Escaping Feature Twist: A Variational Graph Auto-Encoder for Node Clustering

Nairouz Mrabah[1*], **Mohamed Bouguessa**[1] and **Riadh Ksantini**[2]
[1]University of Quebec at Montreal, Montreal, QC, Canada
[2]University of Bahrain, Kingdom of Bahrain
mrabah.nairouz@courrier.uqam.ca, bouguessa.mohamed@uqam.ca, rksantini@uob.edu.bh

Abstract

Most recent graph clustering methods rely on pretraining graph auto-encoders using self-supervision techniques (pretext task) and fine-tuning based on pseudo-supervision (main task). However, the transition from self-supervision to pseudo-supervision has never been studied from a geometric perspective. Herein, we establish the first systematic exploration of the latent manifolds' geometry under the deep clustering paradigm; we study the evolution of their intrinsic dimension and linear intrinsic dimension. We find that the embedded manifolds undergo coarse geometric transformation under the transition regime: from curved low-dimensional to flattened higher-dimensional. Moreover, we find that this inappropriate flattening leads to clustering deterioration by twisting the curved structures. To address this problem, which we call Feature Twist, we propose a variational graph auto-encoder that can smooth the local curves before gradually flattening the global structures. Our results show a notable improvement over multiple state-of-the-art approaches by escaping Feature Twist.

1 Introduction

The past few years have witnessed the emergence of deep clustering as a promising paradigm for clustering large-scale, high-dimensional, and high-semantic datasets. In essence, it amounts to performing joint clustering and feature learning using neural networks. Recent improvements in deep clustering can be mainly attributed to the seminal advances in self-supervision [Liu *et al.*, 2021] and pseudo-supervision [Mrabah *et al.*, 2020]. On the one hand, self-supervision involves solving a pretext task, which requires a high-level understanding of the data semantics. For instance, random walk [Perozzi *et al.*, 2014], adjacency reconstruction [Kipf and Welling, 2016], and mutual information between local and global representations [Veličković *et al.*, 2019] are among the most relevant self-supervision tasks for graph representation learning. On the other hand, pseudo-supervision aims at solving the primary task by jointly learning pseudo-labels using a clustering algorithm, and supplying these pseudo-labels as a supervisory signal to learn clustering-oriented features.

Most existing deep clustering models follow a two-step strategy to make up for the absence of supervisory signals. They pretrain using a self-supervision loss, namely, the pretraining phase, and fine-tune using pseudo-supervision or a linear combination between pseudo-supervision and self-supervision, namely, the clustering phase. In a recent work, Mrabah et al. [2021] studied the relationship between self-supervision and pseudo-supervision in the clustering phase. They have shown that this relation is governed by the trade-off between Feature Randomness and Feature Drift. Furthermore, they have proved that the graph convolutional operation strengthens the effect of Feature Drift. Nevertheless, the transition from pretraining to clustering has never been studied. Particularly, it is important to understand this transition from a geometric perspective to identify potential deterioration (e.g., twisted manifolds) to the learned structures.

Among the principal ways to assess the geometric configuration of the latent representations is to compute their *Intrinsic Dimension* (ID) and *Linear Intrinsic Dimension* (LID). Practically, the ID metric measures the dimension of the embedded manifolds, and LID measures the dimension of the best subspace (minimal rank) that can enclose the latent manifolds. However, curved manifolds with non-uniformly distributed points render the ID estimation a challenging task. To solve this problem, Facco et al. [2017] have proposed TwoNN, an efficient estimator that requires two nearest neighbours of each point to assess ID. In a recent work, Ansuini et al. [2019] capitalized on this estimator and proposed an estimator for LID based on PCA (Principal Component Analysis) to show that training a neural network supervisedly gives birth to low-dimensional and curved latent manifolds.

We investigate the evolution of ID and LID for graph auto-encoders under the deep clustering paradigm. The results of this investigation are available in Appendix A (all appendices are provided in the Supplementary Material[†]). We find that the pretraining phase gives birth to low-dimensional and curved latent manifolds. However, introducing pseudo-supervision in the clustering phase coarsely flattens and raises the ID of the latent structures. Additionally, we find that the

[*]Contact Author

[†]https://github.com/nairouz/FT-VGAE

coarse flattening of the latent manifolds under the transition regime twists the curved structures, and hence degrades the clustering results. We call this problem Feature Twist (FT).

We propose a variational graph auto-encoder model trained in three phases to avoid the coarse transition from pretraining to clustering. We start by minimizing the vanilla reconstruction cost in the first phase, similar to previous variational auto-encoder methods [Kipf and Welling, 2016], [Hui *et al.*, 2020]. The second training phase flattens the local curvatures while emphasizing the globally curved shape. Finally, the third training phase gradually flattens the global structures while preserving local configurations. We derive the objective functions of the three phases in a principled way following the variational framework. Thus, each optimized objective constitutes a lower bound of the input graph log-likelihood. Moreover, our edge decoding strategies exploit the neighbourhood-level and the cluster-level information.

