Deep Unsupervised Active Learning on Learnable Graphs
Handong Ma, Changsheng Li, Xinchu Shi, Ye Yuan, and Guoren Wang

Abstract—Recently, deep learning has been successfully applied to unsupervised active learning. However, the current method attempts to learn a nonlinear transformation via an auto-encoder while ignoring the sample relation, leaving huge room to design more effective representation learning mechanisms for unsupervised active learning. In this brief, we propose a novel deep unsupervised active learning model via learnable graphs, named ALLGs. ALLG benefits from learning optimal graph structures to acquire better sample representation and select representative samples. To make the learned graph structure more stable and effective, we take into account k-nearest neighbor graph as a priori and learn a relation propagation graph structure. We also incorporate shortcut connections among different layers, which can alleviate the well-known over-smoothing problem to some extent. To the best of our knowledge, this is the first attempt to leverage graph structure learning for unsupervised active learning. Extensive experiments performed on six datasets demonstrate the efficacy of our method.

Index Terms—Active learning, graph structure, unsupervised learning.

I. INTRODUCTION

Active learning is an active research topic in machine learning and computer vision communities. Its goal is to choose informative or representative samples to be labeled, in order to reduce the costs of annotating but guarantee the performance of the model trained on these labeled samples. Due to its huge potentiality, active learning has been widely applied to various tasks, such as image classification [1], recommendation systems [2], object detection [3], semantic segmentation [4], and so on.

According to whether involving labeled data during sample selection, active learning can be roughly categorized into two groups: supervised and unsupervised ones. Supervised active learning methods usually pretrain one or several models with limited labeled data to iteratively select informative samples according to some criteria, such as uncertainty, diversity, and so on. Due to the limit of space, we recommend referring [5] for a more detailed introduction to deep supervised active learning. Unsupervised active learning targets selecting representative samples by taking advantage of the structure information of data. In this brief, we focus on unsupervised active learning, since it is more challenging due to the lack of label information.

Currently, most unsupervised approaches [6]–[10] intend to minimize the data reconstruction loss with different structure regularization terms for selecting representative samples. These methods assume that each data point can be represented by a linear combination of a selected sample subset, thus failing in modeling data with nonlinear structures. To remedy this issue, a kernel-based method [11] incorporates the manifold structure of data into the reproducing kernel Hilbert space (RKHS). More recently, deep learning has been applied to solve the unsupervised active learning problem [12], named DUAL. DUAL attempts to nonlinearly map data into a latent space and then performs sample selection in the learned space.

The key to success for DUAL stems from learning a nonlinear transformation to obtain new feature representations by leveraging deep learning. However, it does not explicitly take advantage of the relationships among samples during representation learning. Recently, graph neural networks (GNN) have attracted much attention [13]–[15], where the sample relation has been proven to be helpful for learning good sample representations. Thus, it ought to be beneficial to sample selection, if we can leverage graph structure information of data, and aggregate neighbor information of samples to learn better representations. But another question arises: how to construct an optimal graph structure for non-graph data still remains an open problem.

Based on the above-mentioned considerations, we propose a novel deep unsupervised active learning model based on learnable graphs, called ALLG. Specifically, ALLG first utilizes an auto-encoder framework to map samples into a latent space. Without predefining the graph structure of samples, ALLG devises a novel adjacent matrices learning module to automatically learn an optimal graph structure among samples for jointly refining sample representations and sample selection. In this module, we incorporate the k-nearest neighbor graph as a priori to learn a stable graph structure. Considering that the relation among samples may happen to evolve as the sample representations change in different network layers, we attempt to learn a series of relation propagated adjacent matrices, in the hope of capturing more precise graph structures. Moreover, we add shortcut connections among different adjacent matrices learning layers, which can alleviate the well-known over-smoothing problem to some extent.

Finally, a self-selection layer is employed to select representative samples based on the learned sample representation.

The contributions of this brief are summarized as follows.

1) ALLG builds a connection between unsupervised active learning and graph structure learning. To the best of our knowledge, this is the first attempt to leverage graph structure learning for unsupervised active learning.

2) ALLG attempts to learn precise sample representation by leveraging the graph and devises a novel mechanism to dynamically learn a series of graph adjacent matrices.

