Spring-Electrical Models For Link Prediction

Yana Kashinskaya
Yandex School of Data Analysis
Moscow, Russia
kashin.yana@gmail.com

Egor Samosvat*
Yandex
Moscow, Russia
sameg@yandex-team.ru

Akmal Artikov
Yandex
Moscow, Russia
aartikov@yandex-team.ru

ABSTRACT

We propose a link prediction algorithm that is based on spring-electrical models. The idea to study these models came from the fact that spring-electrical models have been successfully used for networks visualization. A good network visualization usually implies that nodes similar in terms of network topology, e.g., connected and/or belonging to one cluster, tend to be visualized close to each other. Therefore, we assumed that the Euclidean distance between nodes in the obtained network layout correlates with a probability of a link between them. We evaluate the proposed method against several popular baselines and demonstrate its flexibility by applying it to undirected, directed and bipartite networks.

CCS CONCEPTS

• **Information systems → Link and co-citation analysis**; • **Human-centered computing → Graph drawings**; • **Computing methodologies → Dimensionality reduction and manifold learning**; *Factorization methods*; *Learning latent representations*;

KEYWORDS

Spring-electrical models, Link prediction, Graph embeddings

ACM Reference Format:
Yana Kashinskaya, Egor Samosvat, and Akmal Artikov. 2019. Spring-Electrical Models For Link Prediction. In *The Twelfth ACM International Conference on Web Search and Data Mining (WSDM '19), February 11–15, 2019, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3289600.3290961

1 INTRODUCTION

Link prediction is usually understood as a problem of predicting missed edges in partially observed networks or predicting edges which will appear in the near future of evolving networks [21]. A prediction is based on the currently observed edges and takes into account a topological structure of the network. Also, there may be some side information or meta-data such as node and edge attributes.

*This work is supported by Russian President grant MK-527.2017.1.

WSDM '19, February 11–15, 2019, Melbourne, VIC, Australia

ACM ISBN 978-1-4503-5940-5/19/02…$15.00
https://doi.org/10.1145/3289600.3290961

The importance of link prediction problem follows naturally from a variety of its practical applications. For example, popular online social networks such as Facebook and LinkedIn suggest a list of people you may know. Many e-commerce websites have personalized recommendations which can be interpreted as predictions of links in bipartite graphs [27]. Link prediction can also help in the reconstruction of some partially studied biological networks by allowing researches to focus on the most probable connections [19].

More formally, in order to evaluate the performance of the proposed link prediction method we consider the following problem formulation. The input is a partially observed graph and our aim is to predict the status (existence or non-existence) of edges for unobserved pairs of nodes. This definition is sometimes called structural link prediction problem [21]. Another possible definition suggests to predict future edges based on the past edges but it is limited to time-evolving networks which have several snapshots.

An extensive survey of link prediction methods can be found in [6, 17, 19]. Here we describe some of the most popular approaches that are usually used as baselines for evaluation [21, 28]. The simplest framework of link prediction methods is the similarity-based algorithm, where to each pair of nodes a score is assigned based on topological properties of the graph [19]. This score should measure similarity (also called proximity) of any two chosen nodes. For example, one such score is the number of common neighbours that two nodes share, because usually if nodes have a lot of common neighbours they tend to be connected with each other and belong to one cluster. Other popular scores are Shortest Distance, Preferential Attachment [3], Jaccard [29] and Adamic-Adar score [2].

Another important class of link prediction methods are latent feature models [1, 7, 20, 21, 24]. The basic idea is to assign each node a vector of latent features in such a way that connected nodes will have similar latent features. Many approaches from this class are based on the matrix factorization technique which gained popularity through its successful application to the Netflix Prize problem [14]. The basic idea is to factor the adjacency matrix of a network into the product of two matrices. The rows and columns of these matrices can be interpreted as latent features of the nodes. Latent features can be also the result of a graph embedding [9]. In particular, there are recent attempts to apply neural networks for this purpose [10, 26].

In this paper, we propose to use spring-electrical models to address the link prediction problem. These models have been successfully used for networks visualization [8, 12, 31]. A good network visualization usually implies that nodes similar in terms of network topology, e.g., connected and/or belonging to one cluster, tend to be visualized close to each other [25]. Therefore, we assumed that the Euclidean distance between nodes in the obtained network layout correlates with a probability of a link between them. Thus, our

idea is to use the described distance as a prediction score. We evaluate the proposed method against several popular baselines and demonstrate its flexibility by applying it to undirected, directed and bipartite networks.

The rest of the paper is organized as follows. First, we formalize the considered problem and present standard metrics for performance evaluation in Section 2 and review related approaches which we used as baselines in Section 3. Next, we discuss spring-electrical models and introduce our method for link prediction in Section 4. We start a comparison of methods with a case study discussed in Section 5. Section 6 presents experiments with undirected networks. We modify the basic model to apply it to bipartite and directed networks in Section 7, followed by conclusion in Section 8.

2 PROBLEM STATEMENT

We focus on the structural definition of the link prediction problem [21]. The network with some missing edges is given and the aim is to predict these missing edges. This definition allows us to work with networks having only a single time snapshot. We also assume that there is no side information such as node or edge attributes, thus, we focus on the link prediction methods based solely on the currently observed link structure.

More formally, suppose that we have a network $G = \langle V, E \rangle$ without multiple edges and loops, where V is the set of nodes and E is the set of edges. We assume that G is a connected graph, otherwise we change it to its largest connected component. The set of all pairs of nodes from V is denoted by U.

