Stationary Diffusion State Neural Estimation for Multiview Clustering

Chenghua Liu, Zhuolin Liao, Yixuan Ma, Kun Zhan[⋆]

School of Information Science and Engineering, Lanzhou University
{liuchh20,liaozhl20,mayx2021,kzhan}@lzu.edu.cn

Abstract

Although many graph-based clustering methods attempt to model the stationary diffusion state in their objectives, their performance limits to using a predefined graph. We argue that the estimation of the stationary diffusion state can be achieved by gradient descent over neural networks. We specifically design the Stationary Diffusion State Neural Estimation (SDSNE) to exploit multiview structural graph information for co-supervised learning. We explore how to design a graph neural network specially for unsupervised multiview learning and integrate multiple graphs into a unified consensus graph by a shared self-attentional module. The view-shared self-attentional module utilizes the graph structure to learn a view-consistent global graph. Meanwhile, instead of using auto-encoder in most unsupervised learning graph neural networks, SDSNE uses a co-supervised strategy with structure information to supervise the model learning. The co-supervised strategy as the loss function guides SDSNE in achieving the stationary state. With the help of the loss and the self-attentional module, we learn to obtain a graph in which nodes in each connected component fully connect by the same weight. Experiments on several multiview datasets demonstrate effectiveness of SDSNE in terms of six clustering evaluation metrics.

1 Introduction

Feature diversity is ubiquitous and we live in a world composed of a large amount of multiview content. Multiple views refer to different features of the same instance (Blum and Mitchell 1998). Since multiview features are highly relevant, more and more artificial intelligence tasks involve the processing of multiview data. Our goal is to leverage multiview data to derive clustering algorithms.

Multiview clustering carries out joint feature learning and co-view relationship modeling, aiming to exploit the correlation of different views effectively. Since the essential consensus structure coexists in multiview features, using the structured graphs of different views for unsupervised multiview feature learning is able to optimize clustering performance. Combining structural information from different views can achieve better efficient performance than clustering of any single view.

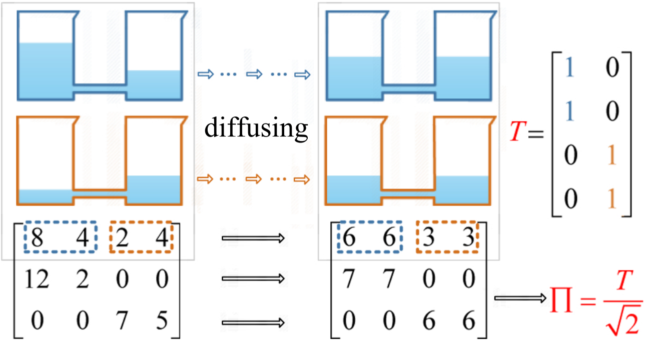


Figure 1: A diffusion example. A cup denotes a node in the graph and the color of cups denotes the class label. Cups have different initial water volumes. After a long time, connected cups with the same color have the same amount of water, which is stationary. All of the different stationary states can be categorized into three cases, i.e., case 1 is that four cups have water as shown; case 2 is blue cups have water while no water in oranges; and case 3 is no water in blues while oranges have water, so we show three cases for example. If all of the different stationary states are orthonormalized to a stationary matrix Π, then Π has the semantic meaning to the indicator matrix T. Then, we argue that the estimation of Π can be achieved by gradient descent over neural networks.

Since discriminative modeling is mostly used when exploring the internal structure of data, the structural relationship between data points is usually expressed in the form of a graph. With predefined graphs, co-learning spectral clustering algorithms are applied firstly to unsupervised multiview learning (Kumar and Daumé 2011; Kumar, Rai, and Daume 2011) due to the well-defined mathematical framework of spectral clustering (SC) (Ng, Jordan, and Weiss 2002). Later, a series of SC or subspace learning-based methods are applied to multiview clustering by exploiting graph information (Xia et al. 2014; Li et al. 2015; Gao et al. 2015; Zhang et al. 2018; Pan and Kang 2021). Although these graph-based multiview clustering methods have made great progress by integrating information from different views, their unified consensus representation is usually obtained by a simple weighted-sum approach. We design a neural estimator that synergistically uses multiple graphs to obtain the consensus graph by explor-

[⋆]Corresponding author.

ing the internal structure of multiview data. In this paper, we learn to obtain a consensus graph and the clustering results can be obtained without performing the eigen decomposition.

How to specifically design a graph neural network for unsupervised multiview learning is a challenging problem. We argue that the estimation of the stationary state as described in Fig. 1 can be achieved by gradient descent over neural networks. In Fig. 1, the stationary state is given by Π and the values in Π means the blue cups belong to cluster 1 and the oranges belong to cluster 2, i.e., in the graph of Fig. 1, the blues connect with blues while oranges connect to oranges, which is an ideal structure since each node-connected component belongs to one cluster. We exploit multiple graphs to learn such an ideal structural graph under the motivation of the stationary diffusion state neural estimation (SDSNE). If a multiview system becomes stationary, our intuition is that it needs to share an intrinsical structural graph. With this intuition, we use a shared self-attentional module to model a neural layer. In comparison with graph convolutional network (GCN) (Kipf and Welling 2017), a step of the diffusion pattern is similar to the layer of GCN, but GCN lacks effective supervision to render it stationary at the ideal state. For designing a good supervision, we use the multiple graphs in a unsupervised manner via adding a co-supervised loss.

For improving the multiview clustering performance, we design SDSNE under considering three observations: (1) An intrinsical consensus graph sharing between views; (2) the unified global feature is the supervision of each single-view feature and each single-view is also the supervision of the global feature; and (3) for achieving the stationary state, the neural estimated state transition probability matrix is constructed by a learned graph. Inspired by observation 1, the parameter matrix of SDSNE models a graph and the multiview features share the same parameter matrix. From observation 2, we design a co-supervised loss function. The co-supervised loss of SDSNE not only renders the diffusion to become stationary but also makes different views consistent. Observation 3 makes SDSNE different from auto-encoder, i.e., we do not use a reconstructed loss like the auto-encoder for features or graphs. The terms of the co-supervised loss function are learned by SDSNE.