Contributions. **(i)** We establish a systematic exploration of the deep clustering paradigm from a geometric perspective. We focus on the transition regime from self-supervision to pseudo-supervision, and we shed the light on Feature Twist, a problem that has never been tackled by existent deep clustering methods. **(ii)** We introduce a variational auto-encoder trained in three phases to circumvent the coarse geometric transition from pretraining to clustering. More precisely, we flatten the local curvatures of the latent manifolds before gradually flattening the global structures. Our decoding strategy exploits the neighbourhood-level and the cluster-level information. **(iii)** We conduct various experiments to validate the contributions of the proposed approach. The obtained results provide strong evidence that our model can improve the clustering performance over several state-of-the-art deep graph clustering methods by escaping Feature Twist.

2 Related Work

We discuss the broad category of deep graph clustering and the more specific category of variational graph auto-encoders.

2.1 Deep Graph Clustering

ARGAE [Pan *et al.*, 2018] (Adversarially Regularized Graph Auto-Encoder) harnesses adversarial training to impose a Gaussian distribution on the latent representations using a discriminator. After that, the clustering assignments are computed by applying k-means on the embedded codes. Since the latent manifolds are curved at the end of the training process, we do not expect k-means to identify the clustering structures effectively. DAEGC [Wang *et al.*, 2019] (Deep Attentional Embedded Graph Clustering) leverages an attention graph encoding mechanism. Unlike ARGAE, DAEGC is trained to learn clustering-oriented features by minimizing joint clustering and reconstruction after the pretraining phase. However, DAEGC falls short of any consideration to the coarse geometric transformation associated with the transition from pretraining to clustering. AGE [Cui *et al.*, 2020] (Adaptive Graph Encoder) smooths the input feature matrix with a Laplacian filter computed based on the adjacency matrix. Then, the filtered signal feeds a fully connected neural network, which is trained to construct an adaptive and clustering-oriented graph. However, performing pseudo-supervision directly without self-supervision increases the amount of Feature Randomness [Mrabah *et al.*, 2022].

2.2 Variational Graph Auto-Encoders

The previously discussed methods assume a deterministic encoding process to build the node embeddings. Nevertheless, modeling the uncertainty inherent to real-world graphs in the latent space is crucial for solving the clustering task. For example, it is essential to assess the confidence associated with critical predictions on medical graphs [Yang *et al.*, 2021]. The variational graph auto-encoding constitutes a prominent strategy with solid probabilistic foundations to address this problem. It models the uncertainty with random variables from predefined distributions. In another advantageous aspect, this strategy is among the mainstream solutions for generative modeling, such as molecule generation [Liu *et al.*, 2018]. Furthermore, variational graph auto-encoders have theoretically-grounded objective functions by maximizing lower bounds of the input graph log-likelihood. These lower bounds are derived in a principled way, which avoids tuning unrequired hyperparameters for balancing the different components of the objective function.

VGAE [Kipf and Welling, 2016] is the first variational graph auto-encoder model. Although it has found success in many applications, this method has several limitations. First, VGAE does not consider the clustering meta-prior [Bengio *et al.*, 2013] for building the inference and generative models. Consequently, the features learned by this method are not clustering-oriented (no clustering objective derived in the evidence lower bound). Several studies exploit VGAE in solving the clustering task by applying k-means on the embedded codes. However, this strategy is problematic because the latent manifolds are curved at the end of the training process, and there is no systematic way to identify the most relevant metric to capture the latent similarities for any dataset. Therefore, it is critical to flatten the embedded manifolds before applying a Euclidean-based clustering algorithm, such as k-means. Second, VGAE adopts a simplistic decoding style based on the inner-product of the node embedding. This assumption implies that the generated edges do not integrate information from the neighbourhood-level and the cluster-level.

GMM-VGAE [Hui *et al.*, 2020] is a variational graph auto-encoder that can model a multi-category latent structure with Gaussian Mixture Models. Unlike VGAE, GMM-VGAE establishes the clustering meta-prior explicitly by introducing random variables representing the different clusters. Hence, a clustering objective emerges in the lower bound derivation, which in turn promotes the learning of clustering-oriented features. However, the latent codes of this model lie on curved manifolds at the end of the pretraining phase. Thus, performing joint clustering and feature learning based on GMM flattens the embedded manifolds inappropriately: the coarse geometric transformation of the clustering manifolds as shown by our experiments (see Section 4.2). Similar to VGAE, GMM-VGAE has a naive generative model that overlooks the neighbourhood-level and the cluster-level information. This aspect restricts the decoding flexibility.

3 Proposed Method

To address FT, we propose a variational graph auto-encoder (VGAE) trained in three phases. Our proposed model, FT-VGAE (VGAE supplied with a mechanism against FT), is designed to perform embedded clustering without twisting the curved structures while flattening the latent manifolds. We start this section by introducing important notations.

