopping of stochastic

gradient descent (SGD), which implicitly restricts the norm of embedding vectors. Furthermore, with prolonged SGD execution and no proper norm regularization, the embedding vectors can severely overfit the training data.

PAPER OUTLINE

The rest of this paper is organized as follows. In Section 2, we review the definition of graph embedding problem and the general framework of linear graph embedding. In Section 3, we present both theoretical and empirical evidence to support our argument that the generalization of embedding vectors is determined by their norm. In Section 4, we present additional experimental results for a hinge-loss linear graph embedding variant, which further support our argument. In Section 5, we discuss the new insights that we gained from previous results. Finally in Section 6, we conclude our paper. Details of the experiment settings, algorithm pseudo-codes, theorem proofs and the discussion of other related work can all be found in the appendix.

2 PRELIMINIARIES

2.1 THE GRAPH EMBEDDING PROBLEM

We consider a graph $G = (V, E)$, where V is the set of nodes in G, and E is the set of edges between the nodes in V. For any two nodes $u, v \in V$, an edge $(u, v) \in E$ if u and v are connected, and we assume all edges are unweighted and undirected for simplicity[1]. The task of graph embedding is to learn a D-dimensional vector representation $\mathbf{x}_u$ for each node $u \in V$ such that the structure of G can be maximally preserved. These embedding vectors can then be used as features for subsequent applications (e.g., node label classification or link prediction).

2.2 THE LINEAR GRAPH EMBEDDING FRAMEWORK

Linear graph embedding (Tang et al., 2015b; Grover & Leskovec, 2016) is one of the two major approaches for computing graph embeddings [2]. These methods use the inner products of embedding vectors to capture the likelihood of edge existence, and are appealing to practitioners due to their simplicity and good empirical performance. Formally, given a node u and its neighborhood $N_+(u)$ [3], the probability of observing node v being a neighbor of u is defined as:

$$p(v|u) = \frac{\exp(\mathbf{x}_u^T \mathbf{x}_v)}{\sum_{k \in V} \exp(\mathbf{x}_u^T \mathbf{x}_k)}.$$

By minimizing the KL-divergence between the embedding-based distribution and the actual neighborhood distribution, the overall objective function is equivalent to:

$$L = -\sum_{u \in V} \sum_{v \in N_+(u)} \log p(v|u)$$

Unfortunately, it is quite problematic to optimize this objective function directly, as the softmax term involves normalizing over all vertices. To address this issue, the negative sampling (Mikolov et al., 2013) technique is used to avoid computing gradients over the full softmax function. Intuitively, the negative sampling technique can be viewed as randomly selecting a set of nodes $N_-(u)$ that are not connected to each node u as its *negative neighbors*. The embedding vectors are then learned by minimizing the following objective function instead:

[1] All linear graph embedding methods discussed in this paper can be generalized to weighted case by multiplying the weight to the corresponding loss function of each edge. The directed case is usually handled by associating each node with two embedding vectors for incoming and outgoing edges respectively, which is equivalent as learning embedding on a transformed undirected bipartite graph.

[2] The other major approach is to use deep neural network structure to compute the embedding vectors, see the discussion of other related works in the appendix for details.

[3] Note that $N_+(u)$ can be either the set of direct neighbors in the original graph G (Tang et al., 2015b), or an expanded neighborhood based on measures like random walk (Grover & Leskovec, 2016).

$$L = -\sum_{u}\sum_{v\in N_+(u)} \log\sigma(\mathbf{x}_u^T\mathbf{x}_v) - \sum_{u}\sum_{v\in N_-(u)} \kappa\frac{|N_+(u)|}{|N_-(u)|}\log\sigma(-\mathbf{x}_u^T\mathbf{x}_v). \quad (1)$$

where $\sigma(x) = 1/(1+e^{-x})$ is the standard logistic function.

2.3 The Matrix Factorization Interpretation

Although the embedding vectors learned through negative sampling do have good empirical performance, there is very few theoretical analysis of such technique that explains the good empirical performance. The most well-known analysis of negative sampling was done by Levy & Goldberg (2014), which claims that the embedding vectors are approximating a low-rank factorization of the PMI (Pointwise Mutual Information) matrix.

More specifically, the key discovery of Levy & Goldberg (2014) is that when the embedding dimension is large enough, the optimal solution to Eqn (1) recovers exactly the PMI matrix (up to a shifted constant, assuming the asymptotic case where $N_-(u) = V$ for all $u \in V$):

$$\forall u, v, \mathbf{x}_u^T\mathbf{x}_v = \log(\frac{|E|\cdot 1_{(u,v)\in E}}{|N_+(u)||N_+(v)|}) - \log\kappa$$

Based on this result, Levy & Goldberg (2014) suggest that optimizing Eqn (1) under the dimensionality constraint is equivalent as computing a low-rank factorization of the shifted PMI matrix. This is currently the mainstream opinion regarding the intuition behind negative sampling. Although Levy and Goldberg only analyzed negative sampling in the context of word embedding, it is commonly believed that the same conclusion also holds for graph embedding (Qiu et al., 2018).

3 The Importance of Norm Regularization

As explained in Section 2.3, it is commonly believed that linear graph embedding methods are approximating a low-rank factorization of PMI matrices. As such, people often deem the dimensionality constraint of embedding vectors as the key factor to good generalization (Tang et al., 2015b; Grover & Leskovec, 2016). However, due to the sparsity of real-world networks, the explanation of Levy & Goldberg is actually very counter-intuitive in the graph embedding setting: the average node degree usually only ranges from 10 to 100, which is much less than the typical value of embedding dimension (usually in the range of $100 \sim 400$). Essentially, this means that in the context of graph embedding, the total number of free parameters is larger than the total number of training data points, which makes it intuitively very unlikely that the negative sampling model (i.e., Eqn (1)) can inherently guarantee the generalization of embedding vectors in such scenario, and it is much more plausible if the observed good empirical performance is due to some other reason.

