Efficient Algorithms for Constructing Multiplex Networks Embedding*

Pavel Zolnikov[1], Maxim Zubov[2], Nikita Nikitinsky[2], and Ilya Makarov[1]**[0000−0002−3308−8825]

[1] National Research University Higher School of Economics, Moscow, Russia
[2] National University of Science and Technology MISIS, Moscow, Russia
pasha.zolnikov@gmail.com,zubovmv@gmail.com,torselllo@yandex.ru,iamakarov@hse.ru

Abstract. Network embedding has become a very promising technique in analysis of complex networks. It is a method to project nodes of a network into a low-dimensional vector space while retaining the structure of the network based on vector similarity. There are many methods of network embedding developed for traditional single layer networks. On the other hand, multilayer networks can provide more information about relationships between nodes. In this paper, we present our random walk based multilayer network embedding and compare it with single layer and multilayer network embeddings. For this purpose, we used several classic datasets usually used in network embedding experiments and also collected our own dataset of papers and authors indexed in Scopus.

Keywords: Network embedding · Multi-layer network · Machine learning on graphs

1 Introduction

1.1 Network embedding

Many complex and large systems can be represeneted as networks. Examples include social networks, transportation networks, information networks, biological networks, etc. It is a known fact that these networks are often very complicated and because of this they are challenging to analyze. Thus, to deal with them effectively one needs to find an appropriate and effective network representation. This means concise and precise enough representation that will suit well for different kinds of tasks such as analysis and prediction and that will make algorithms performing these tasks time and space efficient.

** Sections 1–3 on network embeddings and data collection were prepared with support by the Russian Science Foundation under grant 19-11-00281. Section 4–6 were prepared with support by the Russian Science Foundation under grant 17-11-01294.

Due to the growing size of modern networks, the traditional ways of network representation such as adjacency matrices, nodes and edges lists has become computationally inefficient. This leads to development of network embedding methods. These methods produce low-dimensional vector representations of the network nodes. The relationship between nodes, which in traditional models was presented as edges between those nodes, in case of network embedding takes form of a distance between corresponding vectors of the nodes in an embedding space. The smaller the distance – the more "close" nodes are considered to each other. Moreover, the nature of embeddings appeared to be able to preserve the structure of input network.

Network embedding is a very promising technique widely used today for many network analysis tasks, such as node classification, node clustering, network alignment and link prediction. It is used in many areas such as social networks analysis, transportation networks analysis, neuroscience, etc. In current work, we focus mainly on a link prediction task.

1.2 Multilayer networks

Standard multilayer or multiplex network can be considered a set of networks (often called layers or subnetworks), which share the set of nodes but differ in set of edges. More formally, a multilayer network is a tuple:

$$MN = (V, E, L), \tag{1}$$

where V is the set of nodes, E is the set of layers and L is the set of multilalyer edges such that:

$$E \subseteq \{(x, y, l) | x, y \in V, l \in L\} \tag{2}$$

Multilayer networks reflect different kind of relationships between nodes. Each layer depicts its own type of relationship. Analyzing multilayer network can produce more promising results in different network analysis tasks because models based on multilayer networks capture information from several types of relationships. A trivial example of such analysis is the following. Consider some user's networks of contacts in different social networks such as Vkontakte, Facebook and Instagram. If we see that this user has a friend in Vkontakte but they are not friends in Facebook we can recommend him to add a friend from VKontakte as a friend in Facebook also, if we have direct node-to-node correspondence between different layers nodes.

Another type of multilayer network, which is not widely used, is a set of networks where layers do not share the set of nodes, but we can associate particular node from one layer with its counterpart from another layer. We consider a multilayer network with two layers:

- Co-authorship of scientific papers;
- Citations between scientific papers.

Obviously, given an author from the first layer we can get all his articles (which are basically its adjacent edges) and thus associate them with a set of nodes from citation networks.

