A Simple Approach to Automated Spectral Clustering

Jicong Fan1,2, Yiheng Tu3,4, Zhao Zhang5, Mingbo Zhao6, Haijun Zhang7

1The Chinese University of Hong Kong, Shenzhen 2Shenzhen Research Institute of Big Data 3Chinese Academy of Science, Beijing 4University of Chinese Academy of Sciences, Beijing 5Hefei University of Technology, Hefei 6Donghua University, Shanghai 7Harbin Institute of Technology, Shenzhen

fanjicong@cuhk.edu.cn yihengtu@gmail.com cszzhang@gmail.com mzhao4@dhu.edu.cn hjzhang@hit.edu.cn

Abstract

The performance of spectral clustering heavily relies on the quality of affinity matrix. A variety of affinity-matrix-construction (AMC) methods have been proposed but they have hyperparameters to determine beforehand, which requires strong experience and leads to difficulty in real applications, especially when the inter-cluster similarity is high and/or the dataset is large. In addition, we often need to choose different AMC methods for different datasets, which still depends on experience. To solve these two challenging problems, in this paper, we present a simple yet effective method for automated spectral clustering. First, we propose to find the most reliable affinity matrix via grid search or Bayesian optimization among a set of candidates given by different AMC methods with different hyperparameters, where the reliability is quantified by the relative-eigen-gap of graph Laplacian introduced in this paper. Second, we propose a fast and accurate AMC method based on least squares representation and thresholding and prove its effectiveness theoretically. Finally, we provide a large-scale extension for the automated spectral clustering method, of which the time complexity is linear with the number of data points. Extensive experiments of natural image clustering show that our method is more versatile, accurate, and efficient than baseline methods.

1 Introduction

Clustering is an important approach to data mining and knowledge discovery. Particularly, spectral clustering [Weiss, 1999; Shi and Malik, 2000; Ng et al., 2002; Von Luxburg, 2007] has superior performance than k-means clustering [Steinhaus and others, 1956], hierarchical clustering [Johnson, 1967], DBSCAN [Ester et al., 1996], and mixtures of probabilistic principal component analyzers [Tipping and Bishop, 1999] in many applications. Roughly speaking, spectral clustering consists of two steps: 1) construct an affinity matrix in which each element denotes the similarity between two data points; 2) perform normalized cut [Shi and Malik, 2000] on the graph corresponding to the affinity matrix. K-nearest neighbors (K-NN) and Gaussian kernel \( k(x, y) = \exp(-\|x-y\|^2/(2\varsigma^2)) \) are two popular methods to construct affinity matrices, where \( k \) and \( \varsigma \) are hyperparameters.

As the performance of spectral clustering heavily relies on the quality of affinity matrix, in recent years, a variety of methods have been proposed to construct or learn affinity matrices for spectral clustering. Many of them are in the framework of self-expressive [Roweis and Saul, 2000; Elhamifar and Vidal, 2013] model, i.e., minimize \( \frac{1}{2} \|X - XC\|^2_F + \lambda \mathcal{R}(C) \). Here the columns of \( X \in \mathbb{R}^{m \times n} \) are the data points drawn from a union of subspaces. \( C \in \mathbb{R}^{n \times n} \) is a coefficient matrix. \( \mathcal{R}(C) \) denotes a regularization operator on \( C \). \( \lambda \) is a hyperparameter to be determined in advance.

*Corresponding author
Elhamifar and Vidal [2013] proposed to use $R(C) = \|C\|_1 := \sum_{i=1}^n \sum_{j=1}^n |c_{ij}|$ under a constraint $\text{diag}(C) = 0$. In [Elhamifar and Vidal, 2013], the affinity matrix for spectral clustering is given by $A = |C| + |C|^\top$. The method is called Sparse Subspace Clustering (SSC). Some theoretical results of SSC can be found in [Wang and Xu, 2013; Soltanolkotabi et al., 2014].

