Interpretable node embeddings with mincut loss

Chi Thang Duong [1] **Quoc Viet Hung Nguyen** [2] **Karl Aberer** [1]

Abstract

To construct node embeddings in an unsupervised manner, most graph embedding techniques follow the homophily principle whereby similar nodes in a graph should have close embeddings. Existing embedding techniques interpret the similarity of nodes from their distances on a graph. In this paper, we propose designating nodes as similar when they exhibit similar membership to different communities. To this end, we propose a new differentiable loss function called the mincut loss designed to minimize the number of connections between communities. Each dimension of the embeddings learned using the mincut loss is interpretable as it represents a probability of assignment of a node to a community, which we demonstrate on a word-adjacency graph. Moreover, we provide empirical evidences that the proposed embedding method remains competitive in node classification and link prediction tasks.

1. Introduction

Graphs serve as a natural representation of relations between entities in complex systems, such as social networks or information networks. To enable inference on graphs, a *graph embedding* may be learned. It comprises *node embeddings*, each being a vector-based representation of a graph's node that incorporates its relations to other nodes (Hamilton et al., 2017; Goyal & Ferrara, 2018). Most popular unsupervised graph embedding approaches are designed to learn node embeddings such that "similar" nodes in a graph have close embeddings. Traditional approaches measure the similarities between nodes mainly based on their distances on a graph (e.g., the probability of cooccurring on a random walk) (Goyal & Ferrara, 2018).

In this paper, we propose a different perspective to capture node similarities based on their community membership. We consider a community as a set of nodes such that there are more intra-edges than inter-edges between communities. The homophily principle can be restated as follows: nodes with similar community membership should have close embeddings. For n communities, the likelihood of a node v being assigned to a community can be measured from membership probability. In other words, the community membership of a node to n communities forms a probability distribution. Intuitively, nodes are similar when their membership distributions are similar. For instance, nodes u and v are similar when their likelihood of assignment to community A is high while it is low for community B (assuming that there are only 2 communities).

As communities are not known beforehand, we propose a new loss function (called *mincut* loss) that allows us to learn the node embeddings and communities at the same time. The mincut loss is defined based on the principle that communities are well-separated when there are few connections between them (Fortunato, 2010). Nodes are assigned to n communities based on node embeddings in such a way that minimizes the mincut loss. As node embeddings are continuous, we propose using Gumbel-Softmax (Maddison et al., 2016; Jang et al., 2016) to sample discrete one-hot vectors from continuous node embeddings representing node membership to communities. This also renders the mincut loss differentiable, which allows us to learn node embeddings and communities in an end-to-end fashion.

In addition, the mincut loss allows us to learn node embeddings such that embedding dimensions are interpretable as the i-th element of a node embedding is the unnormalized probability of a node belonging to the i-th community. The node embeddings learned using the mincut loss can outperform several traditional distannce-based graph embedding approaches(Perozzi et al., 2014; Grover & Leskovec, 2016). In addition, our node embeddings are interpretable. An experiment using a word adjacency graph confirms this property, as words with high embedding values along a dimension belong to the same topic.

The rest of the paper is organized as follows. We discuss related works in Section 2. Our proposed mincut loss is discussed in Section 3 while our framework for learning node embeddings with the mincut loss is described in Section 4. The experimental results are shown in Section 5 and Section 6 concludes the paper.

[1]LSIR, EPFL, Switzerland [2]Griffith University, Australia. Correspondence to: Chi Thang Duong <thang.duong@epfl.ch>.

Presented at the ICML 2019 Workshop on Learning and Reasoning with Graph-Structured Data

2. Related works

Graph embedding. Graph representation learning aims to construct a low-dimensional model of nodes in a graph that incorporates the graph's structure (Hamilton et al., 2017; Goyal & Ferrara, 2018). Although there are several ways to embed nodes (Perozzi et al., 2014; Grover & Leskovec, 2016), the majority of unsupervised learning mainly leverages the skipgram loss (Hamilton et al., 2017) whereby embeddings of nodes positioned close together in a graph have high levels of cosine similarity. In this paper, we propose a new loss function that measures the quality of node embeddings based on their ability to partition a graph into well separated communities. The mincut loss is orthogonal to the skipgram loss, as they can be used together.