1.3 Link prediction

In this work, we focus mainly on a link prediction problem. It is a task of learning a model based on original network, which, given two nodes, produces a probability of an edge between them. Formally, given a multilayer network $MN = (V, E, L)$ we aim to learn a function $f : V \to \mathbb{R}^d$ that embeds nodes from V into a low-dimensional vector space. After learning two nodes u and v representations we compute probability between them based on softmax approximation of their dot product of corresponding embedding vectors $f(u) \cdot f(v)$. Then, standard binary classification framework is trained for link prediction using negative sampling for balancing classes of existing edges and non-connected pairs of nodes.

Link prediction is a very crucial task for social networks. For example, it is used in Vkontakte, Facebook, Instagram to recommend friends to users or in LinkedIn to recommend companies hiring new employees.

2 Network embedding methods overview

2.1 Single layer network embedding

There are many methods of embedding single layer networks, the most widely known are LINE[17], DeepWalk[16], APP[19] and node2vec[11]. They are mostly based on random walks, which are quite fast among structural only models. Their usual process is to generate a set of random walks (this is where they differ) and then train a skipgram model on these random walks.

2.2 Multilayer network embedding methods

The commonly used models include traditional centrality measures extensions, matrix factorization, random walk, deep neural networks and their variations.

Traditional centrality measures extensions As an extension of traditional centrality measures there are works that incorporate cross-layer degree centrality [10][8], multilayer local clustering coefficient (MLCC), cross-layer clustering coefficient(CLCC) [13][9] and use them on multilayer networks. These methods can do well in some cases but they can not capture all types of relationship existing between layers of a multilayer network.

Matrix factorization The adjacency matrix (or tensor in multilayer case) is a natural way to embed network nodes. In that case, an embedding of each node is simply a row or a column of adjacency matrix (or it is a matrix in multilayer case). But that embedding space has a dimension of $|V|$ too large to be processed in effective way. Because of this methods of matrix factorisation were developed.

These methods are based on factorisation of a multilayer adjacency tensor of a multilayer network. For example, in MULTITENSOR [6] this tensor is factorized into three tensors of a special kind and the model is learned afterwards based on these obtained tensors.

Another method based in matrix factorization is MANE[14]. In this method each layer's adjacency matrix is factorized into product of several matrices and then a target function of these products is optimized.

Methods based on matrix factorization can do well in some cases (usually when analyzed networks are small) but they have one big drawback. When large networks are analyzed (which is usually the case because modern networks are extremely large) these methods can lead to computational errors and are very slow.

Random walk based methods

One of the way to use random-walks based methods on a multilayer networks is to merge all layers into a single network and then use random-walks based methods for single-layer networks. But during the process of merging layers sufficient part of information about nodes relationship in different layers is lost.

Besides that there are already invented random-walks based methods for multilayer networks. One of them is PMNE(c)[15]. In this method, the idea of node2vec is extended to be used for multilayer networks. Main difference of this method from classic node2vec is that on each step, there is a probability that a random walk will "jump" to another layer. The rest of it is the classic node2vec.

Another random-walk based method is OhmNet[20]. The model is computed in two phases. In the first phase, node2vec is used to construct neighborhoods for each node in each layer. In the second phase, OhmNet uses an iterative approach, in which features associated with each object in the layers hierarchy are iteratively updated by fixing the rest of the features. The two phases of OhmNet are executed sequentially. The OhmNet algorithm scales to large multilayer networks because each phase is parallelizable and executed asynchronously.

There are many other network embedding methods using random walks, each of them generates these walks in its own way. Examples are Scalable Multiplex Network Embedding [18], Levy random walks on multiplex networks [12], MultiNet: A Scalable Multiplex Network Embedding Framework [7], etc.