In this paper, we provide a different explanation to the good empirical performance of linear graph embedding methods: we argue that the good generalization of linear graph embedding vectors is due to their small norm, which is in turn caused by the vanishing gradients during the stochastic gradient descent (SGD) optimization procedure. We provide the following evidence to support this argument:

- In Section 3.1, we theoretically analyze the generalization error of two linear graph embedding variants: one has the standard dimensionality constraints, while the other restricts the vector norms. Our analysis shows that:
  - The embedding vectors can generalize well to unseen data if their average squared l_2 norm is small, and this is always true regardless of the embedding dimension choice.
  - Without norm regularization, the embedding vectors can severely overfit the training data if the embedding dimension is larger than the average node degree.
- In Section 3.2, we provide empirical evidence that the generalization of linear graph embedding is determined by vector norm instead of embedding dimension. We show that:
  - In practice, the average norm of the embedding vectors is small due to the early stopping of SGD and the vanishing gradients.
  - The generalization performance of embedding vectors starts to drop when the average norm of embedding vectors gets large.
  - The dimensionality constraint is only helpful when the embedding dimension is very small (around $5 \sim 10$) and there is no norm regularization.

3.1 GENERALIZATION ANALYSIS OF TWO LINEAR GRAPH EMBEDDING VARIANTS

In this section, we present a generalization error analysis of linear graph embedding based on the uniform convergence framework (Bartlett & Mendelson, 2002), which bounds the maximum difference between the training and generalization error over the entire hypothesis space. We assume the following statistical model for graph generation: there exists an unknown probability distribution $\mathcal{Q}$ over the Cartesian product $V \times U$ of two vertex sets V and U. Each sample (a, b) from $\mathcal{Q}$ denotes an edge connecting $a \in V$ and $b \in U$.The set of (positive) training edges E_+ consists of the first m i.i.d. samples from the distribution $\mathcal{Q}$, and the negative edge set E_- consists of i.i.d. samples from the uniform distribution $\mathcal{U}$ over $V \times U$. The goal is to use these samples to learn a model that generalizes well to the underlying distribution $\mathcal{Q}$. We allow either $V = U$ for homogeneous graphs or $V \cap U = \emptyset$ for bipartite graphs.

Denote $E_\pm = \{(a, b, +1) : (a, b) \in E_+\} \cup \{(a, b, -1) : (a, b) \in E_-\}$ to be the collection of all training data, and we assume that data points in $E_\pm$ are actually sampled from a combined distribution $\mathcal{P}$ over $V \times U \times \{\pm 1\}$ that generates both positive and negative edges. Using the above notations, the training error $\mathcal{L}_t(\mathbf{x})$ and generalization error $\mathcal{L}_g(\mathbf{x})$ of embedding $\mathbf{x} : (U \cup V) \to \mathbb{R}^D$ are defined as follows:

$$\mathcal{L}_t(\mathbf{x}) = \frac{1}{|E_\pm|} \sum_{(a,b,y) \in E_\pm} -\log \sigma(y \mathbf{x}_a^T \mathbf{x}_b) \qquad \mathcal{L}_g(\mathbf{x}) = -\mathbb{E}_{(a,b,y)\sim\mathcal{P}} \log \sigma(y \mathbf{x}_a^T \mathbf{x}_b)$$

In the uniform convergence framework, we try to prove the following statement:

$$\mathbf{Pr}(\sup_{\mathbf{x} \in \mathcal{H}} (\mathcal{L}_g(\mathbf{x}) - \mathcal{L}_t(\mathbf{x})) \leq \epsilon) \geq 1 - \delta$$

which bounds the maximum difference between $\mathcal{L}_t(\mathbf{x})$ and $\mathcal{L}_g(\mathbf{x})$ over all possible embeddings $\mathbf{x}$ in the hypothesis space $\mathcal{H}$. If the above uniform convergence statement is true, then minimizing the training error $\mathcal{L}_t(\mathbf{x})$ would naturally lead to small generalization error $\mathcal{L}_g(\mathbf{x})$ with high probability.

Now we present our first technical result, which follows the above framework and bounds the generalization error of linear graph embedding methods with norm constraints:

Theorem 1. *[Generalization of Linear Graph Embedding with Norm Constraints] Let* $E_\pm = \{(a_1, b_1, y_1), (a_2, b_2, y_2), \ldots, (a_{m+m'}, b_{m+m'}, y_{m+m'})\}$ *be i.i.d. samples from a distribution* $\mathcal{P}$ *over* $V \times U \times \{\pm 1\}$. *Let* $\mathbf{x} : (U \cup V) \to \mathbb{R}^D$ *to be the embedding for nodes in the graph. Then for any bounded* 1*-Lipschitz loss function* $l : \mathbb{R} \to [0, B]$ *and* $C_U, C_V > 0$*, with probability* $1 - \delta$ *(over the sampling of* $E_\pm$*), the following inequality holds*

$$\mathbb{E}_{(a,b,y)\sim\mathcal{P}} l(y \mathbf{x}_a^T \mathbf{x}_b) \leq \frac{1}{m + m'} \sum_{i=1}^{m+m'} l(y_i \mathbf{x}_{a_i}^T \mathbf{x}_{b_i}) + \frac{2\sqrt{C_U C_V}}{m + m'} \mathbb{E}_\sigma ||A_\sigma||_2 + 4B \sqrt{\frac{2 \ln(4/\delta)}{m + m'}} \quad (2)$$

for all embeddings $\mathbf{x}$ *satisfying*

$$\sum_{u \in U} \|\mathbf{x}_u\|^2 \leq C_U, \sum_{v \in V} \|\mathbf{x}_v\|^2 \leq C_V$$

where $||A_\sigma||_2$ *is the spectral norm of the randomized adjacency matrix* A_σ *defined as follows:*