3) Inspired by the idea of residual learning, ALLG adds shortcut connections among different adjacent matrices learning layers to alleviate the over-smoothing problem.

Extensive experiments are performed on six publicly available datasets, and experimental results demonstrate the effectiveness of ALLG, compared with the state-of-the-art.

II. RELATED WORK

In this section, we will briefly review some work on unsupervised active learning and graph structure learning.
A. Unsupervised Active Learning

Unsupervised active learning has attracted much attention in recent years. At the earlier stage, [6] proposes to utilize transductive experimental design (TED) to select a sample subset and obtain a greedy solution. Active Learning via Neighborhood Reconstruction (ALNR) [16] performs sample selection by considering the neighborhood relation of samples. Robust Representation and Structured Sparsity (RRSS) [17] proposes a convex formulation by introducing a structured sparsity-inducing norm and a robust sparse representation loss. To select complimentary samples, [9] proposes a diverse loss function that is an extension of TED. Local structure reconstruction (LSR) [18] was proposed via local structure reconstruction to select representative data points. Active Learning with Feature Selection (ALFS) [10] builds a connection between unsupervised active learning and feature selection and proposes a convex formulation to select samples and features simultaneously. Active Learning via Subspace Learning (ALSL) [19] finds that the high-dimensional features of samples drawn from low-dimensional subspaces often contain noise. Therefore, ALSL tries to discover and leverage the low-rank structures of data to perform sample selection.

Most recently, owing to the powerful representation capability and great success of deep learning, a deep model has been explored to solve the unsupervised active learning problem [12], [20]. Li et al. [12] utilize deep auto-encoders to embed data to a latent space, and then select the most representative samples to best reconstruct both the whole dataset and clustering centroids. [20] devises structure-guided self-supervised strategies to further enhance the performance of unsupervised active learning. Inspired by the idea of matrix sketching, [21] is proposed to reconstruct the sketch of datasets, which can not only preserve the major information of samples but also reduce the model parameters. Although deep learning-based methods have achieved impressive results, they ignore the relationships among data points during sample representation learning and thus have inferior results to the methods which can leverage the sample relation.

B. Graph Structure Learning

For the sake of applying GNNs to nongraph structured data, many graph structure learning methods have been proposed in recent years. Dong et al. [22] and Egilmez et al. [23] explore to learn the graphs from data without associating it with the downstream tasks. More recently, [24]–[27] aim to dynamic construct graphs toward the downstream tasks. Tang et al. [28], [29] modeled the multiview or cross-view problem as graph learning processing to fuse the information from different views. However, these methods are task-specific ones, which depend on the supervised information. Cai et al. [30] presented a method to actively learn the graph embedding and reduce the cost of annotating each graph node.

Different from these models, we propose to optimize the learning of graphs and active learning simultaneously in an unsupervised manner. Specifically, [30] used active learning to reduce the annotating of graph nodes, and in this brief, we propose to take the advantage of graph structure learning for unsupervised active learning.

III. Method

The overall architecture of our method is illustrated in Fig. 1. Our network mainly consists of the following components: an encoder and decoder module is used to learn a nonlinear mapping. An adjacent matrix learning module aims to learn a relation propagation regularized graph structure for precise sample representation. A shortcut connection is introduced to alleviate the over-smoothing problem. A self-selection layer is used to select representative samples.

**A. Encoder and Decoder**

In order to learn a nonlinear transformation $\Theta$, we utilize an auto-encoder architecture to map the data into a latent space because $\Theta$ is a non-linear transformation by considering the sample relation to learn better feature representations of samples. To this end, we first learn a latent space with a deep auto-encoder. We then attempt to learn the graph structure of the data in the latent space and leverage it for learning a good representation. Based on the learned representation, we can select the most representative samples via a self-selection layer. We attempt to optimize them in a joint framework.
weights and bias associated with the $l$th hidden layer, respectively. \( \sigma(\cdot) \) is a nonlinear activation function. Then, we can define the latent representation as

\[
Z' = \Theta(X) = [\Theta(x_1), \Theta(x_2), \ldots, \Theta(x_n)] \in \mathbb{R}^{d' \times n}
\]

(2)

where $d'$ denotes the dimension of the latent representation.

