Density-adaptive Local Edge Representation Learning with Generative Adversarial Network Multi-label Edge Classification

Yang Zhou[†], Sixing Wu[‡], Chao Jiang[†], Zijie Zhang[†], Dejing Dou[§], Ruoming Jin[¶], Pengwei Wang[‖]
[†]Auburn University [‡]Peking University [§]University of Oregon [¶]Kent State University [‖]Alibaba
{yangzhou, czj0043, zzz0092}@auburn.edu, wusixing@pku.edu.cn, dou@cs.uoregon.edu
jin@cs.kent.edu, hoverwang.wpw@alibaba-inc.com

***Abstract*—Traditional network representation learning techniques aim to learn latent low-dimensional representation of vertices in graphs. This paper presents a novel edge representation learning framework,** GANDLERL**, that combines generative adversarial network based multi-label classification with density-adaptive local edge representation learning for producing high-quality low-dimensional edge representations. First, we design a generative adversarial network based multi-label edge classification model to classify rarely labeled edges in graphs with a large amount of noise data into K classes. A four-player zero sum game model, with the mixed training of true and real-looking fake edges as well as a contrastive loss containing a similar-loss and a dissimilar-loss, is proposed to improve the classification quality of unlabeled edges. Second, a local autoencoder edge representation learning method is developed to design K local representation learning models, each with individual parameters and structure to perform local representation learning on each of K classification-based subgraphs with unique local characteristics and jointly optimize the loss functions within and across classes. Third but last, we propose a density-adaptive edge representation learning method with the optimization at both edge and subgraph levels to address the representation learning of graph data with highly imbalanced vertex degree and edge distribution.**

***Index Terms*—generative adversarial network multi-label edge classification, density-adaptive local edge representation learning**

I. Introduction

Network representation learning, which aims to learn latent low-dimensional representation of vertices in graphs, has received increasing attention in recent years [1]–[13]. Network representation learning techniques have great potential for understanding of large-scale graphs and their underlying processes. However, how to learn latent low-dimensional representation of edges in graphs for further improving the efficiency and scalability of compute-intensive edge analysis tasks has not been adequately investigated yet.

Edge analysis tasks with high time complexity. As online social media and online shopping sites become ubiquitous, we have witnessed an increased interest in edge data analysis, such as link prediction [14], [15], edge clustering [16]–[18], edge classification [19], edge-weight anonymization [20], etc. Scaling computation on large graphs with millions or billions of edges is widely recognized as a challenging big data research problem.

Data Sparsity and Imbalanced Graph Distribution. Almost all real-world graphs are extremely sparse. As shown in Tables I, all graph datasets with small density values are very sparse since #edges in these graphs is much smaller than #edges in their corresponding complete graphs with the density value of 1. Also, real sparse graphs commonly have highly skewed vertex degree and edge distribution. Concretely, a small number of high-degree vertices connect to a large fraction of graph, but a large number of low-degree vertices have very few or no edges connected to other vertices. In reality, edge analysis applications often fail to analyze edges associated with low-degree vertices or sparse subgraphs due to inadequate information. Running global edge representation learning on the whole graphs without considering density-based local characteristics of edges and subgraphs, such as vertex degree distribution, edge distribution, and size and density of subgraphs, may lead to sub-optimal edge representations. On one hand, if moderate global edge representation learning is performed to learn low-dimensional sparse representation, then it may be insufficient to produce dense enough representations for edges associated with low-degree vertices and sparse subgraphs, and thus fail to capture the underlying correlations between edges. On the other hand, it is unnecessary for edges associated with high-degree vertices and dense subgraphs to execute excessive global representation learning targeting low-dimensional dense representation, since there is already enough information for future prediction. In addition, excessive global edge representation learning may introduce huge reconstruction errors or random noise and could result in serious efficiency deterioration.

Rarely labeled and noise data. A straightforward way of local edge representation learning is to divide the whole graph into K disjoint subgraphs, in which edges within each subgraph have similar local characteristics, and use K local edge representation learning models to train K subgraphs and optimize the objective function on each subgraph. The unsupervised and supervised learning approaches are two important types of techniques to solve the problem of graph partitioning. In comparison with unsupervised learning, supervised learning can often achieve better graph partitioning result with the help

978-1-5386-9159-5/18/$31.00 ©2018 IEEE
DOI 10.1109/ICDM.2018.00203

IEEE computer society

of labeled training data. However, the reality is that real-world graphs often contain a small portion of labeled data with a large portion of unlabeled data.

This paper makes the following contributions to local edge representation learning for sparse graphs with noise data.

A **generative adversarial network (GAN) based multi-label edge classification** model is proposed to classify rarely labeled edges in graphs with a large amount of noise data into K classes. We introduce additional classifier and classification learner to the GAN model and build a four-player zero sum game model to handle the issue of multi-label edge classification. In order to ensure the robustness to noise, we train the classifier by mixing true edges and generated real-looking fake edges. Also, a contrastive loss function is proposed to capture the underlying edge homophily property in graph data as well as improve the classification quality of unlabeled edges.