We consider a non-directed attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, where $\mathcal{V} = \{v_1, v_2, ..., v_N\}$ is a set of N nodes, $e_{ij} \in \mathcal{E}$ indicates that there is an edge between the i^{th} and j^{th} nodes, and $X \in \mathbb{R}^{N \times J}$ is the attribute matrix. Each row of this matrix $x_i \in \mathbb{R}^J$ denotes the vector associated with the i^{th} node, and J is the dimension of the input space. We consider that the set $\mathcal{V}$ can be grouped into N_c clusters. Let $A = (a_{ij}) \in \mathbb{R}^{N \times N}$ be the adjacency matrix of $\mathcal{G}$ and D the degree matrix of A. Thus, $a_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$ and $a_{ij} = 0$ otherwise. We adopt GCN [Kipf and Welling, 2017] layers for mapping the input graph to low-dimensional representations $Z \in \mathbb{R}^{N \times d}$ according to the layer-wise mechanism:

$$Z^{(l)} = f_\phi(Z^{(l-1)}, A|W^{(l)}) = \phi(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}Z^{(l-1)}W^{(l)}), \quad (1)$$

where $\widetilde{A} = A + I$, $\widetilde{D} = D + I$, and $I \in \mathbb{R}^{N \times N}$ is the identity matrix. $Z^{(l)}$ represents the output matrix of the l^{th} layer and $Z^{(0)} = X$. ϕ denotes the activation function and $W = \{W^{(l)}\}$ is the set of trainable weights. The architecture of our model consists of two encoding heads specified as:

$$Z^{(1)} = f_{\text{ReLU}}(X, A|W^{(0)}), \quad (2)$$

$$Z^{(2)}_\mu = f_{\text{Linear}}(Z^{(1)}, A|W^{(1)}_\mu) \in \mathbb{R}^{N \times d}, \quad (3)$$

$$Z^{(2)}_\sigma = f_{\text{Linear}}(Z^{(1)}, A|W^{(1)}_\sigma) \in \mathbb{R}^{N \times d}. \quad (4)$$

Following the variational auto-encoder framework, our approach maximizes an evidence lower bound of the input graph log-likelihood for each training phase. These bounds depend on the designed inference and generative models. We explain the different phases in the following subsections.

3.1 First Phase

For this phase, we develop a slight variant of the variational model proposed by Kipf and Welling in [2016]. The inference model is defined by the distribution $q(Z|X, A)$ parameterized by two encoding layers:

$$q(Z|X, A) = \prod_{i=1}^{N} q(z_i|X, A) = \prod_{i=1}^{N} \mathcal{N}(z_i|\mu_{z_i}, \text{diag}(\sigma^2_{z_i})), \quad (5)$$

where $\mu_{z_i} = Z^{(2)}_\mu[i, :]$ and $\sigma^2_{z_i} = Z^{(2)}_\sigma[i, :]$ are the mean vector and the variance vector, respectively, of a multivariate Gaussian distribution associated with z_i. The generative model of this phase is defined by the distribution $p(A, Z)$ factorized in a decoding style $p(A, Z) = p(A|Z)p(Z)$ such that:

$$p(Z) = \prod_{i=1}^{N} p(z_i) = \prod_{i=1}^{N} \mathcal{N}(z_i|0, I), \quad (6)$$

$$p(A|Z) = \prod_{i=1}^{N}\prod_{j=1}^{N} p(a_{ij}|z_i, z_j) = \prod_{i=1}^{N}\prod_{j=1}^{N} \mathcal{B}\text{er}(\beta_{ij}), \quad (7)$$

where $\beta_{ij} = \text{Sigmoid}(z_i^T z_j)$ is the parameter of a Bernoulli distribution $\mathcal{B}\text{er}(\beta_{ij})$ associated with each generated edge.

Given the design choices of the generative and inference models, we formulate a variational lower bound of the input graph log-likelihood as follows:

$$\begin{aligned}\mathcal{L}_1 = &\sum_{i,j=1}^{N} \mathbb{E}_{z_i, z_j \sim q(.|X,A)}\Big[\log\big(p(a_{ij}|z_i, z_j)\big)\Big] \\ &- 2\,N \sum_{i=1}^{N} KL\big(q(z_i|X, A)||p(z_i)\big).\end{aligned} \quad (8)$$

The first term of $\mathcal{L}_1$ is the adjacency reconstruction. The gradient of this term can be estimated by Monte Carlo sampling and the *reparameterization trick* [Kingma and Welling, 2014]. The second term of is a regularization function. The full derivation of this bound is provided in Appendix B.