3 Scopus dataset

3.1 Dataset description

For conducting experiments in this work, a new dataset was collected. With the use of Elsevier API[2], te information about scientific papers published in HSE was scraped from the Scopus database[4]. The dataset consists of two networks:

- A network of co-authorship between authors. It is an undirected graph, in which nodes are authors and each edge means that two authors have a paper published in co-authorship;
- A network of citations between papers. Nodes in this network are papers and edges represent relationship "cited by" between papers

Brief description of a Scopus dataset is presented in Table 3.1:

Layers	#nodes	#edges	atributes
co-authorship	14749	62060	on edges: 'artId' – article id from citations layer
citations	12191	91820	

Table 1. Information on Scopus dataset

3.2 Scopus dataset acquisition

In order to obtain two networks described in previous sections, we needed to find a way to interact with Scopus site and with Elsevier's API to Scopus database. There were several options on how to make this interaction:

- Interacting with Elsevier's API to Scopus database. This API provides several functions to retrieve data from Scopus. The most universal function is "Scopus search". With use of this function one can send http requests to an url "https://API.elsevier.com/content/search/Scopus" and retrieve needed information about published articles. Benefits of this method is its flexibility. There are many configurable parameters that ease the process of acquiring data. Main drawback of this method is restrictions that exist for a user, which does not have sufficient privileges. Any user who registers at Elseveir will have those insufficient privileges. But a user with a subscription to Scopus is able to download needed data. Another drawback is the restriction on the number of requests to Elsevier's API. Each user can have up to 10 API keys but each key allows one to send 20000 requests. After sending this number of requests one has to wait for a week to be able to use this API key again.

– There is a library called elsapy[1] which is basically a wrapper on the most popular Elsevier API functions. It has the same benefits and drawbacks as a previous method but it has its own drawbacks including restrictions on the number of requests.
– Using Selenium[5] and other similar tools. With usage of these tools, one can simulate work with internet browser and download articles data from Scopus as it has been made from browser. The largest drawback, which made this method unacceptable, is speed of work. Simulating browser work is slow and we needed many articles to form networks.

After research the decision was made to use first method.

4 Proposed model

We propose a model of embedding a Scopus multilayer network using random-walk approach. The algorithm consists of three steps. In step one, random walk is generated from a co-authorship network. Then, in step two, starting node of this random walk is considered. All its adjacent edges are taken and then their 'artId' attribute is considered. Set of these attributes is now a new starting node on a citations layer. Short random walks (of size 1-2) are generated on citations layer. After that, terminating nodes of these walks are considered. Edges from co-authorship layer having 'artId' in set of terminating nodes are taken. Ends of these edges are added to the end of random walk generated on step one. Steps one and two are repeated until desired number of random walks is reached. At step three, skipgram model is trained on obtained random walks.

This model allows us to use the topology of the second layer of multilayer network and thus consider relationship between nodes in it. An algorithm of our method is presented below.

Data: $MultilayernetworkMNwithtwolayers:$
$co-authorsiplayerandcitationslayer$, $numWalks$, $walkLength$, $shortWalkLength$
Result: $Networkembeddingf forMN$
Initialize $walkList$ to empty;
$walkList := generateRandomWalks(MN(Layer1), walkLength)$;
for i *from 1* **to** $numWalks$ **do**
  $startingNode = getStartingNode(walkList(i))$;
  $secondLayerStartingNodes =$
   $getAttributes(adjacentEdges(startingNode), artId)$
  $walkListSecondLayer = generateRandomWalks(MN(Layer2),$
  $secondLayerStartingNodes, shortWalkLength)$;
  $terminatingNodes =$
   $getTerminatingNodes(secondLayerStartingNodes)$;
  $walkList(i) += terminatingNodes$
end
$f \leftarrow node2vecSGD(walkList)$

Algorithm 1: Algorithm for proposed model

5 Experiments

To demonstrate effectiveness of multilayer network embedding algorithms firstly we apply baseline methods on classic multilayer networks datasets. We choose PMNE method (all three of its variations) to demonstrat superiority of multilayer embedding over single layer embedding. Then we apply baseline methods including PMNE and our method to Scopus dataset.

5.1 Datasets

For our experiments we use datasets of multilayer networks from site of Manlio De Domenico [3]. We choose three multilayer networks, the details of them are presented further.

Vickers :
This network is based on the survey of 29 seventh grade students from Victoria, Australia. They have been asked three questions. Each question is a different type of relationship in the network.

CKM :
This network is based on the survey of physicians from Illinois, Bloomington, Quincy, and Galesburg. They have been asked three questions. Each question is a different type of relationship in the network.