A **local autoencoder edge representation learning** approach is proposed to design K local representation learning models, each with individual parameters and structure, including weights, biases, #neurons, #hidden layers, and convergence, to perform local representation learning on each of K classification-based subgraphs with unique local characteristics and jointly optimize the loss functions within and across classes. This framework helps reduce the reconstruction error between the output and the input, preserve classification and network structure, and make edges within/across classes have similar/dissimilar representation features.

A **density-adaptive edge representation learning** method with two-level optimization is proposed to address the representation learning of sparse graph data with highly imbalanced vertex degree and edge distribution. At edge level, based on imbalanced degree distribution of two associated vertices, a unique density control parameter is introduced and optimized for each edge to control the activation of hidden neurons in the learning process and the density of its final representation, in case of overdense or too sparse final representation. At subgraph level, in response to imbalanced size and density of K subgraphs, we also introduce a density control parameter for each local representation learning model to control the density of the representation of the corresponding subgraph. This allows to learn the dense representations of edges associated with low-degree vertices and sparse subgraphs to introduce more auxiliary information for them, and thus help improve the prediction quality. On the other hand, the sparse representations of edges associated with high-degree vertices and dense subgraphs are learnt to reduce information to be processed, and thus avoid huge reconstruction errors as well as help improve the prediction efficiency.

II. Related Work

Network representation learning or **network embedding** has become an active research field that learns low-dimensional vertex representations from network structure [1]–[13]. DeepWalk [1] learns latent social representations of vertices, by using local information obtained from truncated random walks and treating walks as the equivalent of sentences. LINE [2] designes two loss functions to preserve both the first-order and second-order proximities. node2vec [3] is a scalable network feature learning framework by maximizing the likelihood of preserving network neighborhoods of nodes. M-NMF [5] combines network embedding and community detection into a unified model to perform global network embedding on the whole graph. TransNet [7] is a translation-based network representation learning model to model and predict social relations between vertices with translation mechanism. MCGE [8] is a multi-view clustering framework on graph instances with graph embedding by taking advantages of the consensus and complimentary information from multiple views to partition multiple graphs into clusters. Abu-El-Haija et al. [15] propose a link prediction method by modeling an edge as a function of the embeddings of two vertices and utilizing the graph likelihood technique to determine whether there exists an edge between these two vertices. GraphGAN [12] is a link prediction method to use the GAN model to learn the connectivity distribution in graphs.

Node classification/clustering in networked data has attracted active research in the last decade [16], [18], [19], [26]–[34]. DYCOS [35] exhibited a node classification model in dynamic information networks with both text content and links. RankClass [36] integrates classification and ranking in a mutually enhancing process to provide class summaries for heterogeneous information networks. HCC [37] is a collective classification algorithm for analyzing heterogeneous information networks, by capturing different dependencies among instances with respect to different meta paths. SCRN [17] improves the classification quality by integrating both the network topology and the social context features extracted together. SNOC [38] is a node classification method for streaming networks that integrates network structure and node labels to find an optimal subset of features to represent the network.

III. Problem Definition

We formally define our research problem as follows. A *graph* is denoted as $G = (V, E)$, where V is the set of M vertices and E is the set of N edges. A vertex $v_i \in V (1 \leq i \leq M)$ represents an entity in G and each edge $x_s = (v_i, v_j) \in E (1 \leq s \leq N)$ is associated with two vertices $v_i, v_j \in V$ and denotes the relationships between two corresponding entities. Based on our previously proposed the edge-centric random walk model [18], the original representation of an edge x_s, denoted by $\mathbf{x}_s$, is defined as the s^{th} row in the transition matrix $\mathbf{T}$ on the corresponding edge-centric graph of G. The problem of **Generative Adversarial Network multi-label edge classification based Density-adaptive Local Edge Representation Learning** (GANDLERL) consists of the following two analysis tasks.

Let $C = \{c_1, c_2, \cdots, c_K\}$ be a finite set of K possible class labels. Given a graph $G = (V, E)$ with a small set of labeled edges $E_L \subset E$, a large set of unlabeled edges $E_U = E - E_L$, and $|E_L| << |E_U|$. The task of **semi-supervised multi-label**

edge classification is to use all edges in E_L and some edges in E_U as the training data to classify other testing edges in E_U into K classes. We further convert the multi-label edge classification result to the multi-class edge classification result.

