

Article

A Spectral Clustering Algorithm for Non-Linear Graph Embedding in Information Networks

Li Ni ^{1,*},[†] , Peng Manman ^{2,†} and Wu Qiang ^{2,†}

¹ College of Information and Electronic Engineering, Hunan City University, Yiyang 413000, China

² College of Information and Engineer, Hunan University, Changsha 410008, China;

pengmanman@hnu.edu.cn (P.M.); wuqiang@hnu.edu.cn (W.Q.)

* Correspondence: lini@hncu.edu.cn; Tel.: +86-159-0737-8098

[†] These authors contributed equally to this work.

Abstract: With the development of network technology, information networks have become one of the most important means for people to understand society. As the scale of information networks expands, the construction of network graphs and high-dimensional feature representation will become major factors affecting the performance of spectral clustering algorithms. To address this issue, in this paper, we propose a spectral clustering algorithm based on similarity graphs and non-linear deep embedding, named *SEG_SC*. This algorithm introduces a new spectral clustering model that explores the underlying structure of graphs through sparse similarity graphs and deep graph representation learning, thereby enhancing graph clustering performance. Experimental analysis with multiple types of real datasets shows that the performance of this model surpasses several advanced benchmark algorithms and performs well in clustering on medium- to large-scale information networks.

Keywords: information networks; spectral clustering; deep embedding



Citation: Ni, L.; Manman, P.; Qiang, W. A Spectral Clustering Algorithm for Non-Linear Graph Embedding in Information Networks. *Appl. Sci.* **2024**, *14*, 4946. <https://doi.org/10.3390/app14114946>

Academic Editor: Danhuai Guo

Received: 11 April 2024

Revised: 4 June 2024

Accepted: 5 June 2024

Published: 6 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Information networks, an essential form for expressing complex relationships between objects, are ubiquitous in our daily lives and work. They include social information networks formed by social media platforms such as WeChat and semantic information networks constituted by webpages on the Internet.

Spectral clustering, detailed in numerous studies [1–4], comprises three main steps: constructing similarity graphs, transforming graphs into low-dimensional vectors, and utilizing the K-means clustering algorithm. For example, Cai et al. [1,2] proved that a similarity graph based on the representative point strategy could be successfully applied to spectral clustering for information networks. Luo et al. [3] used K-means to select representative points from original data and constructed a multi-layer kernel-free similarity graph based on representative points. Shanham et al. [4] used the deep learning network approach to overcome the scalability and generalization issues of low-dimensional embedding in spectral clustering. In summary, at present, the optimization of spectral clustering algorithms mainly consists of the construction of similarity graphs and low-dimensional embedding representations.

With the development of information networks, designing effective algorithms to construct information network graphs to represent their semantic information has become a focal research issue. Traditional methods [5,6] generally adopt high-dimensional sparse vectors, representing the structure and semantic information of information networks through adjacency matrices and involving the design of various application algorithms based on the above. However, high-dimensional sparse representation requires more computational space and time as the network scale increases, leading to poor scalability and accuracy in a graph structure.

The k-nearest neighbor (KNN) graph is a manifold graph construction learning method, which includes methods such as Isomap [7] and local linear embedding [8]. This particular model has good generalization capabilities and can map topological manifolds. If high-dimensional manifolds are embedded in low-dimensional manifold spaces, the properties of Euclidean space still exist locally. The KNN algorithm is known for its stability, simplicity, and efficiency. However, it presents limitations when handling large datasets or high-dimensional data, which can result in increased computational complexity and limited clustering results. In recent years, considering nearest neighbor search methods, D. Cai et al. [1,3,9] improved the KNN graph by introducing a KNN graph clustering algorithm that utilizes representative points. These representative points represent each cluster during computation, and noise has minimal influence on this approach. Moreover, for KNN graph construction, the representative point graph has been applied in spectral clustering. By computing a low-rank approximation of the affinity tensor graph, the initial unified affinity matrix is refined with reliable information from the confidence affinity matrix.

For low-dimensional vector representation, there are many effective methods available. Most models [10,11] that learn graph construction first use linear encoding and then apply singular value decomposition (SVD) [12] for linear dimension reduction. Linear encoding [6,13] produces sparse representation, where each data point is represented as a linear representation of a small number of base vectors. Mathematically, linear encoding is defined as finding two matrices U and Z , whose product can best approximate the data $X \approx UZ$. Large-scale sparse coding clustering (LSC) [9] combines linear encoding, SVD, and K-means into a spectral clustering method. It has a solid theoretical basis and has contributed to significant advancements in clustering research on large-scale data. GraRep [14] is a novel model designed for learning weighted graph vertex representations by generating low-dimensional vectors to represent graphs. However, these methods all involve the use of SVD for linear dimension reduction, and more effective non-linear dimension reduction techniques have yet to be explored.

In recent years, deep learning [4,15] has been widely applied in graph representation learning. Implementing a greedy layer training strategy can improve computational performance. Stacked autoencoders [16] are considered an effective method for mapping from the original representation space to low-dimensional spaces, with the capability to produce highly non-linear projections.

More recently, sparse autoencoders have been used to replace the SVD step in spectral clustering. Shao et al. [17] proposed a deep architecture called DLC for fast graph clustering with linear encoders. Their method combines feature embedding and linear encoding and was implemented through a deep hierarchical architecture. Cao et al. [18] proposed the DNGR model for learning graph representations. The model utilizes pointwise mutual information (PMI) to represent the original graph. It then projects this low-dimensional representation using stacked denoising autoencoders. The authors theoretically and empirically demonstrate the effectiveness of the model in clustering and classification processes. Tian et al. [13] indicated in their study that the computational complexity of autoencoders, $O(n)$, is considerably lower than the characteristic decomposition time, $O(n^3)$, of spectral clustering. They proposed a non-linear spectral clustering algorithm based on deep learning. In this method, the graph is embedded in deep learning through autoencoders, followed by clustering using the K-means algorithm. This study on non-linear mapping spectral clustering algorithms made significant contributions to the field; however, the study presented the following issues: (1) for large-scale data, directly representing graph structures with high-dimensional similarity matrices does not reflect the essential features of graph structures; and (2) the authors simply propose a framework for non-linear spectral clustering algorithms without providing theoretical proof.