Given the network G, we actually do not know its missing edges. Thus, we hide a random subset of edges $E_{pos} \subset E$, while keeping the network connected. The remaining edges are denoted by E_{train}. Also we randomly sample unconnected pairs of nodes E_{neg} from $U \backslash E$. In this way, we form $E_{test} = E_{pos} \cup E_{neg}$ such that $|E_{neg}| = |E_{pos}|$ and $E_{test} \cap E_{train} = \emptyset$. We train models on the network $G' = \langle V, E_{train} \rangle$ and try to find missing edges E_{pos} in E_{test}.

We assume that each algorithm provides a list of scores for all pairs of nodes $(u, v) \in E_{test}$. The $score(u, v)$ characterizes similarity of nodes u and v. The higher the $score(u, v)$, the higher probability of these nodes to be connected is.

To measure the quality of algorithms we use the area under the receiver operating characteristic curve (AUC) [11]. From a probabilistic point of view AUC is the probability that a randomly selected pair of nodes from E_{pos} has higher score than a randomly selected pair of nodes from E_{neg}:

$$\sum_{(u_p, v_p) \in E_{pos}} \sum_{(u_n, v_n) \in E_{neg}} \frac{I\left[score(u_p, v_p) > score(u_n, v_n)\right]}{|E_{pos}| \cdot |E_{neg}|},$$

where $I[\cdot]$ denotes an indicator function.

We repeat the evaluation several times in order to compute the mean AUC as well as the standard deviation of AUC.

3 RELATED WORK

In this section we describe some popular approaches to link prediction problem. The mentioned methods will be used as baselines during our experiments.

3.1 Local Similarity Indices

Local similarity-based methods calculate $score(u, v)$ by analyzing direct neighbours of u and v based on different assumptions about link formation behavior. We use $\delta(u)$ to denote the set of neighbours of u.

The assumption of *Common Neighbours index* is that a pair of nodes has a higher probability to be connected if they share many common neighbours

$$CN(u, v) := |\delta(u) \cap \delta(v)|.$$

Adamic-Adar index is a weighted version of Common Neighbours index

$$AA(u, v) := \sum_{z \in \delta(u) \cap \delta(v)} \frac{1}{|\delta(z)|}.$$

The weight of a common neighbour is inversely proportional to its degree.

Preferential Attachment index is motivated by Barabási–Albert model [3] which assumes that the ability of a node to obtain new connections correlates with its current degree,

$$PA(u, v) := |\delta(u)| \cdot |\delta(v)|.$$

Our choice of these three local similarity indices is based on the methods comparison conducted in [17, 21].

3.2 Matrix Factorization

Matrix factorization approach is extensively used for link prediction problem [1, 7, 20, 21, 24]. The adjacency matrix of the network is approximately factorized into the product of two matrices with smaller ranks. Rows and columns of these matrices can be interpreted as latent features of the nodes and the predicted score for a pair of nodes is a dot-product of corresponding latent vectors.

A truncated singular value decomposition (Truncated SVD) of matrix $A \in R^{m \times n}$ is a factorization of the form $A_r = U_r \Sigma_r V_r^T$, where $U_r \in R^{m \times r}$ has orthonormal columns, $\Sigma_r = diag(\sigma_1, ..., \sigma_r) \in R^{r \times r}$ is diagonal matrix with $\sigma_i \geq 0$ and $V_r \in R^{n \times r}$ also has orthonormal columns [13]. Actually it solves the following optimization problem:

$$\min_{A_r : rank(A_r) \leq r} \|A - U_r \Sigma_r V_r^T\|_F = \sqrt{\sigma_{r+1}^2 + ... + \sigma_n^2},$$

where $\sigma_1, ..., \sigma_n$ are singular values of the matrix A. To cope with sparse matrices we use `scipy.sparse.linalg.svds`[1] implementation of Truncated SVD based on the implicitly restarted Arnoldi method [30].

Another popular approach for training latent features is a non-negative matrix factorization (NMF). NMF with r components is a group of algorithms where a matrix $A \in R^{n \times m}$ is factorized into two matrices $W_r \in R^{n \times r}$ and $H_r \in R^{m \times r}$ with the property that all three matrices have non-negative elements [18]:

$$\min_{W_r, H_r : W_r \geq 0, H_r \geq 0} \|A - W_r H_r^T\|_F.$$

These conditions are consistent with the non-negativity of the adjacency matrix in our problem. We take as a baseline alternating non-negative least squares method with coordinate descent optimization approach [5] from `sklearn.decomposition.NMF`[2].

[1] https://docs.scipy.org/doc/scipy-0.18.1/reference/generated/scipy.sparse.linalg.svds.html
[2] http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html

3.3 Neural Embedding

Several attempts to apply neural networks for graph embedding, such as DeepWalk [26] and node2vec [10], were motivated by word2vec, a widely used algorithm for extracting vector representations of words [23]. The general idea of adopting word2vec for graph embedding is to treat nodes as "words" and generate "sentences" using random walks.

The objective is to maximize likelihood of observed nodes co-occurrences in random walks. Probability of nodes u and v with latent vectors x_u and x_v to co-occur in a random walk is estimated using a softmax:

$$P(u|v) = \frac{\exp(x_u \cdot x_v)}{\sum_{w \in V} \exp(x_w \cdot x_v)}.$$

In practice a direct computation of the softmax is infeasible, thus, some approximations, such as a "negative sampling" or a "hierarchical softmax", are used [22].