3.2 Second Phase

In the first phase, our model learns low-dimensional and curved embedded manifolds. However, because of the curved aspect, the latent clusters are hard to discover using Euclidean-based methods. Unlike previous approaches, we avoid flattening the latent structures by performing embedded clustering directly after the first phase. Instead of that, our second phase smooths the local curvatures while emphasizing the global curved structures. To achieve this goal, local neighbourhoods are merged following the assumption:

$$\forall (i, l) \in [|1, N|] \times [|1, N_m|] \; q(z_i|X, A) = q(\lambda_l(z_i)|X, A), \quad (9)$$

where N_m denotes the neighbourhood size, and λ_l is a function that returns for each embedded point z_i its l^{th} nearest neighbour in the latent space. Similar to the first phase, we keep the same inference and generative models, and we establish another lower bound of the input graph log-likelihood. We derive the second bound by enforcing the neighbourhood assumption. The obtained function is formulated as follows:

$$\begin{aligned}\mathcal{L}_2 = &\frac{1}{N_m}\sum_{i,j=1}^{N}\sum_{l=1}^{N_m} \mathbb{E}_{z_i, z_j \sim q(\lambda_l(.)|X,A)}\Big[\log\Big(p(a_{ij}|z_i, z_j)\Big)\Big] \\ &- 2\,\frac{N}{N_m}\sum_{i=1}^{N}\sum_{l=1}^{N_m} KL\Big(q(\lambda_l(z_i)|X, A)||p(z_i)\Big).\end{aligned} \quad (10)$$

The first term of the second bound constructs each edge e_{ij} from the input graph by considering the local neighbourhood of the embedded points z_i and z_j. Whereas for the first phase, only z_i and z_j are considered for generating the edge e_{ij}. Similar to the first phase, the gradient of the first term can be estimated by Monte Carlo sampling and the reparameterization trick. The second term is a regularization. The full derivation of this bound is provided in Appendix C.

3.3 Third Phase

In the third phase, we alternate between maximizing two objective functions $\mathcal{L}_3$ and $\mathcal{L}_4$. Both of them constitute distinct lower bounds of the input graph log-likelihood, and they are derived based on different assumptions on the inference and generative models. The goal is to perform embedded clustering without twisting the curved structures while flattening the latent manifolds. We assess the level of the transformation by monitoring the ID of the different clusters.

$\mathcal{L}_3$ combines embedded over-clustering and adjacency reconstruction simultaneously. Let $O = (o_1, ..., o_N)$ be a sequence of N *i.i.d.* discrete random variables, where o_i describes the over-clustering assignments of the variable z_i. $\mathcal{L}_3$ is optimized by simultaneously learning the encoder training weights and a set of over-clustering centers $\{\Omega_j\}_{j=1}^{N_o}$. N_o denotes the number of over-clustering centers, which are initialized based on k-means. The generative model of $\mathcal{L}_3$ is described by the distribution $p(A, O, Z)$ that factorizes as:

$$p(A, O, Z) = p(A|Z)\, p(O|Z)\, p(Z), \tag{11}$$

where $p(Z)$ and $p(A|Z)$ are defined similar to the first phase according to Eq. (6) and Eq. (7), respectively. $p(O|Z)$ is defined by the product of Student's t-distributions to capture the similarities between the latent codes and the over-clustering centers. The distribution $p(O|Z)$ is articulated as:

$$p(O|Z) = \prod_{i=1}^{N} p(o_i|z_i), \tag{12}$$

$$p(o_i = j|z_i) = \frac{(1 + \|z_i - \Omega_j\|^2)^{-1}}{\sum_{j'} (1 + \|z_i - \Omega_{j'}\|^2)^{-1}}. \tag{13}$$

The inference model for $\mathcal{L}_3$ is described by the distribution $q(Z, O|X, A) = q(O|Z)\, q(Z|X, A)$, such that $q(Z|X, A)$ is expressed according to Eq. (5). As for $q(O|Z)$, it enlarges the top assignment scores of $p(O|Z)$ similar to [Xie *et al.*, 2016]:

$$q(O|Z) = \prod_{i=1}^{N} q(o_i|z_i), \tag{14}$$

$$q(o_i = j|z_i) = \frac{p(o_i = j|z_i)^2 / \sum_{i'} p(o_{i'} = j|z_{i'})}{\sum_{j'} \left(p(o_i = j'|z_i)^2 / \sum_{i'} p(o_{i'} = j'|z_{i'})\right)}. \tag{15}$$

Given the specified inference and generative models, we formulate a lower bound of the input graph log-likelihood in Eq. (16). The full derivation of this bound is provided in Appendix D. The third term is the over-clustering objective.

$$\begin{aligned}\mathcal{L}_3 = &\sum_{i,j=1}^{N} \mathbb{E}_{z_i, z_j \sim q(.|X,A)} \Big[\log\big(p(a_{ij}|z_i, z_j)\big)\Big] \\ &- 2N \sum_{i=1}^{N} KL\big(q(z_i|X, A)||p(z_i)\big) \\ &- 2N \sum_{i=1}^{N} \mathbb{E}_{z_i \sim q(.|X,A)} \Big[KL\Big(q(o_i|z_i)||p(o_i|z_i)\Big)\Big].\end{aligned} \tag{16}$$

