HAL
open science

Exponential Family Graph Embeddings

Abdulkadir Çelikkanat, Fragkiskos D. Malliaros

To cite this version:

Abdulkadir Çelikkanat, Fragkiskos D. Malliaros. Exponential Family Graph Embeddings. AAAI 2020 - Thirty-Fourth AAAI Conference on Artificial Intelligence, Feb 2020, New York, United States. hal-03088917

HAL Id: hal-03088917

https://hal.science/hal-03088917v1

Submitted on 29 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Exponential Family Graph Embeddings

Abdulkadir Çelikkanat
CentraleSupélec and Inria Saclay
University of Paris-Saclay
Gif-Sur-Yvette, France
abdulkadir.celikkanat@centralesupelec.fr

Fragkiskos D. Malliaros
CentraleSupélec and Inria Saclay
University of Paris-Saclay
Gif-Sur-Yvette, France
fragkiskos.malliaros@centralesupelec.fr

Abstract

Representing networks in a low dimensional latent space is a crucial task with many interesting applications in graph learning problems, such as link prediction and node classification. A widely applied network representation learning paradigm is based on the combination of random walks for sampling context nodes and the traditional *Skip-Gram* model to capture center-context node relationships. In this paper, we emphasize on exponential family distributions to capture rich interaction patterns between nodes in random walk sequences. We introduce the generic *exponential family graph embedding* model, that generalizes random walk-based network representation learning techniques to exponential family conditional distributions. We study three particular instances of this model, analyzing their properties and showing their relationship to existing unsupervised learning models. Our experimental evaluation on real-world datasets demonstrates that the proposed techniques outperform well-known baseline methods in two downstream machine learning tasks.

Introduction

Graphs or networks have become ubiquitous as data from diverse disciplines can naturally be represented as graph structures. Characteristics examples include social, collaboration, information and biological networks, or even networks that are generated by textual information. Besides, graphs are not only useful as models for data representation but can be proven valuable in prediction and learning tasks. For example, one might wish to recommend new friendship relationships in social networks such as Facebook and LinkedIn, predict the missing structure or the role of a protein in a protein-protein interaction graph, or even to discover missing relations between entities in a knowledge graph. To that end, the tasks of learning and analyzing large-scale real-world graph data drive several important applications, but also pose a plethora of challenges.

The major challenge in machine learning on graphs is how to incorporate information about its structure in the learning model. For example, in the case of friendship recommendations in social networks (also known as the *link prediction* problem), in order to determine whether two unlinked users are similar, we need to obtain an informative representation of the users and their proximity — that potentially is not fully captured by graph statistics (e.g., centrality criteria) (Chakrabarti, Faloutsos, and McGlohon 2010), kernel functions (Vishwanathan et al. 2010), or more generally other handcrafted features extracted from the graph (Liben-Nowell and Kleinberg 2007). To deal with these challenges, a recent paradigm in network analysis, known as *network representation learning* (NRL), aims at finding vector representations of nodes (i.e., *node embeddings*), in such a way that the structure of the network and its various properties are preserved in lower-dimensional representation space. The problem is typically expressed as an optimization task, where the goal is to optimize the mapping function from the graph space to a low-dimensional space, so that proximity relationships in the learned space reflect the structure of the original graph (Hamilton, Ying, and Leskovec 2017; Goyal and Ferrara 2018; Cai, Zheng, and Chang 2018). Furthermore, in most cases, the feature learning approach is purely *unsupervised*. That way, after obtaining embeddings, the learned features can further be used in any downstream machine learning task, such as classification and prediction.

Initial studies in network representation learning mostly focused on classical dimensionality reduction techniques via matrix factorization (e.g., (Cao, Lu, and Xu 2015; Tang and Liu 2009)). Although the success of such approaches in capturing the structural properties of a network, they tend to be, unfortunately, not efficient for large scale networks. Therefore, the community has concentrated on developing alternative approaches, and inspired by the field of natural language processing (NLP) (Mikolov et al. 2013), *random-walk based* methods have become a prominent line of research for network representation learning. Characteristic examples of such approaches constitute DEEPWALK (Perozzi, Al-Rfou, and Skiena 2014) and NODE2VEC (Grover and Leskovec 2016) models. Typically, those methods sample a set of random walks from the input graph, treating them as the equivalent of sentences in natural language, while the nodes visited by the walk are considered as the equivalent of words. The point that essentially differentiates these methods, concerns the strategy that is followed to generate (i.e., sample) node sequences. The idea mainly aims to model

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

center-context node relationships, examining the occurrence of a node within a certain distance with respect to another node (as indicated by the random walk); this information is then utilized to represent the relationship between a pair of nodes. Then, widely used NLP models, such as the *Skip-Gram* model (Mikolov et al. 2013), are used to learn node latent representations, examining *simple co-occurrence relationships* of nodes within the set of random walk sequences.

Nevertheless, *Skip-Gram* models the conditional distribution of nodes within a random walk based on the *softmax* function, which might prohibit to capture richer types of interaction patterns among nodes that co-occur within a random walk. Motivated by the aforementioned limitation of current random walk-based NRL methodologies, we argue that considering more expressive *conditional probability models* to relate nodes within a random walk sequence, might lead to more informative latent node representations.

In particular, we capitalize on *exponential family distribution* models to capture interactions between nodes in random walks. Exponential families correspond to a mathematically convenient parametric set of probability distributions, which is flexible in representing relationships among entities. More precisely, we introduce the concept of *exponential family graph embeddings* (EFGE), that generalizes random walk NRL techniques to exponential family conditional distributions. We study three particular instances of the proposed EFGE model that correspond to widely known exponential family distributions, namely the Bernoulli, Poisson and Normal distributions. The extensive experimental evaluation of the proposed models in the tasks of node classification and link prediction suggests that, the proposed EFGE models can further improve the predictive capabilities of node embeddings, compared to traditional *Skip-Gram*-based and other baseline methods. In addition, we further study the objective function of the proposed parametric models, providing connections to well-known unsupervised graph learning models under appropriate parameter settings.

Contributions. The main contributions of the paper can be summarized as follows:

- We introduce a novel representation learning model, called EFGE, which generalizes classical *Skip-Gram*-based approaches to exponential family distributions, towards more expressive NRL methods that rely on random walks. We study three instances of the model, namely the EFGE-BERN, EFGE-POIS and EFGE-NORM models, that correspond to well-known distributions.
- We show that the objective functions of existing unsupervised and representation learning models, including word embedding in NLP (Mikolov et al. 2013) and overlapping community detection (Yang and Leskovec 2013), can be re-interpreted under the EFGE model.
- In a thorough experimental evaluation, we demonstrate that the proposed exponential family graph embedding models generally outperform widely used baseline approaches in various learning tasks on graphs. In addition, the running time to learn the representations is similar to other *Skip-Gram*-based models.

Source code. The implementation of the proposed models is provided in the following website: https://abdcelikkanat.github.io/projects/EFGE/.

Preliminary Concepts

Random Walk-based Node Embeddings

Let $G = (\mathcal{V}, \mathcal{E})$ be a graph, where $\mathcal{V}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denote the vertex and edge sets respectively. Random-walk based node embedding methods (Perozzi, Al-Rfou, and Skiena 2014; Grover and Leskovec 2016; Çelikkanat and Malliaros 2018; Çelikkanat and Malliaros 2019) generate a set of node sequences by simulating random walks that can follow various strategies; node representations are then learned relying on these generated sequences. We use an ordered sequence of nodes $\mathbf{w} = (w_1, ..., w_L) \in \mathcal{W}$ to denote a *walk*, if every pair (w_l, w_{l+1}) belongs to the edge set $\mathcal{E}$ for all $1 \leq l < L$. Then, the notation $\mathcal{W}$ will represent the set of walks of length L.