LAZEGA :
This is a social multiplex network. There are three types of relationships (Co-work, Friendship, and Advice) between partners and associates of a corporate partnership.
Description of datasets statistics is available at Table 5.1.

Dataset	#layers	#nodes	#edges
Vickers	3	29	740
CKN	3	246	1551
LAZEGA	3	71	2223

5.2 Baseline methods

Our model will be compared to the following baseline methods:

DeepWalk :
As it was stated before, this method generates several random walks on the network and then uses skipgram model to learn embeddings.

LINE :
This model also generates random walks as the DeepWalk but adds its own term to cost function. Also is considers second order neighbors along with first order neighbors to incorporate higher order relations.

Node2Vec :
Node2Vec uses two additional parameters to control generation of random walks. It performs better than DeepWalk in some of the cases.

Principled Multilayer Network Embedding :
In their work, authors of PMNE suggested three models of merging multilayer network in order to produce embeddings for each node. They're usually denoted as PMNE (n) (network aggregation), PMNE (r) (results aggregation), and PMNE (layer co-analysis)

Apart from baseline embedding methods our model will also be compared to well-known network structure methods used for link prediction.

Jaccard Coefficient (JC):
Given two nodes u and v JC is computed as follows:

$$\frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|} \tag{3}$$

where $N(.)$ is the neighborhood of a node.

Adamic/Adar (AA):
AA acts almost like JC but it weights nodes with fewer neighbors more than JC. It is computed as follows:

$$\sum_{z \in N(u) \cap N(v)} \frac{1}{log(|N(z)|)} \tag{4}$$

5.3 Evaluation metrics

As we were considering link prediction task, we used ROC-AUC as the criteria. It is very common practice to use this metric. Higher value of ROC-AUC means that model has great performance in link prediction. Value one means that the model perfectly predicts all the links.

5.4 Model parameters

For all baseline methods, we set dimension of embedding space to 200. For random walk methods, window parameter is chosen to be 10, negative sampling size is 5. For LINE model, the dimension of embedding space is 100 as it creates two embeddings and merges them into one. For Node2Vec, as stated in their work, best parameters are $p = 2$ and $q = 0.5$. For PMNE model, methods the parameters are chosen according to the original paper.

5.5 Experiment results for link prediction

In our experiment, we use five-fold cross-validation. Also we chose negative samples of non-existing edges in the network. Size of negative sample is 20% of testing sample size.

The results are stated in Table 5.5.

Model	Vickers	CKM	LAZEGA	Scopus
DeepWalk	0.821	0.781	0.780	0.936
LINE	0.676	0.637	0.695	0.93
Node2Vec	0.821	0.781	0.780	0.932
PMNE(n)	0.810	0.917	0.792	0.986
PMNE(r)	0.844	0.904	0.813	0.992
PMNE(c)	0.837	0.847	0.797	0.968
Jaccard Coefficient	0.778	0.873	0.826	0.711
Adamic/Adar	0.803	0.875	0.814	0.499
Our method	0.813	0.832	0.783	0.981

Table 2. Results of experiments

From the results, we can see that in case of large enough multilayer networks, methods of embedding single layer networks perform worse when compared to multilayer network embedding methods, such as PMNE (all variants) and our method. Also we can state that our method performs equally or even slightly better than PMNE on several of selected datasets.

Still PMNE performs better than our method. One of the interpretations of this result can be the fact that PMNE uses information from "merged" network, i.e., it uses larger neighborhoods from all of the layers whereas our method use full random walk on the first layer producing large neighborhoods but it does not use all nodes from the second layer, retaining only terminal nodes.

We also made an attempt to interpret why does our method outperforms single layer network models. We took node2vec method, which was run on co-authorship layer of Scopus network. As it was stated earlier, some edges were removed from network to obtain test set. We have searched for an edge in this set, which was not predicted by node2vec but was predicted by our method. It is the edge ('57189500027', '8592870000'), numbers here are Scopus ids of authors. Then we looked at articles published by these authors in the citations layer. There was edge ('85029806407', '85028755866'), so article '85029806407' was published by '57189500027', and article '85028755866' was published by '8592870000'. Because our method considers short paths in citations layer, edge ('85029806407', '85028755866') was predicted by it and terminal node '85028755866' was explored. Then all authors of '85028755866' were added to random walk, including '8592870000'. This is a good example of how our method incorporates network relations from second layer to make predictions in the first layer.