We then reorganize the matrix $\mathbf{T}$ into K disjoint subgraphs $\mathbf{X}_1, \cdots, \mathbf{X}_K$, such that edges with the same labels lie together, i.e., $\mathbf{T} = [\mathbf{X}_1 \dots \mathbf{X}_K]^T$. Each $\mathbf{X}_k$ corresponds to the transition probabilities from edges belonging to class c_k to all edges in G. The **density-adaptive local edge representation learning** is to map the original representation $\mathbf{x}_s$ of each edge $x_s \in E$ on each subgraph $\mathbf{X}_k$ to a low-dimensional representation $\mathbf{y}_s$, i.e., $\mathbf{x}_s : \mathbb{R}^N \mapsto \mathbf{y}_s : \mathbb{R}^D$ and $D << N$.

IV. GAN-based Multi-label Edge Classification

Generative adversarial network (GAN) model was proposed to generate photographs that look authentic to human observers [39]. Essentially, GAN can be treated as a two-class classification model that determines whether an input image is real or fake. One advantage of the GAN model is robust to noise data, especially real-looking fake data. This motivates us to extend it to a four-player game model to handle the multi-label classification of rarely labeled edges with noise data.

Concretely, given the empirical distributions $p(x)$ and $p(y)$ of real edges x and true labels y, our GAN-based multi-label classification model contains four players: (1) a discriminator $\mathbf{D}(x, y)$ determines whether a sample edge-label pair (x, y) comes from the true distribution $p(x, y)$ rather than the prediction distribution $p_{\mathbf{C}}(x, y)$ and the fake distribution $p_{\mathbf{G}}(x, y)$; (2) a generator $x = \mathbf{G}(y)$ takes a real label y as input and generates a fake edge sample x based on the fake conditional distribution $p_{\mathbf{G}}(x|y)$, and then the generated fake edge $x = \mathbf{G}(y)$ is fed into the discriminator $\mathbf{D}(x, y)$ or the classification learner $\mathbf{L}(x, y)$. Thus, the fake edge-label pair is a sample from the joint distribution $p_{\mathbf{G}}(x, y) = p(y)p_{\mathbf{G}}(x|y)$; (3) a multi-label classifier $y = \mathbf{C}(x)$ predicts the class labels of a real edge x following the class conditional distribution $p_{\mathbf{C}}(y|x)$; and (4) a classification learner $\mathbf{L}(x, y)$ is used to decide whether the sample edge-label pair (x, y) comes from the prediction distribution $p_{\mathbf{C}}(x, y)$ or not. As (x, y) is sampled by drawing x from $p(x)$ and drawing y from the classifier $y = \mathbf{C}(x)$ following $p_{\mathbf{C}}(y|x)$. Thus, the joint distribution $p_{\mathbf{C}}(x, y) = p(x)p_{\mathbf{C}}(y|x)$.

The goal of GAN-based multi-label edge classification is to match the prediction distribution $p_{\mathbf{C}}(x, y)$ produced by the classifier $y = \mathbf{C}(x)$ with the true distribution $p(x, y)$, i.e, $p_{\mathbf{C}}(x, y) = p(x, y)$.

$$\min_{\mathbf{C},\mathbf{G}} \max_{\mathbf{D},\mathbf{L}} \mathcal{V}(\mathbf{D}, \mathbf{L}, \mathbf{C}, \mathbf{G}) = \mathbb{E}_{(x,y)\sim p(x,y)} \log \mathbf{D}(x, y) + \mathbb{E}_{x\sim p(x), y\sim p_{\mathbf{C}}(y|x)} \log \mathbf{L}(x, \mathbf{C}(x)) + \mathbf{KL}\Big(p(x, y)|p_{\mathbf{C}}(x, y)\Big) + \mathbb{E}_{y\sim p(y), x\sim p_{\mathbf{G}}(x|y)} \log \Big(1 - \mathbf{D}(\mathbf{G}(y), y) - \mathbf{L}(\mathbf{G}(y), y)\Big) \quad (1)$$

We further use the softmax assignment to normalize the classifier output and produce the class-membership probabilities of edges on each class.

$$p(x, y = k) = \frac{e^{p_{\mathbf{C}}(x,y=k)}}{\Sigma_{k=1}^{K} e^{p_{\mathbf{C}}(x,y=k)}}, x \in E, k \in \{1, \cdots, K\} \quad (2)$$

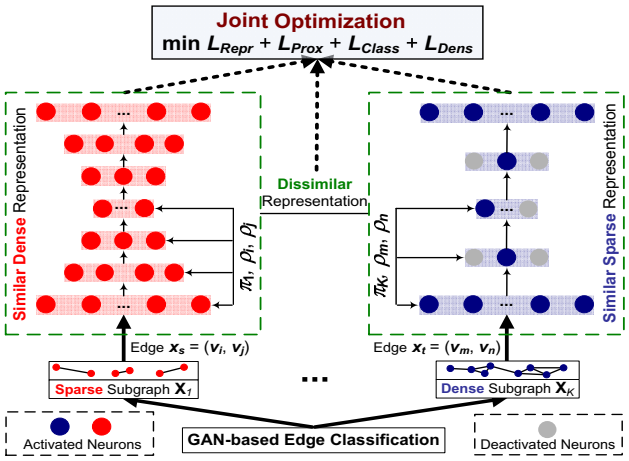


Fig. 1. Density-adaptive Local Autoencoder Edge Representation Learning

Many vertex-centric graph mining algorithms are based on the existence of vertex homophily. This motivates us to propose the concept of edge homophily, i.e., similar edges in nature share common vertices, to further improve the accuracy of multi-label edge classification. We introduce a contrastive loss $\mathcal{L}$ containing a similar-loss and a dissimilar-loss to capture the edge homophily.