Being inspired from the the field of natural language processing and the *Skip-Gram* model (Mikolov et al. 2013) for word embeddings, each walk is considered as a sentence in a document and similarly the surrounding vertices appearing at a certain distance from each node in a walk are defined as the *context set* of this particular node, which is also called *center* in our description. More formally, we will use $\mathcal{N}_\gamma^{\mathbf{w}}(w_l) := \{w_{l+j} \in \mathcal{V} : -\gamma \leq j \leq \gamma,\ j \neq 0\}$ to denote the context sequence of node w_l in the random walk $\mathbf{w} \in \mathcal{W}$. Representations of nodes are learned by optimizing the relationship between these center and context node pairs under a certain model. More formally, the objective function of *Skip-Gram* based models for network representation learning is defined in the following way:

$$\arg\max_{\Omega} \frac{1}{N \cdot L} \sum_{\mathbf{w} \in \mathcal{W}} \sum_{1 \leq l \leq L} \sum_{v \in \mathcal{N}_\gamma(w_l)} \log p(y_{w_l,v}; \Omega), \quad (1)$$

where y_{w_l,v_j} represents the relationship between the center w_l and context node v in the walk $\mathbf{w} \in \mathcal{W}$, N is the number of walks, L is length of walks and $\Omega = (\alpha, \beta)$. Note that, we typically learn two embedding vectors $\alpha[v]$ and $\beta[v]$ for each node $v \in \mathcal{V}$, where $\beta[v]$ corresponds to the vector if the node is interpreted as a center node and $\alpha[v]$ denotes the vector if v is considered as the context of other nodes. In all downstream machine learning applications, we only consider $\alpha[v]$ to represent the embedding vector of v.

Generally speaking, random walk-based network representation learning methods can use different approaches to sample the context of a particular node. For instance, DEEPWALK performs uniform truncated random walks, while NODE2VEC is based on second order random walks to capture context information. Another crucial point related to *Skip-Gram* based models, has to do with the way that the relationship among center and context nodes in Eq. (1) is modeled. In particular, DEEPWALK uses the *softmax* function to model the conditional distribution $p(\cdot)$ of a context node for a given center, in such a way that nodes occurring in similar contexts tend to get close to each other in the latent representation space. In a similar way, NODE2VEC adopts the

negative sampling strategy, where it implicitly models co-occurrence relationships of context and center node pairs. As we will present shortly, in our work we rely on exponential family distributions, in order to further extend and generalize random-walk NRL models to conditional probability distribution beyond the *softmax* function — towards capturing richer types of node interaction patterns.

Exponential Families

In this paragraph, we introduce the *exponential family distributions*, a parametric set of probability distributions that includes among others the Gaussian, Binomial and Poisson distributions. A class of probability distributions is called exponential family distributions if they can be expressed as

$$p(y) = h(y) \exp\Big(\eta T(y) - A(\eta)\Big), \tag{2}$$

where h is the *base measure*, η are the *natural parameters*, T is the *sufficient statistic* of the distribution and $A(\eta)$ is the *log-normalizer* or *log-partition* function (Andersen 1970). Different choices of base measure and sufficient statistics lead us to obtain different probability distributions. For instance, the base measure and sufficient statistic of the *Bernoulli* distribution are $h(y) = 1$ and $T(y) = y$ respectively, while for the *Poisson* distribution we have $h(y) = 1/y!$ and $T(y) = y$.

As we mentioned above, exponential families contain a wide range of commonly used distributions, providing a general class of models by re-parameterizing distributions in terms of the natural parameters η. That way, we will use the natural parameter η to design a set of network representation learning models, defining $\eta_{v,u}$ as the product of context and center vectors in the following way:

$$\eta_{v,u} := f\big(\beta[v]^\top \cdot \alpha[u]\big),$$

where f is called the *link function*. As we will present shortly in the following section, we have many alternative options for the form of the link function $f(\cdot)$.

Proposed Approach

In this section, we will introduce the proposed *exponential family graph embedding* models, referred to as EFGE. The main idea behind this family of models is to utilize the expressive power of exponential family distribution towards conditioning context nodes with respect to the center node of interest. Initially, we will describe the formulation of the general objective function of the EFGE model, and then we will present particular instances of the model based on different exponential family distributions.

Let $\mathcal{W}$ be a collection of node sequences generated by following a random walk strategy over a given graph G, as defined in the previous section. Based on that, we can define a generic objective function to learn node embeddings in the following way:

$$\mathcal{L}(\alpha, \beta) := \arg\max_{\Omega} \sum_{\mathbf{w} \in \mathcal{W}} \sum_{1 \le l \le L} \sum_{v \in \mathcal{V}} \log p(y^l_{w_l,v}; \Omega), \tag{3}$$

where $y^l_{w_l,v}$ is the observed value indicating the relationship between the center w_l and context node v. Here, we aim to find embedding vectors $\Omega = (\alpha, \beta)$ by maximizing the log-likelihood function in Eq. (3). Note that, the objective function in Eq. (3) is quite similar to the one of the *Skip-gram* model (Mikolov et al. 2013) presented in Eq. (1), except that we also include nodes that are not belonging to context sets.

Instead of restricting ourselves to the *Sigmoid* or *Softmax* functions in order to model the probability in the objective function of Eq. (3), we provide a generalization assuming that each $y_{w_l,v}$ follows an exponential family distribution. That way, the objective function used to learn node embedding vector sets $\Omega = (\alpha, \beta)$ can be rewritten as follows:

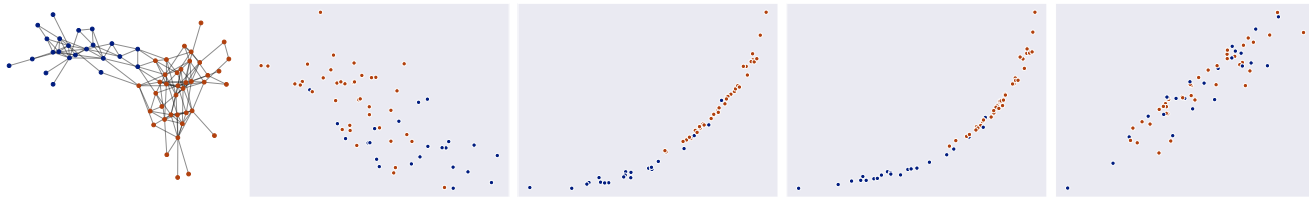
$$\arg\max_{\Omega} \sum_{\mathbf{w} \in \mathcal{W}} \sum_{1 \le l \le L} \sum_{v \in \mathcal{V}} \log h(y_{w_l,v}) + \eta_{w_l,v} T(y_{w_l,v}) - A(\eta_{w_l,v}). \tag{4}$$

As we can observe, Eq. (4) which is the objective function of the generic EFGE graph embeddings model, generalizes *Skip-Gram*-based models to exponential family conditional distributions described in Eq. (2). That way, EFGE models have the additional flexibility to utilize a wide range of exponential distributions, allowing them to capture more complex types of node interactions beyond simple co-occurrence relationships. It is also important to stress out that, the first term of Eq. (4) does not depend on parameter $\eta_{w_l,v}$; this will bring an advantage during the optimization process.

Initially, we sample a set of N random walks based on a chosen walk strategy. This strategy can be any context sampling process, such as uniform random walks (as in DEEPWALK) or biased random walks (as in NODE2VEC). Then, based on the chosen instance of the EFGE model, we learn center and context embedding vectors. In this paper, we have examined three particular instances of the EFGE model, that represent well known exponential family distributions. In particular, we utilize the Bernoulli, Poisson, and Normal distributions leading to the corresponding EFGE-BERN, EFGE-POIS and EFGE-NORM models. For illustration purposes, Fig. 1 depicts the *Dolphins* network composed by two communities and the embeddings in two dimensions as computed by different models. As we can observe, for this particular toy example, the proposed EFGE-BERN and EFGE-POIS models learn representations that are able to differentiate nodes with respect to their communities. In the following sections, we analyze the properties of these models in detail.