In our study, we utilized sparse representation for the graph similarity matrix, which typically enhances clustering accuracy by eliminating noise information that can impact clustering results. In representation learning, deep structures are used to construct features. Deep structures can benefit from hierarchical training for fine-tuning, which helps in finding

improved local solutions to meet the expectation maximization objectives. The method proposed in this paper is called sparse encoder graph spectral clustering (*SEG_SC*).

The *SEG_SC* algorithm proposed in this paper makes the following contributions:

1. In this study, we study the manifold learning similarity matrix and linear low-dimensional mapping of semantic information networks. We define a model structure using the K-means clustering algorithm, similar to spectral algorithms.
2. Theoretically, the results presented in this paper prove that the model adopts a clustering method consistent with the spectral clustering objective function and captures the non-linear features of semantic information networks.
3. Empirically, our results demonstrate that the model can enhance the clustering performance of semantic information networks by capturing meaningful semantic, relational, and structural information. Our experimental results show that the model can effectively complete clustering tasks.

The remainder of this paper is arranged as follows: in Section 2, we review related works; in Section 3, we describe the proposed framework, its theoretical analysis, computation complexity analysis, and the optimization algorithm; in Section 4, we report the experimental results; and finally, we conclude the paper in Section 5.

2. Related Work

2.1. Spectral Clustering

Spectral clustering is an algorithm derived from graph theory and has been widely utilized in clustering. The primary aim is to represent all nodes using a Laplacian matrix. In this matrix, the edge weights between nodes that are far apart are lower; in contrast, those that are closer are higher. The Laplacian matrix is then subjected to eigenvalue decomposition to identify its k-smallest eigenvalues, which are utilized to derive a reduced-dimension representation of the network. Finally, the K-means algorithm is employed for clustering. The process is illustrated in Figure 1.

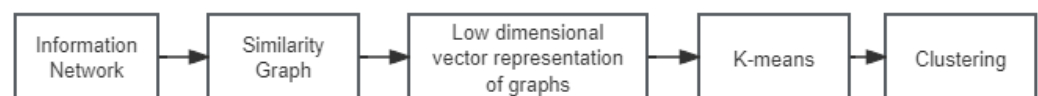


Figure 1. Structural diagram of the spectral clustering algorithm.

There are three methods for constructing the similarity graph in the Laplacian matrix: the adjacency method, the KNN method, and the fully connected method. The adjacency method sets a distance threshold and then measures the similarity between any two nodes using the Euclidean distance. The KNN algorithm traverses all sample points, selecting the nearest k points as neighbors for each sample in order to calculate their similarity. The fully connected method calculates the similarity values between all points.

The objective of spectral clustering is to partition the graph into k-disconnected subgraphs while minimizing the graph cut. The normalized cut criterion [19] is defined as follows:

Definition 1. (Normalized Cut Criterion)

$$\text{Normalized - Cut}(C_1, C_2, \dots, C_k) = \sum_{i=1}^k \frac{\text{cut}(C_i, \overline{C_i})}{\text{vol}(C_i)}, \quad (1)$$

where $\text{vol}(C_i) = \sum_{v \in C_i} d_v$, $C_i \cap C_j = \Phi$, $\cup_{i=1}^k C_i = V$, $\overline{C_i}$ denotes the complement of C_i and $\text{cut}(C_i, \overline{C_i})$ represents the number of connections between the two subgraphs C_i and $\overline{C_i}$.

Solving the normalized cut set is an NP-hard problem; according to *Normalized—Cut* $(C_1, C_2, \dots, C_k) = \sum_{i=1}^k h_i^T L h_i = \sum_{i=1}^k (H^T L H)_{ii} = \text{tr}(H^T L H)$, its approximation can be described as a trace function minimization Formula (2).

$$\arg \min_H \text{tr}(H^T L H) \text{ s.t. } H^T D H = I, \quad (2)$$

where $L = D - S$, $D = \text{diag}(d_1, d_2, \dots, d_n)$, and properties of the Laplacian matrix L are as follows:

- L is a symmetric and positive semi-definite matrix;
- The number of graph connections equals the number of eigenvalues equal to zero, i.e., $\text{Rank}(L) = n - k$;
- For vector $f \in R^n$, there is $f^T L h = \frac{1}{2} \sum_{i,j=1}^n S_{ij} (f_i - f_j)^2$.

This objective function calculates the eigenvectors of the matrix L . According to the matrix computation Theorem 1, H is composed of the eigenvectors corresponding to the k -smallest non-negative singular values of the normalized Laplacian matrix. The final clustering result is obtained by running the K-means algorithm on the graph's embedding matrix H .

Theorem 1. (Eckart–Young–Mirsky Theorem) Given an r -rank matrix P , its SVD: $P = V \Sigma U^T$, where $V^T V = I$, $U^T U = I$ and then

$$\arg \min_{\text{rank}(\vec{P})=k} \|P - \vec{P}\|_F = V \vec{\Sigma} U^T, \quad (3)$$

where $\vec{\Sigma}$; besides only containing the k largest singular values and replacing other singular values with 0, the rest remains the same as the matrix Σ . This theorem indicates that reconstructing the matrix using the k largest singular vectors from singular value decomposition provides the best rank- k approximation of the original matrix.

The effectiveness of spectral clustering depends on the similarity matrix, making it very effective for clustering sparse data, a task that is challenging for the K-means algorithm. Moreover, due to the utilization of dimensionality reduction techniques, spectral clustering handles the complexity of high-dimensional clustering more effectively than traditional clustering methods such as K-means.

2.2. Linear Coding

Linear coding (LC) [9] is a matrix decomposition technique that represents the original data as a combination of a set of base vectors and their corresponding “compressed” representations. Its data representation form is

$$X \approx UZ, \quad (4)$$

where $X = (x_1, x_2, \dots, x_n) \in R^{m \times n}$, $U \in R^{m \times p}$, and the column vectors of U are the basis vectors. $Z \in R^{p \times n}$ and the column vectors of Z represent p -dimensional vectors with respect to the base vectors of U . Therefore, the matrix decomposition of LC is defined as the following optimization function problem:

$$\arg \min_{U, Z} \|X - UZ\|^2. \quad (5)$$

LC adds sparse constraints to each column of Z in the optimization problem described in Formula (5). LC implements a data-sparse representation based on matrix decomposition, where each data point is a linear combination of a few base vectors. This approach allows for a more accurate interpretation of each data point.