$$A_\sigma(i, j) = \begin{cases} \sigma_{ij} & \exists y, (u_i, v_j, y) \in E_\pm \\ 0 & \forall y, (u_i, v_j, y) \notin E_\pm \end{cases}$$

in which σ_{ij} *are i.i.d. Rademacher random variables.*

The proof can be found in the appendix. Intuitively, Theeorem 1 states that with sufficient norm regularization, linear graph embedding can generalize well regardless of the embedding dimension (note that term D does not appear in Eqn (2) at all). Theorem 1 also characterizes the importance of choosing proper regularization in lnorm restricted inear graph embedding: in Eqn (2), both the training error term $\frac{1}{m+m'} \sum_{i=1}^{m+m'} l(y_i \mathbf{x}_{a_i}^T \mathbf{x}_{b_i})$ and the gap term $\frac{2\sqrt{C_U C_V}}{m+m'} \mathbb{E}_\sigma ||A_\sigma||_2$ are dependent on the value of C_U and C_V. With larger C values (i.e., weak norm regularization), the training error would be smaller due to the less restrictive hypothesis space, but the gap term would larger, meaning that the model will likely overfit the training data. Meanwhile, smaller C values (i.e., strong norm

regularization) would lead to more restrictive models, which will not overfit but have larger training error as trade-off. Therefore, choosing the most proper norm regularization is the key to achieving optimal generalization performance. A rough estimate of $\mathbb{E}_\sigma||A_\sigma||_2$ can be found in the appendix for interested readers.

On the other hand, if we restrict only the embedding dimension (i.e., no norm regularization on embedding vectors), and the embedding dimension is larger than the average degree of the graph, then it is possible for the embedding vectors to severely overfit the training data. The following example demonstrates this possibility on a d-regular graph, in which the embedding vectors can always achieve zero training error even when the edge labels are randomly placed:

Claim 1. *Let $G = (V, E)$ be a d-regular graph with n vertices and $m = nd/2$ labeled edges (with labels $y_i \in \{\pm 1\}$):*

$$V = \{v_1, \ldots, v_n\} \quad E = \{(a_1, b_1, y_1), \ldots, (a_m, b_m, y_m)\}$$

Then for each of the 2^m possible label combinations, there exists a embedding $\mathbf{x} : V \to \mathbb{R}^D$ with dimension $D = d$ that achieves perfect classification accuracy on the randomized training dataset:

$$\forall (y_1, \ldots, y_m) \in \{\pm 1\}^m, \exists \mathbf{x} : V \to \mathbb{R}^D$$

$$\textbf{\textit{s.t.}} \quad \forall i \in \{1, \ldots, m\}, \quad y_i \mathbf{x}_{a_i}^T \mathbf{x}_{b_i} > 1$$

The proof can be found in the appendix. In other words, without norm regularization, the number of training samples required for learning D-dimensional embedding vectors is at least $\Omega(nD)$. Considering the fact that many large-scale graphs are sparse (with average degree < 20) and the default embedding dimension commonly ranges from 100 to 400, it is highly unlikely that the the dimensionality constraint by itself could lead to good generalization performance.

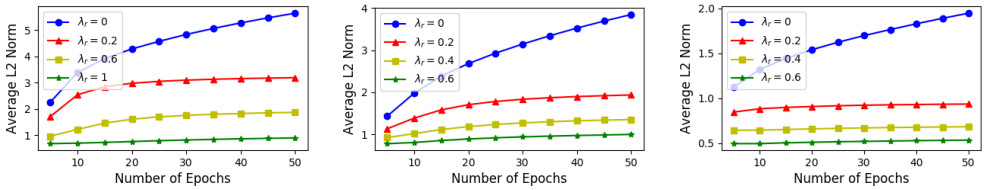
3.2 Empirical Evidence on the Cause of Generalization

In this section, we present several sets of experimental results for the standard linear graph embedding, which collectively suggest that the generalization of these methods are actually determined by vector norm instead of embedding dimension.

Experiment Setting: We use stochastic gradient descent (SGD) to minimize the following objective:

$$L = -\lambda_{+1} \sum_{(u,v)\in E_+} \log \sigma(\mathbf{x}_u^T \mathbf{x}_v) - \lambda_{-1} \sum_{(u,v)\in E_-} \log \sigma(-\mathbf{x}_u^T \mathbf{x}_v) + \lambda_r \sum_{v \in V} ||\mathbf{x}_v||_2^2$$

Here E_+ is the set of edges in the training graph, and E_- is the set of negative edges with both ends sampled uniformly from all vertices. The SGD learning rate is standard: $\gamma_t = (t + c)^{-1/2}$. Three different datasets are used in the experiments: `Tweet`, `BlogCatalog` and `YouTube`, and their details can be found in the appendix. The default embedding dimension is $D = 100$ for all experiments unless stated otherwise.

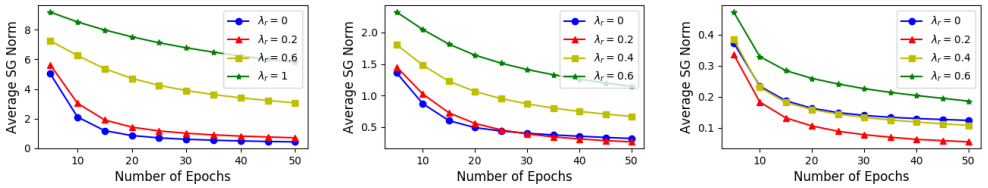


(a) `Tweet` (b) `BlogCatalog` (c) `YouTube`

Figure 1: Average l_2 Norm during SGD Optimization

SGD Optimization Results in Small Vector Norm: Figure 1 shows the average l_2 norm of the embedding vectors during the first 50 SGD epochs (with varying value of λ_r). As we can see, the average norm of embedding vectors increases consistently after each epoch, but the increase rate gets slower as time progresses. In practice, the SGD procedure is often stopped after $10 \sim 50$ epochs (especially for large scale graphs with millions of vertices[4]), and the relatively early stopping time would naturally result in small vector norm.