The EFGE-BERN Model

Our first model is the EFGE-BERN model, in which we assume that each $y_{w_l,v}$ follows a Bernoulli distribution which is equal to 1 if node v appears in the context set of w_l in the walk $\mathbf{w} \in \mathcal{W}$. It can be written as $y_{w_l,v} = x^{l-\gamma}_{w_l,v} \vee \cdots \vee x^{l-1}_{w_l,v} \vee x^{l+1}_{w_l,v} \vee \cdots \vee x^{l+\gamma}_{w_l,v}$, where $x^{l+j}_{w_l,v}$ indicates the appearance of v in the context of w_l at the specific position $l + j$ $(-\gamma \le j \le \gamma)$. We can express the objective function of the EFGE-BERN model, $\mathcal{L}_B(\alpha, \beta)$, by dividing Eq. (4) into two parts with respect to the values of $y_{w_l,v}$ and $x^{l+j}_{w_l,v}$:



(a) Network (b) DEEPWALK (c) EFGE-BERN (d) EFGE-POIS (e) EFGE-NORM

Figure 1: The *Dolphins* network composed by 2 communities and the corresponding embeddings for $d = 2$.

$$\mathcal{L}_B = \sum_{\mathbf{w}\in\mathcal{W}} \sum_{1\le l\le L} \left[\sum_{v\in\mathcal{N}_\gamma(w_l)} \log p(y_{w_l,v}) + \sum_{v\notin\mathcal{N}_\gamma(w_l)} \log p(y_{w_l,v}) \right]$$

$$= \sum_{\mathbf{w}\in\mathcal{W}} \sum_{1\le l\le L} \left[\sum_{\substack{|j|\le\gamma \\ u^+:=w_j}} \log p(x^{l+j}_{w_l,u^+}) + \sum_{\substack{|j|\le\gamma \\ u^-:\neq w_j}} \log p(x^{l+j}_{w_l,u^-}) \right]$$

Note that, the exponential form of a Bernoulli distribution with a parameter π is $\exp(\eta x - A(\eta))$, where the log-normalizer $A(\eta)$ is $\log(1+\exp(\eta))$ and the parameter π is the sigmoid function $\sigma(\eta) = 1/\big(1-\exp(-\eta))\big)$. Therefore, we can rewrite the objective function $\mathcal{L}_B(\alpha,\beta)$ as follows:

$$\sum_{\mathbf{w}\in\mathcal{W}} \sum_{1\le l\le L} \left[\sum_{\substack{|j|\le\gamma \\ u^+:=w_j}} \log\sigma(\eta_{w_l,u^+}) + \sum_{\substack{|j|\le\gamma \\ u^-:\neq w_j}} \log\sigma(-\eta_{w_l,u^-}) \right]$$

We choose the identity map for the link function $f(\cdot)$, so $\eta_{v,u}$ becomes equal to the product of vectors $\alpha[v]$ and $\beta[u]$.

Relationship to negative sampling. Although the *negative sampling* strategy (Mikolov et al. 2013) was proposed to approximate the objective function of the *Skip-Gram* model for node representation, any rigorous theoretical argument showing the connection between them has not been provided. In Lemma 1, we show that the log-likelihood $\mathcal{L}_B(\alpha,\beta)$ of the EFGE-BERN model in fact converges to the objective function of negative sampling given in Eq. (5). In our implementation, we adopt negative sampling in order to improve the efficiency of the computation.

Lemma 1. *The log-likelihood function $\mathcal{L}_B$ converges to*

$$\sum_{w\in\mathcal{W}} \sum_{1\le l\le L} \sum_{|j|\le\gamma} \left[\log p\big(x^{l+j}_{w_l,w_{l+j}}\big) + \sum_{s=1}^{k} \mathop{\mathbb{E}}_{u\sim q^-} \log p\big(x^{l+j}_{w_l,u}\big) \right] \quad (5)$$

for large values of k.

Proof. Please see the appendix. □

The EFGE-POIS Model

In this model, we will use the Poisson distribution to capture the relationship between context and center nodes in a random walk sequence. Let $y_{w_l,v}$ be a value indicating the number of occurrences of node v in the context of w_l. We assume that $y_{w_l,v}$ follows a Poisson distribution, with the mean value $\tilde{\lambda}_{w_l,v}$ being the number of appearances of node v in the context $\mathcal{N}^{\mathbf{w}}_\gamma(w_l)$. Similar to the previous model, it can be expressed as $y_{w_l,v} = x^{l-\gamma}_{w_l,v} + \cdots + x^{l-1}_{w_l,v} + x^{l+1}_{w_l,v} + \cdots + x^{l+\gamma}_{w_l,v}$, where $x^{l+j}_{w_l,v} \sim Pois(\lambda_{w_l,v})$ for $-\gamma \le j \le \gamma$. That way, we obtain $\tilde{\lambda}_{w_l,v} = \sum_{j=-\gamma}^{\gamma} \lambda^{l+j}_{w_l,v}$, since the sum of independent Poisson random variables is also Poisson. By plugging the exponential form of the Poisson distribution into Eq. (3), we can derive the objective function $\mathcal{L}_P(\alpha,\beta)$ of the model as:

$$\sum_{\mathbf{w}\in\mathcal{W}} \sum_{1\le l\le L} \sum_{v\in\mathcal{V}} \Big[\log h(y_{w_l,v}) + \Big(\eta_{w_l,v} y_{w_l,v} - \exp(\eta_{w_l,v}) \Big) \Big],$$

where the base measure $h(y_{w_l,v})$ is equal to $1/y_{w_l,v}!$. Note that, the number of occurrence $y_{w_l,v}$ is equal to 0 if v does not appear in the context of $w_l \in \mathcal{V}$. Following a similar strategy as in the EFGE-BERN model, the equation can be split into two parts for the cases where $y_{w_l,v} > 0$ and $y_{w_l,v} = 0$. That way, we can adopt the negative sampling strategy (given in Eq. (5)) as follows:

$$\sum_{\mathbf{w}\in\mathcal{W}} \sum_{1\le l\le L} \sum_{\substack{|j|\le\gamma \\ u:=w_j}} \Big[-\log(x^{l+j}_{w_l,u}!) + \eta_{w_l,u} x^{l+j}_{w_l,u} - \exp(\eta_{w_l,u}) \Big]$$
$$+ \sum_{\substack{|j|\le\gamma \\ u:\neq w_j}} \Big[-\exp(\eta_{w_l,u}) \Big].$$

Note that, in the EFGE-POIS model, we do not specify any particular link function — thus, the natural parameter is equal to the product of the embeddings vectors.

Relationship to overlapping community detection. It can be seen that the objective function of the widely used BIGCLAM overlapping community detection method by Yang and Leskovec (Yang and Leskovec 2013), can be obtained by unifying the objectives of the EFGE-BERN and EFGE-POIS models. The relationship is shown in Lemma 2. Besides, one can say that each entry of the embedding vectors correspond to a value indicating the membership of a node to

a community — in this case, BIGCLAM restricts the vectors to non-negative values.

Lemma 2. *Let $Z_{w_l,v}$ be independent random variables following Poisson distribution with natural parameter $\eta_{w_l,v}$ defined by $\log(\beta[w_l] \cdot \alpha[v])$. Then, the objective function of* EFGE-BERN *model becomes equal to*

$$\sum_{\boldsymbol{w}\in\mathcal{W}} \sum_{1\le l\le L} \Bigg[\sum_{v\in\mathcal{N}_\gamma(w_l)} \log\Big(1-\exp\big(-\beta[w_l]^\top \cdot \alpha[v]\big)\Big) - \sum_{v\notin\mathcal{N}_\gamma(w_l)} \beta[w_l]^\top \cdot \alpha[v] \Bigg]$$

if the model parameter $\pi_{w_l,v}$ defined by $p(Z_{w_l,v} > 0)$.

Proof. Please see the appendix. □

The EFGE-NORM Model

If a node v appears in the context of w_l more frequently with respect to other nodes, we can say that v has a higher interaction with w_l than the rest ones. Therefore, we will consider each $y_{w_l,v}$ in this model as an edge weight indicating the relationship between the nodes w_l and v. We assume that $x^{l+j}_{w_l,v} \sim \mathcal{N}(1,\sigma^2_+)$ if $v \in \mathcal{N}_\gamma(w_l)$, and $x^{l+j}_{w_l,v} \sim \mathcal{N}(0,\sigma^2_-)$ otherwise. Hence, we obtain that $y_{w_l,v} \sim \mathcal{N}(\tilde{\mu},\tilde{\sigma^2})$, where $\tilde{\mu}$ is the number of occurrences of v in the context if we follow a similar assumption $y_{w_l,v} = \sum_{j=-\gamma}^{\gamma} x^{l+j}_{w_l,v}$ as in the previous model. The definition of the objective function, $\mathcal{L}_N(\alpha,\beta)$, for the EFGE-NORM model is defined as follows:

$$\sum_{\mathbf{w}\in\mathcal{W}} \sum_{1\le l\le L} \sum_{\substack{|j|\le\gamma \\ u:=w_j}} \left[\log h(x^{l+j}_{w_l,u}) + \left(x^{l+j}_{w_l,u} \frac{\eta_{w_l,u}}{\sigma^+} - \frac{\eta^2_{w_l,u}}{2} \right) \right] + \sum_{\substack{|j|\le\gamma \\ u:\neq w_j}} \left[\log h(x^{l+j}_{w_l,u}) + \left(x^{l+j}_{w_l,u} \frac{\eta_{w_l,u}}{\sigma^-} - \frac{\eta^2_{w_l,u}}{2} \right) \right],$$

where the base measure $h(x_{w_l,u})$ is $\exp(-x^2_{w_l,u}/2\sigma^2)/\sqrt{2\pi}\sigma$ for known variance. In this model, we choose the link function as $f(x) = \exp(-x)$, so $\eta_{w_l,u}$ is defined as $\exp(-\beta[w_l]^\top \alpha[u])$.