In this paper we consider node2vec which has shown a good performance in the link prediction [10]. This method generates 2nd order random walks $c_1, \ldots, c_n$ with a transition probability defined by the following relation:

$$P\left(c_i = x | c_{i-1} = v, c_{i-2} = t\right) \propto \begin{cases} 0, \text{ if } (v, x) \notin E \\ \frac{1}{p}, \text{ else if } d_{tx} = 0 \\ 1, \text{ else if } d_{tx} = 1 \\ \frac{1}{q}, \text{ else if } d_{tx} = 2 \end{cases}$$

where d_{tx} is the graph distance between nodes t and x. The parameters p and q allows to interpolate between walks that are more akin to breadth-first or depth-first search. Generated random walks are given as input to word2vec. Finally, for each node u a vector x_u is assigned.

In order to estimate $score(u, v)$ we compute the dot-product of the corresponding latent vectors:

$$node2vec(u, v) := x_u \cdot x_v.$$

We have used a reference implementation of node2vec[3] with default parameters unless stated otherwise.

4 SPRING-ELECTRICAL MODELS FOR LINK PREDICTION

Currently the main application of spring-electrical models to graph analisys is a graph visualization. The basic idea is to represent a graph as a mechanical system of like charges connected by springs [8]. In this system between each pair of nodes act repulsive forces and between adjacent nodes act attractive forces. In an equilibrium state of the system, the edges tend to have uniform length (because of the spring forces), and nodes that are not connected tend to be drawn further apart (because of the electrical repulsion).

Actually, in practice edge attraction and vertex repulsion forces may be defined using functions that are not precisely based on the Hooke's and Coulomb's laws. For instance, in [8], the pioneering work of Fruchterman and Reingold, repulsive forces are inversely proportional to the distance and attractive forces are proportional to the square of the distance. In [31] and [12] spring-electrical models were further studied and the repulsive force got new parameters C, K and p, which we will discuss later. In our research, we will also use their modification with the following forces:

$$\begin{aligned} f_r(u, v) &= -CK^{1+p}/||x_u - x_v||^p, \quad p > 0, u \neq v; u, v \in V, \\ f_a(u, v) &= ||x_u - x_v||^2/K, \quad (u, v) \in E; u, v \in V. \end{aligned} \tag{1}$$

Here we denote by $\|x_u - x_v\|$ the Euclidean distance between coordinate vectors of the nodes u and v in a layout, and by $f_r(u, v)$ and $f_a(u, v)$ we denote values of repulsive and attractive forces, respectively. Figure 1 illustrates the forces acting on one of the nodes in a simple star graph.

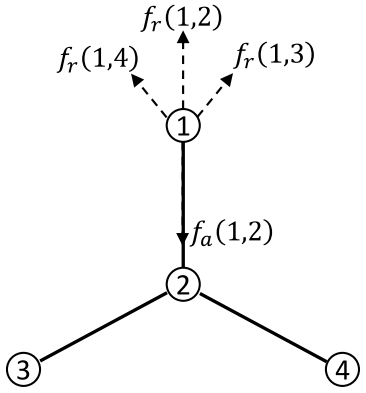


Figure 1: Spring-electrical model

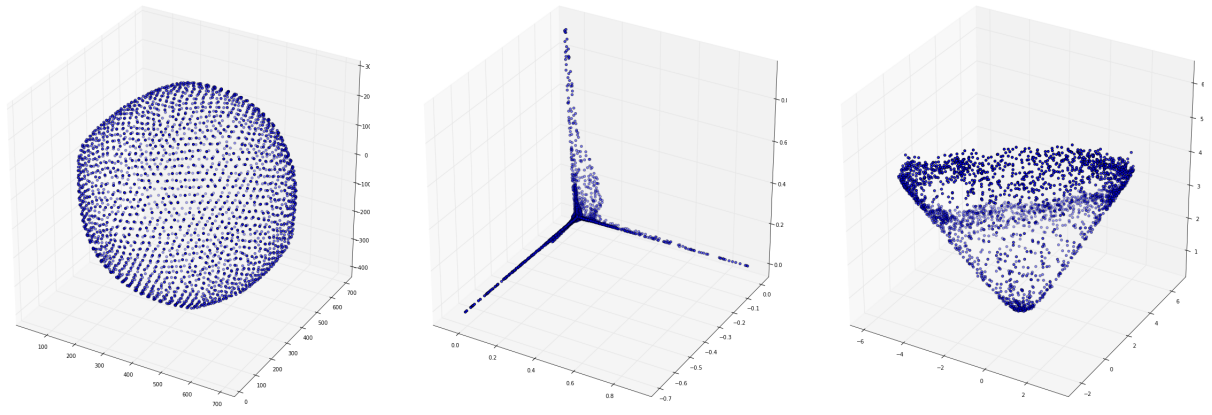
By exploiting that force is the negative gradient of energy, the force model can be transformed into an energy model, such that force equilibria correspond to (local) energy minima [25]. An optimal layout is achieved in the equilibrium state of the system with the minimum value of the system energy. Thus, finding of an optimal layout is actually the following optimization problem:

$$\min_{\{x_w, w \in V\}} \left(\sum_{(u,v) \in E} \frac{||x_u - x_v||^3}{3K} + \sum_{\substack{u, v \in V \\ u \neq v}} \frac{\frac{1}{p-1} CK^{1+p}}{||x_u - x_v||^{p-1}} \right). \tag{2}$$

A lot of algorithms were proposed to find the optimal layout. All of them meet two main challenges: (i) a computational complexity of the repulsive forces, (ii) a slow convergence speed and trapping in local minima. Note that local minima of the energy might lead to layouts with bad visualization characteristics. We will use Scalable Force Directed Placement (SFDP) algorithm described in [12], which is able to overcome both challenges.