2.3. Low-Dimensional Embedding Learning of Information Networks

Dimension reduction is the process of transforming the original high-dimensional attribute space into a lower-dimensional subspace data representation through mathematical transformations. At present, dimension reduction techniques are mainly divided into two categories: linear and non-linear methods.

Classic linear dimension reduction methods include principal component analysis (PCA), multi-dimensional scaling (MDS), linear discriminant analysis (LDA), etc. PCA [20] projects data into a low-dimensional space representation through linear transformation. Its optimal objective function is to minimize Formula (6):

$$\arg \min_W -\text{trace}(W^T X X^T W) \text{ s.t. } W^T W = I. \quad (6)$$

The goal of MDS is to construct a low-dimensional space where the similarity relationships between nodes can be represented. In this space, the distances are consistent with those in the original data, taking into account the global Euclidean distance of samples during sample dimension reduction. LDA minimizes differences between nodes in the same cluster and maximizes differences between nodes in different clusters through linear projection.

Recently, neural information network learning has achieved significant success in various applications, including image classification, speech recognition, and natural language processing. Collobert et al.'s research [21–25] made training in deep architectural structures possible and effective. One of the strategies employed in their research was the greedy layered unsupervised pre-training strategy, whose aim was to learn a useful representation layer by layer and then set the output features as inputs for the next layer. Each layer in this process involves some form of non-linearity, such as non-linear activation functions (e.g., sigmoid or tanh) or regularization of features (e.g., sparse constraints). By stacking these non-linear layers together, deep learning can be used to generate improved representations. The process from the input layer to the hidden layer is called encoding, and the process from the hidden layer to the output layer is called decoding, as illustrated in the model process in Figure 2.

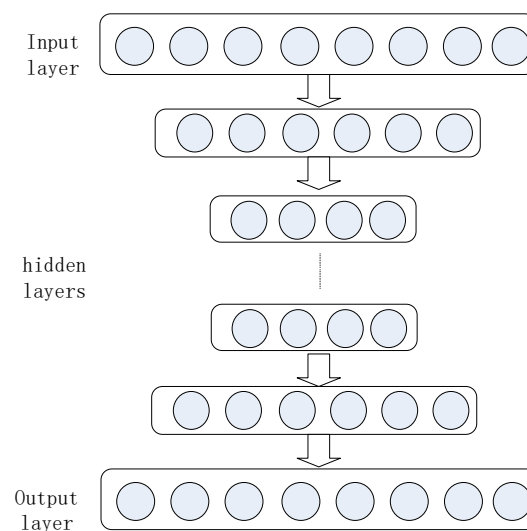


Figure 2. Multi-layer deep architecture.

3. Model Framework

The model proposed in this paper is driven by a three-step scheme of spectral clustering, which includes the construction of similarity graphs, low-dimensional graph embedding, and the K-means clustering algorithm. In our study, we first implemented the

reconstruction of the similarity matrix for spectral clustering, followed by creating a low-dimensional embedded graph. Our spectral clustering model was utilized for large-scale clustering. The basis of this model is to design an effective method for constructing a similarity graph and performing Laplacian matrix eigen decomposition. Specifically, in the similarity graph S , we attempted to design it as symmetric and positive semi-definite, satisfying the following form:

$$S = Z^T Z, \quad (7)$$

where $Z \in R^{p \times n}$ represents the p -dimensional representation of the original graph, $p \ll n$.

The purpose of the *SEG_SC* spectral clustering model is to first represent the information network as a graph $G = (V, S)$, where S is the node similarity graph. Next, it embeds S into a low-dimensional graph embedding matrix R and then identifies a disjoint partition $C = \{C_1, C_2, \dots, C_k\}$ using the K-means algorithm, where k represents the number of clusters.

3.1. Similarity Matrix

KNN [26] is a commonly used manifold learning method. By utilizing a test sample, we integrated the concept of k -nearest neighbor information and distance measurement with the normalized Laplacian matrix of spectral clustering to derive a suitable weighted information network similarity matrix. This matrix can be represented by the following Formula (8):

$$S(v_i, v_j) = \frac{A_{ij}}{\sum_{v_t \in N(v_i)} A_{it}}, \quad (8)$$

where v_t is the k -nearest neighbor node of v_i .

S defines the similarity relationship between each node and all its adjacent nodes; however, this representation is very time-consuming, especially for large-scale data. According to the SC method, for any data point v_i , our approximation can be represented as shown in Formula (9):

$$\hat{v}_i = \sum_{j=1}^p Z_{ji} U_j, \quad (9)$$

where the value of U is randomly selected from the information network graph to create a set of p landmark nodes, generating the landmark node set $U = \{u_1, u_2, \dots, u_p\}$. Calculating the relationship values of U with the k -nearest neighbor nodes, we obtained the similarity graph. According to the Nadaraya–Watson method [27], the sparse representation Z is defined as shown in Formula (10):

$$Z(u_j, v_i) = \frac{K(v_i, u_j)}{\sum_{u_t \in U, t=\{i_1, i_2, \dots, i_k\}} K(v_i, u_t)}, \quad (10)$$

where $K(\cdot)$ is the Gaussian kernel function. Therefore, the definition of the similarity matrix S is as shown in Formula (11):

$$S = Z^T Z. \quad (11)$$

3.2. Non-Linear Low-Dimensional Graph Embedding

First, in the present paper, we analyze the non-linear spectral clustering properties of the single-layer network. Second, the model representation was studied, and we describe the deep embedding graph of the multi-layer network. Third, we analyze the computation process of the multi-layer network based on the objection function. Finally, we analyze its time performance and the algorithm.