[4]Each epoch of SGD has time complexity $O(|E|D)$, where D is the embedding dimensionality (usually around 100). Therefore in large scale graphs, even a single epoch would require billions of floating point operations.



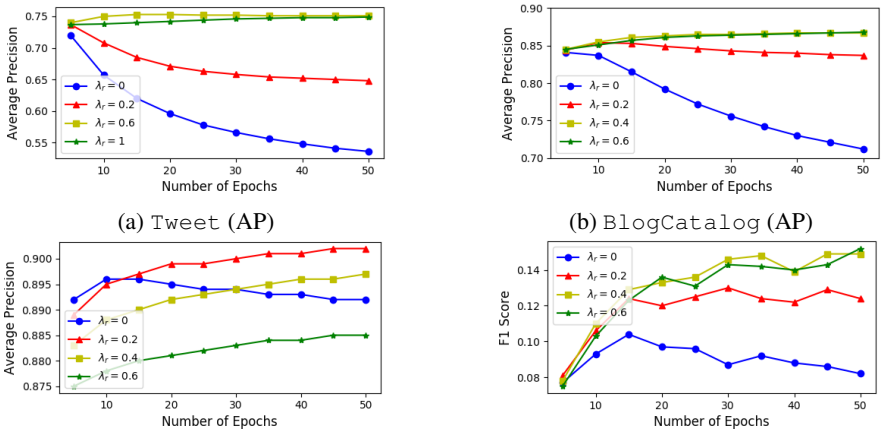
(a) `Tweet` (b) `BlogCatalog` (c) `YouTube`

Figure 2: Average Norm of Stochastic Gradients during SGD Optimization

The Vanishing Gradients: Figure 2 shows the average l_2 norm of the stochastic gradients $\partial L/\partial \mathbf{x}_u$ during the first 50 SGD epochs:

$$\frac{\partial L}{\partial \mathbf{x}_u} = -\sum_{v \in N_+(u)} \sigma(-\mathbf{x}_u^T \mathbf{x}_v)\mathbf{x}_v + \sum_{v \in N_-(u)} \sigma(\mathbf{x}_u^T \mathbf{x}_v)\mathbf{x}_v + 2\lambda_r \mathbf{x}_u \tag{3}$$

From the figure, we can see that the stochastic gradients become smaller during the later stage of SGD, which is consistent with our earlier observation in Figure 1. This phenomenon can be intuitively explained as follows: after a few SGD epochs, most of the training data points have already been well fitted by the embedding vectors, which means that most of the coefficients $\sigma(\pm \mathbf{x}_u^T \mathbf{x}_v)$ in Eqn (3) will be close to 0 afterwards, and as a result the stochastic gradients will be small in the following epochs.



(a) `Tweet` (AP) (b) `BlogCatalog` (AP)

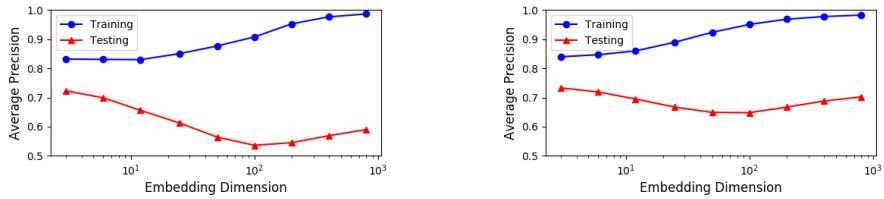
(c) `YouTube` (AP) (d) `BlogCatalog` (F1)

Figure 3: Generalization Performance During SGD

Regularization and Early Stopping: Figure 3 shows the generalization performance of embedding vectors during the first 50 SGD epochs, in which we depicts the resulting average precision (AP) score[5] for link prediction and F1 score for node label classification. As we can see, the generalization performance of embedding vectors starts to drop after $5 \sim 20$ epochs when λ_r is small, indicating that they are overfitting the training dataset afterwards. The generalization performance is worst near the end of SGD execution when $\lambda_r = 0$, which coincides with the fact that embedding vectors in that case also have the largest norm among all settings. Thus, Figure 3 and Figure 1 collectively suggest that the generalization of linear graph embedding is determined by vector norm.

Impact of Embedding Dimension Choice: Figure 4 shows the generalization AP score on `Tweet` dataset with varying value of λ_r and embedding dimension D after 50 epochs. As we can see in Figure 4, without any norm regularization ($\lambda_r = 0$), the embedding vectors will overfit the training dataset for any D greater than 10, which is consistent with our analysis in Claim 1. On the other hand, with larger λ_r, the impact of embedding dimension choice is significantly less noticeable, indicating that the primary factor for generalization is the vector norm in such scenarios.

[5]Average Precision (AP) evaluates the performance on ranking problems: we first compute the precision and recall value at every position in the ranked sequence, and then view the precision $p(r)$ as a function of recall r. The average precision is then computed as $AveP = \int_0^1 p(r)dr$.



(a) $\lambda_r = 0$ (b) $\lambda_r = 0.2$

(c) $\lambda_r = 0.6$ (d) $\lambda_r = 1$

Figure 4: Impact of Embedding Dimension

4 Demonstrating the Importance of Norm Regularization via Hinge-Loss Linear Graph Embedding

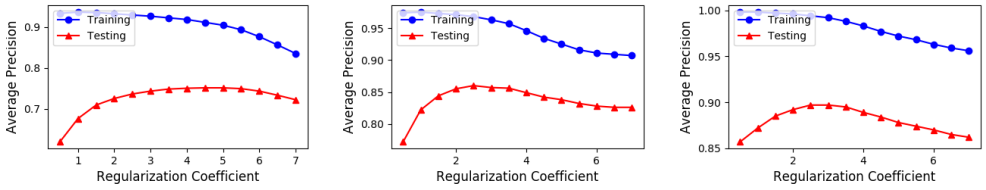
In this section, we present the experimental results for a non-standard linear graph embedding formulation, which optimizes the following objective:

$$L = \lambda_{+1} \sum_{(u,v)\in E_+} h(\mathbf{x}_u^T \mathbf{x}_v) + \lambda_{-1} \sum_{(u,v)\in E_-} h(-\mathbf{x}_u^T \mathbf{x}_v) + \frac{\lambda_r}{2} \sum_{v\in V} ||\mathbf{x}_v||_2^2 \tag{4}$$