$$\mathcal{L} = \min_{\mathbf{C}} \gamma \sum_{x_s, x_t \in E, x_s \wedge x_t \neq \emptyset} \Big(1 - cosine\big(p_{\mathbf{C}}(x_s, y_s), p_{\mathbf{C}}(x_t, y_t)\big)\Big) + (1 - \gamma) \sum_{x_s, x_t \in E, x_s \wedge x_t = \emptyset} \max\Big\{0, cosine\big(p_{\mathbf{C}}(x_s, y_s), p_{\mathbf{C}}(x_t, y_t)\big) - m\Big\} \quad (3)$$

where $x_s \wedge x_t$ denotes whether two edges x_s and x_t in E share common vertices or not, and $cosine\big(p_{\mathbf{C}}(x_s, y_s), p_{\mathbf{C}}(x_t, y_t)\big)$ specifies the cosine similarity between two edges regarding their class-membership distributions y_s and y_t on K classes. A positive margin value m indicates that the similarity between two edges without common vertices that is below the margin will not contribute to the loss.

Finally, we transform the multi-label edge classification result into the multi-class edge classification result as follow.

$$y = \arg\max_{k} \; p(x, y = k), \forall k, 1 \leq k \leq K \quad (4)$$

V. Density-adaptive Local Edge Representation Learning

We transmit each of K classification-based subgraphs with unique density-based local characteristics to one of K local representation learning models with exclusive hyperparameters and structure, including weights, biases, #neurons, and #hidden layers, to individually perform local edge representation learning with the joint objective function by considering the correlations within and across classes. In response to imbalanced vertex degree distribution, edge distribution, and size and density of subgraphs, we develop a density-adaptive edge representation learning method with the optimization at both edge and subgraph levels. as shown in Figure 1.

A. Local Edge Representation Learning

At subgraph level, in response to imbalanced edge distribution and size and density of subgraphs, each subgraph $\mathbf{X}_k$ has its own local autoencoder edge representation learning model and neural network with one input layer (Layer 1), multiple hidden layers, one output layer (Layer $2L_k-1$), unique weight and bias parameters. Each local autoencoder model performs individual feedforward representation learning to map the original representation $\mathbf{x}_s$ of each edge on each $\mathbf{X}_k$ to a low-dimensional representation $\mathbf{y}_s$. We choose less neurons, less hidden layers, and coarse-grained convergence to decrease data transformation in autoencoder, and thus reduce the reconstruction errors as well as improve the overall computational cost of local representation learning on dense subgraphs, while selecting more neurons, more hidden layers, and fine-grained convergence for sparse subgraphs to introduce more auxiliary information and thus help improve the prediction quality. In addition, we introduce a density control parameter for each subgraph to control the density of its final representation, i.e., perform dense/sparse representation learning on sparse/dense subgraphs to learn the dense/sparse representations of edges. At edge level, based on imbalanced degree distribution of two associated vertices, a unique density control parameter is introduced and optimized for each edge to control the activation of hidden neurons in the learning process and the density of its final representation. We control the activation of the representation of each edge in each layer, to learn the dense/sparse representations of edges with low/high-degree vertices and make edges on the same subgraphs have similar representation features.

Concretely, we use $\mathbf{x}_s$ as the input of an edge x_s in class c_k to learn its low-dimensional representation $\mathbf{y}_s$ and reconstruction representation $\mathbf{x}'_s$. The k^{th} neural network for training edges on subgraph $\mathbf{X}_k$ has $2L_k-1$ layers: layer 1 inputs the original representation $\mathbf{x}_s$, layer L_k produces the low-dimensional representation $\mathbf{y}_s$, and layer $2L_k-1$ outputs the reconstruction $\mathbf{x}'_s$ of edge x_s To compute the $\mathbf{y}_s$, and $\mathbf{x}'_s$, we need to compute all hidden representations from layers 2 to $2L_k-1$ below.

$$\begin{aligned} \mathbf{y}_s^{(2)} &= f(\mathbf{W}_k^{(1)}\mathbf{x}_s + \mathbf{b}_k^{(1)}) \\ &\cdots \\ \mathbf{y}_s^{(l)} &= f(\mathbf{W}_k^{(l-1)}\mathbf{y}_s^{(l-1)} + \mathbf{b}_k^{(l-1)}) \end{aligned} \tag{5}$$

where $f(x)$ denotes the element-wise sigmoid function $\frac{1}{1+e^{-x}}$, $\mathbf{W}_k^{(l)}$ denotes the weights of the links between layers l and $l+1$, and $\mathbf{b}_k^{(l)}$ is the bias associated with layer $l+1$. For ease of presentation, we let symbol $\mathbf{y}_s$ replace the final representation $\mathbf{y}_s^{(L_k)}$ and use $\mathbf{x}'_s$ to substitute the reconstruction $\mathbf{y}_s^{(2L_k-1)}$.