3.2.1. Spectral Analysis on Single-Layer Graph

Using the method of graph dimension reduction based on non-linear projection, the projection of sample points in the low-dimensional coordinate system is as follows:

$$R_{ij} = f(W_j^T S_i), \quad (12)$$

where S_i is the i -th row vector of the similarity matrix S , W_j is the j -th row vector of parameter W , and $R_i = (R_{i1}, \dots, R_{id})$ is the low-dimensional embedding landmark of S_i . R_i approximates the reconstruction of X_i , which can be represented as

$$\hat{S}_i = \sum_{j=1}^d g(R_{ij} W_j). \quad (13)$$

The objective formula for the difference in distance between the original sample S_i and the projected reconstructed sample $\hat{S}_i$ is as follows:

$$\begin{aligned} & \sum_{i=1}^n \left\| \sum_{j=1}^d g(R_{ij} W_j) - S_i \right\|_2^2 \\ &= \sum_{i=1}^n g(R_i^T W) g(W^T R_i) - 2 \sum_{i=1}^n g(R_i^T W) S_i + \text{const} \\ &\propto -\text{tr} \left(\sum_{i=1}^n g(W R_i^T) g(R_i W^T) \right), \end{aligned} \quad (14)$$

where $W = (w_1, w_2, \dots, w_d)$; hence, the objective formula solving process can be approximated as

$$\arg \max_{W, R_i} \text{tr} \left(\sum_{i=1}^n g(W R_i^T) g(R_i W^T) \right) \text{ s.t. } W^T W = I. \quad (15)$$

Therefore, the model ensures that the distance to the sample points on the hyperplane is sufficiently close, and the projections of the sample points on the hyperplane are separated as far as possible.

Notably, when the function $g()$ is a linear function, this objective model transforms into the objective function of regular spectral clustering, as shown in Formula (16):

$$\max_S \text{tr}(W S W^T), \text{ s.t. } W^T W = I. \quad (16)$$

Therefore, Formula (15) is defined as the non-linear objective model of spectral clustering.

3.2.2. Deep Graph Embedding on a Multi-Layer Graph

As previously stated, we can consider S as a training set containing n nodes. We consider multi-layer information networks in each layer. The objective formula is as follows:

$$\begin{aligned} & \arg \max_{W^{(l)}, R_i} \sum_{l=1}^m \text{tr} \left(\sum_{i=1}^n g(W^{(l)} (R_i^{(l)})^T) g(R_i^{(l)} (W^{(l)})^T) \right) \\ & \text{ s.t. } W^T W = I. \end{aligned} \quad (17)$$

Assuming the multi-layer information network deep architecture as shown in Figure 2, the input of the first layer is denoted as $X^{(1)} = S$. More hidden layers need to be added to the multi-layer architecture.

It is widely known that multi-layer information networks are divided into an input layer, hidden layers, and an output layer; the structure of the i -th neuron in layer $(l + 1)$ is shown in Figure 3.

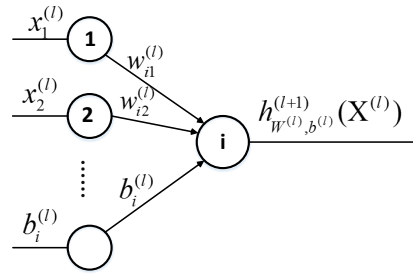


Figure 3. Neuron structure.

The formula for $h_{W^{(l+1)}, b^{(l)}}^{(l+1)}(X^{(l)})$ is:

$$h_{W^{(l+1)}, b^{(l)}}^{(l+1)}(X^{(l)}) = f((W^{(l+1)})^T X) = f\left(\sum_{j=1}^q w_{ij}^{(l)} x_j^{(l)} + b_i^{(l)}\right), \quad (18)$$

where f is the neuron activation function, which is generally a sigmoid or tanh function. $X^{(l)}$ represents the data of the layer l , assuming that the number of neural units in layer l is $q + 1$.

Assuming there are r training samples $X_1, \dots, X_r$ through the deep neural information networks, if the input sample is X_i , the output of the output layer is $h_{W,b}(X_i)$, and the actual output should be Y_i . Therefore, a loss function with regularization terms is defined as

$$J(W, b) = \frac{1}{r} \sum_{i=1}^r J(W, b, X_i, Y_i) + \frac{\lambda}{2} \sum_{l=1}^{n_l-1} \sum_{i=1}^{s_{l+1}} \sum_{j=1}^{s_l} (w_{ij}^{(l)})^2, \quad (19)$$

where $J(W, b, X_i, Y_i) = \frac{1}{2} \|h_{W,b}(X_i) - Y_i\|^2$ is the sum of squared errors of the output layer. Here, n_l denotes the number of layers in the entire information network, and s_l indicates the number of units in layer l , excluding bias units. λ represents the regularization term parameter.

3.2.3. Computation Analysis Based on Deep Graph Embedding

The regularization term stabilizes the algorithm and suppresses overfitting. $J(W, b)$ is a non-convex function; thus, using the gradient descent method can easily lead to local optimization results. In addition, the initialization of each parameter $w_{ij}^{(l)}$ and $b_i^{(l)}$ must be set randomly and not all set to zero. If this is not performed, the output will be the same for any input.

The step formula for parameters $w_{ij}^{(l)}$ and $b_i^{(l)}$ is

$$\begin{aligned} w_{ij}^{(l)} &= w_{ij}^{(l)} - \lambda \frac{\partial J(W, b)}{\partial w_{ij}^{(l)}} \\ b_i^{(l)} &= b_i^{(l)} - \lambda \frac{\partial J(W, b)}{\partial b_i^{(l)}}, \end{aligned} \quad (20)$$

where λ is the step learning parameter.

$$\frac{\partial J(W, b, X_i, Y_i)}{\partial w_{ij}^{(l)}} = \frac{\partial J(W, b, X_i, Y_i)}{\partial I_i^{(l+1)}} \frac{\partial I_i^{(l+1)}}{\partial w_{ij}^{(l)}},$$

where $I_i^{(l+1)} = w_{ij}^{(l)} x_j^{(l)} + b_i^{(l)}$. Assuming $e_i^{(l+1)} = \frac{\partial J(W, b, X_i, Y_i)}{\partial I_i^{(l+1)}}$, then

$$\frac{\partial J(W, b, X_i, Y_i)}{\partial w_{ij}^{(l)}} = x_j^{(l)} e_i^{(l+1)}.$$