$\mathcal{L}_4$ formulates a clustering-oriented lower bound of the input graph log-likelihood. Let $C = (c_1, ..., c_N)$ be a sequence of N i.i.d. discrete random variables, where c_i describes the clustering assignments of z_i. $\mathcal{L}_4$ is optimized by simultaneously learning the training weights and a set of clustering centers $\{\Phi_j\}_{j=1}^{N_c}$. The clustering centers are initialized based on k-means. The generative model of $\mathcal{L}_4$ is described by the distribution $p(A, C, Z) = p(A|Z)\, p(C|Z)\, p(Z)$, such that $p(Z)$ and $p(A|Z)$ keep the same definitions provided for $\mathcal{L}_3$, $p(C|Z) = \prod_{i=1}^{N} p(c_i|z_i)$, and $p(c_i|z_i)$ is expressed as:

$$p(c_i = j|z_i) = \frac{(1 + \|z_i - \Phi_j\|^2)^{-1}}{\sum_{j'} (1 + \|z_i - \Phi_{j'}\|^2)^{-1}}. \tag{17}$$

The inference model of $\mathcal{L}_4$ is defined by the distribution $q(Z, C|X, A) = q(C|Z)\, q(Z|X, A)$, such that $q(Z|X, A)$ is defined in Eq. (5). Moreover, we reduce the intracluster variance by enforcing the following clustering assumption:

$$\forall i \in [|1, N|] \quad q(z_i|X, A) = q(\gamma(z_i)|X, A), \tag{18}$$

where γ is a function that returns for each latent code the corresponding clustering center. The distribution $q(C|Z)$ factorizes as $q(C|Z) = \prod_{i=1}^{N} q(c_i|z_i)$ and $q(c_i|z_i)$ is expressed as:

$$q(c_i = j|z_i) = \frac{p(c_i = j|z_i)^2 / \sum_{i'} p(c_{i'} = j|z_{i'})}{\sum_{j'} \left(p(c_i = j'|z_i)^2 / \sum_{i'} p(c_{i'} = j'|z_{i'})\right)}. \tag{19}$$

Given the specified inference and generative models, we formulate the fourth lower bound in Eq. (20). The full derivation of this bound is provided in Appendix E. The first term constructs the input graph edges by considering the similarities between the latent centers. The second term is a regularization, and the third term represents the clustering objective.

$$\begin{aligned}\mathcal{L}_4 = &\sum_{i,j=1}^{N} \mathbb{E}_{z_i, z_j \sim q(\gamma(.)|X,A)} \Big[\log\Big(p(a_{ij}|z_i, z_j)\Big)\Big] \\ &- 2N \sum_{i=1}^{N} KL\Big(q(\gamma(z_i)|X, A)||p(z_i)\Big) \\ &- 2N \sum_{i=1}^{N} \mathbb{E}_{z_i \sim q(.|X,A)} \Big[KL\Big(q(c_i|z_i)||p(c_i|z_i)\Big)\Big].\end{aligned} \tag{20}$$

3.4 Algorithm

For the first phase, we train for a fixed number of iterations T_1. For the second phase, we train for T_2 iterations. At the end of this phase, we load the training weights of the iteration with the lowest average ID value. For the third phase, we train the model until the average ID becomes equal to LID. At the end of the third phase, we load the training weights of the iteration with the lowest average ID value among the ten iterations with the best clustering quality according to the unsupervised metric DBI [Davies and Bouldin, 1979]. Eventually, we perform spectral clustering [Ng *et al.*, 2002] on the final latent codes to get the clustering assignments. The full algorithm and its complexity analysis are provided in Appendix F and Appendix G, respectively, due to page limit restrictions.

Method	Cora			Citeseer			Pubmed			USA Air-Traffic			Europe Air-Traffic			Brazil Air-Traffic		
	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
GAE	61.3	44.4	38.1	48.2	22.7	19.2	63.2	24.9	24.6	43.9	13.6	11.8	47.6	19.9	12.7	62.6	37.8	30.8
ARGE	64.0	44.9	35.2	57.3	35.0	34.1	68.1	27.6	29.1	48.5	25.1	17.6	50.1	23.7	16.6	67.2	43.8	38.0
ARVGE	63.8	45.0	37.4	54.4	26.1	24.5	63.5	23.2	22.5	48.2	24.7	20.0	51.1	22.2	16.2	63.4	43.8	38.5
DGI	71.3	56.4	51.1	68.8	44.4	45.0	53.3	18.1	16.6	52.2	22.9	21.7	48.6	16.1	12.3	64.9	31.0	30.4
AGC	68.9	53.7	48.6	67.0	41.1	41.9	69.8	31.6	31.9	48.1	20.6	14.2	33.3	09.5	04.0	39.7	24.1	08.1
DAEGC	70.4	52.8	49.6	67.2	39.7	41.0	67.1	26.6	27.8	46.4	27.2	18.4	53.6	30.9	23.3	71.0	47.4	41.2
GIC	72.5	53.7	50.8	69.6	**45.3**	46.5	67.3	31.9	29.1	49.7	22.1	19.9	40.4	09.4	06.2	40.5	23.5	14.1
AGE	76.1	59.9	54.5	70.1	44.3	45.4	71.1	31.6	33.4	50.9	22.5	18.2	54.2	30.8	19.6	40.9	20.0	16.2
VGAE	64.7	43.4	37.5	51.9	24.9	23.8	69.6	28.6	31.7	45.8	23.6	15.7	49.9	23.5	16.7	64.1	38.0	30.7
GMM-VGAE	71.5	53.1	47.4	67.5	40.7	42.4	71.1	29.9	33.0	48.1	21.9	13.2	51.1	27.5	21.7	70.2	46.0	41.9
FT-VGAE	**77.4**	**61.0**	**58.2**	**70.8**	44.5	**46.7**	**72.5**	**33.0**	**35.5**	**53.7**	**27.5**	**22.8**	**55.1**	**31.4**	**26.3**	**73.3**	**47.6**	**44.4**

Table 1: Comparing the clustering performance between different graph clustering methods. Best methods in bold and second best underlined.