The computational complexity challenge in SFDP is solved by Barnes-Hut optimization [4]. As a result, the total complexity of all repulsive forces calculation reduces to $O(|V| \log |V|)$, compared to a straightforward method with complexity $O(|V|^2)$. The second challenge is solved by a multilevel approach in combination with an adaptive cooling scheme [12, 31]. The idea is to iteratively coarse the network until the network size falls below some threshold.

[3] https://github.com/aditya-grover/node2vec



(a) SFDP (b) SVD (c) node2vec

Figure 2: Triangulation of 3d-sphere (3d latent features)

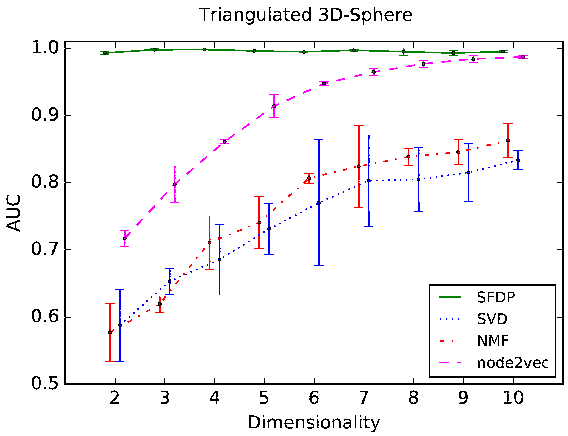


Figure 3: Triangulation of 3d-sphere (AUC scores)

Once the initial layout for the coarsest network is found, it is successively refined and extended to all the levels starting with the coarsest networks and ending with the original.

Let us now discuss how the model parameters influence on the equilibrium states of the system. According to Theorem 1 in [12], parameters K and C do not change possible equilibriums and only scale them, however, they influence on the speed of convergence to an equilibrium. As it follows from equation (1), a parameter p controls the strength of repulsive forces. For small values of p, nodes on the periphery of a layout are affected strongly by repulsive forces of central nodes. This leads to so-called "peripheral effect": edges at the periphery are longer than edges at the center [12]. On the other hand, despite larger values of p reduce "peripheral effect", too small repulsive forces might lead to clusters collapsing [25]. We study influence of the repulsive force exponent p on the performance of our method in Section 6.

A good network visualization usually implies that nodes similar in terms of network topology, e.g., connected and/or belonging to one cluster, tend to be visualized close to each other [25]. Therefore, we assumed that the Euclidean distance between nodes in the obtained network layout correlates with a probability of a link between them. Thus, to address the link prediction problem, we first find a network layout using SFDP and then use the distance between nodes as a prediction score:

$$\mathrm{SFDP}(u, v) = \|x_u - x_v\|.$$

Our method can be interpreted as a latent feature approach to link prediction.

5 CASE STUDY

Before discussing experiments with real-world networks, we consider a graph obtained by triangulation of a three-dimensional sphere. This case study reveals an important difference between SFDP and other baselines which use latent feature space.

First, we have trained three-dimensional latent vectors and visualized them (see Figure 2). SFDP arranges latent vectors on the surface of a sphere as one might expect. SVD's latent vectors form three mutually perpendicular rays and node2vec places latent vectors on the surface of a cone. The reason of such a different behavior is that SFDP uses the Euclidean distance in its loss function (see equation (2)), while node2vec and SVD relies on the dot-product. One can see that dot-product based methods fail to express the fact that all nodes in the considered graph are structurally equivalent.

The difference between the dot-product and the Euclidean distance is in the way they deal with latent vectors corresponding to completely unrelated nodes. The dot-product tends to make such vectors perpendicular, while the Euclidean distance just places them "far away". The number of available mutually perpendicular direction is determined by the dimensionality of the latent feature space. As the result, dimensionality becomes restrictive for dot-product based methods.