The main contributions of this paper are listed as follows:

- We introduce the Stationary Diffusion State Neural Estimation (SDSNE) for multiview clustering, which is trainable via back-propagation.
- We show that the multiview clustering utility of this estimator is derived from the shared parameter between views and the unified global feature.
- We design a co-supervised loss to guide SDSNE in achieving the stationary state. The co-supervision of the global feature and each single view renders them achieve the stationary state.

Thus, SDSNE for multiview clustering can learn a good graph representation, and we use the graph representation to obtain the clustering label. Extensive experiments on real-world datasets are conducted to validate the superiority of SDSNE in terms of different clustering evaluation metrics.

2 Related Works

Superficially, the objective of SDSNE is similar to SC (Shi and Malik 2000; Ng, Jordan, and Weiss 2002). SDSNE uses the learned graph in its objective function but SC exploits a predefined graph. SC-based methods are very difficult to avoid the eigen decomposition of a Laplacian matrix constructed by the raw feature or the embedding. SDSNE achieves its objective by utilizing the learned multiview structural graphs synergistically. A synergism co-supervised loss function is designed to render each single view stationary. In existing traditional methods, clustering with adaptive neighbors (CAN) (Nie, Wang, and Huang 2014) also learns to obtain exact k connected components in the graph (k is the cluster number.). Inspired by CAN, Zhan (Zhan et al. 2018, 2019) designed MVGL and MCGC to obtain a graph with k connected components for multiview clustering. The objective of MVGL or MCGC is divided into subproblems and is solved alternately. In MVGL and MCGC, the number of components is determined by the number of the multiplicity of 0 as an eigenvalue of the Laplacian matrix. Thus, they need to perform the eigen decomposition in each iteration, which results in high complexity. Although SpectralNet (Shaham et al. 2018) and MvSCN (Huang et al. 2019) use the concept of spectral clustering in their titles, but the representation learning of them is mainly derived from Siamese networks (Hadsell, Chopra, and LeCun 2006; Chopra, Hadsell, and LeCun 2005) or DEC (Xie, Girshick, and Farhadi 2016). The core representation learning of SpectralNet and MvSCN do not use the objective of SC. We build a neural estimator to effectively achieve such an objective in an end-to-end way effectively.

For the multiview clustering, O2MAC (Fan et al. 2020) employs GCN for attributed multiview graph clustering. The graph auto-encoder of O2MAC exploits a reconstruction loss between input graphs and decoded graphs. Besides the loss, O2MAC uses a self-supervised loss. In the beginning, GCN was used for semi-supervised clustering (Kipf and Welling 2017). If GCN has a good prediction on unlabelled data, its predicted labels are similar to the clustering indicator matrix. Since the GCN layer is quite similar to a step of the diffusion process, GCN can be regarded as using gradient descent over GCN layer for obtaining the stationary state.

Cross-diffusion process (CDP) (Wang et al. 2012) is inspired by co-training (Blum and Mitchell 1998) algorithm. CDP is applied to biomedical research for clustering cancer subtypes (Wang et al. 2014) and the multiview features are extracted for each patient, i.e., DNA methylation, mRNA expression, and microRNA (miRNA) expression (Wang et al. 2014). Variants of CDP are widely applied to image retrieval (Bai et al. 2017a,b). CDP has a good explanation in (Bai et al. 2017a,b). Based on (Bai et al. 2017a,b), CGD (Tang et al. 2020) uses cross-view diffusion on multiple graphs and the final clustering result is obtained by SC.

SDSNE for multiview clustering is a unified end-to-end neural network. The unique purpose of SDSNE is to estimate the stationary state. SDSNE learns to obtain a graph: nodes in each connected component connect with the same weight. With the representation, it can obtain clustering with the k-means clustering algorithm. Although SDSNE, SC, CAN, and GCN have a similar objective, SDSNE models the stationary state directly and explores multiview clustering under the co-supervision of multiple graphs.

3 Method

3.1 Denotation

Let $\mathcal{X} = \{X^{(1)}, X^{(2)}, \ldots, X^{(n_v)}\}$ be a multiview dataset with n_v different views. Feature matrix is denoted by $X^{(v)} \in \mathbb{R}^{n \times d_v}$ where n is the number of data points and d_v denotes the dimension of the v-th view. We suppose that n instances belong to k categories. $I \in \mathbb{R}^{n \times n}$ is an identity matrix. We build affinity graphs $A^{(v)} = [a_{ij}^{(v)}]$ for each view with the Gaussian kernel. We model to obtain a unified graph $H = f(A^{(v)}, \forall\, v|W)$ for multiview clustering, where $f(\cdot|W)$ denotes a neural network with parameter W.

3.2 Stationary Diffusion State

An undirected graph is regarded as a Markov chain (Brin and Page 1998). Graph diffusion process usually starts from a predefined affinity graph $A = [a_{ij}] \in \mathbb{R}^{n \times n}, \forall\, ij, a_{ij} \geqslant 0$ (Zhou and Burges 2007). An element a_{ij} denotes a pairwise affinity between nodes i and j. The Markov transition matrix P can be deduced from A. In this paper, we define $P = [p_{ij}], \forall\, ij, p_{ij} = p_{ji} \geqslant 0, \sum_i p_{ij} = 1$, and the graph diffusion process is given by

$$\boldsymbol{h} \leftarrow P\boldsymbol{h} \tag{1}$$

where $\boldsymbol{h}$ denotes the state of nodes in the graph.