In order to define a density control parameter for each subgraph $\mathbf{X}_k$, we first compute its density value below.

$$D(\mathbf{X}_k) = \frac{nnz(\mathbf{X}_k)}{|c_k| \times N} \tag{6}$$

where $nnz(\mathbf{X}_k)$ denotes the number of non-zero elements in the submatrix $\mathbf{X}_k$, $|c_k|$ is the number of edges belonging to c_k, and the density $D(\mathbf{X}_k)$ is bounded in the range $[0, 1]$.

A density control parameter π_k for each local edge representation learning model to control the density of the intermediate and final representations of the corresponding subgraph $\mathbf{X}_k$.

$$\pi_k = \begin{cases} 1 - \dfrac{D(\mathbf{X}_k)}{\max\limits_{1\le m\le K} D(\mathbf{X}_m)}, & D(\mathbf{X}_k) \neq \max\limits_{1\le m\le K} D(\mathbf{X}_m), \\ \varepsilon, & otherwise. \end{cases} \tag{7}$$

where $\max\limits_{1\le m\le K} D(\mathbf{X}_m)$ is the largest density among K subgraphs and ε is a small positive number to avoid zero activation of the subgraph with the largest density. Intuitively, the larger the $D(\mathbf{X}_k)$ is, the smaller the π_k is.

We also define a density control parameter ρ_s for each vertex $x_s = (v_i, v_j)$ to control the activation of hidden neurons in each layer in the learning process and the density values of the final representations of x_s.

$$\rho_s = \begin{cases} 1 - \dfrac{d_i + d_j}{2N}, & d_i, d_j \neq N, \\ \eta, & otherwise. \end{cases} \tag{8}$$

where d_x is the degree of v_x and η is a small positive number to avoid zero activation of the vertices with the full degree of N. The larger the d_i and d_j are, the smaller the ρ_s is.

The first goal of our GANDLERL method is to minimize the reconstruction error between the output and the input. $\mathcal{L}_{Repr}$ is thus defined below.

$$\mathcal{L}_{Repr} = \Sigma_{k=1}^{K}\Sigma_{x_s\in c_k}||\mathbf{x}'_s - \mathbf{x}_s||_2^2 + \omega_1\Sigma_{k=1}^{K}\Sigma_{l=1}^{2L_k-2}||\mathbf{W}_k^{(l)}||_F^2 \tag{9}$$

where the first term denotes the reconstruction error between the output $\mathbf{x}'_s$ and the input $\mathbf{x}_s$ of edge x_s belonging to class c_k, and the second term is a regularization term that decreases the magnitude of the weights and prevents the overfitting.

In order to maintain the property of edge homophily, i.e., similar edges in nature share common vertices, the goal of proximity preservation aims to make edges with/without common vertices have similar/dissimilar representation features.

$$\begin{aligned} \mathcal{L}_{Prox} = \omega_2 &\sum_{x_s,x_t\in E, x_s\wedge x_t\neq\emptyset} ||\mathbf{y}_s - \mathbf{y}_t||_2^2 + \\ (1-\omega_2) &\sum_{x_s,x_t\in E, x_s\wedge x_t=\emptyset} \max\left\{0, m - ||\mathbf{y}_s - \mathbf{y}_t||_2^2\right\} \end{aligned} \tag{10}$$

The classification loss $\mathcal{L}_{Class}$ is defined as the difference between the intra-class distance and the inter-class distance about edge representation features.

$$\begin{aligned} \mathcal{L}_{Class} =& \omega_3\Sigma_{k=1}^{K}\Sigma_{x_s,x_t\in c_k}\frac{1}{|c_k||c_k|}||\mathbf{y}_s - \mathbf{y}_t||_2^2 - \\ & \omega_4\Sigma_{k=1}^{K}\Sigma_{m=1}^{K}\Sigma_{x_s\in c_k, x_t\in c_m, k\neq m}\frac{1}{|c_k||c_m|}||\mathbf{y}_s - \mathbf{y}_t||_2^2 \end{aligned} \tag{11}$$

where ω_1 and ω_2 are two user-defined weighting parameters to control the term importance.