As for the decoder, it learns another nonlinear mapping to reconstruct the original data, which guides the training of the encoder. The decoder has a symmetric structure with the encoder, which consists of $L$ fully connected layers as well. Then, the reconstruction loss of auto-encoder is defined as

\[
\mathcal{L}_r = \sum_{i=1}^{n} \|X_i - \bar{X}_i\|^2_F = \|X - \bar{X}\|^2_F
\]

(3)

where $\bar{X}$ denotes the reconstruction of $X$. Actually, the input $X$ plays a role of self-supervisor to guide the learning of auto-encoder.

B. Graph Structure Learning

After obtaining the latent representation $Z'$ of the input $X$, we intend to learn the graph structure of the input $X$ for generating a more precise sample representation.

1) Adjacent Matrix Learning: As aforementioned, taking advantage of the relationships among data points can have a positive effect on sample representation learning, which has been verified in GNNs [31]. However, many GNN algorithms are developed to deal with graph data and assume that the adjacent relationship is given, which cannot be directly applied to non-graph data. To deal with it, there are some algorithms, which attempt to construct a human-estimated graph structure, e.g., $k$-nearest neighbor graph. However, such methods cannot guarantee the graph structure is optimal. Based on these considerations, we aim to learn a data-driven optimal graph structure. Considering that some human-estimated graph structures can still reveal some prior information, we can integrate such information into our framework to regularize graph structure learning. In all, we propose the following regularization terms to learn the graph structure:

\[
\mathcal{L}_a = \alpha\|\mathbf{A}_1\|_F^2 + \beta\|\mathbf{A}_1 \cdot \mathbf{A}_0\|_F^2
\]

(4)

where $\mathbf{A}_0$ denotes the $k$-nearest neighbor graph, and $\mathbf{A}_1$ denotes the learned graph, i.e., the adjacent matrix, $\alpha$ and $\beta$ are positive tradeoff parameters.

In (4), the first regularization term aims to reduce the complexity of the learned adjacent matrix. The second term imposes a constraint on $\mathbf{A}_1$, making it not deviate from the $k$-nearest neighbor graph $\mathbf{A}_0$, such that the prior information can be integrated. One can control how close the learned adjacent matrix is to the prior by modifying the parameter $\beta$. Actually, the graph adjacent matrix $\mathbf{A}_1$ can be regarded as the parameters of a fully connected layer without bias. Therefore, it can be updated through a standard backpropagation procedure during training.

After obtaining the adjacent matrix $\mathbf{A}_1$, we can obtain the new sample representations $\mathbf{S}_i$ based on $Z'$ as

\[
\mathbf{S}_1 = \sigma(Z' \cdot \mathbf{A}_1)
\]

(5)

where $\sigma(\cdot)$ is a nonlinear activation function. Based on the aforementioned equation, we can see that the new sample representation $\mathbf{S}_1$ can be obtained based upon a linear combination of $Z'$ with the weight matrix $\mathbf{A}_1$, thereby the relation among samples can be incorporated into the process of representation learning.

2) Relation Propagation: It is worth noting that only learning one adjacent matrix may be suboptimal for learning the graph structure of data since the representations of samples are from different layers during the process of training. In order to learn a more stable and effective optimal graph structure, we intend to learn multiple adjacent matrices with a relation regularized term. In order to enable the learning of multiple adjacent matrices, we propose another regularization terms as

\[
\mathcal{L}_r = \sum_{i=2}^{N} \left( \alpha'\|\mathbf{A}_i\|_F^2 + \beta'\|\mathbf{A}_i - \mathbf{A}_{i-1}\|_F^2 \right)
\]

(6)

where $\mathbf{A}_i$ denotes the learned adjacent matrix of the $i$th matrix learning layer, and $\alpha'$ and $\beta'$ are positive tradeoff parameters.

In (6), the first term aims to lower the complexities of the learned adjacent matrices, while the second term utilizes the former learned adjacent matrix to regularize the latter adjacent matrix. By this means, the relation among samples can be smoothly propagated. After learning multiple adjacent matrices, we can obtain another sample representation for sample selection as

\[
\mathbf{S}_{i+1} = \sigma(\mathbf{S}_i \cdot \mathbf{A}_i), \quad i = 1, 2, \ldots, N - 1
\]

(7)

where $\mathbf{S}_i$ denotes the latent sample representations in the $i$th graph Laplacian layer.