Similarly,

$$\frac{\partial J(W, b, X_i, Y_i)}{\partial b_i^{(l)}} = \frac{\partial J(W, b, X_i, Y_i)}{\partial I_i^{(l+1)}} \frac{\partial I_i^{(l+1)}}{\partial b_i^{(l)}} = e_i^{(l+1)},$$

calculate $e_i^{(l+1)}$,

$$e_i^{(l+1)} = \frac{\partial J(W, b, X_i, Y_i)}{\partial I_i^{(l+1)}} = \frac{\partial J(W, b, X_i, Y_i)}{\partial O_i^{(l+1)}} \frac{\partial O_i^{(l+1)}}{\partial I_i^{(l+1)}},$$

where $O_i^{(l+1)} = f(I_i^{(l+1)})$; hence, the output layer $e_i^{(l+1)} = \frac{\partial J(W, b, X_i, Y_i)}{\partial O_i^{(l+1)}} f'(I_i^{(l+1)})$; thus, the entire output vector $E^{(l+1)} = \nabla_O J(W, b, X_i, Y_i) \odot f'(I^{(l+1)})$, where $\nabla_O J(W, b, X_i, Y_i) = \{\frac{\partial J(W, b, X_i, Y_i)}{\partial O_1^{(l+1)}}, \dots, \frac{\partial J(W, b, X_i, Y_i)}{\partial O_n^{(l+1)}}\}$ and $\odot$ represents the Hadamard product. Therefore, the middle layer is

$$\begin{aligned} e_i^{(l+1)} &= \frac{\partial J(W, b, X_i, Y_i)}{\partial I_i^{(l+1)}} \\ &= \sum_{k=1}^r \frac{\partial J(W, b, X_i, Y_i)}{\partial I_k^{(l+2)}} \frac{\partial I_k^{(l+2)}}{\partial O_i^{(l+1)}} \frac{\partial O_i^{(l+1)}}{\partial I_k^{(l+1)}} \\ &= \sum_{k=1}^r e_k^{(l+2)} \frac{\partial (w_k i^{(l+2)} O_i^{(l+1)} + b_k^{(l+2)})}{\partial O_i^{(l+1)}} f'(I_k^{(l+1)}) \\ &= \sum_{k=1}^r e_k^{(l+2)} w_k^{(l+2)} f'(I_k^{(l+1)}), \end{aligned} \quad (21)$$

where $E^{(l+1)} = ((W^{(l+2)})^T E^{(l+2)}) \odot f'(I^{(l+1)})$. Thus, the parameters were trained using the gradient descent method.

$$W^{(l)} = W^{(l)} - \frac{\lambda}{r} \sum_{i=1}^r E^{(l)}(X_i) X^{(l-1)}(X_i) \quad (22)$$

$$b^{(l)} = b^{(l)} - \frac{\lambda}{r} \sum_{i=1}^r E^{(l)}(X_i), \quad (23)$$

where r represents the number of samples.

3.2.4. Computation Complexity Analysis of an Algorithm on a Multi-Layer Graph

The results presented in this paper refer to the deep model as Sparse Encoding Graph Spectral Clustering (SEG_SC), and its implementation process is shown in Algorithm 1.

Suppose we possess the original data $X \in R^{n \times d}$ and use p landmark points; we need $O(pdn)$ to construct the similarity matrix and $O(n)$ to complete the multi-layer graph embedding. K-means requires $O(nkt)$, where t is the number of iterations. In Algorithm 1, the computational complexity is $O(pdn + n + nkt)$.

Algorithm 1 *SEG_SC algorithm*

Require: Original information network X , number of landmark nodes p , and m -layer information network structure;
Ensure: Clustering partitions $C_1, C_2, \dots, C_k$
 Calculate the similarity matrix S according to Formula (11);
for $i = 1$ to n **do**
 1. Build a multi-layer model as shown in Figure 2 based on the similarity matrix S . Let $X^{(1)} = S$ be the input layer;
 2. Train the m -layer model by optimizing the target Formula (17). The node parameter calculation process is illustrated in Formulas (22) and (23), obtaining the activation h_i of the hidden layer;
end for
 Execute K-means for $R^{(\frac{m}{2})}$

4. Experiment

In the following section, we report on the experiments conducted in our study. First, we introduce the datasets used in the experiments, the evaluation metrics, and the benchmark algorithms. Subsequently, we demonstrate the effectiveness of the proposed *SEG_SC* algorithm through empirical testing.

4.1. Experimental Data

To evaluate model performance, in this study we tested the model on various real-world datasets obtained from the UCI Machine Learning Repository and social information network datasets. A brief description of the datasets used can be found below as follows:

- **Wine:** The Wine dataset, sourced from the UCI Machine Learning Repository (Asuncion and Newman, 2007), comprises 178 instances and 13 features. These data are the results of chemical analyses of mature wine grapes. All samples are labeled with three categories of wine. In our study, we established a similarity graph.
- **Pendigit:** This dataset is a handwritten digit dataset consisting of 3498 samples and 16 features created by 44 writers.
- **Mnist:** This dataset consists of handwritten digits and is available on Yann LeCun's webpage. Each image is represented as a 784-dimensional vector and there are 4000 images in total.
- **Blogdata:** This dataset is a social relationship information network provided by blog authors. This information network contains 10,321 nodes, 333,983 edges, and 39 different cluster labels. These labels represent the inferred and provided blog interests of the blog authors.
- **20 newsgroups:** This dataset is a news social network containing 18,846 newsgroup documents and 26,214 features divided into 20 different groups.
- **Seimic:** This dataset is used to classify different types of mobile vehicles, selecting 98,000 samples, 50 features, and 3 classifications.
- **Covtype:** This dataset is a forest cover-type dataset containing 581,012 data, 54 features, and 2 classes. This dataset contains tree observations from four areas of the Roosevelt National Forest in Colorado, USA.

Detailed information on all of these semantic networks is summarized in Table 1.

Table 1. *SEG_SC* experimental data information.

Dataset	Node Count	Feature Count	Cluster Count
Wine	178	13	3
Pendigit	3498	16	10
Mnist	4000	784	10
Blogdata	10,312	1	39
20 newsgroups	18,846	26,214	20

Table 1. Cont.