We control the activation of the representation of each edge in each layer, in case of overdense or too sparse final representations.

$$\begin{aligned} \mathcal{L}_{Dens} =& \omega_5\Sigma_{k=1}^{K}\Sigma_{l=2}^{L_k}\mathbf{KL}(\pi_k|\hat{\pi}_{kl}) + \\ & \omega_6\Sigma_{k=1}^{K}\Sigma_{x_s=(v_i,v_j)\in c_k}\Sigma_{l=2}^{L_k}\mathbf{KL}(\rho_s|\hat{\rho}_{sl}) \end{aligned} \tag{12}$$

where $\hat{\rho}_{sl}$ denotes the average activation of hidden neurons in layer l on the neural network to train edge x_s in class c_k, i.e., the average of the components of the activation vector $\mathbf{y}_s^{(l)}$.

TABLE I
EXPERIMENT DATASETS

Graph	#Vertices	#Edges	Density
BLOGCATALOG	10,312	667,966	1.3×10^{-2}
FLICKR	80,513	11,799,764	3.6×10^{-3}
YOUTUBE	1,138,499	5,980,886	9.2×10^{-6}

$\hat{\pi}_{kl}$ specifies the average of the activation vectors $\mathbf{y}_s^{(l)}$ for all edges in c_k, i.e., $\hat{\pi}_{kl} = 1/|c_k|\Sigma_{x_s \in c_k}\hat{\rho}_{sl}$.

When we try to minimize $\mathcal{L}_{Dens}$, the KL divergence tends to achieve the minimum of 0 at $\hat{\pi}_{kl} = \pi_k$ and $\hat{\rho}_{sl} = \rho_s$. As shown in Eq.(7), we force a dense/sparse subgraph $\mathbf{X}_k$ with a small/large π_k. As a result, few/many hidden neurons in layer l will be activated. Thus, it is highly possible that the representations of $\mathbf{X}_k$ are sparse/dense. As for edges $x_s = (v_i, v_j)$ in c_k, we perform sparse/dense representation learning on edges associated with high/low-degree vertices with a small/large ρ_s to learn their sparse/dense representations.

Therefore, the overall loss function of the density-adaptive local edge representation learning is given as follow.

$$\mathcal{L}_{Total} = \mathcal{L}_{Repr} + \mathcal{L}_{Prox} + \mathcal{L}_{Class} + \mathcal{L}_{Dens} \tag{13}$$

VI. EXPERIMENTAL EVALUATION

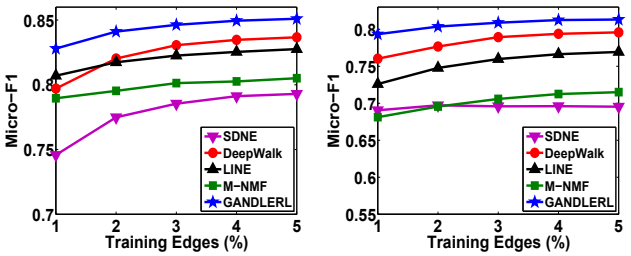
We have performed extensive experiments on three real sparse graph datasets, as shown in Table I.

We compare GANDLERL with four recently developed representative algorithms, DeepWalk [1], LINE [2], SDNE [4], and M-NMF [5], which all perform global representation learning on the whole graphs and learn low-dimensional vertex representation. We use the same input of the transition matrix of the edge-centric graph for all five methods.

We use multi-label edge classification and edge clustering to evaluate the quality of representation learning results by five algorithms. We utilize the LIBLINEAR package [40] to train the multi-label classifiers and use Macro-F1 and Micro-F1 [2] to evaluate the classification quality. Also, we choose K-Means [41] to partition edges into clusters and adopt Dunn index [42] to evaluate the clustering quality.

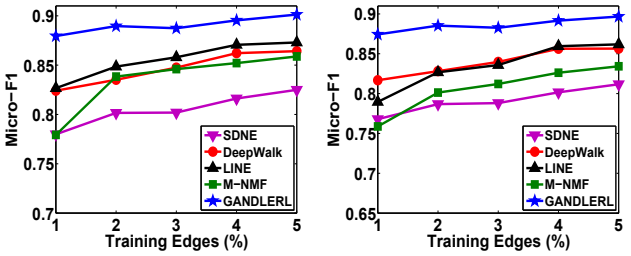
A. Multi-label Edge Classification Quality

Figures 2 and 3 exhibit the quality of multi-label classification on learnt representation results by five network representation learning algorithms on FLICKR and YOUTUBE with different proportions of training data. The low-dimensional representations of edges by network representation learning methods are used as features to classify each edge into a set of K classes. It is observed from Figure 2 that among all five network representation learning models, the classification algorithm on the learnt representation results by GANDLERL achieves the best quality in all experiments. Especially, as shown in Figure 3, GANDLERL significantly outperforms all other models on YOUTUBE, which is very sparse, showing the power of GANDLERL in the presence of data sparsity and imbalanced graph distribution. Compared to all other representation learning results, GANDLERL averagely achieves 4.7% Micro-F1 increase and 9.9% Micro-F1 boost on FLICKR and 6.6% Micro-F1 growth and 8.4% Macro-F1 increase on YOUTUBE respectively. Note that even if



(a) Micro-F1 (b) Macro-F1

Fig. 2. Classification on FLICKR



(a) Micro-F1 (b) Macro-F1

Fig. 3. Classification on YOUTUBE

the proportion of training data is very small, such as 1% and 2%, GANDLERL still can achieve considerable accuracy improvement. It demonstrates that GANDLERL is robust to rarely labeled data. This advantage is very important for graph classification since the labeled data is usually scarce.