Theorem 1. *The number k of connected components of the graph is equal to the multiplicity of 1 as an eigenvalue of P.*

Proof. If the multiplicity of 1 is k, their corresponding orthonormal eigenvectors are $\boldsymbol{\pi}_1$, $\boldsymbol{\pi}_2, \ldots$, and $\boldsymbol{\pi}_k$. For each $\boldsymbol{\pi}$, we have $\boldsymbol{\pi}^\top\boldsymbol{\pi} - \boldsymbol{\pi}^\top P\boldsymbol{\pi} = 0$ from $P\boldsymbol{\pi} = 1\boldsymbol{\pi}$, then, $\boldsymbol{\pi}^\top\boldsymbol{\pi} - \boldsymbol{\pi}^\top P\boldsymbol{\pi} = \frac{1}{2}\sum_{i,j=1}^n p_{ij}(\pi_i - \pi_j)^2 = 0$ if and only if $\pi_i = \pi_j$ is constant on each connected component. □

Theorem 2. *The following statements are equivalent for the Markov chain determined by a transition probability matrix P*

1. *The Markov chain is stationary at the state of $\boldsymbol{\pi}$.*
2. *$\boldsymbol{\pi} = P\boldsymbol{\pi}$.*
3. *$\sum_{i,j=1}^n p_{ij}(\pi_i - \pi_j)^2 = 0$.*

Proof. If $\boldsymbol{h}$ is unknown, the solution of $\boldsymbol{h} = P\boldsymbol{h}$ is an eigenvector corresponding to the eigenvalue 1 of P or is obtained by the Gauss-Seidel method. With the Gauss-Seidel method, Eq. (1) iterates until $\boldsymbol{h}$ does not change, which means that an initial state without loss of generality is able to render it stationary at the state of $\boldsymbol{\pi}$, i.e., *1* and *2* are equivalents. From the proof of Theorem 1, we find that *2* and *3* are equivalent. □

Theorem 3. *The k eigenvectors corresponding to eigenvalue 1 of P constructs the matrix $\Pi = [\boldsymbol{\pi}_i]$, and $H = [\boldsymbol{h}_i]$ is constrained by $H^\top H = I$. Then, the inequality $0 = \sum_{i=1}^k \boldsymbol{\pi}_i^\top (I - P)\boldsymbol{\pi}_i = \min_{H^\top H = I} \mathrm{Tr}(H^\top (I - P)H) \leqslant \sum_{i=1}^k \boldsymbol{h}_i^\top (I - P)\boldsymbol{h}_i$ holds.*

Proof. See Zhan et al. 2019. □

From Theorem 3, we have

$$\mathrm{Tr}(\Pi^\top (I - P)\Pi) \leqslant \mathrm{Tr}(H^\top (I - P)H) \tag{2}$$

and we argue that the estimation of the stationary state described in Theorem 1 can be achieved by gradient descent over neural networks, i.e., we use $\mathrm{Tr}(H^\top (I - P)H)$ as its loss function and define Eq. (1) to be its neural layer.

In comparison with GCN (Kipf and Welling 2017), the diffusion, Eq. (1), is similar to the layer of GCN. The layer of GCN can be described by

$$H \leftarrow \hat{A}H\Theta \tag{3}$$

where Θ denotes the model parameter, and $\hat{A}$ is a normalized graph with self-loop. $\hat{A}$ has the same semantic meaning of the state transition probability matrix P.

With the guidance of loss functions, if the final representation of GCN, H, tends to be a good prediction, it is similar to Π. Π is the ideal clustering indicator, which implies that the output of Eq. (3) is directly obtained from the stationary state after learning with the gradient descent over GCN.

3.3 SDSNE for Multiview Clustering

Given different transition matrices, $P^{(1)}$ and $P^{(2)}$, in two views, we construct a hyper-transition matrix with them,

$$\boldsymbol{P} = P^{(1)} \otimes P^{(2)} \tag{4}$$

where $\otimes$ denotes the Kronecker product. The detail of why we design such a hyper transition matrix refers to §3.4.

Then, the diffusion with $\boldsymbol{P}$ is given by

$$\boldsymbol{g} \leftarrow \boldsymbol{P}\boldsymbol{g} = \mathrm{vec}\big(P^{(2)} S (P^{(1)})^\top\big) \tag{5}$$

where $\mathrm{vec}(\cdot)$ denotes the vectorization by stacking columns one by one and $\mathrm{vec}(S) = \boldsymbol{g}$.

If a multiview system becomes stationary, our intuition is to share an intrinsical structural graph. From Eq. (5), we model the consensus graph S for different views by,

$$S \leftarrow P^{(v)} S (P^{(u)})^\top, \forall\, u, v \in \{1, 2, \ldots, n_v\}\,. \tag{6}$$

Eq. (6) implies that all views share a consensus graph S. Note that S can also be regarded as a graph and the detail of why S is a graph refers to §3.4.

We use a neural network to learn directly with the gradient descent algorithm to obtain such a stationary state. Since the consensus feature is modeled by S which is shared between different views, P in Eq. (6) can be from the same view. We model the diffusion as a layer of the neural network and we share model parameter W in different views,

$$H^{(v)} \leftarrow P^{(v)} W (P^{(v)})^\top, \forall\, v \in \{1, 2, \ldots, n_v\}\,. \tag{7}$$

Then, we fuse $H^{(v)}$ to obtain a unified global feature by,

$$H = \alpha \sum_{v=1}^{n_v} H^{(v)} + (1 - \alpha)I \tag{8}$$

where α is a trade-off hyper parameter.

Note that $H^{(v)}$ can be also regarded as a graph and the detail of why $H^{(v)}$ is a graph refers to §3.4. With Eq. (7), we obtain different graphs $H^{(v)}$ and $H^{(v)}$ is normalized to attain $\hat{P}^{(v)}$ for each view.

According to Theorems 2 and 3, the loss function guides SDSNE in obtaining a stationary state by minimizing,

$$\mathcal{L}_{\mathrm{sds}} = \sum_{v=1}^{n_v} \mathrm{Tr}\big(H^\top (I - \hat{P}^{(v)})H\big) \, . \tag{9}$$

If a hyper graph achieves the stationary state, edges in each connected component tend to have the same value in the output graph $H^{(v)}$, i.e., nodes in each component connect with each other by the same edge weight. We need a graph in which different values in different components, rather than a graph in which a component is marked by a non-zero value while others are zeros, so we add an ℓ_2-regularization loss. The overall loss is given by,

$$\mathcal{L} = \mathcal{L}_{\mathrm{sds}} + \mu \sum_{v=1}^{n_v} \mathrm{Tr}\big((H^{(v)})^\top H^{(v)}\big) \, . \tag{10}$$

We summarize the SDSNE algorithm in Algorithm 1. We use the symmetric Laplacian matrix rather than the random walk Laplacian since the former usually has better performance (Chung 1997; Shi and Malik 2000; Ng, Jordan, and Weiss 2002; Von Luxburg 2007).