4 Experiments

We select six datasets for our experiments: three citation networks [Sen *et al.*, 2008] (Cora, Citeseer, and Pubmed) and three air traffic networks [Sen *et al.*, 2008] (Brazil Air Traffic, US Air Traffic, and Europe Air Traffic). The data description and preprocessing are discussed in Appendix I. Our comparison covers ten deep graph clustering models: GAE, VGAE [Kipf and Welling, 2016], ARGE, ARVGE [Pan *et al.*, 2018], DGI [Veličković *et al.*, 2019], AGC [Zhang *et al.*, 2019], DAEGC [Wang *et al.*, 2019], GMM-VGAE [Hui *et al.*, 2020], GIC [Mavromatis and Karypis, 2021], and AGE [Cui *et al.*, 2020]. The clustering results of DGI are obtained by running k-means on the latent codes. For each baseline, we use the code of the original paper, and we set the hyper-parameters as suggested by the authors or we tune them if no recommendations are provided. All experiments are performed under the same hardware and software environments as described in Appendix J. For evaluation, we employ three standard clustering metrics, namely ACC, NMI, and ARI, and two geometric ones, namely ID and LID. We provide a full description of ID and LID in Appendix K. The architecture, learning rates, and all the other hyper-parameters of FT-VGAE are specified and discussed in Appendix H. The training layers are initialized from uniform distributions. Our code is available on https://github.com/nairouz/FT-VGAE.

4.1 Clustering Results

In Table 1, we compare FT-VGAE with several state-of-the-art graph clustering methods on six datasets. For each model, we report the best results among ten trials. Initially, we focus on the first part of the table (i.e., the first 8 rows), which considers methods with different architectures and objective functions. As we can see, FT-VGAE yields better results than these models. Despite the differences in various factors, the improvement brought by FT-VGAE suggests the importance of tackling FT for solving the graph clustering problem. In the second part (i.e., the last three rows) of Table 1, we perform a more conclusive comparison between three variational graph auto-encoders (VGAE, GMM-VGAE, and FT-VGAE). VGAE (one training phase), GMM-VGAE (two training phases), and FT-VGAE (three training phases) have the same architecture, they maximize similar objective functions (i.e., lower bounds of the input graph log-likelihood). We observe from Table 1 that FT-VGAE outperforms VGAE and GMM-VGAE by a considerable margin. For instance, the difference in clustering performance between FT-VGAE and GMM-VGAE is higher than 10% in terms of ARI on Cora. Unlike VGAE and GMM-VGAE, FT-VGAE stands out by its capacity to tackle the FT problem. Further analysis and comparison with these variational models in terms of execution time are available in Appendix L.

4.2 Feature Twist

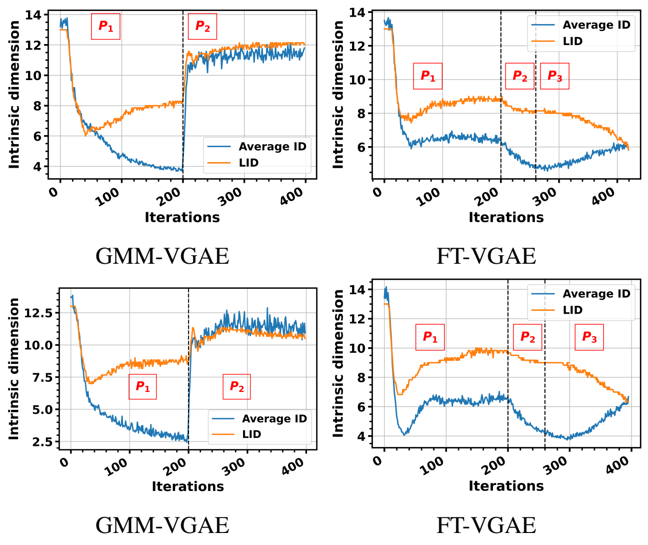

Figure 1: ID and LID of GMM-VGAE and FT-VGAE on Cora (first row) and Citeseer (second row). Average ID: average ID of the clustering manifolds. LID: number of dimensions that can capture 90% of the covariance matrix (linear correlations) estimated based on PCA (Principal Component Analysis). P_i indicates the i^{th} phase.