3) Shortcut Connection: In GNNs, there is a well-known problem, i.e., over smoothing. The over-smoothing problem means that repeated graph Laplacian eventually makes node embeddings indistinguishable. Empirically, a shortcut connection would bring more discriminative features from the former layers to alleviate this problem. The shortcut connection is illustrated in Fig. 2. Mathematically, it can be described as

\[
\mathbf{S}^{out} = r\mathbf{S}_k + (1-r)\mathbf{S}_N
\]

(8)

where $r$ is a tradeoff parameter to control the contribution of the $k$th graph structure learning layer to the final sample representation.

C. Self-Selection Layer

After obtaining the final latent sample representations, we perform sample selection by introducing a self-selection layer, as shown in Fig. 1.

Self-selection layer takes the output of the adjacent matrix learning layers as the input. In order to select samples to best reconstruct all
ones, we use the loss function presented as
\[ \mathcal{L}_i = \sum_{i=1}^{n} \| S_{\text{out}}^i - S_{\text{out}}^i Q \|_2 + \lambda \| Q \|_{1, \infty} \] 
(9)

where \( Q \in \mathbb{R}^{d \times n} \) is the reconstruction coefficients for the samples. \( \| Q \|_{1, \infty} = \sum_{i=1}^{n} | q_i |_{\infty} \) where \( q_i \) is the \( i \)th row vector of \( Q \) and \( | \cdot |_{\infty} \) denotes the supnorm of a vector, defined as \( | a |_{\infty} = \max_{1 \leq j \leq d} | a_j | \).

The first term in (9) aims to pick out \( m \) samples to reconstruct the whole dataset in the latent space, while the second term is a regularization term to enforce the coefficient matrix \( Q \) row-sparse. To minimize (9), similarly, \( Q \) can be regarded as the parameters of a fully connected layer without bias and nonlinear activations and solved jointly through a standard backpropagation procedure.

After the inputs passing by the self-selection layers, we then feed them into the decoder as its inputs as
\[ \hat{x}_n = S_{\text{out}} Q \in \mathbb{R}^{d \times n}. \] 
(10)

D. Overall Model and Training

Based on (3), (4), (6), and (9), the final loss function is
\[ \mathcal{L}_{\text{total}} = \mathcal{L}_r + \mathcal{L}_s + \mathcal{L}_p + \mathcal{L}_q. \] 
(11)

To jointly optimize (11), we use a two-stage training strategy following [12]. First, we only pretrain the encoder–decoder without considering the matrices learning module and self-selection layer, minimizing the loss in (3). By this means, we obtain good initial parameters for fine-tuning the whole model. Specifically, three full connection layers are used in the encoder, and the decoder has a symmetric structure with it. After that, we update all the parameters by minimizing (11) through a standard backpropagation procedure. The ReLU is used in our method as the activations. And we optimize (11) with the Adam optimizer where we set the learning rate to \( 1.0 \times 10^{-3} \). In the final model, ALLG utilizes two adjacent matrices learning layers. As for the shortcut connection, we set the \( k = 1 \) and \( r = 0.3 \) in (8) during training. The hyperparameters \( \alpha \) and \( \alpha' \) are set to be the same so as the \( \beta \) and \( \beta' \) during training. The optimization procedure for ALLG can also be seen in Algorithm 1.

E. Postprocessing

Once the model converges, we can get the parameter \( Q \) based on the self-selection layer. As \( Q \) is row sparse after training, the row values of \( Q \) can be regarded as the contributions of each data point to reconstruct other data points. Thus, we calculate the \( \ell_2 \)-norm of the rows and sort them in descending order to get a rank of samples. Then we can select the top-\( m \) data points as the most representative samples to be labeled.

IV. EXPERIMENTS

In this section, we will introduce the experimental results to demonstrate the effectiveness of the proposed method. We also conduct some ablation studies and analyses on the proposed ALLG.