6 Conclusion

In this work, we presented new model of embedding multilayer networks. It was shown that it performs equally and in some cases better than PMNE, a well known method for embedding multiplex networks. Also it was shown that single layer networks embedding methods perform worse than multilayer network embedding methods when multilayer network is given and is a part of highly correlated relations between corresponding nodes across the layers.

References

1. Elsevier, https://dev.elsevier.com
2. Elsevier api library, https://github.com/ElsevierDev/elsapy
3. Manlio de domenico's personal page, https://comunelab.fbk.eu/manlio/index.php
4. Scopus, https://scopus.com
5. Selenium test automation, https://docs.seleniumhq.org
6. Bacco, C.D., Power, E.A., Larremore, D.B., Moore, C.: Community detection, link prediction, and layer interdependence in multilayer networks. Physical Review E **95**(4) (Apr 2017). https://doi.org/10.1103/physreve.95.042317, https://doi.org/10.1103/physreve.95.042317
7. Bagavathi, A., Krishnan, S.: Multi-net: A scalable multiplex network embedding framework. In: Studies in Computational Intelligence, pp. 119–131. Springer International Publishing (Dec 2018). https://doi.org/10.1007/978-3-030-05414-4_10, https://doi.org/10.1007/978-3-030-05414-4_10
8. Bródka, P., Kazienko, P., Musial, K., Skibicki, K.: Analysis of neighbourhoods in multi-layered dynamic social networks. CoRR **abs/1207.4293** (2012), http://arxiv.org/abs/1207.4293
9. Bródka, P., Musial, K., Kazienko, P.: A method for group extraction in complex social networks. In: Knowledge Management, Information Systems, E-Learning, and Sustainability Research, pp. 238–247. Springer Berlin Heidelberg (2010). https://doi.org/10.1007/978-3-642-16318-0_27, https://doi.org/10.1007/978-3-642-16318-0_27
10. Bródka, P., Skibicki, K., Kazienko, P., Musial, K.: A degree centrality in multi-layered social network. CoRR **abs/1210.5184** (2012), http://arxiv.org/abs/1210.5184
11. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864. ACM (2016)
12. Guo, Q., Cozzo, E., Zheng, Z., Moreno, Y.: Lévy random walks on multiplex networks. Scientific Reports **6**(1) (Nov 2016). https://doi.org/10.1038/srep37641, https://doi.org/10.1038/srep37641
13. Kazienko, P., Brodka, P., Musial, K.: Individual neighbourhood exploration in complex multi-layered social network. In: 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. IEEE (Aug 2010). https://doi.org/10.1109/wi-iat.2010.313, https://doi.org/10.1109/wi-iat.2010.313
14. Li, J., Chen, C., Tong, H., Liu, H.: Multi-layered network embedding. In: 2018 SIAM International Conference on Data Mining (SDM) 2018. pp. 684–692 (1 2018)

15. Liu, W., Chen, P.Y., Yeung, S., Suzumura, T., Chen, L.: Principled multilayer network embedding. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW). pp. 134–141. IEEE (2017)
16. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710. ACM (2014)
17. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web. pp. 1067–1077. International World Wide Web Conferences Steering Committee (2015)
18. Zhang, H., Qiu, L., Yi, L., Song, Y.: Scalable multiplex network embedding. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden. pp. 3082–3088 (2018). https://doi.org/10.24963/ijcai.2018/428, https://doi.org/10.24963/ijcai.2018/428
19. Zhou, C., Liu, Y., Liu, X., Liu, Z., Gao, J.: Scalable graph embedding for asymmetric proximity. In: Thirty-First AAAI Conference on Artificial Intelligence (2017), https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14696
20. Zitnik, M., Leskovec, J.: Predicting multicellular function through multi-layer tissue networks. Bioinformatics **33**(14), 190–198 (2017)