Algorithm 1: SDSNE for multiview clustering.

```
Input: $\mathcal{X} = \{X^{(1)}, X^{(2)}, \ldots, X^{(n_v)}\}$ .
Output: $H$ .
Initialization: $\alpha$, $\mu$, $W$, and $epoch_{\max}$ .
for $v \in \{1, 2, \ldots, n_v\}$ do
    Construct $A^{(v)}$ by the Gaussian kernel with $X^{(v)}$ .
    Calculate the degree matrix $D^{(v)}$ of $A^{(v)}$ .
    Normalize $A^{(v)}$ by $P^{(v)} = (D^{(v)})^{-\frac{1}{2}} A^{(v)} (D^{(v)})^{-\frac{1}{2}}$ .
end for
while $epoch \leqslant epoch_{\max}$ do
    for $v \in \{1, 2, \ldots, n_v\}$ do
        $H^{(v)} \leftarrow P^{(v)} W (P^{(v)})^\top$ .
        Calculate the degree matrix $\hat{D}^{(v)}$ of $H^{(v)}$ .
        $\hat{P}^{(v)} = (\hat{D}^{(v)})^{-\frac{1}{2}} H^{(v)} (\hat{D}^{(v)})^{-\frac{1}{2}}$ .
    end for
    Update $H$ by Eq. (8) .
    Update $\mathcal{L}$ by Eq. (10) .
    Update $W$ by the gradient descent algorithm.
    if The loss $\mathcal{L}$ converges. then
        Break.
    end if
    $epoch = epoch + 1$ .
end while
```

3.4 Analysis of SDSNE

First, we give the reason why we use a hyper graph. Referring to Theorem 3 and Eq. (5), the third statement in Theorem 2 can be reached by minimizing,

$$\boldsymbol{g}^\top (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{g} = \frac{1}{2} \sum_{i,j,k,l=1}^{n} p_{ij}^{(1)} p_{kl}^{(2)} (s_{ik} - s_{jl})^2 \tag{11}$$

where $\boldsymbol{I} \in \mathbb{R}^{n^2 \times n^2}$ is an identity matrix and $\boldsymbol{g} = \mathrm{vec}(S)$. $S = [\boldsymbol{s}_i]$ is a symmetric similarity matrix. In the right side of Eq. (11), if fixing p_{kl}, minimizing it makes $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ similar when p_{ij} is large; if fixing p_{ij}, $\boldsymbol{s}_k$ and $\boldsymbol{s}_l$ are similar when p_{kl} is large; if both p_{ij} and p_{kl} are large, the affinity s_{ik} tends to the same value of s_{jl}. Thus, by minimizing $\sum_{i,j,k,l=1}^{n} \hat{p}_{ij}^{(1)} \hat{p}_{kl}^{(2)} (h_{ik} - h_{jl})^2$, SDSNE achieves a graph $H^{(v)}$ in which nodes in each connected component connect with each other by the same edge weight. Since minimizing $\sum_{i,j,k,l=1}^{n} \hat{p}_{ij}^{(1)} \hat{p}_{kl}^{(2)} (h_{ik} - h_{jl})^2$ is equal to minimizing $\hat{p}_{kl}^{(v)} \sum_{i,j=1}^{n} \hat{p}_{ij}^{(v)} (h_{ik} - h_{jl})^2 = p_{kl}^{(v)} \mathrm{Tr}\big(H^\top (I - \hat{P}^{(v)})H\big)$, we use Eq. (9) as the loss function to render each view stationary.

Second, we answer why S or $H^{(v)}$ is regarded as a graph. Since Eq. (7) is a self-attentional module, we regard S or $H^{(v)}$ as a graph. We suppose that using $P^{(v)}$ learns a query $Q^{(v)}$ and a key $K^{(v)}$,

$$Q^{(v)} = P^{(v)} W_1 \, , \tag{12}$$

$$K^{(v)} = P^{(v)} W_2 \tag{13}$$

where W_1 and W_2 are two learnable weight matrices. We use $Q^{(v)}$ and $K^{(v)}$ to construct a new self-attentional graph, the attentional coefficient matrix can be given by Eq. (7), i.e.,

$$H^{(v)} \leftarrow Q^{(v)} (K^{(v)})^\top = P^{(v)} W (P^{(v)})^\top \, . \tag{14}$$

Third, we analyze the reason why we use $\hat{P}^{(v)}$ rather than $P^{(v)}$ in the loss. We suppose that the stationary $H_\infty^{(v)} = \Pi^{(v)}$ achieves, then we have $\Pi^{(v)} = \hat{P}^{(v)} H_0^{(v)}$, where $H_0^{(v)}$ is the initial state. This stationary state also can be derived from $\Pi^{(v)} = P^{(v)} P^{(v)} \cdots P^{(v)} H_0^{(v)}$ without loss of generality. It means that SDSNE models $W = P^{(v)} P^{(v)} \cdots P^{(v)} H_0^{(v)}$ directly by gradient descent algorithm and W is shared between views. If we use the fixed $P^{(v)}$, it guides SDSNE in staying at the first step of diffusion.

Fourth, we analyze the effect of the self-loop. We add a weighted self-loop in Eq. (8). The loss, Eq. (9), also is a structural view-consistent loss, and the self-loop in H guides $\hat{P}^{(v)}$ in learning a self-loop. As shown in Fig. 1, if the transition probability to others is 1 and the probability to itself is 0, all water moves to others unreservedly at the first step of diffusion, and all will return at the next step. To avoid such oscillations, we add a weighted self-loop in Eq. (8) .

3.5 Convergence and Complexity Analysis

Since Eq. (10) is convex, i.e., the Laplacian matrix is positive semi-definite (Chung 1997), minimizing it by the gradient descent algorithm renders SDSNE converged.