A. Experimental Setting

1) Dataset: To demonstrate that datasets from different domains can benefit from ALLG, we conduct experiments on six publicly available datasets from different domains, and the details of the datasets are summarized in Table I.1 In the datasets, “Splice-junction” and “Epileptic Seizure Recognition (ESR)” are from biology, while “ESR” is present as time–series. “Plant species leaves” and “letter recognition” are generated from images. “Waveform” is a physical dataset, while “Gas Sensor Array Drift Dataset (GSAD)” is from sensors utilized in simulations.

2) Baseline2: We compare our method with several typical unsupervised active learning algorithms, including RRSS [17], ALNR [16], Manifold Adaptive Experimental Design (MAED) [11], ALFS [10], and DUAL [12]. We also compare with a matrix column subset selection algorithm, deterministic column sampling (DCS) [32], which can be used for unsupervised active learning. In addition, we take K-means as another baseline, in which we choose the samples which are closest to the cluster centers as the most representative samples, and we set \( K = 5 \) in experiments.

3) Experimental Protocol: Following [10] and [12], we randomly select 50% of the samples as the candidate set, and the rest is the testing set. Different active learning algorithms are performed on the same candidate set to query the most representative \( m \) samples. To verify the quality of samples selected by these methods, we train two classifiers by using these selected samples as the training data: an SVM classifier with a linear kernel and \( C = 100 \), as well as a logistic regression (LR) classifier. We search the tradeoff parameters in our algorithm from \([0.1, 1, 10]\). For a fair comparison, the parameters of RRSS, ALNR, MAED, ALFS, and DUAL are all searched from the same space. Each experiment is run five times, and the result is reported in terms of the average accuracy.

B. Experimental Result

1) General Performance: Figs. 3 and 4 show the results of different methods combined with an SVM classifier and an LR classifier, respectively. The values of the \( x \)-axis in Figs. 3 and 4 denote the number of queries, also presented as the number of the selected samples \( m \).

We can observe that ALLG outperforms all other baselines in almost all queries. We use two different classifiers to illustrate that the quality of selected samples by ALLG is agnostic to classifiers. It is worth noting that ALLG achieves marked improvement compared with deep learning-based DUAL, with about 3% average improvement on different numbers of the query. It is verified that learning graphs based on 

1These datasets are all downloaded from the University of California, Irvine (UCI) machine learning repository: https://archive.ics.uci.edu/ml/datasets.php

2All source codes are obtained from the authors of the corresponding papers, except K-means and ALNR.
Fig. 3. Performance comparisons of different active learning methods combined with SVM on six benchmark. (a) Splice-junction. (b) Plant species leaves. (c) Waveform. (d) ESR. (e) GSAD. (f) Letter recognition.

Fig. 4. Performance comparisons of different active learning methods combined with LR on six benchmark. (a) Splice-junction. (b) Plant species leaves. (c) Waveform. (d) ESR. (e) GSAD. (f) Letter recognition.

Note that we do not perform ALFS and RRSS on larger datasets, because of their unaffordable computational complexities.

2) Ablation Study: We perform an ablation study on the Splice-junction dataset to gain a further understanding of the proposed method. The experimental setting is as follows.

1) Training Classifiers Using the Original Features: For ALLG and DUAL, they embed samples to a latent space and use the new representation to train classifiers. To eliminate the influence of new representations, we use original features to train classifiers after obtaining the selected samples by ALLG and DUAL. We denote them as ALLG_O and DUAL_O,
respectively. The results are shown in Table II. In general, ALLG_O and DUAL_O still achieve better performance than other methods. Meanwhile, ALLG_O has a better result than DUAL_O, which verifies the effectiveness of our method once again. In addition, ALLG is better than ALLG_O, and DUAL achieves better performance than DUAL_O. This illustrates that learning a nonlinear representation can be helpful for active learning.