Optimization

For the optimization we use *Stochastic Gradient Descent* (SGD) (Bottou 1991) to learn representations $\Omega = (\alpha,\beta)$. Since we use exponential family distributions, we have a general form of the objective function given in Eq. (4). As it is computationally very expensive to compute gradients for each node pair, we take advantage of the fact that we have formulated the objective function of each model in such a way that it could be divided into two parts according to the values of $x^{l+j}_{w_l,u}$; thus, we adopt the negative sampling strategy, setting sampling size to $k = 5$ in all the experiments. For the update of learning parameters and for generating negative samples, we follow the approach described in (Perozzi, Al-Rfou, and Skiena 2014; Mikolov et al. 2013).

	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{K}\rvert$	$\lvert\mathcal{C}\rvert$	**Avg. Degree**	**Density**
CiteSeer	3,312	4,660	6	438	2.814	0.0009
Cora	2,708	5,278	7	78	3.898	0.0014
DBLP	27,199	66,832	4	2,115	4.914	0.0002
AstroPh	17,903	19,7031	-	1	22.010	0.0012
HepTh	8,638	24,827	-	1	5.7483	0.0007
Facebook	4,039	88,234	-	1	43.6910	0.0108
GrQc	4,158	13,428	-	1	6.4589	0.0016

Table 1: Statistics of network datasets used in the experiments. $|\mathcal{V}|$: number of nodes, $|\mathcal{E}|$: number of edges, $|\mathcal{K}|$: number of labels and $|\mathcal{C}|$: number of connected components.

Experimental Evaluation

In this section, we evaluate the performance of the proposed models with respect to several node embedding baseline techniques in the node classification and link prediction tasks over various networks shown in Table 1.

Baseline methods. We evaluate the three proposed EFGE models against five state-of-the-art NRL techniques. (i) DEEPWALK (Perozzi, Al-Rfou, and Skiena 2014) generates a set of node sequences by choosing a node uniformly at random from the neighbours of the node it currently resides. (ii) NODE2VEC (Grover and Leskovec 2016) relies on a biased random walk strategy, introducing two additional parameters which are used to determine the behaviour of the random walk in visiting nodes close to the one currently residing at. We simply set these parameters to 1.0. (iii) LINE (Tang et al. 2015) learns embeddings that are based on first-order and second-order proximity (each one of length $d/2$). (iv) HOPE (Ou et al. 2016) is a matrix factorization method which aims at extracting feature vectors by preserving higher order patterns of the network (in our experiments, we have used the *Katz* index). (v) NETMF (Qiu et al. 2018) aims at factorizing the matrix approximated by the pointwise mutual information of center and context pairs. In our experiments, we have used walk length $L = 10$, number of walks $N = 80$ and window size $\gamma = 10$ for all models and the variants of EFGE model are fed with the same node sequences produced by NODE2VEC.

Node Classification

Experimental setup. In the classification task, we aim to predict the correct labels of nodes having access to a limited number of training labels (i.e., nodes with known label). In our experiments, we split the nodes into varying training ratios, from 2% up to 90% in order to better evaluate the models. We perform our experiments applying an one-vs-rest logistic regression classifier with L_2 regularization[1], computing the Micro-F_1 score (the Macro-F_1 score over a wide range of training ratios is also presented in the Appendix). We repeat the experiments for 50 times and report the average score for each network.

Experiment results. Table 2a shows the classification performance on the *CiteSeer* network. In all cases, the proposed models outperform the baselines, with the EFGE-

[1]We have used the *scikit-learn* package in the implementation.

	2%	4%	6%	8%	10%	30%	50%	70%	90%
DEEPWALK	0.416	0.460	0.489	0.505	0.517	0.566	0.584	0.595	0.592
NODE2VEC	0.450	0.491	0.517	0.530	0.541	0.585	0.597	0.601	0.599
LINE	0.323	0.387	0.423	0.451	0.466	0.532	0.551	0.560	0.564
HOPE	0.196	0.205	0.210	0.204	0.219	0.256	0.277	0.299	0.320
NETMF	0.451	0.496	0.526	0.540	0.552	0.590	0.603	0.604	0.608
EFGE-BERN	0.461	0.493	0.517	0.536	0.549	0.588	0.603	0.609	0.609
EFGE-POIS	0.484	0.514	0.537	0.551	**0.562**	0.595	**0.606**	0.611	0.613
EFGE-NORM	**0.493**	**0.525**	**0.542**	**0.553**	0.561	**0.596**	**0.606**	**0.612**	**0.616**

(a) *CiteSeer*

	2%	4%	6%	8%	10%	30%	50%	70%	90%
DEEPWALK	0.621	0.689	0.715	0.732	0.747	0.802	0.819	0.826	0.833
NODE2VEC	0.656	0.714	0.743	0.757	0.769	0.815	0.831	0.839	0.841
LINE	0.450	0.544	0.590	0.633	0.661	0.746	0.765	0.774	0.775
HOPE	0.277	0.302	0.299	0.302	0.302	0.301	0.302	0.303	0.302
NETMF	0.636	0.716	0.748	0.767	0.773	**0.821**	**0.834**	**0.841**	**0.844**
EFGE-BERN	0.668	0.720	0.743	0.759	0.767	0.808	0.823	0.834	0.838
EFGE-POIS	0.680	0.733	0.746	0.759	0.765	0.802	0.814	0.820	0.825
EFGE-NORM	**0.682**	**0.743**	**0.760**	**0.770**	**0.780**	0.810	0.824	0.827	0.839

(b) *Cora*

	2%	4%	6%	8%	10%	30%	50%	70%	90%
DEEPWALK	0.545	0.585	0.600	0.608	0.613	0.626	0.628	0.628	0.633
NODE2VEC	0.575	0.600	0.611	0.619	0.622	0.636	0.638	0.639	0.639
LINE	0.554	0.580	0.590	0.597	0.603	0.618	0.621	0.623	0.623
HOPE	0.379	0.378	0.379	0.379	0.379	0.379	0.379	0.378	0.380
NETMF	0.577	0.589	0.596	0.601	0.605	0.617	0.620	0.623	0.623
EFGE-BERN	0.573	0.598	0.610	0.617	0.622	0.634	0.638	0.638	0.638
EFGE-POIS	0.588	0.605	0.614	0.620	0.624	0.635	0.637	0.636	0.638
EFGE-NORM	**0.603**	**0.614**	**0.622**	**0.624**	**0.628**	**0.637**	**0.640**	**0.642**	**0.641**

(c) *DBLP*

Table 2: Micro-F_1 scores for the node classification experiment for varying training sizes of networks.

NORM and EFGE-POIS models being the best performing ones. The EFGE-NORM model shows the best performance among the three EFGE models, for most training sizes. The percentage gain for Micro-F_1 score of our best model with respect to the highest baseline score, is varying from 0.61% up to 9.33%. For the results on the *Cora* network shown in Table 2b, the EFGE-NORM model outperforms the baseline methods for small training set sizes of up to 10%. The EFGE-POIS also shows similar characteristics, while the EFGE-BERN model has comparable performance to NODE2VEC. The EFGE-NORM model has a gain of 4.0% against the best of baselines. For large training sets above 30%, NETMF is the best performing model over the *Cora* network. Lastly, moving on the results on the *DBLP* network shown in Table 2c, the EFGE-NORM model shows the best performance in all cases under the Micro-F_1 scores. The highest Micro-F_1 gain of our proposed models against the best performing baseline is around 4.51%.