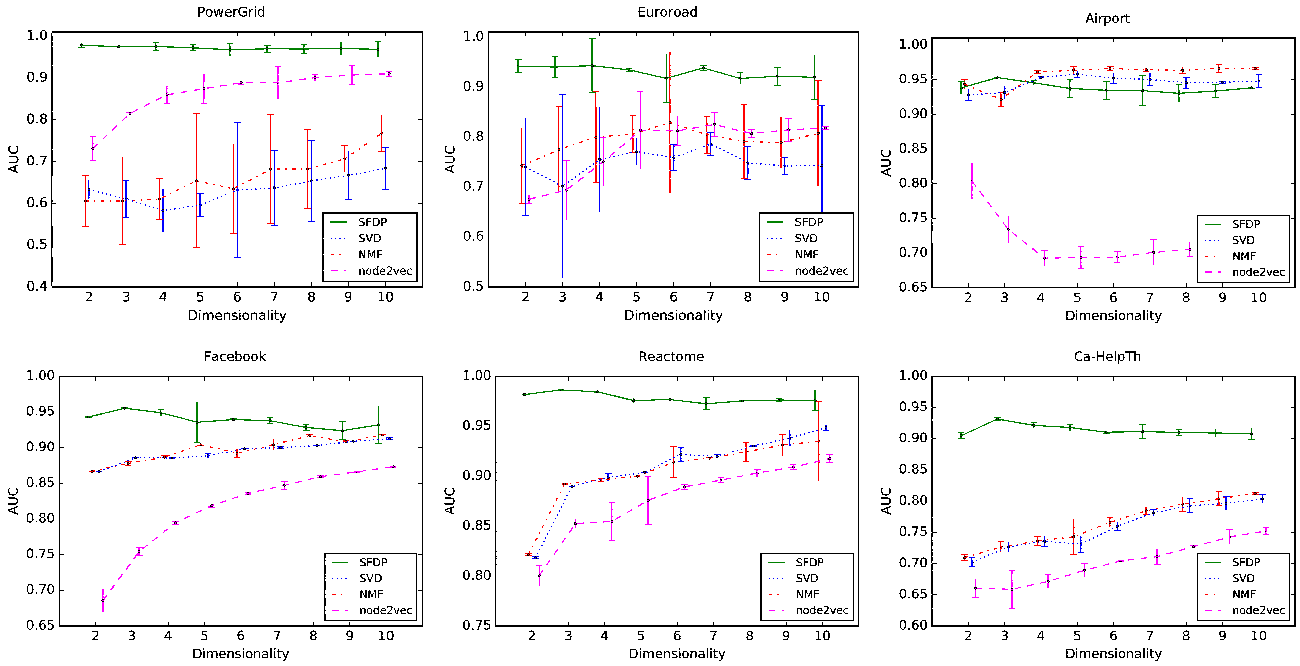


Figure 4: Influence of dimensionality

In order to further support this observation we have evaluated AUC score of the discussed methods depending on the dimensionality of the latent feature space. The results are presented on Figure 3. SFDP has good quality starting from very low dimensions. Node2vec achieves a reasonable quality in the ten-dimensional latent feature space, while SVD and NMF needs about 100 dimensions.

Our experiments with real-world networks, described above, confirm that SFDP might have a competitive quality of link prediction even in very low dimensions. This advantage might lead to some practical applications as many problems related to vector embedding are much easier in low dimensions, e.g., searching of the nearest neighbors.

6 EXPERIMENTS WITH UNDIRECTED NETWORKS

First, we have chosen several undirected networks in which geographical closeness correlates with the probability to obtain a connection. Thus, the ability to infer a distance feature can be tested on them.

- "PowerGrid" [15] is an undirected and unweighted network representing the US electric powergrid network. There are 4,941 nodes and 6,594 supplies in the system. It is a sparse network with average degree 2.7.
- "Euroroad" [15] is a road network located mostly in Europe. Nodes represent cities and an edge between two nodes denotes that they are connected by a road. This network consists of 1,174 vertices (cities) and 1,417 edges (roads).
- "Airport" [15] has information about 28,236 flights between 1,574 US airports in 2010. Airport network has hubs, i.e. several busiest airports. Thus, the connections occur not only because of geographical closeness, but also based on the airport sizes.

We have also chosen several undirected networks of other types.

- "Facebook" [15] is a Facebook friendship network consist of 817,035 friendships and 63,731 users. This network is a subset of full Facebook friendship graph.
- "Reactome" [15] has information about 147,547 interactions between 6,327 proteins.
- "Ca-HepTh" [16] is a collaboration network from the arXiv High Energy Physics - Theory section from January 1993 to April 2003. The network has 9,877 authors and 25,998 collaborations.

All the datasets and our source code are available in our GitHub repository[4]. In our experiments we have used two implementations of SFDP, from `graphviz`[5] and `graph_tool`[6] libraries, with the default parameters unless stated otherwise.

Following the discussion in Section 5, we have first analyzed the behavior of latent feature methods in low dimensions. On the sparse datasets "PowerGrid" and "Euroroad" we hide 10% of edges in order to keep a train network connected. On other datasets 30% of edges were included in a test set. We repeat this process several times and report mean AUC scores as well as 95% confidence intervals. The results are presented on Figure 4.

[4]https://github.com/KashinYana/link-prediction
[5]http://www.graphviz.org
[6]https://graph-tool.skewed.de/

Table 1: Comparison with latent features models

Dataset	SFDP	SVD	NMF	node2vec
PowerGrid	2d **0.978**±0.005	30d 0.848±0.007	40d 0.913±0.009	50d **0.931**±0.011
Euroroad	2d **0.941**±0.012	7d 0.785±0.023	6d 0.829 ± 0.037	75d 0.871±0.021
Airport	3d **0.953**±0.000	5d **0.957**±0.005	6d **0.966**± 0.003	2d 0.804±0.026
Facebook	3d **0.951**±0.000	20d 0.922±0.000	500d **0.959**±0.001	150d 0.935±0.000
Reactome	3d **0.986**±0.000	100d **0.987** ±0.001	125d **0.993** ±0.000	100d 0.954±0.000
Ca-HepTh	3d **0.931**±0.004	100d 0.856± 0.005	150d **0.921**±0.007	125d 0.884±0.006

As it was expected the dot-product based methods have a clear growing trend on the most of networks, with lower performance in low dimensions. In contrast, SFDP has a good quality starting from two dimensions usually with a slight increase at dimensionality equals three and a slowly decreasing trend after this point.