2) Adjacent Matrices Learning Module: We also verify the effectiveness of the graph structure learning module, i.e., adjacent matrix learning. We denote the following. 1) w/o Smoothing: ALLG trained without graph structure learning. 2) ALLG_knn: ALLG using $k$-nearest neighbor graph as the only adjacent matrix. 3) ALLG_one: ALLG trained with one learned adjacent matrix. 4) ALLG_ts: ALLG trained with two learned adjacent matrices but forcing them to be the same. 5) ALLG_td: ALLG trained with two different learned adjacent matrices (i.e., it will become ALLG when shortcut connections are added). The results are reported in Table III. We find that ALLG_knn achieves better results than w/o smoothing, illustrating that leveraging the graph structure of data is good for learning a better sample representation. ALLG_one is better than ALLG_knn, demonstrating that learning a graph structure can achieve superior results, compared with a human estimated one. ALLG_td outperforms ALLG_one and ALLG_ts. This illustrates that learning multiple different adjacent matrices can be beneficial to representation learning. Finally, ALLG beats ALLG_td, showing that shortcut connection is effective for unsupervised active learning.

3) Parameter Study: We study the sensitivity of our algorithm in terms of the tradeoff parameters $\lambda$, $\alpha$, and $\beta$ on the Splice-junction dataset. We fix the number of queries to 125. The results are shown in Fig. 5. Our method is insensitive to the parameters with a relatively wide range. And we can observe that when these three tradeoff parameters are set to 1, our method can reach the best results.

4) Convergence Analysis: We further show the convergence curves of ALLG on the Splice-junction dataset. The results are shown in Fig. 6. As the number of epochs increases, the total loss and the supnorm loss of $Q$ are gradually decreased until convergent. From Fig. 6, the algorithm converges when the epoch reaches around 2000.
Fig. 6. Convergence analysis of ALLG. (a) Total loss. (b) Supnorm loss of Q.

Fig. 7. Visualization by t-SNE on Splice-junction dataset. The red circles denote the selected samples and other color solid circles denote different classes.

5) Visualization: We use t-SNE [33] to visualize sample selection on the Splice-junction dataset shown in Fig. 7, the selected samples are distributed uniformly and can better represent the whole dataset.

V. CONCLUSION

In this brief, we proposed a novel unsupervised active learning model by learning optimal graph structures of data. A novel adjacent matrices learning module was devised to learn a series of optimal graph structures of data through relation propagation regularization without predefining the graph structures. In the meantime, we took the k-nearest neighbor graph prior to assist in graph structure learning. Finally, we utilized a shortcut connection to alleviate the over-smoothing problem. Experimental results demonstrated the effectiveness of our method.

REFERENCES

[1] K. Wang, D. Zhang, Y. Li, R. Zhang, and L. Lin, “Cost-effective active learning for deep image classification,” IEEE Trans. Circuits Syst. Video Technol., vol. 27, no. 12, pp. 2591–2600, Jun. 2016.

[2] M. Elahi, F. Ricci, and N. Rubens, “A survey of active learning in collaborative filtering recommender systems,” Comput. Sci. Rev., vol. 20, pp. 29–50, May 2016.

[3] H. H. Aghdam, A. Gonzalez-Garcia, A. Lopez, and J. Weijer, “Active learning for deep learning neural networks,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 3672–3680.

[4] Y. Siddiqui, J. Valentín, and M. Niessner, “ViewAL: Active learning with viewpoint entropy for semantic segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 9433–9443.

[5] P. Ren et al., “A survey of deep active learning,” ACM Comput. Surv., vol. 54, no. 9, pp. 1–40, 2021.

[6] K. Yu, J. Bi, and V. Tresp, “Active learning via transductive experimental design,” in Proc. 23rd Int. Conf. Mach. Learn. (ICML), 2006, pp. 1081–1088.

[7] L. Zhang, C. Chen, J. Bu, D. Cai, X. He, and T. S. Huang, “Active learning based on locally linear reconstruction,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 10, pp. 2026–2038, Oct. 2011.

[8] F. Zhu, B. Fan, X. Zhu, Y. Wang, S. Xiang, and C. Pan, “10,000+ times accelerated robust subset selection,” in Proc. AAAI Conf. Artif. Intell., 2015, pp. 3217–3224, vol. 29, no. 1, pp. 3217–3224.

[9] L. Shi and Y.-D. Shen, “Diversifying convex transductive experimental design for active learning,” in Proc. IJCAI, 2018.

[10] C. Li, H. Ma, Z. Kang, Y. Yuan, X.-Y. Zhang, and G. Wang, “On deep unsupervised active learning,” in Proc. 29th Int. Joint Conf. Artif. Intell., Jul. 2020, pp. 2626–2632.