SDSNE does not perform the eigen decomposition. The complexity of the eigen decomposition is $O(n^3)$ where n is the number of data points. The layer of SDSNE costs $O(n^3)$ complexity too since it needs to perform the matrix multiplication. The matrix multiplication is easier to calculate with a parallel algorithm than the eigen decomposition. Since the existing cross-diffusion methods need $O(n^4)$ to compute the objective, SDSNE has lower complexity than cross-diffusion-based methods.

4 Experiments

4.1 Datasets

Six benchmark datasets are used to demonstrate the effectiveness of the proposed method, including

BBC Sport[1]: The document dataset contains 544 documents in five classes, such as athletics, cricket, football, rugby, tennis. Two different features are extracted for each document (Xia et al. 2014).

MSRC-v1[2]: The image dataset consists of seven classes: tree, building, airplane, cow, face, car, and bicycle. It contains 30 images in each category and each image has six views.

100 Leaves[3]: The image dataset consists of 100 classes of leaves, there are 16 images in each class, and three different features are extracted: shape, margin, and texture.

Three Sources[4]: The document dataset has 169 stories reported in BBC, Reuters, and the *Guardian*. Each story was manually annotated with one of the six topical labels: business, entertainment, health, politics, sport, and technology.

Scene-15 (Fei-Fei and Perona 2005): It consists of 4485 images in total, which has 15 scene categories with both indoor and outdoor environments. For every image, three features, including GIST, PHOG, and LBP are extracted.

Reuters[5]: We use a subset of Reuters that consists of 18,758 articles in six classes and each article has five views, i.e., English, French, German, Italian, and Spanish.

4.2 Experimental Setup

We evaluate the performance of SDSNE on six multiview datasets.We compare SDSNE with six state-of-the-art algorithms. The baseline can be coarsely categorized into three groups.

The Best Single-view:

1. $\mathrm{SC}_{\mathrm{best}}$ (Ng, Jordan, and Weiss 2002) is performed for each single-view feature and we report the best.
2. $\mathrm{LRR}_{\mathrm{best}}$ (Liu et al. 2013) uses low-rank representation to solve the subspace clustering problem and we report the best single-view results.

Graph-based:

3. MCGC (Zhan et al. 2019) imposes a rank constraint on the Laplacian matrix and utilizes a new disagreement cost function for regularizing graphs from different views to learn a consensus graph.
4. GMC (Wang, Yang, and Liu 2020) fuses the multiple graphs to generate a unified graph under consideration to the view weights.
5. CGD (Tang et al. 2020) learns a unified graph for multiview clustering via cross-view graph diffusion.

GNN-based:

6. O2MAC (Fan et al. 2020) assumes that there is a dominated view. Using GCN processes the dominated view feature to obtain a unique latent feature. With the latent feature, O2MAC reconstructs multiview graphs.

For a fair comparison, we run each method 10 times and report the mean of performance as well as the standard deviation. For SDSNE, we set the seed of the pseudo-random generator as in GCN (Kipf and Welling 2017) to eliminate the fluctuation of clustering results. The learning rate is set to be 10^{-4} or 10^{-5}. Without loss of generality, the Gaussian kernel function with Euclidean distance is used to generate initial view-specific graphs, and the σ in the Gaussian kernel function is set to 0.5. In experiments, we employ six widely used metrics to measure the clustering performance: normalized mutual information (NMI), clustering accuracy (ACC), adjusted rand index (ARI), F_1-score, precision, and purity. Note that a higher value indicates better performance for the six metrics. During training, we use an early stop strategy with the patience of 10 and stop training when the loss function drops dramatically. We perform the k-means clustering and SC to obtain the clustering results.

4.3 Experimental Results

Clustering performance is summarized in Table 1. $\mathrm{SDSNE}_{\mathrm{km}}$ means that we perform the k-means clustering on H, and $\mathrm{SDSNE}_{\mathrm{sc}}$ means that SC is performed on H.

In Table 1, we obtain the following observations: (1) On all six multiview datasets, results of $\mathrm{SDSNE}_{\mathrm{sc}}$ are higher than state-of-the-art methods. It implies that SDSNE learns the shared information well between multiple views to improve the clustering performance. (2) In most cases, SDSNE outperforms other state-of-the-art methods on both large multiview datasets, e.g., Reuters, and small datasets, e.g., Three Sources. Some methods, e.g., O2MAC, only perform well on few datasets. It means that SDSNE is efficient for multiview clustering. (3) The accuracy of $\mathrm{SDSNE}_{\mathrm{sc}}$ on Three Sources is over 15.4% than the best of other methods. At the same time, SDSNE is much better than other methods in other datasets and metrics, which shows that SDSNE effectively learns to obtain a good representation. In the textual and image datasets, SDSNE obtains better performance. The performance mainly depends on the quality of raw input data but does not depend on the types.

Figs. 2(a)-(g) show the t-SNE (Maaten and Hinton 2008) visualization of the raw feature and the output H of SDSNE on MSRC-v1. SDSNE integrates multiple graphs into a unified graph H and obtains better results than others. The t-SNE visualization and similarity graph of SDSNE show that SDSNE learns to obtain a high-quality graph representation since both $\mathrm{SDSNE}_{\mathrm{sc}}$ and $\mathrm{SDSNE}_{\mathrm{km}}$ perform well. The