Graph partitioning. The closest to our work is (Nazi et al., 2019) in which the authors propose a partition loss function for graph partitioning. The loss function aims for balanced partitions based on the number of *nodes*, which is not applicable in our setting. In addition, the loss function is susceptible to a local optima where all nodes are assigned to one partition. By using Gumbel-softmax, our mincut loss is free of this degenerate case. On the other hand, graph partitioning can be considered as a special case of community detection problem (Fortunato, 2010). While the latter allows for different community sizes in term of nodes, the former does not. Most popular community detection algorithms are based on the modularity metric, which can be captured by the modularity loss function. Our empirical evaluation shows our mincut loss to perform better than the modularity loss for different datasets.

3. Mincut loss

3.1. Preliminaries

Graph model. We represent a graph $G = \{V, E\}$ by its nodes $V = \{v_i\}$ and edges $E = \{(v_i, v_j) | v_i \in V \wedge v_j \in V\}$. A graph can also be modeled from its adjacency matrix $\mathbf{A}$ where each row/column represents a node in G and a cell $A(i, j)$ corresponding to the edge weight between v_i, v_j.

Edge cut. A graph G can be divided into m disjoint communities $\mathbb{C} = \{C_1, C_2, \cdots, C_m\}$ where $\bigcup_{i=1,m} C_i = V$ and $\forall C_i \neq C_j, C_i \cap C_j = \emptyset$. We also denote $\bar{C}_i = V \setminus C_i$ as the complement of C_i. Given a community C and its complement $\bar{C}$, its edge-cut $cut(C)$ is defined as $cut(C) = \sum_{(v_i,v_j)\in E} \mathbb{1}_{v_i \in C, v_j \in \bar{C}}$. For a set of communities $\mathbb{C} = \{C_1, \cdots, C_m\}$, its edge-cut is defined as follows:

$$c(\mathbb{C}) = c(C_1, \cdots, C_m) = \frac{1}{2} \sum c(C_i)$$

To form well-defined communities, nodes are assigned to communities such that the edge-cut $c(\mathbb{C})$ is minimized.

3.2. Mincut loss

The mincut loss takes the node embeddings as input and measures how well embeddings can be used to separate the graphs into communities. In the following, we discuss how to compute the edgecut given the node embeddings. The process for computing the mincut loss is shown in Figure 1.

We assume an embedding function $f_\theta : V \to \mathbb{R}^m$ embeds a node to an m-dimensional space where θ denotes its parameters. We denote the embedding matrix as $\mathbf{E} \in \mathbb{R}^{n \times m}$ where each row denotes a node embedding.

Computing edge cut. The assignment of nodes to communities can be captured by *membership matrix* $\mathbf{P} \in \{0, 1\}^{n \times m}$ with rows representing nodes in G and columns representing communities in $\mathbb{C}$. As each node is only assigned to one community, the rows of $\mathbf{P}$ are one-hot vectors where $\mathbf{P}_{ij} = 1$ when node v_i is assigned to community C_j.

Intuitively, the assignment of a node to a community is based on the node and its role in the graph. We assume that node embedding can capture this information. As a result, we propose computing membership matrix $\mathbf{P}$ based on embedding matrix $\mathbf{E}$. To convert $\mathbf{E} \in \mathbb{R}^{n \times m}$ into $\mathbf{P} \in \{0, 1\}^{n \times m}$, we use the Gumbel-softmax with τ as the temperature hyperparameter. The Gumbel-softmax allows us to sample the discrete matrix $\mathbf{P}$ from the continuous $\mathbf{E}$ while it is also differentiable, which is important for learning the parameters of f_θ using backprop.