In Figure 1, we study the training process of FT-VGAE and GMM-VGAE on Cora and Citeseer from a geometric perspective based on two metrics ID and LID. We observe an abrupt transition in ID for GMM-VGAE when the training goes from the self-supervision task (i.e., phase one) to the pseudo-supervision task (i.e., phase two). Specifically, the embedded manifolds undergo substantial transformation in a few iterations: from curved (strong difference between ID and LID) low-dimensional (much lower than the embedded space dimension) manifolds to flattened higher-dimensional structures. Such an abrupt transition brings geometric deterioration caused by the inappropriate flattening (twisting) of the curved structures as explained in Appendix A. As opposed to GMM-VGAE, FT-VGAE has a smooth transition in ID between the different phases. The second phase of FT-VGAE makes the average ID decrease by merging the local neighbourhoods while globally maintaining the curved structures (see the difference between ID and LID in the second phase in Figure 1). The third phase flattens the latent manifolds slowly by combining clustering and over-clustering to avoid twisting the curved structures.

4.3 Ablation Study

Dataset	Metrics	P_1	P_1 & P_2	P_1&P_3	P_1&P_2&P_3
Cora	ACC	60.1	55.6	76.4	77.4
	NMI	47.1	46.4	59.4	61.0
	ARI	23.9	22.9	55.8	58.2
Citeseer	ACC	61.9	70.0	70.5	70.8
	NMI	34.4	43.2	44.3	44.5
	ARI	34.0	45.5	46.5	46.7
Pubmed	ACC	70.7	72.1	70.1	73.3
	NMI	31.3	32.6	32.1	47.6
	ARI	32.9	35.1	32.5	44.4

Table 2: Ablation study of FT-VGAE. P_i indicates the i^{th} phase.

In Table 2, we show the results of four ablation experiments conducted on three datasets to tease out the effectiveness of our contributions. First, we observe that performing $P_1\&P_2$ does not necessarily give better results than P_1 alone. In fact, P_2 widens the gap between the average ID and LID, which can make the clustering structures more difficult to discover even with spectral clustering. Second, we observe that performing $P_1\&P_3$ does not necessarily yield better results than $P_1\&P_2$ or even P_1 alone. This result suggests that combining clustering and over-clustering without the second phase is not necessarily sufficient to escape FT. Third, we observe that performing $P_1\&P_2\&P_3$ gives consistently better results than $P_1\&P_2$ and $P_1\&P_3$. These results substantiate the suitability of performing P_3 after P_2. Hence, it is important to flatten the local curves during the second phase before slowly flattening the global structures during the third phase.

4.4 Sensitivity

We study the sensitivity of FT-VGAE to the hyperparameters $N_o(\ll N)$ and $N_m(\ll N)$ on Cora and Citeseer. We expect that too large values for these hyper-parameters leads to lower clustering results. Therefore, we select N_o and N_m from the ranges $[3, 5, 7, 9, 11]$ and $[50, 100, 150, 200, 250]$, respectively. As we can see in Figure 2, our model yields strong results in terms of ACC and NMI for a wide range of values.

4.5 Visualisation

As we can see in Figure 3, the second phase promotes the separation between the different clusters. Furthermore, we observe that the third phase makes the clustering structures more pronounced by introducing pseudo-supervision.

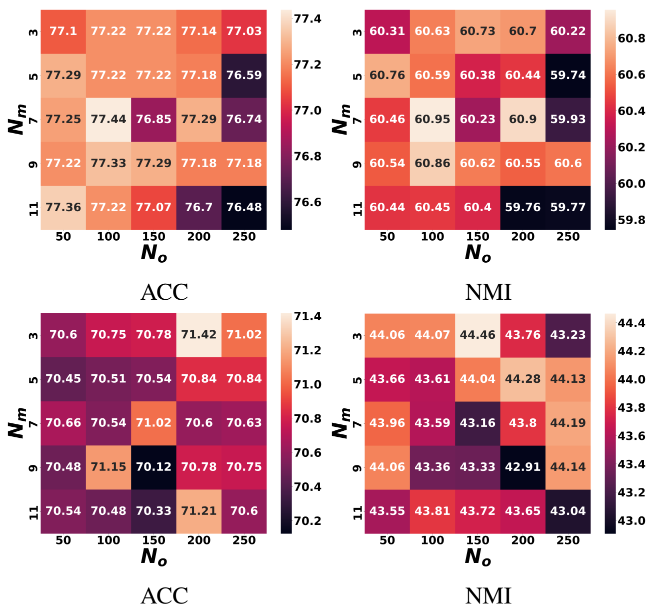

Figure 2: Sensitivity of FT-VGAE to N_m and N_o in terms of ACC and NMI. Top row: results on Cora; bottom row: results on Citeseer.