Dataset	Node Count	Feature Count	Cluster Count
Seimic	98,000	50	3
Covtype	581,012	54	2

4.2. Benchmark Algorithms

To demonstrate the effectiveness of *SEG_SC*, the following methods were used as benchmark algorithms. A brief description of each method is found below as follows:

- PCA [20]: A common matrix decomposition method used for dimensionality reduction or feature extraction. PCA defines a loss function based on the distance between the original sample and the reconstructed sample after projection. PCA is a linear graph embedding method. The clustering result is then obtained using the K-means algorithm.
- SC [2]: Spectral clustering consists of an anchor graph based on set-to-set distances derived from local covariance matrix representation.
- GraphEncoder [13]: A method for learning graph representations. This method explores non-linear graph embedding using deep learning and obtains clustering results by running the K-means algorithm on the embedding.
- DCN (deep clustering network) [28]: The DCN performs joint dimensionality reduction and clustering. Reduction is achieved by training a deep neural information network to approximate any non-linear function.

4.3. Evaluation Criteria

In this paper, we evaluate the performance of the aforementioned clustering algorithms using the accuracy (ACC) and normalized mutual information (NMI) of the clustering results. ACC evaluates the performance of the algorithm by comparing the clustering labels it produces with the actual clustering labels, examining the clustering status of all pairs of nodes to determine if they belong to the same cluster. ACC is defined as follows:

$$ACC = \frac{\sum_{i=1}^n (cluster_label_i == truth_label_i)}{n}, \quad (24)$$

where n represents the number of nodes, $cluster_label_i$ represents the clustering label result of the i -th node by the algorithm, and $truth_label_i$ represents the real clustering label of the i -th node.

NMI is the mutual information entropy between the algorithm's clustering labels and the actual clustering labels. Mutual information entropy considers partitions as the probability distribution of nodes falling into a cluster. It is defined as follows:

$$NMI(\pi_a, \pi_b) = \frac{\sum_{i=1}^{K^a} \sum_{j=1}^{K^b} n_{i,j} \log(\frac{n_{i,j}}{n_i^a n_j^b})}{\sqrt{(\sum_{i=1}^{K^a} n_i^a \log(\frac{n_i^a}{n})) (\sum_{j=1}^{K^b} n_j^b \log(\frac{n_j^b}{n}))}}, \quad (25)$$

where π_a and π_b represent two types of clustering partitioning, $n_{i,j}$ represents the number of nodes belonging to both cluster C_i^a and cluster C_j^b , n_i^a indicates the number of nodes belonging to the i -th cluster C_i^a , and n_j^b indicates the number of nodes belonging to the j -th cluster C_j^b . NMI's value ranges from 0 to 1, with a value of 1 being given when the results of a and b are the same.

4.4. Parameter Settings

In our experiments, each sample from all datasets was normalized to possess the same unit length. We set the neighborhood number in the sparse KNN graph search to five. We detected the relationship between parameters p (the number of landmark points)

and m (depth level lay) and performance indicators ACC and NMI on real small- and medium-sized data from various sources.

Experiments were conducted on the *SEG_SC* model, setting the landmark number parameter p from 200 to 1200, increasing in intervals of 100, with the derived ACC results as shown in Figure 4. From Figure 4, it can be seen that the ACC value increases with the change in landmark value, which is attributed to the random selection method. Evidently, having more landmarks can enhance the final ACC performance; however, the change was not found to be significant in our study. Figure 5 illustrates the relationship between the number of landmarks and NMI. The NMI value generally increases as the number of landmarks increases but with limited magnitude.

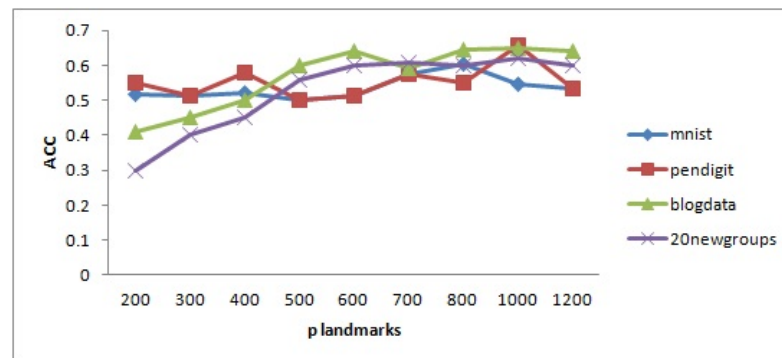


Figure 4. Relationship graph between ACC and the number of landmarks (Parameter p) on the dataset.

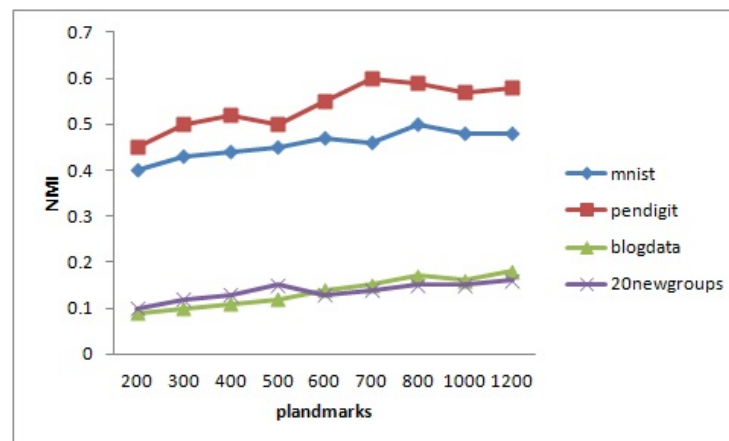


Figure 5. Relationship graph between NMI and the number of landmarks (parameter p) on the dataset.

Examining the settings of the depth structure layer (lay), the ACC value of the first layer of *SGE_SC* on the dataset in Figure 6 was obtained by directly applying K-means to the input normalized sparse similarity graph. It can be observed from Figure 6 that as the layers become deeper, the ACC value increases. This suggests that deep structures play a crucial role in generating high-quality clustering results. In Figure 7, as the number of layers increases, the NMI values for large-scale datasets such as blogdata and 20 newgroup increase.

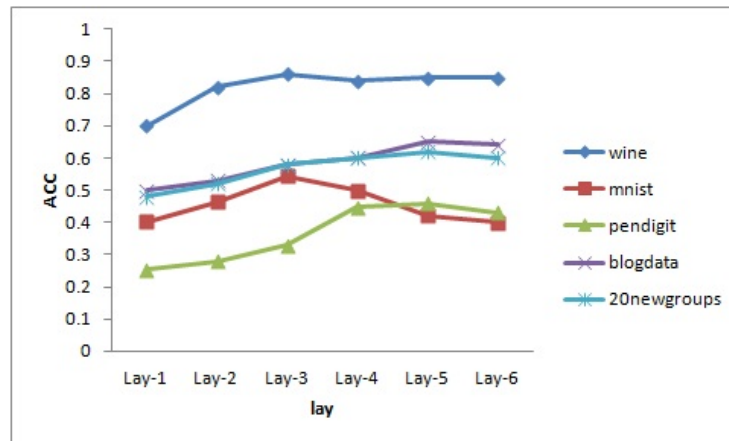


Figure 6. Relationship graph between ACC and depth level lay (parameter m) on the dataset.