[1] http://mlg.ucd.ie/datasets/bbc.html

[2] https://www.microsoft.com/en-us/research/project/image-understanding/

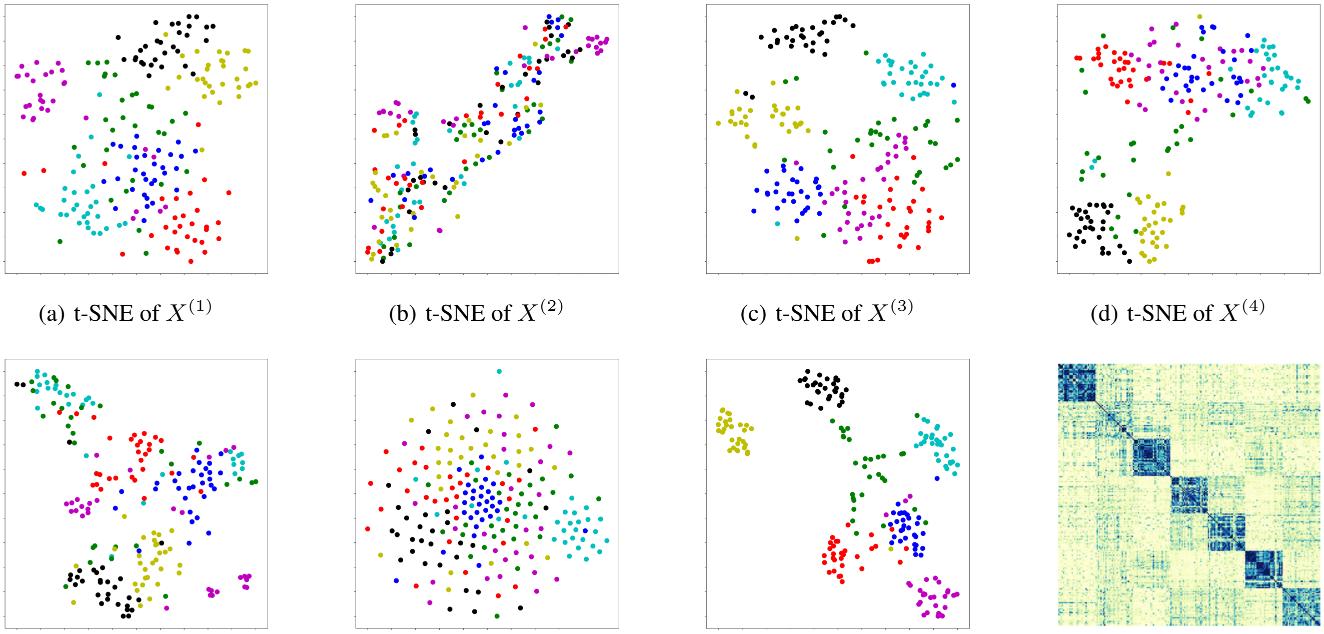
[3] https://achieve.ics.uci.edu/ml/datasets/One-hundred+plant+species+leaves+data+set

[4] http://mlg.ucd.ie/datasets/3sources.html

[5] http://ama.liglab.fr/~amini/DataSets/Classification/Multiview/ReutersMutliLingualMultiView.htm

Methods	NMI	ACC	ARI	F_1-score	Precision	Purity
BBC Sport						
SC$_{\text{best}}$	0.022±0.005	0.360±0.003	0.005±0.003	0.386±0.002	0.241±0.002	0.362±0.001
LRR$_{\text{best}}$	0.775±0.002	0.904±0.003	0.747±0.001	0.812±0.002	0.754±0.001	0.904±0.001
MCGC	0.112±0.000	0.421±0.000	0.049±0.000	0.401±0.000	0.258±0.000	0.444±0.000
GMC	0.705±0.000	0.739±0.000	0.601±0.000	0.721±0.000	0.573±0.000	0.763±0.000
CGD	0.910±0.003	0.974±0.004	0.931±0.002	0.947±0.001	0.943±0.003	0.974±0.002
O2MAC	0.891±0.018	0.964±0.008	0.906±0.019	0.965±0.009	0.959±0.011	0.964±0.008
SDSNE$_{\text{km}}$	0.899±0.000	0.969±0.000	0.918±0.000	0.938±0.000	0.974±0.000	0.969±0.000
SDSNE$_{\text{sc}}$	0.948±0.000	0.985±0.000	0.958±0.000	0.968±0.000	0.991±0.000	0.985±0.000
MSRC-v1						
SC$_{\text{best}}$	0.556±0.000	0.519±0.000	0.289±0.000	0.431±0.000	0.300±0.000	0.523±0.000
LRR$_{\text{best}}$	0.539±0.021	0.681±0.018	0.413±0.019	0.498±0.017	0.476±0.019	0.681±0.018
MCGC	0.692±0.000	0.776±0.000	0.630±0.000	0.685±0.000	0.640±0.000	0.785±0.000
GMC	0.816±0.000	0.895±0.000	0.767±0.000	0.799±0.000	0.786±0.000	0.895±0.000
CGD	0.842±0.004	0.910±0.006	0.790±0.003	0.819±0.004	0.804±0.005	0.910±0.005
O2MAC	0.617±0.011	0.709±0.030	0.525±0.020	0.691±0.026	0.716±0.040	0.715±0.021
SDSNE$_{\text{km}}$	0.898±0.000	0.943±0.000	0.867±0.000	0.886±0.000	0.953±0.000	0.943±0.000
SDSNE$_{\text{sc}}$	0.872±0.000	0.933±0.000	0.845±0.000	0.867±0.000	0.942±0.000	0.933±0.000
100 Leaves						
SC$_{\text{best}}$	0.777±0.002	0.483±0.014	0.203±0.008	0.215±0.008	0.128±0.007	0.520±0.003
LRR$_{\text{best}}$	0.715±0.018	0.488±0.013	0.307±0.011	0.315±0.010	0.274±0.007	0.529±0.009
MCGC	0.834±0.000	0.727±0.000	0.410±0.000	0.418±0.000	0.290±0.000	0.747±0.000
GMC	0.902±0.000	0.824±0.000	0.497±0.000	0.504±0.000	0.352±0.000	0.851±0.000
CGD	0.943±0.007	0.859±0.005	0.821±0.006	0.823±0.004	0.770±0.006	0.881±0.005
O2MAC	0.782±0.003	0.557±0.009	0.432±0.007	0.546±0.010	0.567±0.010	0.586±0.009
SDSNE$_{\text{km}}$	0.979±0.000	0.962±0.000	0.934±0.000	0.935±0.000	0.965±0.000	0.966±0.000
SDSNE$_{\text{sc}}$	0.972±0.000	0.957±0.000	0.913±0.000	0.914±0.000	0.967±0.000	0.957±0.000
Three Sources						
SC$_{\text{best}}$	0.054±0.014	0.331±0.015	0.011±0.012	0.362±0.011	0.228±0.008	0.349±0.013
LRR$_{\text{best}}$	0.525±0.016	0.627±0.009	0.351±0.011	0.555±0.012	0.411±0.013	0.668±0.008
MCGC	0.075±0.000	0.301±0.000	0.037±0.000	0.337±0.000	0.216±0.000	0.384±0.000
GMC	0.548±0.000	0.692±0.000	0.443±0.000	0.605±0.000	0.484±0.000	0.746±0.000
CGD	0.695±0.005	0.781±0.006	0.611±0.005	0.709±0.006	0.651±0.007	0.828±0.003
O2MAC	0.727±0.030	0.755±0.026	0.650±0.040	0.669±0.022	0.667±0.025	0.840±0.020
SDSNE$_{\text{km}}$	0.747±0.000	0.828±0.000	0.741±0.000	0.802±0.000	0.720±0.000	0.846±0.000
SDSNE$_{\text{sc}}$	0.848±0.000	0.935±0.000	0.867±0.000	0.898±0.000	0.927±0.000	0.935±0.000
Scene-15						
SC$_{\text{best}}$	0.384±0.014	0.377±0.013	0.208±0.001	0.272±0.014	0.234±0.014	0.404±0.014
LRR$_{\text{best}}$	0.369±0.002	0.368±0.003	0.201±0.001	0.263±0.002	0.233±0.003	0.395±0.001
MCGC	0.142±0.000	0.179±0.000	0.054±0.000	0.170±0.000	0.096±0.000	0.186±0.000
GMC	0.058±0.000	0.140±0.000	0.004±0.000	0.132±0.000	0.071±0.000	0.146±0.000
CGD	0.419±0.006	0.428±0.004	0.256±0.003	0.315±0.003	0.277±0.002	0.484±0.004
O2MAC	0.325±0.009	0.309±0.013	0.155±0.007	0.306±0.013	0.319±0.018	0.339±0.010
SDSNE$_{\text{km}}$	0.437±0.000	0.443±0.000	0.247±0.000	0.308±0.000	0.505±0.000	0.458±0.000
SDSNE$_{\text{sc}}$	0.438±0.000	0.436±0.000	0.263±0.000	0.325±0.000	0.426±0.000	0.485±0.000
Reuters						
SC$_{\text{best}}$	0.112±0.012	0.296±0.008	0.059±0.000	0.378±0.007	0.238±0.009	0.329±0.007
LRR$_{\text{best}}$	0.206±0.006	0.397±0.003	0.064±0.005	0.324±0.004	0.240±0.005	0.294±0.005
MCGC	0.263±0.000	0.439±0.000	0.072±0.000	0.388±0.000	0.257±0.000	0.349±0.000
GMC	0.274±0.000	0.472±0.000	0.078±0.000	0.391±0.000	0.262±0.000	0.351±0.000
CGD	0.287±0.005	0.492±0.004	0.082±0.003	0.422±0.003	0.279±0.003	0.367±0.003
O2MAC	0.290±0.026	0.459±0.039	0.243±0.053	0.376±0.028	0.394±0.024	0.550±0.039
SDSNE$_{\text{km}}$	0.388±0.000	0.516±0.000	0.210±0.000	0.457±0.000	0.491±0.000	0.581±0.000
SDSNE$_{\text{sc}}$	0.393±0.000	0.522±0.000	0.237±0.000	0.471±0.000	0.484±0.000	0.587±0.000