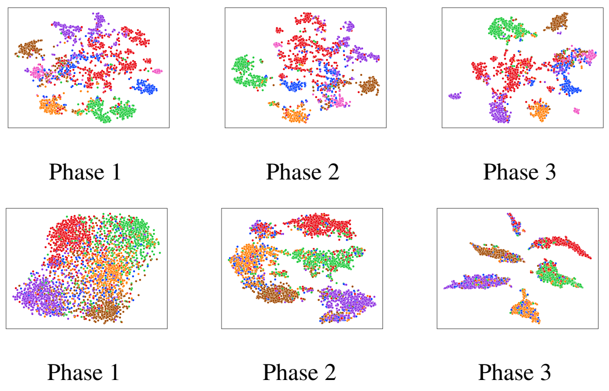

Figure 3: T-SNE visualizations of the latent representations. Top row: results on Cora; bottom row: results on Citeseer.

5 Conclusion

This work studies the variation of the intrinsic dimension under the transition regime from self-supervision to pseudo-supervision. We find that the latent clusters undergo a coarse geometric transformation: from curved low-dimensional to flattened higher-dimensional structures. We establish empirically that this inappropriate flattening, namely, Feature Twist, leads to clustering deterioration. To alleviate this problem, we propose a principled variational auto-encoder framework with three training phases. The first phase learns the curved manifolds. After that, the second phase flattens the local structures while emphasizing the globally curved shape. The third phase then gradually flattens the global structures while preserving the local configurations. Through relevant empirical results, we show that our method outperforms several state-of-the-art approaches by escaping Feature Twist.

Acknowledgments

This work has been supported by Research Grants from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[Ansuini *et al.*, 2019] Alessio Ansuini, Alessandro Laio, Jakob H Macke, and Davide Zoccolan. Intrinsic dimension of data representations in deep neural networks. In *NeurIPS*, pages 6111–6122, 2019.

[Bengio *et al.*, 2013] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE TPAMI*, 35(8):1798–1828, 2013.

[Cui *et al.*, 2020] Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. Adaptive graph encoder for attributed graph embedding. In *KDD*, pages 976–985, 2020.

[Davies and Bouldin, 1979] David L Davies and Donald W Bouldin. A cluster separation measure. *IEEE TPAMI*, 1(2):224–227, 1979.

[Facco *et al.*, 2017] Elena Facco, Maria d'Errico, Alex Rodriguez, and Alessandro Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific Reports*, 7(1):1–8, 2017.

[Hui *et al.*, 2020] Binyuan Hui, Pengfei Zhu, and Qinghua Hu. Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning. In *AAAI*, volume 34, pages 4215–4222, 2020.

[Kingma and Welling, 2014] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Variational graph auto-encoders. In *NeurIPS Workshop*, pages 1–3, 2016.

[Kipf and Welling, 2017] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Liu *et al.*, 2018] Qi Liu, Miltiadis Allamanis, Marc Brockschmidt, and Alexander L Gaunt. Constrained graph variational autoencoders for molecule design. In *NeurIPS*, pages 7806–7815, 2018.

[Liu *et al.*, 2021] Yixin Liu, Shirui Pan, Ming Jin, Chuan Zhou, Feng Xia, and Philip S Yu. Graph self-supervised learning: A survey. *arXiv preprint arXiv:2103.00111*, 2021.

[Mavromatis and Karypis, 2021] Costas Mavromatis and George Karypis. Graph infoclust: Maximizing coarse-grain mutual information in graphs. In *PAKDD*, pages 541–553, 2021.

[Mrabah *et al.*, 2020] Nairouz Mrabah, Naimul Mefraz Khan, Riadh Ksantini, and Zied Lachiri. Deep clustering with a dynamic autoencoder: From reconstruction towards centroids construction. *Neural Networks*, 130:206–228, 2020.

[Mrabah *et al.*, 2021] Nairouz Mrabah, Mohamed Bouguessa, Mohamed Fawzi Touati, and Riadh Ksantini. Rethinking graph auto-encoder models for attributed graph clustering. *arXiv preprint arXiv:2107.08562*, 2021.

[Mrabah *et al.*, 2022] Nairouz Mrabah, Mohamed Bouguessa, and Riadh Ksantini. Adversarial deep embedded clustering: on a better trade-off between feature randomness and feature drift. *IEEE TKDE*, 34(4):1603–1617, 2022.

[Ng *et al.*, 2002] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *NeurIPS*, pages 849–856, 2002.

[Pan *et al.*, 2018] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. In *IJCAI*, pages 2609–2615, 2018.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *KDD*, pages 701–710, 2014.

[Sen *et al.*, 2008] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

[Veličković *et al.*, 2019] Petar Veličković, William Fedus, William L. Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. In *ICLR*, 2019.

[Wang *et al.*, 2019] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. In *IJCAI*, pages 3670–3676, 2019.

[Xie *et al.*, 2016] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *ICML*, pages 478–487, 2016.

[Yang *et al.*, 2021] Fan Yang, Rui Meng, Hyuna Cho, Guorong Wu, and Won Hwa Kim. Disentangled sequential graph autoencoder for preclinical alzheimer's disease characterizations from adni study. In *MICCAI*, pages 362–372, 2021.

[Zhang *et al.*, 2019] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. Attributed graph clustering via adaptive graph convolution. In *IJCAI*, pages 4327–4333, 2019.