B. Edge Clustering Quality

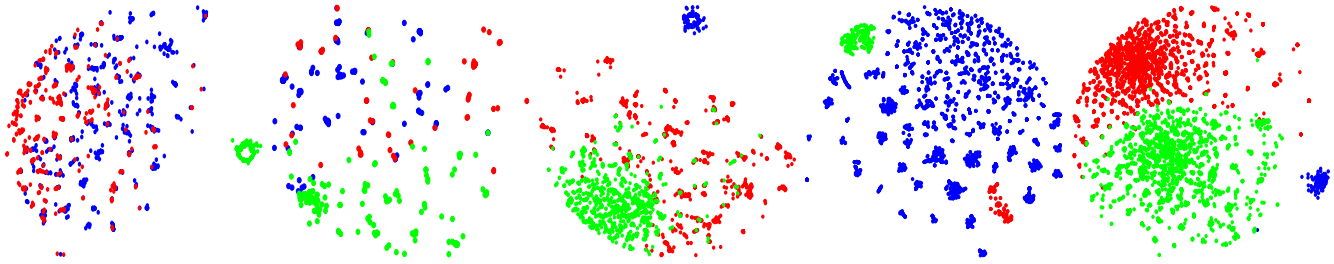
Figure 4 presents the quality of edge clustering on learnt representation results by five network representation learning algorithms on BLOGCATALOG and YOUTUBE. As the performance of K-means is very sensitive to initial centroids, we repeat the clustering ten times, each with different K initial centroids, and reported the average Dunn values. Similar trends are observed for the clustering quality comparison: GANDLERL achieves the largest Dunn values (>0.61), which are obviously better than other four methods. This demonstrates the integration of GAN-based multi-label classification, local representation learning, and density-adapt representation learning is able to make low-dimensional representations robust to graphs with different scales and densities.

C. Graph Visualization Performance

We map the low-dimensional vectors of edges from YOUTUBE to a 2D space with the t-SNE tool [43], as shown in Figure 5. We plot each edge as a colored point and each color denotes one of three randomly selected categories. We have observed that GANDLERL achieves the best visualization performance in terms of community separation and boundary.

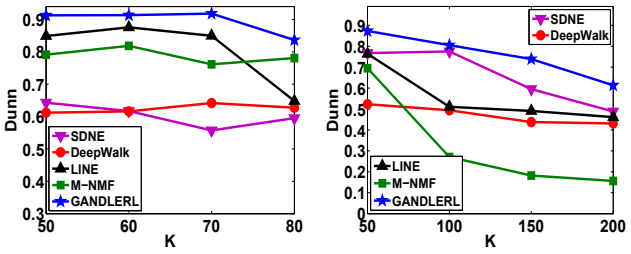
VII. CONCLUSIONS

We have presented our novel GANDLERL model. First, we design a generative adversarial network based multi-label edge classification model to classify rarely labeled edges in graphs with noise data into K classes. Second, a local autoencoder



(a) SDNE (b) DeepWalk (c) LINE (d) M-NMF (e) GANDLERL

Fig. 5. Visualization on YOUTUBE



(a) Dunn on BLOGCATALOG (b) Dunn on YOUTUBE

Fig. 4. Clustering

edge representation learning method is developed to perform local representation learning on each of K subgraphs and jointly optimize the loss functions within and across classes. Finally, we propose a density-adaptive learning method with the optimization at both edge and subgraph levels.

REFERENCES

[1] B. Perozzi, R. Al-Rfou', and S. Skiena, "Deepwalk: online learning of social representations," in *KDD*, 2014, pp. 701–710.

[2] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*, 2015, pp. 1067–1077.

[3] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *KDD*, 2016, pp. 855–864.

[4] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *KDD*, 2016, pp. 1225–1234.

[5] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *AAAI*, 2017, pp. 203–209.

[6] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *WSDM*, 2017, pp. 731–739.

[7] C. Tu, Z. Zhang, Z. Liu, and M. Sun, "Transnet: Translation-based network representation learning for social relation extraction," in *IJCAI*, 2017, pp. 2864–2870.

[8] G. Ma, L. He, C.-T. Lu, W. Shao, P. S. Yu, A. D. Leow, and A. B. Ragin, "Multi-view clustering with graph embedding for connectome analysis," in *CIKM*, 2017, pp. 127–136.

[9] S. Wang, C. C. Aggarwal, J. Tang, and H. Liu, "Attributed signed network embedding," in *CIKM*, 2017, pp. 137–146.