Overall, the classification experiments show that the proposed EFGE-POIS and EFGE-NORM models perform quite well, outperforming most baselines especially on a limited number of training data. This can qualitatively be explained by the fact that, those exponential family distribution models enable to capture the number of occurrences of a node within the context of another one, while learning the embedding vectors. Of course, the structural properties of the network, such as the existence of community structure, might affect the performance of these models. For instance, as we have seen in the toy example of Fig. 1, the existence of well defined communities at the *Dolphins* network, allows the EFGE-POIS model to learn more discriminative embeddings with respect to the underlying communities (as we expect to have repetitions of nodes that belong to the same community while sampling the context of a node based on random walks).

	DEEPWALK	NODE2VEC	LINE	HOPE	NETMF	EFGE-BERN	EFGE-POIS	EFGE-NORM
Citeseer	0.770	0.780	0.717	0.744	0.742	0.815	**0.834**	0.828
Cora	0.739	0.757	0.686	0.712	0.755	0.769	0.797	**0.807**
DBLP	0.919	0.954	0.933	0.873	0.930	0.950	0.950	**0.955**
AstroPh	0.911	0.969	0.971	0.931	0.897	0.963	0.922	**0.973**
HepTh	0.843	0.896	0.854	0.836	0.882	**0.898**	0.885	0.896
Facebook	0.980	**0.992**	0.986	0.975	0.987	0.991	0.991	**0.992**
GrQc	0.921	**0.940**	0.909	0.902	0.928	0.938	0.937	**0.940**

Table 3: Area Under Curve (AUC) scores for link prediction.

Link Prediction

Experimental set-up. In the link prediction task, the goal is to predict the missing edges or to estimate possible future connections between nodes. For this experiment, we randomly remove half of the edges of a given network, keeping the residual network connected. Then, we learn node representations using the residual network. The removed edges as well as a randomly chosen set of the same number of node pairs form the testing set. For the training set, we sample the same number of non-existing edges following the same strategy to have negative samples, and the edges in the residual network are used as positive instances. Since we learn embedding vectors for the nodes of the graph, we use the extracted node representations to build edge feature vectors using the *Hadamard* product operator. Let $a, b \in \mathbb{R}^d$ be the embeddings of two nodes $u, v \in \mathcal{V}$ respectively. Then, under the Hadamard operator, the embedding of the corresponding edge between u and v will be computed as: $[a_1 * b_1, \cdots, a_d * b_d]$. In all experiments, we have used the logistic regression classifier with L_2 regularization over the networks listed in Table 1.

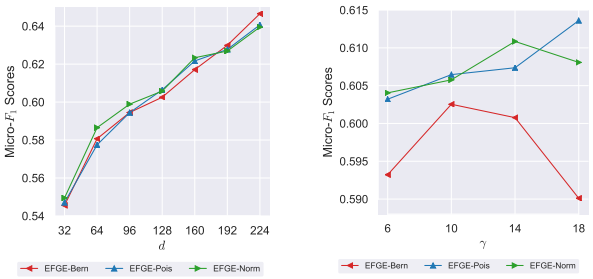
Experiment results. Table 3 shows the area under curve (AUC) scores for the link prediction task. Since the networks used in the node classification experiments consist of disconnected components, we perform the link prediction experiments on the largest connected component. As it can be seen in Table 3, the EFGE-NORM model is performing quite well on almost all different types of networks. Although NODE2VEC is quite effective having similar performance in two datasets, it is outperformed by EFGE-NORM from 0.04% up to 18.29% in the remaining networks.

Parameter Sensitivity

In this subsection, we evaluate how the performance of our models is affected under different parameter settings. In particular, we mainly examine the effect of embedding dimension d and the effect of the window size γ used to sample context nodes. More detailed analysis including the effect

of the standard deviation σ of the EFGE-NORM model is provided in the Appendix.

The effect of dimension size. The dimension size d of embedding vectors is a crucial parameter that can affect the performance of a model. We have conducted experiments examining the effect of embedding dimension d on the *Citeseer* network. As it can be seen in Fig. 2a, the increase in the dimension size has positive affect for all models over Micro-F_1 scores. When the dimension size increases from 32 up to 224, we observe a gain of around 18% for training set constructed from 50% of the network.



(a) Dimension of embeddings (b) Window size

Figure 2: Influence of dimension d and window size γ on the *CiteSeer* network for the training set ratio of 50%.

The effect of window size. Since the appearance or the number of occurrences of a node in the context of a center node is of importance for the EFGE models, we analyze their sensitivity under different window sizes γ on the *CiteSeer* network. Figure 2b depicts the Micro-F_1 scores for training set composed by 50% of the network. As we can observe, both the EFGE-NORM and EFGE-POIS models have the tendency to show better performance for large window sizes, since they directly model the number of occurrences of nodes within a random walk sequence — and potentially are benefited by a large γ value. On the contrary, the performance of the EFGE-BERN model (which in fact captures simple co-occurrence relationships, resembling NODE2VEC) deteriorates for large window sizes.

Related Work

Network representation learning. The traditional unsupervised feature learning methods aim at factorizing some matrix representation of the graph, which has been designed by taking into account the properties of a given network (Hamilton, Ying, and Leskovec 2017). Laplacian Eigenmaps (Belkin and Niyogi 2001) and IsoMap (Tenenbaum, Silva, and Langford 2000) are just some of those approaches targeting to preserve first-order proximity of nodes. More recently, proposed algorithms including GRAREP (Cao, Lu, and Xu 2015) and HOPE (Ou et al. 2016), aim at preserving higher order proximities. Nevertheless, despite the fact that matrix factorization approaches offer an elegant way to capture the desired properties, they mainly suffer from their time complexity. LINE (Tang et al. 2015) and SDNE (Wang, Cui, and Zhu 2016) both optimize more sophisticated objective functions that preserve both first- and second-order proximities at the cost of an increased computational complexity, while VERSE (Tsitsulin et al. 2018) utilizes node similarity measures to learn node representations. In addition, community structure properties can also be taken into account in the NRL process. The authors of (Wang et al. 2017), proposed a matrix factorization algorithm that incorporates the community structure into the embedding process, implicitly focusing on the quantity of modularity.

Random walk-based methods (Hamilton, Ying, and Leskovec 2017) have gained considerable attention, mainly due the efficiency of the *Skip-Gram* model. DEEPWALK performs uniform random walks to sample context nodes, while NODE2VEC and its extensions (Grover and Leskovec 2016; Nguyen and Malliaros 2018) simulate biased-random walks that provide a trade-off between breadth-first and depth-first graph traversals. Following this line of research, distinct random sampling strategies have been proposed and various methods have emerged (Ribeiro, Saverese, and Figueiredo 2017). In all those cases though, the *softmax* function is used model center-context relationships, something that might restrict the performance of the models. More recently, *Skip-Gram*-based methods were extended to multiple vector representations, aiming at capturing multiple roles of nodes in the case of inherent overlapping communities (Epasto and Perozzi 2019). In addition, it was recently shown that DEEPWALK and NODE2VEC implicitly perform matrix factorizations (Qiu et al. 2018; Qiu et al. 2019).

Recently, there is an intense research effort on Graph Neural Network (GNN) architectures (Wu et al. 2019), including graph convolutional networks, autoencoders and diffusion models. Most of these approaches are supervised or semi-supervised, requiring labeled data in the training step, while here we are interested in unsupervised models.

Exponential families. In the related literature, exponential family distributions have been utilized to learn embeddings for high-dimensional data of different types (e.g., market basket analysis) (Rudolph et al. 2016; Liu et al. 2017; Rudolph et al. 2017). As we have presented, our approach generalizes exponential family embedding models to graphs.

Conclusions

In this paper, we introduced exponential family graph embeddings (EFGE), proposing three instances (EFGE-BERN, EFGE-POIS and EFGE-NORM) that generalize random walk approaches to exponential families. The benefit of these models stems from the fact that they allow to utilize exponential family distributions over center-context node pairs, going beyond simple co-occurrence relationships. We have also examined how the objective functions of the models can be expressed in a way that negative sampling can be applied to scale the learning process. The experimental results have demonstrated that instances of the EFGE model are able to outperform widely used baseline methods. As future work, we plan to further generalize the model to other exponential family distributions.

References

[Andersen 1970] Andersen, E. 1970. Sufficiency and exponential families for discrete sample spaces. *Journal of the American Statistical Association.* 65(331):1248–1255.

[Belkin and Niyogi 2001] Belkin, M., and Niyogi, P. 2001. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, 585–591.