We have also studied higher dimensions (up to 500 dimensions). In Table 1 for each method and dataset one can find an optimal dimension with the corresponding AUC score, standard deviation values smaller than 0.0005 are shown as zero. Surprisingly SFDP demonstrates competitive quality in comparison even with high-dimensional dot-product based methods. This observation suggests that real networks might have a lower inherit dimensionality than one might expect.

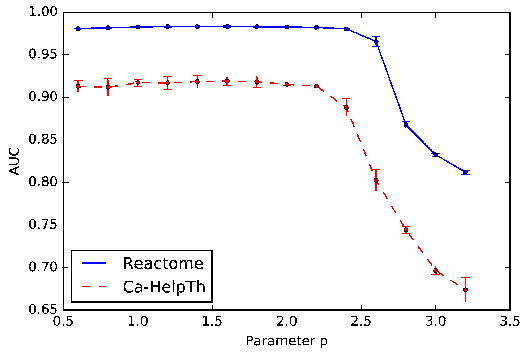


Figure 5: Influence of the repulsive force exponent

As the influence of dimensionality on the performance of SFDP is not very significant we further focused on the two-dimensional SFDP. Speaking about other parameters, according to Section 4 parameters C and K do not change SFDP performance regarding link prediction since they only scale the optimal layout. Thus, we tried to vary the parameter p. Based on Figure 5, we have decided to continue using of the default value of the repulsive force exponent p which equals 2.

Table 2: Comparison with local similarity indices

Dataset	SFDP 2d	PA	CN	AA
PowerGrid	**0.978**±0.005	0.576±0.005	0.625±0.006	0.625±0.006
Euroroad	**0.941**±0.012	0.432±0.015	0.535±0.011	0.534±0.011
Airport	0.938±0.001	0.949 ±0.000	**0.959**±0.000	**0.962**±0.001
Facebook	**0.943**±0.000	0.887±0.000	**0.915**±0.000	**0.915**±0.000
Reactome	**0.981**±0.000	0.899±0.001	**0.988**±0.000	**0.989**±0.000
Ca-HepTh	**0.905**±0.001	0.787±0.000	0.867±0.002	0.867±0.002

Finally, we have compared SFDP with local similarity indices. The results can be found in Table 2. SFDP has shown the highest advance on the geographical networks "PowerGrid" and "Euroroad". This observation supports our hypothesis that SFDP can infer geographical distance. The result of SFDP on the "Airport" network is not so good and we link this fact to the presence of distinct hubs, we will return to this network in Section 7.2. On other datasets spring-electrical approach has shown superior or competitive quality.

7 MODEL MODIFICATIONS

The basic spring-electrical model can be adapted to different networks types. In this section we will present possible modifications for bipartite and directed networks.

7.1 Bipartite Networks

A bipartite network is a network which nodes can be divided into two disjoint sets such that edges connect nodes only from different sets. It is interesting to study this special case because the link prediction in bipartite networks is close to a collaborative filtering problem.

We use the following bipartite datasets in our experiments:

- "Movielens" [15] dataset contains information how users rated movies on the website http://movielens.umn.edu/. The version of the dataset which we used has 9, 746 users, 6, 040 movies and 1 million ratings. Since we are not interested in a rating score, we assign one value to all edge weights.
- "Frwiki" [15] is a network of 201, 727 edit events done by 30, 997 users in 2, 884 articles of the French Wikipedia.
- "Condmat" [15] is an authorship network from the arXiv condensed matter section (cond-mat) from 1995 to 1999. It contains 58, 595 edges which connect 16, 726 publications and 38, 741 authors.

Let us consider the "Movielens" dataset. This network has two types of nodes: users and movies. When applying SFDP model to this network we expect movies of the same topic to be placed nearby. Similarly, we expect users which may rate the same movie to be located closely. In the basic spring-electrical models the repulsive forces are assigned between all nodes. It works good for visualization purpose, but it hinders formation of cluster by users and movies. Therefore, we removed repulsive forces between nodes of the same type.

Consider a bipartite network $G = \langle V, E \rangle$, which nodes are partitioned into two subsets $V = L \sqcup R$ such that $E \subset L \times R$. In our modification of SFDP model for bipartite networks, denoted Bi-SFDP,

Table 3: AUC scores for bipartite datasets, $|E_{pos}|/|E| = 0.3$

Dataset	Bi-SFDP 2d	SFDP 2d	PA	NMF 100d	SVD 100d	node2vec 100d
Movielens	0.758 ±0.001	0.755±0.005	0.773±0.000	**0.870**±0.002	**0.876**±0.000	0.725 ±0.000
Condmat	0.682±0.007	**0.910**±0.002	0.617±0.001	0.819±0.005	0.787±0.004	**0.912** ±0.001
Frwiki	**0.828**±0.005	0.745±0.002	**0.800**±0.000	0.544±0.027	0.571±0.005	0.508 ± 0.003

the following forces are assigned between nodes.

$$f_r(u,v) = -CK^{(1+p)}/||x_u - x_v||^p, \quad p > 0, u \in L, v \in R,$$
$$f_a(u,v) = ||x_u - x_v||^2/K, \quad (u,v) \in E; u \in L, v \in R. \tag{3}$$

To carry out experiments with Bi-SFDP model we have written a patch for `graph_tools` library.