The membership matrix presents a property wherein by right- and left-multiplying $\mathbf{P}$ and its transpose with the adjacency matrix $\mathbf{A}$ of G respectively, we obtain a matrix $\mathbf{D} = \mathbf{P}^T\mathbf{A}\mathbf{P}$. Matrix $\mathbf{D}$ of size $m \times m$ can be considered the adjacency matrix of the *quotient graph* where the nodes are communities. Elements $\mathbf{D}(i, i)$ on the diagonal of $\mathbf{D}$ capture the number of intra-edges in community C_i while nondiagonal elements $\mathbf{D}_{i,j}$ capture the number of cross-edges between communities C_i and C_j. It is worth noting that as matrix $\mathbf{P}$ is extremely sparse, the above equation can be efficiently computed via sparse matrix multiplication.

Based on adjacency matrix $\mathbf{D}$, the number of edge-cuts between communities can be computed as follows:

$$\mathcal{L}_c = \sum_{i \neq j}^{m} \mathbf{D}_{ij} \tag{1}$$

Minimizing $\mathcal{L}_c$ in Equation 1 is equivalent to minimizing the following loss function:

$$\mathcal{L}_c = -Tr(\mathbf{P}^T\mathbf{A}\mathbf{P}) \tag{2}$$

where $Tr(\mathbf{X})$ is the trace of matrix $\mathbf{X}$. Note that if we replace $\mathbf{A}$ by the modularity matrix (Newman, 2006), Equation 2 then captures the modularity loss (Fortunato, 2010).

Mincut loss. While communities with a minimal edge-cut

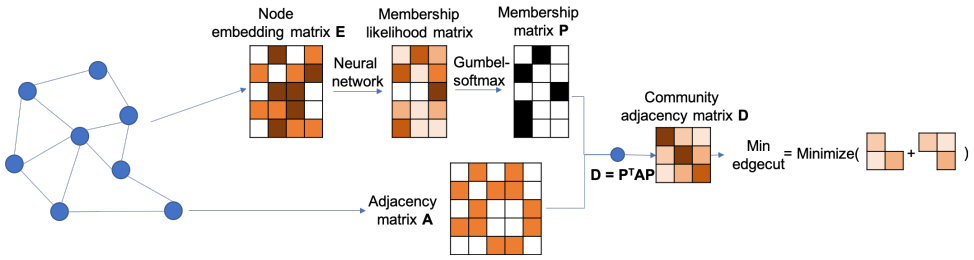

Figure 1. Interpretable node embeddings with mincut

can be identified by minimizing Equation 1, they are usually ill-formed, as some may contain only one node with a small node degree. To have "meaningful" communities, traditional approaches enforce them to have the same number of nodes (Nazi et al., 2019). In this paper, we propose measuring community size by the number of edges. Although this diverges from traditional definition, it leads to better empirical results.

The edge-balance requirement is captured by the following loss function:

$$\mathcal{L}_b = \sum_{i=1}^{m} (\mathbf{D}_{ii} - \frac{\sum_j \mathbf{D}_{jj}}{m})^2$$

Intuitively, we want the number of edges in community C_i (which is $\mathbf{D}(i,i)$) to be equal to the total number of edges in all community divided by the number of communities.

By combining the above loss functions, we obtain the *mincut loss* function:

$$\mathcal{L} = \lambda \sum_{i \neq j}^{k} \mathbf{D}_{ij} + (1-\lambda) \sum_{i=1}^{m} (\mathbf{D}_{ii} - \frac{\sum_j \mathbf{D}_{jj}}{m})^2$$

where λ is a control parameter that sets the weight for the balance and edgecut loss.

4. Learning embeddings with mincut loss

In the following section, we discuss one application of the mincut loss where it is used to learn node embeddings. With the loss function chosen, the practitioner now only needs to select the embedding function f_θ. There are two main types of embedding functions: shallow or deep. A shallow embedding function performs an embedding lookup on the embedding matrix $\mathbf{E}$ while a deep embedding function also takes into account the local neighborhood of a node. However, a deep embedding function requires nodes to have features and takes longer to train. In this paper, we focus on a shallow embedding function, as we do not assume the availability of node features. The architecture of our framework is illustrated in Figure 1.