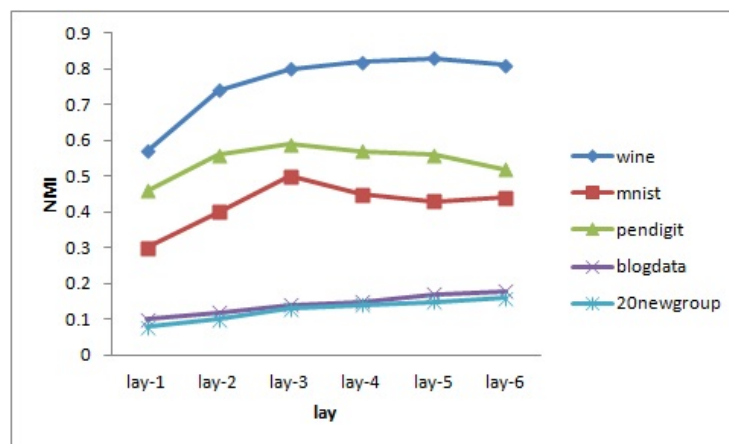


Figure 7. Relationship graph between NMI and depth level lay (parameter m) on the dataset.

In summary, the results presented in this paper suggest using five layers for medium-to large-scale datasets; small-scale datasets, in contrast, do not require more than three layers. The information presented in Table 2 shows the number of nodes in each layer. Moreover, without the loss of generality, in this study we set the number of landmarks to 1000.

Table 2. Neural information network layer structure.

Dataset	Layer Structure
Wine	178-128-64
Pendigit	3498-2400-560
Mnist	4000-1280-640
Blogdata	10,312-5096-2048-1280-640
20 newgroup	18,846-5096-2048-1280-640
Seismic	98,000-59,048-10,080-5090-2948
Covtype	581,012-180,000-98,000-59,000-18,000

All neural networks use the sigmoid function as the activation function for each layer, and the sparse target (the target value for hidden layer activation) is 0.05. Specific parameter information is shown in Table 3.

Table 3. Experimental parameter table.

Parameters	Range	Setpoints
p	200, 300, 400, 500, 600, 700, 800, 1000, 1200	1000
g	sigmoid function, tanh function	sigmoid function
m	1, 2, 3, 4, 5	3 or 5

4.5. Algorithm Comparison

Table 4 presents the average running time of the *SEG_SC* algorithm in comparison with similar deep neural network models such as Graphencoder, indicating that their running times are approximately equal. This result meets the purpose of sparse embedding representation time optimization.

Table 4. *SEG_SC* experimental time.

	Wine	Pendigit	Mnist	Blogdata	20 Newgroups	Seimic	Covtype
Graphencoder	2	250	325	524	620	2794	10,124
<i>SEG_SC</i>	10	228	314	548	612	2641	9942

In terms of cluster performance, Table 5 summarizes the clustering results of seven datasets, indicating that *SEG_SC* offers relatively higher ACC. Table 6 displays the comparison values of various algorithms on large-scale data, demonstrating stronger NMI performance advantages as the data scale increases.

Table 5. Cluster comparison of ACC results.

Algorithm	Wine	Pendigit	Mnist	Blogdata	20 Newgroups	Seimic	Covtype
PCA	0.73	0.54	0.50	0.59	0.59	0.43	0.2
SC	0.81	0.62	0.62	0.58	0.6	0.5	0.27
Graphencoder	0.84	0.6	0.38	0.59	0.51	0.47	0.23
DCN	0.83	0.63	0.83	0.64	0.44	0.65	0.21
<i>SEG_SC</i>	0.86	0.65	0.6	0.65	0.62	0.54	0.24

Table 6. Cluster comparison of NMI results in large-scale data.

Algorithm	Blogdata	20 Newgroups	Seimic	Covtype
Graphencoder	0.12	0.14	0.15	0.07
PCA	0.13	0.14	0.16	0.07
SC	0.13	0.16	0.18	0.08
<i>SEG_SC</i>	0.14	0.16	0.17	0.09

From the results shown in Tables 5 and 6, it can be seen that the model proposed in this paper exhibits superior performance compared to the other models. As shown in Table 5, *SEG_SC* achieved higher ACC. This improvement was significant compared to the other algorithms, attributed to the optimization of graph similarity. In addition, as the graph uses a non-linear projection, it can be observed from Table 6 that in the various types of large-scale datasets, the NMI values tend to be optimal.

The advantage of this model lies in using sparse similarity graphs to replace the original graph's similarity matrix and adopting non-linear encoding schemes instead of feature decomposition. During the final step in particular, the low-dimensional features inputted into K-means are complemented by spectral clustering. Furthermore, the non-linear encoding scheme reduces the dimensionality of the data.

5. Conclusions and Future Research

In the present paper, a spectral clustering model based on deep neural networks named *SEG_SC* is proposed. First, the model utilizes a sparse similarity graph based

on landmark points and the KNN method, introducing a deep graph representation that encodes each vertex into a low-dimensional vector representation. Secondly, the model theoretically demonstrates the effectiveness of *SEG_SC*. Experiments involving the use of real data demonstrate that spectral clustering using deep learning can enhance clustering performance and reduce time consumption in various types of large-scale datasets of information networks.

In future research, we plan to extend this model from general information networks to multi-dimensional information networks, enhancing its scalability on large-scale datasets. In addition, we plan to strengthen its impact by including case studies or real-world applications where the proposed method has been implemented. Conducting such studies will demonstrate its practical viability.