By replacing logistic loss with hinge-loss, it is now possible to apply the dual coordinate descent (DCD) method (Hsieh et al., 2008) for optimization, which circumvents the issue of vanishing gradients in SGD, allowing us to directly observe the impact of norm regularization. More specifically, consider all terms in Eqn (4) that are relevant to a particular vertex u:

$$L(u) = \sum_{(\mathbf{x}_i, y_i)\in D} \frac{\lambda_{y_i}}{\lambda_r} \max(1 - y_i \mathbf{x}_u^T \mathbf{x}_i, 0) + \frac{1}{2}\|\mathbf{x}_u\|^2. \tag{5}$$

in which we defined $D = \{(\mathbf{x}_v, +1) : v \in N_+(u)\} \cup \{(\mathbf{x}_k, -1) : k \in N_-(u)\}$. Since Eqn (5) takes the same form as a soft-margin linear SVM objective, with $\mathbf{x}_u$ being the linear coefficients and $(\mathbf{x}_i, y_i)$ being training data, it allows us to use any SVM solver to optimize Eqn (5), and then apply it asynchronously on the graph vertices to update their embeddings. The pseudo-code for the optimization procedure using DCD can be found in the appendix.

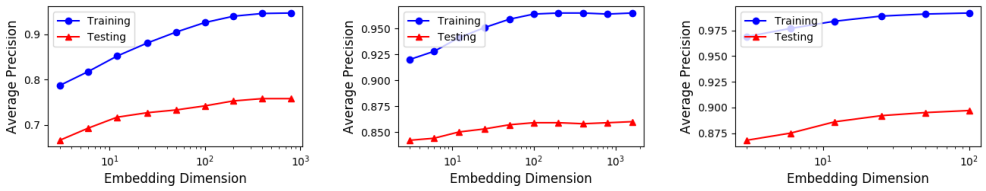


(a) `Tweet` (b) `BlogCatalog` (c) `YouTube`

Figure 5: Generalization Average Precision with Varying λ_r ($D = 100$)

Impact of Regularization Coefficient: Figure 5 shows the generalization performance of embedding vectors obtained from DCD procedure ($\sim$ 20 epochs). As we can see, the quality of embeddings vectors is very bad when $\lambda_r \approx 0$, indicating that proper norm regularization is necessary for generalization. The value of λ_r also affects the gap between training and testing performance, which is consistent with our analysis that λ_r controls the model capacity of linear graph embedding.

Impact of Embedding Dimension Choice: The choice of embedding dimension D on the other hand is not very impactful as demonstrated in Figure 6: as long as D is reasonably large ($\geq$ 30), the exact choice has very little effect on the generalization performance. Even with extremely large embedding dimension setting ($D = 1600$). These results are consistent with our theory that the generalization of linear graph embedding is primarily determined by the norm constraints.



(a) Tweet (b) BlogCatalog (c) YouTube

Figure 6: Generalization Average Precision with Varying D ($\lambda_r = 3$)

5 Discussion

So far, we have seen many pieces of evidence supporting our argument, suggesting that the generalization of embedding vectors in linear graph embedding is determined by the vector norm. Intuitively, it means that these embedding methods are trying to embed the vertices onto a small sphere centered around the origin point. The radius of the sphere controls the model capacity, and choosing proper embedding dimension allows us to control the trade-off between the expressive power of the model and the computation efficiency.

Note that the connection between norm regularization and generalization performance is actually very intuitive. To see this, let us consider the semantic meaning of embedding vectors: the probability of any particular edge (u, v) being positive is equal to

$$\mathbf{Pr}(y = 1|u, v) = \sigma(\mathbf{x}_u^T \mathbf{x}_v) = \sigma(\frac{\mathbf{x}_u^T \mathbf{x}_v}{||\mathbf{x}_u||_2 ||\mathbf{x}_v||_2} ||\mathbf{x}_u||_2 ||\mathbf{x}_v||_2)$$

As we can see, this probability value is determined by three factors:

- $\mathbf{x}_u^T \mathbf{x}_v / (||\mathbf{x}_u||_2 ||\mathbf{x}_v||_2)$, the cosine similarity between $\mathbf{x}_u$ and $\mathbf{x}_v$, evaluates the degree of agreement between the directions of $\mathbf{x}_u$ and $\mathbf{x}_v$.
- $||\mathbf{x}_u||_2$ and $||\mathbf{x}_v||_2$ on the other hand, reflects the degree of confidence we have regarding the embedding vectors of u and v.

Therefore, by restricting the norm of embedding vectors, we are limiting the confidence level that we have regarding the embedding vectors, which is indeed intuitively helpful for preventing overfitting.

It is worth noting that our results in this paper do not invalidate the analysis of Levy & Goldberg (2014), but rather clarifies on some key points: as pointed out by Levy & Goldberg (2014), linear graph embedding methods are indeed approximating the factorization of PMI matrices. However, as we have seen in this paper, the embedding vectors are primarily constrained by their norm instead of embedding dimension, which implies that the resulting factorization is not really a standard low-rank one, but rather a low-norm factorization:

$$\mathbf{x}_u^T \mathbf{x}_v \approx \mathit{PMI}(u, v) \quad \mathit{s.t.} \quad \sum_u ||\mathbf{x}_u||_2^2 \leq C$$

The low-norm factorization represents an interesting alternative to the standard low-rank factorization, and our current understanding of such factorization is still very limited. Given the empirical success of linear graph embedding methods, it would be really helpful if we can have a more in-depth analysis of such factorization, to deepen our understanding and potentially inspire new algorithms.