Table 1: Clustering performance between SDSNE and other state-of-the-art methods.



(a) t-SNE of $X^{(1)}$

(b) t-SNE of $X^{(2)}$

(c) t-SNE of $X^{(3)}$

(d) t-SNE of $X^{(4)}$

(e) t-SNE of $X^{(5)}$

(f) t-SNE of $X^{(6)}$

(g) t-SNE of the learned H

(h) Heat map of H

Figure 2: The visualization of the raw features and the learned H of SDSNE on MSRC-v1.

k-means is performed well on the linear superable subspace, while SC does well in the local-closed structural data. The distribution of data points determines that they are more suitable for performing the k-means or SC. If their results of SDSNE are comparable, it implies the learned graph representation is of high quality as shown Fig. 2(h). That is the output similarity graph H of MSRC-v1 and it is a good block-diagonal matrix.

4.4 Ablation Study

We also explore multi-layer SDSNE. The following layer after the fist layer is given by

$$H_l^{(v)} \leftarrow H_{l-1}^{(v)} W_l (H_{l-1}^{(v)})^\top \, , \forall v \in \{1, 2, \ldots, n_v\} \, . \qquad (15)$$

where $l > 1$ is the layer index and W_l is the parameter matrix. We add 1 to 15 layers and obtain the same results as only one layer of Eq. (7) . Since we share W between views, we obtain good results. If we use different $P^{(v)}$ from different views in Eq. (7), we obtain little bit lower results than Eq. (7). E.g, we use $P^{(1)}$ and $P^{(2)}$ at the first layer, $P^{(2)}$ and $P^{(3)}$ at the second layer, and $P^{(2)}$ and $P^{(1)}$ at the third for Three Sources dataset. SDSNE aims to learn a graph in which nodes in each connected component fully connect with each other by the same edge weight. Using the same $P^{(v)}$ rather than using cross-view diffusion obtains the target easily. Results of MSRC-v1 are shown in Table 2 .

Method	NMI	ACC	ARI
Cross-view	0.860	0.924	0.824
Multiple Layers	0.872	0.933	0.845
SDSNE$_{\text{sc}}$	0.872	0.933	0.845

Table 2: SDSNE performance using different layers.

5 Conclusion

Since SDSNE aims to learn a graph in which nodes in each connected component fully connect by the same edge weight, the learned graph quality is better than other related methods. The advantages of SDSNE and the reasons why it obtains a better graph are the following: (1) Its layer is based on a diffusion step of a hypergraph. SDSNE learns a graph in which nodes in each connected component connect by the same weight. (2) With co-supervision between different views, the loss function guides SDSNE in achieving the stationary state. When SDSNE achieves the stationary state, the learned graph tends to be a structure in which nodes in each connected component fully connect by the same weight.