Interpretable embeddings. It is worth noting that it is possible to learn an embedding function f_θ that creates node embeddings of size $d > m$ and to then apply linear transformations to obtain matrix $\mathbf{E}' \in \mathbb{R}^{n \times m}$ as an input to the mincut loss. However, by feeding matrix $\mathbf{E} \in \mathbb{R}^{n \times m}$ to the mincut loss directly, node embeddings exhibit a favorable property whereby an embedding value at the i-th index represents the membership probability of the node in community C_i. In other words, dimensions of the embeddings are interpretable, as they represent the communities.

Auxiliary loss. In practice, we also learn node embeddings with an auxiliary loss function that encodes the intuition that a node and its neighbors are more likely to be of the same community while non-neighbors belong to different communities. In particular, we also minimize $\mathcal{L}_s = \sum_{v_i \in V} (\sum_{v_j \in N(v_i)} - \log(\mathbf{E}_{i,:}, \mathbf{E}_{j,:}) + \sum_{v_j \in V \setminus N(v_i)} \log(\mathbf{E}_{i,:}, \mathbf{E}_{j,:}))$ where $N(v)$ denotes the neighbors of v. In the experiments, we set the weight of the auxiliary loss as very low (at 0.01) to clearly demonstrate the benefits of the mincut loss.

5. Experiments

The aims of our experiments are to 1) evaluate the interpretability of node embeddings and to 2) evaluate the performance of node embeddings learned with the mincut loss for node classification and link prediction.

5.1. Experimental setup

Datasets and Metrics. For node classification, we use datasets for which node labels are available. After node embeddings are learned, we train a classifier using 50% of nodes while 50% are used for validation. For link prediction, for each dataset, we remove a fraction of edges and construct classifier to predict these missing edges. We select positive and negative links similar to node2vec (Grover & Leskovec, 2016). Statistics for the datasets sre shown in Table 1. We use the micro/macro F1-score for node classification and AP and ROC for link prediction tasks.

Baselines. For the distance-based loss function, we draw comparison with DeepWalk (Perozzi et al., 2014), node2vec (Grover & Leskovec, 2016) which uses random walks to measure the similarities between nodes. For the community-based loss function, we apply the same

Table 2. Node classification results

	ppi		wiki		blogCatalog	
	mic-f1	mac-f1	mic-f1	mac-f1	mic-f1	mac-f1
deepwalk	0.203	0.177	**0.688**	**0.580**	0.37	0.251
node2vec	0.190	0.166	0.650	0.530	0.386	0.258
partition	0.185	0.151	0.599	0.454	0.252	0.131
modularity	0.201	0.159	0.612	0.482	0.283	0.127
mincut	0.207	0.175	0.650	0.512	0.321	0.175
modul. + aux.	0.221	0.200	0.676	0.561	0.393	**0.272**
mincut + aux.	**0.224**	**0.207**	0.667	0.549	**0.400**	0.271

Table 3. Link prediction results

	facebook		ppi	
	ROC	AP	ROC	AP
deepwalk	0.963	0.945	0.557	0.541
node2vec	0.942	0.912	0.536	0.540
partition	0.803	0.783	0.507	0.491
modularity	0.876	0.799	0.467	0.477
mincut	0.961	0.935	0.507	0.491
modul. + aux.	0.984	0.982	0.771	0.810
mincut + aux.	**0.984**	**0.987**	**0.774**	**0.823**

learning node embedding process to a different loss function: 1) partition loss (Nazi et al., 2019), 2) modularity loss (Girvan & Newman, 2002), and 3) our mincut loss. We also consider two variants of modularity loss and mincut loss where we also use the auxiliary loss with a weight of 0.01.