Author Contributions: L.N.: Conceptualization, methodology, software, investigation, formal analysis, writing—original draft preparation, writing—review and editing, funding acquisition; P.M.: visualization, validation, data curation, supervision, funding acquisition; W.Q.: investigation, formal analysis, resources. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key R&D Program of China (Grant 2017YFB0202901, 2017YFB0202905), the Hunan Natural Science Foundation Project (No. 2023JJ50353).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cai, Z.; Li, R.; Wu, H. Learning unified anchor graph based on affinity relationships with strong consensus for multi-view spectral clustering. *Multimed. Syst.* **2023**, *29*, 261–273. [\[CrossRef\]](#)
2. Qin, Y.; Quan, S.; Wei, C.; Ni, W.; Li, K.; Dong, X.; Ye, Y. Spectral clustering with anchor graph based on set-to-set distances for large-scale hyperspectral images. *Int. J. Remote Sens.* **2022**, *43*, 2438–2460. [\[CrossRef\]](#)
3. Luo, X.; He, X.; Yang, X. Fast Spectral Clustering Based on Anchor Point Extraction with Bisecting k-means. *Comput. Eng. Appl.* **2023**, *59*, 74–81.
4. Shaham, U.; Stanton, K.; Li, H.; Nadler, B.; Basri, R.; Kluger, Y. SpectralNet: Spectral Clustering using Deep Neural Networks. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018; pp. 1–20.
5. Ng, A.Y.; Jordan, M.I.; Weiss, Y. On spectral clustering: Analysis and an algorithm. In Proceedings of the International Conference on Neural Information Processing Systems: Natural and Synthetic, Cambridge, MA, USA, 3–8 December 2001; pp. 849–856.
6. Donoho, D.L.; Elad, M. Optimally Sparse Representation in General (Nonorthogonal) Dictionaries via Minimization. *Proc. Natl. Acad. Sci. USA* **2003**, *100*, 2197–2202. [\[CrossRef\]](#) [\[PubMed\]](#)
7. Tenenbaum, J.B.; De, S.V.; Langford, J.C. A global geometric framework for nonlinear dimensionality reduction. *Science* **2000**, *29*, 2319–2323. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Mcivor, R.T.; Humphreys, P.K. A case-based reasoning approach to the make or buy decision. *Integr. Manuf. Syst.* **2000**, *11*, 295–310. [\[CrossRef\]](#)
9. Cai, D.; Chen, X. Large Scale Spectral Clustering Via Landmark-Based Sparse Representation. *IEEE Trans. Cybern.* **2015**, *45*, 1669–1680. [\[PubMed\]](#)
10. Zhang, Y.; Zhou, Z.H. Multilabel dimensionality reduction via dependence maximization. *ACM Trans. Knowl. Discov. Data* **2010**, *4*, 1–21. [\[CrossRef\]](#)
11. Hou, J.; Nayak, R. Robust clustering of multi-type relational data via a heterogeneous manifold ensemble. In Proceedings of the IEEE International Conference on Data Engineering, Seoul, Republic of Korea, 13–17 April 2015; pp. 615–626.
12. Golub, G.H.; Reinsch, C. Singular value decomposition and least squares solutions. *Numer. Math.* **1970**, *14*, 403–420. [\[CrossRef\]](#)
13. Tian, F.; Gao, B.; Cui, Q.; Chen, E.; Liu, T.Y. Learning deep representations for graph clustering. In Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, Québec, QC, Canada, 27–31 July 2014; pp. 1293–1299.
14. Cao, S.; Lu, W.; Xu, Q. GraRep: Learning Graph Representations with Global Structural Information. In Proceedings of the ACM International Conference on Information and Knowledge Management, Melbourne, Australia, 18–23 July 2015; pp. 891–900.
15. Gupta, A.; Katarya, R. Deep embedding for mental health content on socialmedia using vector spacemodel with feature clusters. *Concurr. Comput. Pract. Exp.* **2022**, *34*, 1–21. [\[CrossRef\]](#)

16. Chen, M.; Xu, Z.; Weinberger, K.; Fei, S. Marginalized Denoising Autoencoders for Domain Adaptation. In Proceedings of the 29th International Conference on International Conference on Machine Learning, Edinburgh, UK, 26 June–1 July 2012; pp. 1627–1634.
17. Shao, M.; Li, S.; Ding, Z.; Fu, Y. Deep Linear Coding for Fast Graph Clustering. In Proceedings of the Twenty-Fourth IJCAI Conference, Buenos Aires, Argentina, 25–31 July 2015; pp. 3798–3804.
18. Cao, S.; Lu, W.; Xu, Q. Deep neural networks for learning graph representations. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; pp. 1145–1152.
19. Shi, J.; Malik, J.M. Normalized Cuts and Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2000**, *22*, 888–905.
20. Abdi, H.; Williams, L.J. Principal component analysis. *Wiley Interdiscip. Rev. Comput. Stat.* **2010**, *2*, 433–459. [[CrossRef](#)]
21. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural Language Processing (almost) from Scratch. *J. Mach. Learn. Res.* **2011**, *12*, 2493–2537.
22. Chen, M.; Weinberger, K.; Sha, F.; Bengio, Y. Marginalized denoising auto-encoders for nonlinear representations. In Proceedings of the International Conference on International Conference on Machine Learning, Beijing, China, 21–26 June 2014; pp. 1476–1484.
23. Kang, Z.; Peng, C.; Cheng, Q. Twin Learning for Similarity and Clustering: A Unified Kernel Approach. In Proceedings of the International Conference on Association for the Advancement of Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; pp. 2080–2086.
24. Hinton, G.E.; Zemel, R.S. Autoencoders, minimum description length and Helmholtz free energy. In Proceedings of the International Conference on Neural Information Processing Systems, Denver, CO, USA, 29 November–2 December 1993; Morgan: San Francisco, CA, USA, 1993; pp. 3–10.
25. Bourlard, H.; Kamp, Y. Auto-association by multilayer perceptrons and singular value decomposition. *Biol. Cybern.* **1988**, *59*, 291–294. [[CrossRef](#)] [[PubMed](#)]
26. Altman, N.S. An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *Am. Stat.* **1992**, *46*, 175–185. [[CrossRef](#)]
27. Jones, M.C. Applied Nonparametric Regression. *J. R. Stat. Soc. Ser. C Appl. Stat.* **1992**, *41*, 431–432. [[CrossRef](#)]
28. Yang, B.; Fu, X.; Sidiropoulos, N.D.; Hong, M. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 3861–3870.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.