We specifically design the Stationary Diffusion State Neural Estimation (SDSNE) approach to the stationary state. SDSNE fuses synergistically multiview structural information by a parameter-shared attentional module and learns to attain multiple graphs eventually. Using the learned graphs, we propose a structure level co-supervised learning strategy which is utilized by SDSNE to achieve the stationary state. We use the structure-level co-supervised learning strategy as the loss function which guides SDSNE in capturing consensus information. The unified consensus graph is obtained by the fusion of all learned graphs. Experiments on six real-world datasets show that SDSNE achieves state-of-the-art results for unsupervised multiview clustering.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant No. 62176108, the Natural Science Foundation of Gansu Province of China under Grant No. 20JR5RA246, the Natural Science Foundation of Qinghai Province of China under Grant No. 2022-ZJ-929, and the Fundamental Research Funds for the Central Universities under Grant No. lzujbky-2021-ct09. The main contribution of ideas and writing was Kun Zhan who led the author team. The four authors worked together cohesively and hardly. We thank Chang Tang for sharing datasets. We thank Chenxiao Zhan for the help of Fig. 1.

References

Bai, S.; Bai, X.; Tian, Q.; and Latecki, L. J. 2017a. Regularized diffusion process for visual retrieval. In *AAAI*, 3967–3973.

Bai, S.; Zhou, Z.; Wang, J.; Bai, X.; Jan Latecki, L.; and Tian, Q. 2017b. Ensemble diffusion for retrieval. In *ICCV*, 774–783.

Blum, A.; and Mitchell, T. 1998. Combining labeled and unlabeled data with co-training. In *COLT*, 92–100.

Brin, S.; and Page, L. 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7): 107–117.

Chopra, S.; Hadsell, R.; and LeCun, Y. 2005. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 539–546.

Chung, F. R. 1997. *Spectral Graph Theory*. American Mathematical Society.

Fan, S.; Wang, X.; Shi, C.; Lu, E.; Lin, K.; and Wang, B. 2020. One2multi graph autoencoder for multi-view graph clustering. In *WWW*, 3070–3076.

Fei-Fei, L.; and Perona, P. 2005. A bayesian hierarchical model for learning natural scene categories. In *CVPR*, volume 2, 524–531.

Gao, H.; Nie, F.; Li, X.; and Huang, H. 2015. Multi-view subspace clustering. In *ICCV*, 4238–4246.

Hadsell, R.; Chopra, S.; and LeCun, Y. 2006. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 1735–1742.

Huang, Z.; Zhou, J. T.; Peng, X.; Zhang, C.; Zhu, H.; and Lv, J. 2019. Multi-view spectral clustering network. In *IJCAI*, 2563–2569.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Kumar, A.; and Daumé, H. 2011. A co-training approach for multi-view spectral clustering. In *ICML*, 393–400.

Kumar, A.; Rai, P.; and Daume, H. 2011. Co-regularized multi-view spectral clustering. In *NeurIPS*, 1413–1421.

Li, Y.; Nie, F.; Huang, H.; and Huang, J. 2015. Large-scale multi-view spectral clustering via bipartite graph. In *AAAI*, 2750–2756.

Liu, G.; Lin, Z.; Yan, S.; Sun, J.; Yu, Y.; and Ma, Y. 2013. Robust recovery of subspace structures by low-rank representation. *TPAMI*, 35(1): 171–184.

Maaten, L. v. d.; and Hinton, G. 2008. Visualizing data using t-SNE. *JMLR*, 9(11): 2579–2605.

Ng, A. Y.; Jordan, M. I.; and Weiss, Y. 2002. On spectral clustering: Analysis and an algorithm. In *NeurIPS*, 849–856.

Nie, F.; Wang, X.; and Huang, H. 2014. Clustering and projected clustering with adaptive neighbors. In *KDD*, 977–986.

Pan, E.; and Kang, Z. 2021. Multi-view contrastive graph clustering. In *NeurIPS*.

Shaham, U.; Stanton, K.; Li, H.; Nadler, B.; Basri, R.; and Kluger, Y. 2018. SpectralNet: Spectral clustering using deep neural networks. In *ICLR*.

Shi, J.; and Malik, J. 2000. Normalized cuts and image segmentation. *TPAMI*, 22(8): 888–905.

Tang, C.; Liu, X.; Zhu, X.; Zhu, E.; Luo, Z.; Wang, L.; and Gao, W. 2020. CGD: Multi-view clustering via cross-view graph diffusion. In *AAAI*, 5924–5931.

Von Luxburg, U. 2007. A tutorial on spectral clustering. *Statistics and Computing*, 17(4): 395–416.

Wang, B.; Jiang, J.; Wang, W.; Zhou, Z.-H.; and Tu, Z. 2012. Unsupervised metric fusion by cross diffusion. In *CVPR*, 2997–3004.

Wang, B.; Mezlini, A. M.; Demir, F.; Fiume, M.; Tu, Z.; Brudno, M.; Haibe-Kains, B.; and Goldenberg, A. 2014. Similarity network fusion for aggregating data types on a genomic scale. *Nature Methods*, 11(3): 333–337.

Wang, H.; Yang, Y.; and Liu, B. 2020. GMC: Graph-based multi-view clustering. *TKDE*, 32(6): 1116–1129.

Xia, R.; Pan, Y.; Du, L.; and Yin, J. 2014. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *AAAI*, 2149–2155.

Xie, J.; Girshick, R.; and Farhadi, A. 2016. Unsupervised deep embedding for clustering analysis. In *ICML*, 478–487.

Zhan, K.; Nie, F.; Wang, J.; and Yang, Y. 2019. Multiview consensus graph clustering. *TIP*, 28(3): 1261–1270.

Zhan, K.; Zhang, C.; Guan, J.; and Wang, J. 2018. Graph learning for multiview clustering. *TCyb*, 48(10): 2887–2895.

Zhang, C.; Fu, H.; Hu, Q.; Cao, X.; Xie, Y.; Tao, D.; and Xu, D. 2018. Generalized latent multi-view subspace clustering. *TPAMI*, 42(1): 86–99.

Zhou, D.; and Burges, C. J. 2007. Spectral clustering and transductive learning with multiple views. In *ICML*, 1159–1166.