6 Conclusion

We have shown that the generalization of linear graph embedding methods are not determined by the dimensionality constraint but rather the norm of embedding vectors. We proved that limiting the norm of embedding vectors would lead to good generalization, and showed that the generalization of existing linear graph embedding methods is due to the early stopping of SGD and vanishing gradients. We experimentally investigated the impact embedding dimension choice, and demonstrated that such choice only matters when there is no norm regularization. In most cases, the best generalization performance is obtained by choosing the optimal value for the norm regularization coefficient, and in such case the impact of embedding dimension case is negligible. Our findings combined with the analysis of Levy & Goldberg (2014) suggest that linear graph embedding methods are probably computing a low-norm factorization of the PMI matrix, which is an interesting alternative to the standard low-rank factorization and calls for further study.

REFERENCES

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pp. 585–591, 2001.

Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *KDD*, pp. 855–864, 2016.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pp. 1024–1034, 2017.

Cho-Jui Hsieh, Kai-Wei Chang, Chih-Jen Lin, S. Sathiya Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *ICML*, pp. 408–415, 2008.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Joseph B Kruskal and Myron Wish. *Multidimensional scaling*, volume 11. Sage, 1978.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pp. 2177–2185, 2014.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pp. 3111–3119, 2013.

Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and Analysis of Online Social Networks. In *Proceedings of the 5th ACM/Usenix Internet Measurement Conference (IMC'07)*, San Diego, CA, October 2007.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *KDD*, pp. 701–710, 2014.

Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 459–467. ACM, 2018.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pp. 1329–1336, 2005.

Jian Tang, Meng Qu, and Qiaozhu Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *KDD*, pp. 1165–1174, 2015a.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *WWW*, pp. 1067–1077, 2015b.

Joshua B Tenenbaum, Vin De Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *KDD*, pp. 1225–1234. ACM, 2016.

Sheng Wang, Hyunghoon Cho, ChengXiang Zhai, Bonnie Berger, and Jian Peng. Exploiting ontology graph for predicting sparsely annotated gene function. *Bioinformatics*, 31(12):i357–i364, 2015.

R. Zafarani and H. Liu. Social computing data repository at ASU, 2009. URL `http://socialcomputing.asu.edu`.

APPENDIX

DATASETS AND EXPERIMENTAL PROTOCOLS

We use the following three datasets in our experiments:

- `Tweet` is an undirected graph that encodes keyword co-occurrence relationships using Twitter data: we collected $\sim$1.1 million English tweets using Twitter's Streaming API during 2014 August, and then extracted the most frequent 10,000 keywords as graph nodes and their co-occurrences as edges. All nodes with more than 2,000 neighbors are removed as stop words. There are 9,913 nodes and 681,188 edges in total.

- `BlogCatalog` (Zafarani & Liu, 2009) is an undirected graph that contains the social relationships between BlogCatalog users. It consists of 10,312 nodes and 333,983 undirected edges, and each node belongs to one of the 39 groups.

- `YouTube` (Mislove et al., 2007) is a social network among YouTube users. It includes 500,000 nodes and 3,319,221 undirected edges[6].

For each positive edge in training and testing datasets, we randomly sampled 4 negative edges, which are used for learning the embedding vectors (in training dataset) and evaluating average precision (in testing dataset). In all experiments, $\lambda_+ = 1, \lambda_- = 0.03$, which achieves the optimal generalization performance according to cross-validation. All initial coordinates of embedding vectors are uniformly sampled form $[-0.1, 0.1]$.

OTHER RELATED WORKS

In the early days of graph embedding research, graphs are only used as the intermediate data model for visualization (Kruskal & Wish, 1978) or non-linear dimension reduction (Tenenbaum et al., 2000; Belkin & Niyogi, 2001). Typically, the first step is to construct an affinity graph from the features of the data points, and then the low-dimensional embedding of graph vertices are computed by finding the eigenvectors of the affinity matrix.

For more recent graph embedding techniques, apart from the linear graph embedding methods discussed in this paper, there are also methods (Wang et al., 2016; Kipf & Welling, 2016; Hamilton et al., 2017) that explore the option of using deep neural network structures to compute the embedding vectors. These methods typically try to learn a deep neural network model that takes the raw features of graph vertices to compute their low-dimensional embedding vectors: SDNE (Wang et al., 2016) uses the adjacency list of vertices as input to predict their Laplacian Eigenmaps; GCN (Kipf & Welling, 2016) aggregates the output of neighboring vertices in previous layer to serve as input to the current layer (hence the name "graph convolutional network"); GraphSage (Hamilton et al., 2017) extends GCN by allowing other forms of aggregator (i.e., in addition to the mean aggregator in GCN). Interestingly though, all these methods use only 2 or 3 neural network layers in their experiments, and there is also evidence suggesting that using higher number of layer would result in worse generalization performance (Kipf & Welling, 2016). Therefore, it still feels unclear to us whether the deep neural network structure is really helpful in the task of graph embedding.

Prior to our work, there are some existing research works suggesting that norm constrained graph embedding could generalize well. Srebro et al. (2005) studied the problem of computing norm constrained matrix factorization, and reported superior performance compared to the standard low-rank matrix factorization on several tasks. Given the connection between matrix factorization and linear graph embedding (Levy & Goldberg, 2014), the results in our paper is not really that surprising.

[6] Available at http://socialnetworks.mpi-sws.org/data-imc2007.html. We only used the subgraph induced by the first 500,000 nodes since our machine doesn't have sufficient memory for training the whole graph. The original graph is directed, but we treat it as undirected graph as in Tang et al. (2015b).