Figure 6 demonstrates how this modification affects the optimal layout. We consider a small user–movie network of ten users and three movies. Note that on Figure 6 (b) some of the yellow nodes were collapsed. The reason is that if we remove repulsive forces between nodes of the same type, users which link to the same movies will have the same positions. Thus, Bi-SFDP model could assigned close positions to users with the same interests.

During our preliminary experiments with bipartite networks we have found that PA demonstrates too high results. The reason is that the preferential attachment mechanism is too strong in bipartite networks. In order to focus on thinner effects governing links formation, we have changed a way to sample the set of negative pairs of nodes E_{neg}. A half of pairs of nodes in E_{neg} were sampled with probability proportional to the product of its degrees, such pairs of nodes are counterexamples for the preferential attachment mechanism.

The results of the experiments are summarized in Table 3. Baselines CN and AA are not included in the table, because their scores are always equal to zero on bipartite networks.

Bi-SFDP modification has shown the increase in quality compared with the basic SFDP model on "Movielens" and "Frwiki" datasets. It means that our assumption works for these networks. In contrast, on the "Condmat" standard SFDP and node2vec outperforms all other baselines. In general, the results for bipartite graphs are very dataset-dependent.

7.2 Directed Networks

Spring-electrical models are not suitable for predicting links in directed networks because of the symmetry of the forces and the distance function. Therefore, we first propose to transform the original directed network.

Given a directed network $G = \langle V, E\rangle$, we obtain an undirected bipartite network $G' = \langle V', E'\rangle$, $V' = L \sqcup R$, $E' \subset L \times R$ by the following process. Each node $u \in V$ corresponds to two nodes $u_{out} \in L$ and $u_{in} \in R$. One of the nodes is responsible for outgoing connections, the other one for incoming connections. Thus, for each directed edge $(u,v) \in E$ an edge (u_{out}, v_{in}) is added in E'. Figure 7 illustrates the described transformation. As the result, G' has information about the all directed edges of G.

Then Bi-SFDP model can be applied to find a layout of the network G'. Finally, prediction scores for pairs of nodes from the network G can be easily inherited from the layout of the network G'.

We have called this approach Di-SFDP and have tested on the following datasets.

- "Twitter" [15] is a user-user network, where directed edges represent the fact that one user follows the other user. The network contains 23, 370 users and 33, 101 follows.
- "Google+" [15] is also a user-user network. Directed links indicate that one user has the other user in his circles. There are 23, 628 users and 39, 242 friendships in the network.
- "Cit-HepTh" [15] has information about 352, 807 citations among 27, 770 publications in the arXiv High Energy Physics Theory (hep-th) section.

All pairs of nodes (u,v) such that $(v,u) \in E$ but $(u,v) \notin E$ we call *difficult pairs*. They can not be correctly scored by the basic SFDP model. It is especially interesting to validate models on such pairs of nodes. Therefore, in our experiments a half of pairs of nodes in E_{neg} are difficult pairs.

The experiment results are shown in Table 4. The baselines PA, CC and AA can be also calculated on G', but their quality is close to a random predictor. One can see that Di-SFDP outperforms other baselines on two of the datasets and have a competitive quality on the last one. Note that out-of-box node2vec can not correctly score difficult pairs of nodes as it infers only one latent vector for each node, while other methods has two latent vectors, one is responsible for outgoing connections and another one for incoming connections.

Di-SFDP has also helped us to improve quality on the "Airport" dataset. Despite "Airport" is an undirected network, due to presence of hubs it has a natural orientation of edges. Thus, our idea was to first orient edges from the nodes of low degrees to the nodes of high degrees and then apply Di-SFDP. This trick allowed us to improve the mean AUC from 0.938 to 0.972.

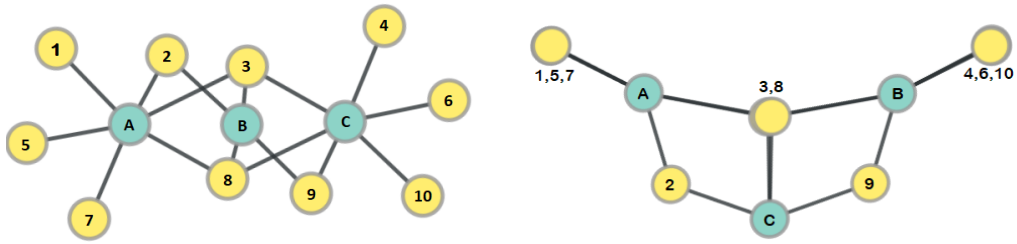
Table 4: AUC scores for directed datasets, $|E_{pos}|/|E| = 0.3$

Dataset	Di-SFDP 2d	NMF 100d	SVD 100d	node2vec 100d
Twitter	**0.952** ±0.002	0.783±0.010	0.694±0.014	0.550 ± 0.001
Google+	**0.998** ±0.000	**0.936**±0.007	0.466±0.033	0.449 ± 0.002
Cit-HepTh	**0.836** ±0.003	**0.842**±0.002	**0.838**±0.001	0.679±0.000