[Bottou 1991] Bottou, L. 1991. Stochastic gradient learning in neural networks. In *Proceedings of Neuro-Nimes. EC2*.

[Cai, Zheng, and Chang 2018] Cai, H.; Zheng, V. W.; and Chang, K. C. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Trans. Knowl. Data Eng.* 30(9):1616–1637.

[Cao, Lu, and Xu 2015] Cao, S.; Lu, W.; and Xu, Q. 2015. Grarep: Learning graph representations with global structural information. In *CIKM*, 891–900.

[Çelikkanat and Malliaros 2018] Çelikkanat, A., and Malliaros, F. D. 2018. TNE: A Latent Model for Representation Learning on Networks. In *NeurIPS Relational Representation Learning (R2L) Workshop*.

[Çelikkanat and Malliaros 2019] Çelikkanat, A., and Malliaros, F. D. 2019. Kernel Node Embeddings. In *GlobalSIP*.

[Chakrabarti, Faloutsos, and McGlohon 2010] Chakrabarti, D.; Faloutsos, C.; and McGlohon, M. 2010. *Graph mining: Laws and generators.* Chapter in *Managing and Mining Graph Data, Aggarwal, C.C., Wang, H. (eds.).* Springer.

[Epasto and Perozzi 2019] Epasto, A., and Perozzi, B. 2019. Is a Single Embedding Enough? Learning Node Representations That Capture Multiple Social Contexts. In *WWW*, 394–404.

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

[Grover and Leskovec 2016] Grover, A., and Leskovec, J. 2016. Node2Vec: Scalable feature learning for networks. In *KDD*, 855–864.

[Hamilton, Ying, and Leskovec 2017] Hamilton, W. L.; Ying, R.; and Leskovec, J. 2017. Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.* 40(3):52–74.

[Liben-Nowell and Kleinberg 2007] Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci. Technol.* 58(7):1019–1031.

[Liu et al. 2017] Liu, L.-P.; Ruiz, F. J. R.; Athey, S.; and Blei, D. M. 2017. Context selection for embedding models. In *NIPS*, 4816–4825.

[Mikolov et al. 2013] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*, 3111–3119.

[Nguyen and Malliaros 2018] Nguyen, D., and Malliaros, F. D. 2018. BiasedWalk: Biased sampling for representation learning on graphs. In *Big Data*, 4045–4053.

[Ou et al. 2016] Ou, M.; Cui, P.; Pei, J.; Zhang, Z.; and Zhu, W. 2016. Asymmetric transitivity preserving graph embedding. In *KDD*, 1105–1114.

[Perozzi, Al-Rfou, and Skiena 2014] Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. DeepWalk: Online learning of social representations. In *KDD*, 701–710.

[Qiu et al. 2018] Qiu, J.; Dong, Y.; Ma, H.; Li, J.; Wang, K.; and Tang, J. 2018. Network Embedding As Matrix Factorization: Unifying DeepWalk, LINE, PTE, and Node2Vec. In *WSDM*, 459–467.

[Qiu et al. 2019] Qiu, J.; Dong, Y.; Ma, H.; Li, J.; Wang, C.; Wang, K.; and Tang, J. 2019. NetSMF: Large-Scale Network Embedding As Sparse Matrix Factorization. In *WWW*, 1509–1520.

[Ribeiro, Saverese, and Figueiredo 2017] Ribeiro, L. F.; Saverese, P. H.; and Figueiredo, D. R. 2017. Struc2Vec: Learning node representations from structural identity. In *KDD*, 385–394.

[Rudolph et al. 2016] Rudolph, M.; Ruiz, F.; Mandt, S.; and Blei, D. 2016. Exponential family embeddings. In *NIPS*. Curran Associates Inc. 478–486.

[Rudolph et al. 2017] Rudolph, M.; Ruiz, F.; Athey, S.; and Blei, D. 2017. Structured embedding models for grouped data. In *NIPS*, 251–261.

[Tang and Liu 2009] Tang, L., and Liu, H. 2009. Relational learning via latent social dimensions. In *KDD*, 817–826.

[Tang et al. 2015] Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. LINE: Large-scale information network embedding. In *WWW*, 1067–1077.

[Tenenbaum, Silva, and Langford 2000] Tenenbaum, J. B.; Silva, V. d.; and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–2323.

[Tsitsulin et al. 2018] Tsitsulin, A.; Mottin, D.; Karras, P.; and Müller, E. 2018. VERSE: Versatile graph embeddings from similarity measures. In *WWW*, 539–548.

[Vishwanathan et al. 2010] Vishwanathan, S. V. N.; Schraudolph, N. N.; Kondor, R.; and Borgwardt, K. M. 2010. Graph kernels. *J Mach Learn Res* 11:1201–1242.

[Wang et al. 2017] Wang, X.; Cui, P.; Wang, J.; Pei, J.; Zhu, W.; and Yang, S. 2017. Community preserving network embedding. In *AAAI*, 203–209.

[Wang, Cui, and Zhu 2016] Wang, D.; Cui, P.; and Zhu, W. 2016. Structural Deep Network Embedding. In *KDD*, 1225–1234.

[Wu et al. 2019] Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Yu, P. S. 2019. A comprehensive survey on graph neural networks. *arXiv*. 1901.00596.

[Yang and Leskovec 2013] Yang, J., and Leskovec, J. 2013. Overlapping community detection at scale: A nonnegative matrix factorization approach. In *WSDM*, 587–596.

Appendix

Proofs of Lemmas

Lemma 1. *The log-likelihood function $\mathcal{L}_B$ converges to*

$$\sum_{\boldsymbol{w}\in\mathcal{W}}\sum_{1\le l\le L}\sum_{|j|\le\gamma}\Big[\log p(x^{l+j}_{w_l,w_{l+j}})+\sum_{s=1}^{k}\mathop{\mathbb{E}}_{u\sim q^-}\big[\log p(x^{l+j}_{w_l,u})\big]\Big]$$

for large values of k.

Proof. Let $q^-(\cdot|w_l)$ be the true conditional distribution of a random walk method for generating context nodes defined over $\mathcal{V}$. Then, it can be written that

$$\begin{aligned}
&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\sum_{|j|\le\gamma}\left[\log p(x^{l+j}_{w_l,v_{l+j}})+\sum_{s=1}^{k}\mathop{\mathbb{E}}_{u\sim q^-}\log p(x^{l+j}_{w_l,u})\right]\\
\approx&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\sum_{|j|\le\gamma}\left[\log p(x^{l+j}_{w_l,v_{l+j}})+k\frac{1}{k}\sum_{\substack{s=1\\u_s\sim q^-}}^{k}\log p(x^{l+j}_{w_l,u_s})\right]\\
=&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\sum_{|j|\le\gamma}\left[\log p(x^{l+j}_{w_l,v_{l+j}})+\sum_{\substack{s=1\\u_s\sim q^-}}^{k}\log p(x^{l+j}_{w_l,u_s})\right]\\
\approx&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\sum_{\substack{|j|\le\gamma\\u:=v_{l+j}}}\log p(x^{l+j}_{w_l,u})+\sum_{\substack{|j|\le\gamma\\u:\neq w_{l+j}}}\log p(x^{l+j}_{w_l,u})\\
=&\mathcal{L}_B(\alpha,\beta),
\end{aligned}$$

where the second line follows from the law of large numbers for the sample size of k, and k is set to $|\mathcal{V}|-1$ in the fourth line. □

Lemma 2. *Let $Z_{w_l,v}$ be independent random variables following Poisson distribution with natural parameter $\eta_{w_l,v}$ defined by $\log(\beta[w_l]\cdot\alpha[v])$. Then the objective function of* EFGE-BERN *model becomes equal to*

$$\sum_{\boldsymbol{w}\in\mathcal{W}}\sum_{1\le l\le L}\Bigg[\sum_{v\in\mathcal{N}_\gamma(w_l)}\log\Big(1-\exp\big(-\beta[w_l]^\top\cdot\alpha[v]\big)\Big)-\sum_{v\notin\mathcal{N}_\gamma(w_l)}\beta[w_l]^\top\cdot\alpha[v]\Bigg],$$

if the model parameter $\pi_{w_l,v}$ is defined by $p(Z_{w_l,v}>0)$.