PROOF OF THEOREM 1

Since $E_{\pm}$ consists of i.i.d. samples from $\mathcal{P}$, by the uniform convergence theorem (Bartlett & Mendelson, 2002; Shalev-Shwartz & Ben-David, 2014), with probability $1-\delta$:

$$\forall \mathbf{x}, \quad s.t. \quad \sum_{u\in U}\|\mathbf{x}_u\|^2 \le C_U, \sum_{v\in V}\|\mathbf{x}_v\|^2 \le C_V,$$

$$\mathbb{E}_{(a,b,y)\sim\mathcal{P}} l(y\mathbf{x}_a^T\mathbf{x}_b) \le \frac{1}{m+m'}\sum_{i=1}^{m+m'} l(y_i\mathbf{x}_{a_i}^T\mathbf{x}_{b_i}) + 2\mathcal{R}(\mathcal{H}_{C_U,C_V}) + 4B\sqrt{\frac{2\ln(4/\delta)}{m+m'}}$$

where $\mathcal{H}_{C_U,C_V} = \{\mathbf{x} : \sum_{u\in U}\|\mathbf{x}_u\|^2 \le C_U, \sum_{v\in V}\|\mathbf{x}_v\|^2 \le C_V\}$ is the hypothesis set, and $\mathcal{R}(\mathcal{H}_{C_U,C_V})$ is the empirical Rademacher Complexity of $\mathcal{H}_{C_U,C_V}$, which has the following explicit form:

$$\mathcal{R}(\mathcal{H}_{C_U,C_V}) = \frac{1}{m+m'}\mathbb{E}_{\sigma_{a,b}\sim\{-1,1\}}\sup_{\mathbf{x}\in\mathcal{H}_{C_U,C_V}}\sum_i \sigma_{a_i,b_i} l(y_i\mathbf{x}_{a_i}^T\mathbf{x}_{b_i})$$

Here $\sigma_{a,b}$ are i.i.d. Rademacher random variables: $\mathbf{Pr}(\sigma_{a,b}=1) = \mathbf{Pr}(\sigma_{a,b}=-1) = 0.5$. Since l is 1-Lipschitz, based on the Contraction Lemma (Shalev-Shwartz & Ben-David, 2014), we have:

$$\begin{aligned}\mathcal{R}(\mathcal{H}_{C_U,C_V}) \le& \frac{1}{m+m'}\mathbb{E}_{\sigma_{a,b}\sim\{-1,1\}}\sup_{\mathbf{x}\in\mathcal{H}_{C_U,C_V}}\sum_i \sigma_{a_i,b_i} y_i\mathbf{x}_{a_i}^T\mathbf{x}_{b_i}\\ =& \frac{1}{m+m'}\mathbb{E}_{\sigma_{a,b}\sim\{-1,1\}}\sup_{\mathbf{x}\in\mathcal{H}_{C_U,C_V}}\sum_i \sigma_{a_i,b_i}\mathbf{x}_{a_i}^T\mathbf{x}_{b_i}\end{aligned}$$

Let us denote X_U as the $|U|d$ dimensional vector obtained by concatenating all vectors $\mathbf{x}_u$, and X_V as the $|V|d$ dimensional vector obtained by concatenating all vectors $\mathbf{x}_v$:

$$X_U = (\mathbf{x}_{u_1}, \mathbf{x}_{u_2}, \ldots, \mathbf{x}_{u_{|U|}}) \quad X_V = (\mathbf{x}_{v_1}, \mathbf{x}_{v_2}, \ldots, \mathbf{x}_{v_{|V|}})$$

Then we have:

$$||X_U||_2 \le \sqrt{C_U} \quad ||X_V||_2 \le \sqrt{C_V}$$

The next step is to rewrite the term $\sum_i \sigma_{a_i,b_i}\mathbf{x}_{a_i}^T\mathbf{x}_{b_i}$ in matrix form:

$$\begin{aligned}&\sup_{\mathbf{x}\in\mathcal{H}_{C_U,C_V}}\sum_i \sigma_{a_i,b_i}\mathbf{x}_{a_i}^T\mathbf{x}_{b_i}\\ =&\sup_{||X_U||_2\le\sqrt{C_U},||X_V||_2\le\sqrt{C_V}} X_U^T[A_\sigma\otimes I_d]X_V\\ =&\sqrt{C_U}\|A_\sigma\otimes I_d\|_2\sqrt{C_V}\end{aligned}$$

where $A\otimes B$ represents the Kronecker product of A and B, and $||A||_2$ represents the spectral norm of A (i.e., the largest singular value of A).

Finally, since $||A\otimes I||_2 = ||A||_2$, we get the desired result in Theorem 1.

PROOF SKETCH OF CLAIM 1

We provide the sketch of a constructive proof here.

Firstly, we randomly initialize all embedding vectors. Then for each $v\in V$, consider all the relevant constraints to $\mathbf{x}_v$:

$$\mathbf{C}_v = \{(a,b,y)\in E : a = v \text{ or } b = v\}$$

Since G is d-regular, $|\mathbf{C}_v| \le d$. Therefore, there always exists vector $b\in\mathbb{R}^d$ satisfying the following $|C_v|$ constraints:

$$\forall (a,b,y)\in C_v, y\mathbf{x}_a\mathbf{x}_b = 1+\epsilon$$

as long as all the referenced embedding vectors are linearly independent.

Choose any vector b' in a small neighborhood of b that is not the linear combination of any other $d-1$ embedding vectors (this is always possible since the viable set is a d-dimensional sphere minus a finite number of $d-1$ dimensional subspaces), and set $\mathbf{x}_v \leftarrow b'$.

Once we have repeated the above procedure for every node in V, it is easy to see that all the constraints $y\mathbf{x}_a^T\mathbf{x}_b : (a,b,y)\in E$ are now satisfied.