Dataset	Di-SFDP 2d	SFDP 2d
Airport	**0.972** ±0.001	0.938±0.001

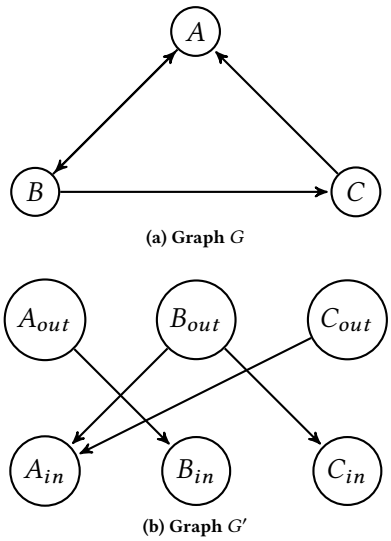
8 CONCLUSION

In this paper we proposed to use spring-electrical models to address the link prediction problem. We first applied the basic SFDP model to the link prediction in undirected networks and then adapted



(a) SFPD (b) Bi-SFDP

Figure 6: The visualization of a bipartite network by SFDP and Bi-SFDP



(a) Graph G

(b) Graph G'

Figure 7: Di-SFDP graph transformation

it to bipartite and directed networks by introducing two novel methods, Bi-SFDP and Di-SFDP. The considered models demonstrates superior or competitive performance of our approach over several popular baselines.

A distinctive feature of the proposed method in comparison with other latent feature models is a good performance even in very low dimensions. This advantage might lead to some practical applications as many problems related to vector embedding are much easier in low dimensions, e.g., searching of the nearest neighbors. On the other hand this observation suggests that real networks might have lower inherit dimensionality than one might expect.

We consider this work as a good motivation towards a new set of research directions. Future research can be focused on choosing an optimal distance measure for latent feature models and deeper analysis of inherit networks dimensionality.

REFERENCES

[1] Evrim Acar, Daniel M Dunlavy, and Tamara G Kolda. 2009. Link prediction on evolving data using matrix and tensor factorizations. In *Data Mining Workshops, 2009. ICDMW'09. IEEE International Conference on*. IEEE, 262–269.

[2] Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks* 25, 3 (2003), 211–230.

[3] Albert-László Barabási and Réka Albert. 1999. Emergence of scaling in random networks. *science* 286, 5439 (1999), 509–512.

[4] Josh Barnes and Piet Hut. 1986. A hierarchical O (N log N) force-calculation algorithm. *nature* 324, 6096 (1986), 446–449.

[5] Andrzej Cichocki and PHAN Anh-Huy. 2009. Fast local algorithms for large scale nonnegative matrix and tensor factorizations. *IEICE transactions on fundamentals of electronics, communications and computer sciences* 92, 3 (2009), 708–721.

[6] William Cukierski, Benjamin Hamner, and Bo Yang. 2011. Graph-based features for supervised link prediction. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 1237–1244.

[7] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5, 2 (2011), 10.

[8] Thomas MJ Fruchterman and Edward M Reingold. 1991. Graph drawing by force-directed placement. *Software: Practice and experience* 21, 11 (1991), 1129–1164.

[9] Palash Goyal and Emilio Ferrara. 2017. Graph Embedding Techniques, Applications, and Performance: A Survey. *arXiv preprint arXiv:1705.02801* (2017).

[10] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 855–864.

[11] James A Hanley and Barbara J McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology* 143, 1 (1982), 29–36.

[12] Yifan Hu. 2005. Efficient, high-quality force-directed graph drawing. *Mathematica Journal* 10, 1 (2005), 37–71.

[13] Virginia Klema and Alan Laub. 1980. The singular value decomposition: Its computation and some applications. *IEEE transactions on automatic control* 25, 2 (1980), 164–176.

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[15] Jérôme Kunegis. 2013. KONECT – The Koblenz Network Collection. In *Proc. Int. Conf. on World Wide Web Companion*. 1343–1350. http://userpages.uni-koblenz.de/~kunegis/paper/kunegis-koblenz-network-collection.pdf

[16] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data. (June 2014).

[17] David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology* 58, 7 (2007), 1019–1031.

[18] Chih-Jen Lin. 2007. Projected gradient methods for nonnegative matrix factorization. *Neural computation* 19, 10 (2007), 2756–2779.

[19] Linyuan Lü and Tao Zhou. 2011. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications* 390, 6 (2011), 1150–1170.

[20] Aditya Krishna Menon and Charles Elkan. 2010. A log-linear model with latent features for dyadic prediction. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 364–373.

[21] Aditya Krishna Menon and Charles Elkan. 2011. Link prediction via matrix factorization. In *Joint european conference on machine learning and knowledge discovery in databases*. Springer, 437–452.

[22] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[24] Kurt Miller, Michael I Jordan, and Thomas L Griffiths. 2009. Nonparametric latent feature models for link prediction. In *Advances in neural information processing systems*. 1276–1284.

[25] Andreas Noack. 2009. Modularity clustering is force-directed layout. *Physical Review E* 79, 2 (2009), 026102.

[26] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.
[27] J Ben Schafer, Joseph Konstan, and John Riedl. 1999. Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*. ACM, 158–166.
[28] Ehsan Sherkat, Maseud Rahgozar, and Masoud Asadpour. 2015. Structural link prediction based on ant colony approach in social networks. *Physica A: Statistical Mechanics and its Applications* 419 (2015), 80–94.
[29] Henry Small. 1973. Co-citation in the scientific literature: A new measure of the relationship between two documents. *Journal of the Association for Information Science and Technology* 24, 4 (1973), 265–269.
[30] Charles F Van Loan. 1996. Matrix computations (Johns Hopkins studies in mathematical sciences). (1996).
[31] Chris Walshaw. 2000. A multilevel algorithm for force-directed graph drawing. In *International Symposium on Graph Drawing*. Springer, 171–182.