Proof. Let $y_{w_l,v}$ follow a Bernoulli distribution with parameter $\pi_{w_l,v}$ and it is equal to 1 if $v\in\mathcal{N}_\gamma(w_l)$, and 0 otherwise. Then, the objective function $\mathcal{L}_B(\alpha,\beta)$ can be divided into parts as follows:

$$\begin{aligned}
\mathcal{L}_{\mathcal{B}}=&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\left[\sum_{v\in\mathcal{N}_\gamma(w_l)}\log p(y_{w_l,v})+\sum_{v\notin\mathcal{N}_\gamma(w_l)}\log p(y_{w_l,v})\right]\\
=&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\Bigg[\sum_{v\in\mathcal{N}_\gamma(w_l)}\log\big(1-p(z_{w_l,v}=0)\big)\\
&\qquad+\sum_{v\notin\mathcal{N}_\gamma(v_i)}\log p(z_{w_l,v}=0)\Bigg]\\
=&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\Bigg[\sum_{v\in\mathcal{N}_\gamma(w_l)}\log\Big(1-\exp\big(-\exp(\eta_{w_l,v})\big)\Big)\\
&\qquad-\sum_{v\notin\mathcal{N}_\gamma(w_l)}\exp(\eta_{w_l,v})\Bigg]\\
=&\sum_{\mathbf{w}\in\mathcal{W}}\sum_{1\le l\le L}\Bigg[\sum_{v\in\mathcal{N}_\gamma(w_l)}\log\Big(1-\exp\big(-\beta[w_l]^\top\cdot\alpha[v]\big)\Big)\\
&\qquad-\sum_{v\notin\mathcal{N}_\gamma(w_l)}\beta[w_l]^\top\cdot\alpha[v]\Bigg]
\end{aligned}$$

□

Dataset Description

Here we provide a detailed description of the graph datasets[2] used in our study.

- *CiteSeer* is a citation network obtained from the *CiteSeer* library, in which each node corresponds to a paper and the edges indicate reference relationships among papers. The labels represent the subjects of the paper.
- *Cora* is another citation network constructed from the publications in the machine learning area; the documents are classified into seven categories.
- *DBLP* is a co-authorship graph, where an edge exists between nodes if two authors have co-authored at least one paper. The labels represent the research areas.
- *AstroPh* is another collaboration network built from the papers submitted to the *ArXiv* repository for the Astro Physics subject area, from January 1993 to April 2003.
- *HepTh* network is constructed in a similar way from the papers submitted to *ArXiv* for the *High Energy Physics - Theory* category.
- *GrQc* is our last collaboration network which has been constructed from the e-prints submitted to the category of *General Relativity and Quantum Cosmology*.
- *Facebook* is a social network extracted from a survey conducted via a *Facebook* application.

[2]The datasets can be found at: https://snap.stanford.edu/data and https://github.com/GTmac/HARP

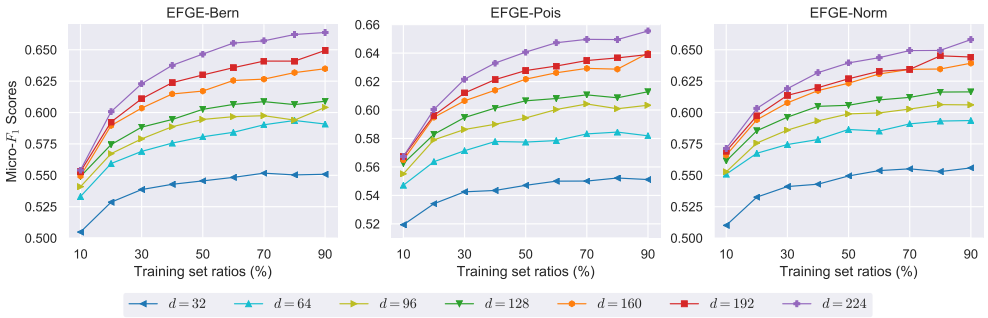


Figure 3: Influence of dimension size over *CiteSeer* network.

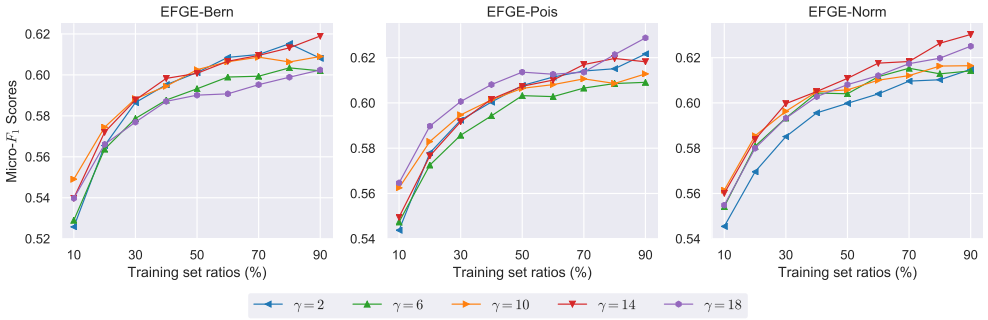


Figure 4: Influence of window size γ for the *CiteSeer* network.

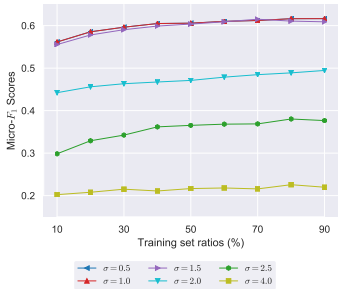


Figure 5: Effect of standard deviation for EFGE-NORM.

Complementary Experimental Results for Node Classification

In this section we present complementary experimental results for node classification by reporting the Micro-F_1 and Macro-F_1 scores for training size ratios varying from 1% up to 90%. The scores are presented in Tables 4, 5 and 6.

Parameter Sensitivity

In this section, we perform some further sensitivity analysis experiments. In particular, we present complementary experiments where we examine the effect of embedding dimension d and the effect of the window size γ used to sample context nodes for the different models over different training set ratios. The results are depicted in Figures 3 and 4.

Effect of standard deviation of EFGE-NORM model The EFGE-NORM model has an extra parameter σ which can influence the performance of the method. To examine the impact of σ, we have chosen six different values, performing experiments over *CiteSeer* network. Figure 5 depicts how the Micro-F_1 scores change for various training set ratios. The results clearly indicate that the model performs well for small values of σ — with the best results obtained for $\sigma = 1$. For this reason, we have set this value for all the experiments conducted in the node classification and link prediction tasks.

		1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	20%	30%	40%	50%	60%	70%	80%	90%
BASELINES	DEEPWALK	0.373	0.416	0.443	0.460	0.471	0.489	0.496	0.505	0.506	0.517	0.548	0.566	0.576	0.584	0.590	0.595	0.591	0.592
		0.318	0.367	0.403	0.418	0.429	0.446	0.454	0.463	0.464	0.474	0.505	0.521	0.529	0.537	0.542	0.547	0.541	0.543
	NODE2VEC	0.405	0.450	0.475	0.491	0.500	0.517	0.524	0.530	0.537	0.541	0.570	0.585	0.590	0.597	0.598	0.601	0.596	0.599
		0.339	0.396	0.424	0.441	0.452	0.470	0.478	0.483	0.491	0.494	0.525	0.537	0.542	0.549	0.550	0.553	0.548	0.551
	LINE	0.273	0.323	0.362	0.387	0.406	0.423	0.440	0.451	0.456	0.466	0.513	0.532	0.543	0.551	0.556	0.560	0.568	0.564
		0.204	0.268	0.311	0.338	0.360	0.372	0.390	0.399	0.407	0.414	0.459	0.480	0.492	0.498	0.505	0.505	0.514	0.513
	HOPE	0.194	0.196	0.202	0.205	0.208	0.210	0.216	0.204	0.216	0.219	0.228	0.256	0.267	0.277	0.293	0.299	0.300	0.320
		0.060	0.060	0.063	0.062	0.066	0.068	0.079	0.064	0.075	0.078	0.094	0.127	0.136	0.150	0.168	0.178	0.183	0.205
	NETMF	0.379	0.451	0.472	0.496	0.515	0.526	0.533	0.540	0.544	0.552	0.578	0.590	0.596	0.603	0.605	0.604	0.611	0.608
		0.315	0.400	0.423	0.445	0.464	0.477	0.486	0.490	0.497	0.503	0.529	0.542	0.546	0.553	0.554	0.552	0.560	0.554
EFGE	EFGE-BERN	0.411	0.461	0.487	0.493	0.513	0.517	0.528	0.536	0.543	0.549	0.574	0.588	0.594	0.603	0.606	0.609	0.606	0.609
		0.345	0.410	0.436	0.446	0.462	0.468	0.481	0.489	0.496	0.502	0.530	0.545	0.549	0.558	0.563	0.564	0.561	0.563
	EFGE-POIS	**0.449**	0.484	0.500	0.514	0.530	0.537	0.544	0.551	0.555	**0.562**	0.583	0.595	0.601	**0.606**	0.608	0.611	0.609	0.613
		0.384	0.425	0.445	0.461	**0.477**	0.482	0.491	**0.500**	**0.504**	**0.512**	0.533	0.547	0.552	0.558	0.559	0.563	0.560	0.564
	EFGE-NORM	0.434	**0.493**	**0.510**	**0.525**	**0.531**	**0.542**	**0.546**	**0.553**	**0.557**	0.561	**0.585**	**0.596**	**0.605**	**0.606**	**0.610**	**0.612**	**0.616**	**0.616**
		0.361	**0.431**	**0.450**	**0.467**	0.474	**0.485**	**0.492**	**0.500**	0.501	0.509	**0.534**	**0.547**	**0.558**	**0.560**	**0.566**	**0.567**	**0.572**	**0.575**