ROUGH ESTIMATION OF $||A_\sigma||_2$ ON ERDOS–RENYI GRAPH

By the definition of spectral norm, $||A_\sigma||_2$ is equal to:

$$||A_\sigma||_2 = \sup_{||\mathbf{x}||_2=||\mathbf{y}||_2=1, \mathbf{x},\mathbf{y}\in\mathbb{R}^n} \mathbf{y}^T A_\sigma \mathbf{x}$$

Note that,

$$\mathbf{y}^T A_\sigma \mathbf{x} = \sum_{(i,j)\in E} \sigma_{ij} y_i x_j$$

Now let us assume that the graph G is generated from a Erdos-Renyi model (i.e., the probability of any pair u, v being directed connected is independent), then we have:

$$\mathbf{y}^T A_\sigma \mathbf{x} = \sum_i \sum_j \sigma_{ij} e_{ij} y_i x_j$$

where e_{ij} is the boolean random variable indicating whether $(i, j) \in E$.

By Central Limit Theorem,

$$\sum_i \sum_j \sigma_{ij} e_{ij} y_i x_j \sim \mathcal{N}(0, \frac{m}{n^2})$$

where m is the expected number of edges, and n is the total number of vertices. Then we have,

$$\mathbf{Pr}(\mathbf{y}^T A_\sigma \mathbf{x} \geq t) \approx O(e^{-\frac{t^2 n^2}{2m}})$$

for all $||\mathbf{x}||_2 = ||\mathbf{y}||_2 = 1$.

Now let S be an ϵ-net of the unit sphere in n dimensional Euclidean space, which has roughly $O(\epsilon^{-n})$ total number of points. Consider any unit vector $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, and let $\mathbf{x}_S, \mathbf{y}_S$ be the closest point of $\mathbf{x}, \mathbf{y}$ in S, then:

$$\begin{aligned}
\mathbf{y}^T A_\sigma \mathbf{x} =& (\mathbf{y}_S + \mathbf{y} - \mathbf{y}_S)^T A_\sigma (\mathbf{x}_S + \mathbf{x} - \mathbf{x}_S) \\
=& \mathbf{y}_S^T A_\sigma \mathbf{x}_S + (\mathbf{y} - \mathbf{y}_S)^T A_\sigma \mathbf{x}_S + \mathbf{y}_S^T A_\sigma (\mathbf{x} - \mathbf{x}_S) + (\mathbf{y} - \mathbf{y}_S)^T A_\sigma (\mathbf{x} - \mathbf{x}_S) \\
\leq& \mathbf{y}_S^T A_\sigma \mathbf{x}_S + 2\epsilon n + \epsilon^2 n
\end{aligned}$$

since $||A_\sigma|| \leq n$ is always true.

By union bound, the probability that at least one pair of $\mathbf{x}_S, \mathbf{y}_S \in S$ satisfying $\mathbf{y}_S^T A_\sigma \mathbf{x}_S \geq t$ is at most:

$$\mathbf{Pr}(\exists \mathbf{x}_S, \mathbf{y}_S \in S : \mathbf{y}_S^T A_\sigma \mathbf{x}_S \geq t) \approx O(\epsilon^{-2n} e^{-\frac{t^2 n^2}{2m}})$$

Let $\epsilon = 1/n, t = \sqrt{8m \ln n/n}$, then the above inequality becomes:

$$\mathbf{Pr}(\exists \mathbf{x}_S, \mathbf{y}_S \in S : \mathbf{y}_S^T A_\sigma \mathbf{x}_S \geq t) \approx O(e^{-n \ln n})$$

Since $\forall \mathbf{x}_S, \mathbf{y}_S \in S, \mathbf{y}_S^T A_\sigma \mathbf{x}_S < t$ implies that

$$\sup_{||\mathbf{x}||_2=||\mathbf{y}||_2=1, \mathbf{x},\mathbf{y}\in\mathbb{R}^n} \mathbf{y}^T A_\sigma \mathbf{x} < t + 2\epsilon n + \epsilon^2 n$$

Therefore, we estimate $||A_\sigma||_2$ to be of order $O(\sqrt{m \ln n/n})$.

PSEUDOCODE OF DUAL COORDINATE DESCENT ALGORITHM

Algorithm 1 shows the full pseudo-code of the DCD method for optimizing the hinge-loss variant of linear graph embedding learning.

Algorithm 1 DCD Method for Hinge-Loss Linear Graph Embedding

function DCDUPDATE($u, N_+(u), N_-(u)$)
  $D = \{(v, +1) : v \in N_+(u)\} \cup \{(v, -1) : v \in N_-(u)\}$
  $w \leftarrow \sum_{(v,s)\in D} \alpha_{uv} s \mathbf{x}_v$
  for $(v, s) \in D$ **do**
    $G \leftarrow sw^T \mathbf{x}_v - 1$
    $U \leftarrow \lambda_s / \lambda_r$
    $$PG \leftarrow \begin{cases} \min(G, 0) & \text{if } \alpha_{uv} = 0 \\ \max(G, 0) & \text{if } \alpha_{uv} = U \\ G & \text{Otherwise} \end{cases}$$
    if $PG \neq 0$ **then**
      $Q = \mathbf{x}_v^T \mathbf{x}_v$
      $\bar{\alpha}_{uv} \leftarrow \alpha_{uv}$
      $\alpha_{uv} \leftarrow \min(\max(\alpha_{uv} - G/Q), 0, U)$
      $w \leftarrow w + (\alpha_{uv} - \bar{\alpha}_{uv}) s x_v$
    end if
  end for
  $\mathbf{x}_u \leftarrow w$
end function
function MAIN($V, E_+, E_-, \lambda_+, \lambda_-, \lambda_r$)
  Randomly initialize $\mathbf{x}_v$ for all $v \in V$
  Initialize $\alpha_{uv} \leftarrow 0$ for all $(u, v) \in E_+ \cup E_-$
  for $t \in 1, \ldots, T$ **do**
    for $u \in V$ **do**
      $N_+(u) \leftarrow \{v \in V : (u, v) \in E_+\}$
      $N_-(u) \leftarrow \{v \in V : (u, v) \in E_-\}$
      DCDUpdate($u, N_+(u), N_-(u)$)
    end for
  end for
end function