[10] Y. Zhang, Y. Xiong, X. Kong, and Y. Zhu, "Learning node embeddings in interaction graphs," in *CIKM*, 2017, pp. 397–406.

[11] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *CIKM*, 2017, pp. 1767–1776.

[12] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, X. Xie, and M. Guo, "GraphGAN: Graph Representation Learning with Generative Adversarial Nets," in *AAAI*, 2018.

[13] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec," in *WSDM*, 2018, pp. 459–467.

[14] D. Liben-Nowell and J. M. Kleinberg, "The link-prediction problem for social networks," *JASIST*, vol. 58, no. 7, pp. 1019–1031, 2007.

[15] S. Abu-El-Haija, B. Perozzi, and R. Al-Rfou, "Learning edge representations via low-rank asymmetric projections," in *CIKM*, 2017.

[16] L. Tang and H. Liu, "Scalable learning of collective behavior based on sparse social dimensions," in *CIKM*, 2009, pp. 1107–1116.

[17] X. Wang and G. Sukthankar, "Multi-label relational neighbor classification using social context features," in *KDD*, 2013.

[18] Y. Zhou, L. Liu, and D. Buttler, "Integrating vertex-centric clustering with edge-centric clustering for meta path graph analysis," in *KDD*, 2015.

[19] Y. Zhou and L. Liu, "Activity-edge centric multi-label classification for mining heterogeneous information networks," in *KDD*, 2014.

[20] L. Liu, J. Wang, J. Liu, and J. Zhang, "Privacy preservation in social networks with sensitive edge weights," in *SDM*, 2009, pp. 954–965.

[21] Y. Webscope, "Yahoo! altavista web page hyperlink connectivity graph, circa 2002," http://webscope.sandbox.yahoo.com/.

[22] P. Boldi, M. Santini, and S. Vigna, "A large time-aware web graph," *ACM SIGIR Forum*, vol. 42, pp. 33–38, 2008.

[23] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *WWW*, 2010, pp. 591–600.

[24] M. Gjoka, M. Kurant, C. T. Butts, and A. Markopoulou, "Walking in facebook: A case study of unbiased sampling of osns," in *INFOCOM*, San Diego, CA, March 2010, pp. 2498–2506.

[25] http://www.informatik.uni-trier.de/∼ley/db/.

[26] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," in *PVLDB*, 2009.

[27] Y. Sun, Y. Yu, and J. Han, "Ranking-based clustering of heterogeneous information networks with star network schema," in *KDD*, 2009.

[28] Y. Zhou, H. Cheng, and J. X. Yu, "Clustering large attributed graphs: An efficient incremental approach," in *ICDM*, 2010.

[29] Y. Zhou and L. Liu, "Clustering analysis in large graphs with rich attributes," in *Data Mining: Foundations and Intelligent Paradigms: Volume 1: Clustering, Association and Classification*, D. E. Holmes and L. C. Jain, Eds. Springer, 2011.

[30] Y. Sun, C. C. Aggarwal, and J. Han, "Relation strength-aware clustering of heterogeneous information networks with incomplete attributes," *PVLDB*, 2012.

[31] Q. Gu, C. Aggarwal, J. Liu, and J. Han, "Selective sampling on graphs for classification," in *KDD*, 2013, pp. 131–139.

[32] Y. Zhou and L. Liu, "Social influence based clustering of heterogeneous information networks," in *KDD*, 2013.

[33] Y. Zhou and L. Liu, "Social influence based clustering and optimization over heterogeneous information networks," *TKDD*, 2015.

[34] P. Yang, P. Zhao, V. W. Zheng, and X. Li, "An aggressive graph-based selective sampling algorithm for classification," in *ICDM*, 2015.

[35] C. C. Aggarwal and N. Li, "On node classification in dynamic content-based networks," in *SDM*, Mesa, AZ, April 28-30 2011, pp. 355–366.

[36] M. Ji, J. Han, and M. Danilevsky, "Ranking-based classification of heterogeneous information networks," in *KDD*, 2011, pp. 1298–1306.

[37] X. Kong, P. S. Yu, Y. Ding, and D. J. Wild, "Meta path-based collective classification in heterogeneous information networks," in *CIKM*, 2012.

[38] T. Guo, X. Zhu, J. Pei, and C. Zhang, "Snoc: Streaming network node classification," in *ICDM*, Shenzhen, China, Dec. 2014, pp. 150–159.

[39] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, Montreal, Canada, Dec. 2014, pp. 2231–2239.

[40] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of Machine Learning Research (JMLR)*, 9:1871–1874, 2008.

[41] S. P. Lloyd, "Least squares quantization in pcm," *TIT*, 28:128–137, 1982.

[42] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu, "Understanding of internal clustering validation measures," in *ICDM*, 2010, pp. 911–916.

[43] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *JMLR*, 2008.