Table 4: Node classification for varying training sizes for the *Citeseer* network. For each method, the first row indicates the Micro-F_1 scores and the second one shows the Macro-F_1 scores.

		1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	20%	30%	40%	50%	60%	70%	80%	90%
BASELINES	DEEPWALK	0.531	0.621	0.667	0.689	0.703	0.715	0.727	0.732	0.744	0.747	0.784	0.802	0.810	0.819	0.822	0.826	0.826	0.833
		0.455	0.571	0.638	0.666	0.681	0.698	0.711	0.717	0.730	0.734	0.774	0.792	0.800	0.809	0.813	0.816	0.815	0.825
	NODE2VEC	0.575	0.656	0.696	0.714	0.731	0.743	0.746	0.757	0.764	0.769	0.799	0.815	0.824	0.831	0.835	0.839	**0.842**	0.841
		0.501	0.605	0.665	0.687	0.713	0.723	0.730	0.741	0.750	0.755	0.786	0.803	0.812	0.819	0.823	0.826	0.830	0.825
	LINE	0.384	0.450	0.506	0.544	0.568	0.590	0.618	0.633	0.648	0.661	0.723	0.746	0.758	0.765	0.770	0.774	0.775	0.775
		0.272	0.364	0.435	0.491	0.523	0.555	0.588	0.607	0.626	0.642	0.713	0.736	0.747	0.755	0.759	0.762	0.766	0.764
	HOPE	0.260	0.277	0.297	0.302	0.304	0.299	0.302	0.302	0.302	0.302	0.303	0.301	0.303	0.302	0.303	0.303	0.303	0.302
		0.065	0.068	0.067	0.066	0.068	0.066	0.066	0.066	0.066	0.066	0.067	0.067	0.067	0.067	0.067	0.068	0.070	0.072
	NETMF	0.534	0.636	0.693	0.716	0.735	0.748	0.757	0.767	0.770	0.773	**0.807**	**0.821**	**0.828**	**0.834**	**0.839**	**0.841**	0.839	**0.844**
		0.461	0.591	0.667	0.694	0.717	0.731	0.741	0.751	0.757	0.760	**0.797**	**0.811**	**0.819**	**0.824**	**0.830**	**0.832**	**0.831**	**0.835**
EFGE	EFGE-BERN	0.593	0.668	0.703	0.720	0.738	0.743	0.751	0.759	0.760	0.767	0.792	0.808	0.815	0.823	0.828	0.834	0.837	0.838
		0.507	0.622	0.672	0.695	0.718	0.723	0.735	0.744	0.743	0.754	0.780	0.798	0.806	0.814	0.819	0.823	0.825	0.824
	EFGE-POIS	**0.605**	0.680	0.714	0.733	0.739	0.746	0.752	0.759	0.761	0.765	0.791	0.802	0.809	0.814	0.817	0.820	0.824	0.825
		0.512	**0.630**	**0.685**	0.709	0.715	0.731	0.737	0.744	0.748	0.752	0.780	0.792	0.798	0.803	0.807	0.810	0.815	0.813
	EFGE-NORM	0.601	**0.682**	**0.720**	**0.743**	**0.754**	**0.760**	**0.765**	**0.770**	**0.776**	**0.780**	0.799	0.810	0.814	0.824	0.824	0.827	0.832	0.839
		0.512	0.626	**0.685**	**0.711**	**0.730**	**0.741**	**0.746**	**0.754**	**0.760**	**0.766**	0.785	0.796	0.800	0.810	0.810	0.812	0.818	0.824

Table 5: Node classification for varying training sizes for the *Cora* network. For each method, the first row indicates the Micro-F_1 scores and the second one shows the Macro-F_1 scores.

		1%	2%	3%	4%	5%	6%	7%	8%	9%	10%	20%	30%	40%	50%	60%	70%	80%	90%
BASELINES	DEEPWALK	0.517	0.545	0.569	0.585	0.593	0.600	0.604	0.608	0.610	0.613	0.622	0.626	0.627	0.628	0.627	0.628	0.629	0.633
		0.462	0.492	0.510	0.522	0.529	0.535	0.537	0.542	0.543	0.545	0.552	0.555	0.556	0.556	0.555	0.556	0.557	0.559
	NODE2VEC	0.557	0.575	0.590	0.600	0.605	0.611	0.615	0.619	0.621	0.622	0.632	0.636	0.638	0.638	0.640	0.639	0.640	0.639
		0.497	0.517	0.532	0.541	0.545	0.552	0.554	**0.558**	0.558	0.560	**0.569**	**0.571**	**0.572**	0.572	**0.574**	0.573	**0.574**	**0.573**
	LINE	0.525	0.554	0.570	0.580	0.587	0.590	0.594	0.597	0.600	0.603	0.613	0.618	0.619	0.621	0.621	0.623	0.623	0.623
		0.434	0.475	0.499	0.510	0.519	0.521	0.526	0.530	0.533	0.535	0.548	0.552	0.554	0.556	0.555	0.557	0.558	0.559
	HOPE	0.376	0.379	0.379	0.378	0.378	0.379	0.379	0.379	0.378	0.379	0.378	0.379	0.379	0.379	0.379	0.378	0.379	0.380
		0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.137	0.138	0.138
	NETMF	0.564	0.577	0.586	0.589	0.593	0.596	0.599	0.601	0.604	0.605	0.613	0.617	0.619	0.620	0.620	0.623	0.623	0.623
		0.463	0.490	0.503	0.506	0.510	0.513	0.517	0.518	0.521	0.522	0.528	0.530	0.532	0.531	0.531	0.533	0.533	0.533
EFGE	EFGE-BERN	0.554	0.573	0.586	0.598	0.604	0.610	0.615	0.617	0.618	0.622	0.631	0.634	0.635	0.638	0.637	0.638	0.638	0.638
		0.494	0.514	0.526	0.539	0.542	0.547	0.551	0.553	0.553	0.556	0.563	0.566	0.566	0.569	0.568	0.569	0.570	0.570
	EFGE-POIS	0.574	0.588	0.598	0.605	0.611	0.614	0.618	0.620	0.623	0.624	0.631	0.635	0.636	0.637	0.638	0.636	0.638	0.638
		0.509	0.528	0.538	0.544	0.549	0.552	0.554	0.556	0.559	0.559	0.565	0.568	0.569	0.569	0.570	0.569	0.570	0.570
	EFGE-NORM	**0.596**	**0.603**	**0.610**	**0.614**	**0.618**	**0.622**	**0.622**	**0.624**	**0.627**	**0.628**	**0.635**	**0.637**	**0.640**	**0.640**	**0.641**	**0.642**	**0.642**	**0.641**
		0.520	**0.533**	**0.544**	**0.548**	**0.552**	**0.556**	**0.555**	**0.558**	**0.560**	**0.562**	0.568	0.569	**0.572**	**0.573**	0.572	**0.574**	0.573	0.572

Table 6: Node classification for varying training sizes for the *DBLP* network. For each method, the first row indicates the Micro-F_1 scores and the second one shows the Macro-F_1 scores.