[11] D. Cai and X. He, “Manifold adaptive experimental design for text categorization,” IEEE Trans. Knowl. Data Eng., vol. 24, no. 4, pp. 707–719, Apr. 2011.

[12] C. Li, H. Ma, Z. Kang, Y. Yuan, X.-Y. Zhang, and G. Wang, “On deep unsupervised active learning,” in Proc. 29th Int. Joint Conf. Artif. Intell., Jul. 2020, pp. 2626–2632.

[13] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” 2016, arXiv:1609.02907.

[14] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, “Graph attention networks,” 2017, arXiv:1710.10903.

[15] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, “How powerful are graph neural networks?” 2018, arXiv:1810.00826.

[16] Y. Hu, D. Zhang, Z. Jin, D. Cai, and X. He, “Active learning via neighborhood reconstruction,” in Proc. IJCAI, 2013, pp. 1415–1421.

[17] F. Nie, W. Hua, H. Huang, and C. Ding, “Early active learning via robust representation and structured sparsity,” in Proc. IJCAI, 2013, pp. 1572–1578.

[18] Q. Li, X. Shi, L. Zhou, Z. Bao, and Z. Guo, “Active learning via local structure reconstruction,” Pattern Recognit. Lett., vol. 92, pp. 81–88, Jun. 2017.

[19] C. Li et al., “Unsupervised active learning via subspace learning,” in Proc. AAAI, 2021, vol. 35, no. 9, pp. 8332–8339.

[20] C. Li, H. Ma, Y. Yuan, G. Wang, and D. Xu, “Structure guided deep neural network for unsupervised active learning,” IEEE Trans. Image Process., vol. 31, pp. 2767–2781, 2022.

[21] C. Li, R. Li, Y. Yuan, G. Wang, and D. Xu, “Deep unsupervised active learning via matrix sketching,” IEEE Trans. Image Process., vol. 30, pp. 9280–9293, 2021.

[22] X. Dong, D. Thanou, P. Frossard, and P. Vanderheynst, “Learning Laplacian matrix in smooth graph signal representations,” IEEE Trans. Signal Process., 2016, vol. 64, no. 23, pp. 6160–6173, Dec. 2016.

[23] H. E. Egilmez, E. Pavez, and A. Ortega, “Graph learning from data under Laplacian and structural constraints,” IEEE J. Sel. Topics Signal Process., vol. 11, no. 6, pp. 825–841, Sep. 2017.

[24] R. Li, S. Wang, F. Zhu, and J. Huang, “Adaptive graph convolutional neural networks,” 2018, arXiv:1801.03226.

[25] E. Chei et al., “Learning the graphical structure of electronic health records with graph convolutional transformer,” 2019, arXiv:1906.04716.

[26] P. Liu, S. Chang, X. Huang, J. Tang, and J. C. K. Cheung, “Contextualized non-local neural networks for sequence learning,” in Proc. AAAI Conf. Artif. Intell., vol. 33, 2019, pp. 6762–6769.

[27] Y. Chen, L. Wu, and M. J. Zaki, “Deep iterative and adaptive learning for graph neural networks,” 2019, arXiv:1912.07832.

[28] C. Tang et al., “Learning a joint affinity graph for multiview unsupervised clustering,” IEEE Trans. Multimedia, vol. 21, no. 7, pp. 1724–1736, Jul. 2018.

[29] C. Tang et al., “Cross-view locality preserved diversity and consensus learning for multi-view unsupervised feature selection,” IEEE Trans. Knowl. Data Eng., early access, Jan. 1, 2021, doi: 10.1109/TKDE.2020.3048678.

[30] H. Cai, V. W. Zheng, and K. C.-C. Chang, “Active learning for graph embedding,” 2017, arXiv:1705.05085.

[31] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A comprehensive survey on graph neural networks,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 1, pp. 4–24, Jan. 2020.

[32] D. Papailiopoulos, A. Kyrillidis, and C. Boutsidis, “Provable deterministic leverage score sampling,” in Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Aug. 2014, pp. 997–1006.

[33] L. van der Maaten and G. Hinton, “Visualizing data using t-SNE,” J. Mach. Learn. Res., vol. 9, pp. 2579–2605, Nov. 2008.