Table 1. Datasets statistics

Dataset	**#Nodes**	**#Edges**
wiki	4,777	184'812
ppi	3,890	76,584
facebook	4,039	88,234
blogCatalog	10,312	333,983

5.2. Node classification results

Table 2 compares the results of community-based loss functions to those of traditional distance-based loss functions. We observe that techniques using community-based losses (except in the case of GAP) achieve comparable or stronger performance than distance-based approaches. For example, our proposed mincut loss outperforms node2vec by 24.6% for the PPI dataset for both micro and micro-f1 scores. Although the f1-scores of community-based losses are lower than those of distance-based loss in the wiki dataset, this could reflect a tradeoff in the interpretability of our node embeddings. For community-based losses, our approach outperforms the modularity loss and partition loss across all datasets and metrics. We also find that adding the auxiliary loss helps increase the f1-score by at most 8%, denoting a superior community structure.

5.3. Link prediction results

For the link prediction task, we find that our mincut loss with additional auxiliary loss outperforms the other approaches. This result is expected as our mincut loss is designed to form communities with a large number of edges. Without the auxiliary loss, the mincut loss still outperforms all community-based losses. For example, the mincut loss provides gains of 19.6% and 10% over the partition and modularity loss respectively. On the other hand, the mincut loss without auxiliary loss differs from that of DeepWalk by only 0.002 while it still outperforms node2vec.

5.4. Interpretability

To analyze the interpretability of our embeddings, we construct a word adjacency graph of 5000 words from the Text8 corpus (Mahoney, 2011). The edge between two words is weighted by their Positive Pointwise Mutual Information (Yin & Shen, 2018). We then use the mincut loss to learn the node(word) embeddings of size 128. We report the top words with the largest embedding values (in parentheses) along a dimension in Table 4.

Table 4. Top words of each dimension relating to a specific topic

Dim #1	Dim #3	Dim #9
trial (1.54)	luther (1.56)	miles (1.47)
murder (1.52)	practices (1.55)	paved (1.43)
judge (1.51)	protestant (1.55)	km (1.41)
copyright (1.51)	churches (1.55)	total (1.33)
guilty (1.5)	muslims (1.54)	main (1.3)
crimes (1.5)	catholic (1.54)	its (0.6)
relating (1.49)	muhammad (1.54)	including (0.38)

We observe that for the top words in each dimension, we can infer the meaning behind each dimension. For instance, the 1st dimension is related to the judicial system while the 3rd dimension focuses on religion. In addition, for the 9th dimension, we find that words related to the topic captured by the dimension have high unnormalized probabilities while unrelated words have low probabilities. There is a large gap between them (e.g. between the word "main" and the word "its"). This clearly shows that both the embedding dimensions and embedding values are meaningful.

6. Conclusion

We introduced a new loss function for embedding learning in an unsupervised manner. The embeddings learned using our proposed mincut loss still perform well in several downstream tasks such as link prediction and node classification. In addition, the embeddings are highly interpretable as the dimensions are meaningful, which is extremely useful in explaining models that use these embeddings.

An interesting direction for future work is to understand the connection between number of communities and embedding dimensionality. Another is to apply mincut loss to applications such as image segmentation or hierarchical community detection.

References

Fortunato, S. Community detection in graphs. *Physics reports*, 486(3-5):75–174, 2010.

Girvan, M. and Newman, M. E. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

Goyal, P. and Ferrara, E. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78–94, 2018.

Grover, A. and 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.

Hamilton, W. L., Ying, R., and Leskovec, J. Representation learning on graphs: Methods and applications. *IEEE Data Engineering Bulletin*, 2017.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Mahoney, M. Large text compression benchmark. *URL: http://www. mattmahoney. net/text/text. html*, 2011.

Nazi, A., Hang, W., Goldie, A., Ravi, S., and Mirhoseini, A. Gap: Generalizable approximate graph partitioning framework. *arXiv preprint arXiv:1903.00614*, 2019.

Newman, M. E. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

Perozzi, B., Al-Rfou, R., and 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.

Yin, Z. and Shen, Y. On the dimensionality of word embedding. In *Advances in Neural Information Processing Systems*, pp. 887–898, 2018.