Following the self-expressive framework, [Liu et al., 2013] let $R(C) = \|C\|_*$ (nuclear norm of $C$) and proposed a Low-Rank Representation (LRR) method for subspace clustering. [Lu et al., 2012] and [Pan Ji et al., 2014] used the least squares representation (LSR) model for subspace clustering. A few variants of LRR and SSC can be found in [Patel and Vidal, 2014; Li and Vidal, 2015; Patel et al., 2015; Shen and Li, 2016; Li and Vidal, 2016; Fan and Chow, 2017; Fan et al., 2018; Lu et al., 2018; Pan and Kang, 2021; Kang et al., 2022]. Recently, deep learning methods were also used to learn affinity matrices for spectral clustering [Ji et al., 2017; Zhang et al., 2019b; a; Lv et al., 2021] and have achieved state-of-the-art performance on many benchmark datasets.

One common limitation of these spectral or subspace clustering methods is that they have at least one hyperparameter to determine. In the codes of SSC$^\dagger$ and its variants provided by their authors, there is usually one more thresholding parameter for affinity matrix, which affects the clustering accuracy a lot. In the deep learning clustering methods such as [Ji et al., 2017] and [Zhang et al., 2019b], we need to determine the network structures and regularization parameters, which is much more difficult. Since clustering is an unsupervised learning problem, the hyperparameters cannot be tuned by cross-validation widely used in supervised learning. Thus we have to tune the hyperparameters in spectral clustering by experience, which is difficult when the dataset is quite different from those in our experience and/or the inter-class similarity is high compared to the intra-cluster similarity.

Note that SSC, LRR, and their kernel or deep learning extensions have quadratic or even cubic time complexity (per iteration), which further increases the difficulty of hyperparameter selection in clustering large datasets, though there have been a few works improving the computational efficiency [Peng et al., 2013; Cai and Chen, 2014; Wang et al., 2014; Peng et al., 2015; You et al., 2016a,b; Li and Zhao, 2017; You et al., 2018; Matsushima and Brbic, 2019; Li et al., 2020; Chen et al., 2020; Kang et al., 2020; Fan, 2021; Cai et al., 2022]. On the other hand, different datasets often require different AMC methods, which is hard to tackle by experience.

This paper aims at model and hyperparameter selection for spectral clustering and wants to improve the convenience, accuracy, and efficiency of spectral clustering. Our contributions are as follows.

- We propose a relative-eigen-gap based automated spectral clustering (AutoSC) method. It finds the Laplacian matrix with largest relative-eigen-gap among a set of candidates constructed by different models with different hyperparameters.
- We also implement the AutoSC method via Bayesian optimization. The method can select the possibly best model and optimize the hyperparameters automatically. Note that any AMC methods (e.g. SSC) can be included in the framework of AutoSC.
- To improve the accuracy and efficiency of AutoSC, we propose a new AMC method based on least squares representation and thresholding and prove its effectiveness theoretically.
- We provide an extension for AutoSC to cluster large-scale datasets.

Experiments on seven benchmark image datasets demonstrate the effectiveness of our method. Particularly, our method outperforms state-of-the-art methods of large-scale clustering.

2 Related work

Exploiting eigenvalue information for clustering As the number of zero eigenvalues of a Laplacian matrix is equal to the number of connected components of the graph [Von Luxburg, 2007], a few researchers took advantage of eigenvalue information in spectral clustering [Meila et al., 2005; Meila and Shortreed, 2006; Ji et al., 2015; Hu et al., 2017; Lu et al., 2018]. For instance, [Ji et al., 2015] utilized eigen-gap to determine the rank of the Shape Interaction Matrix. But the method requires determining another hyperparameter $\gamma$ beforehand and needs to perform spectral clustering multiple times. The methods of [Meila et al., 2005; Hu et al., 2017; Lu et al., 2018] are based on

\[C_{ij} = \frac{1}{\sqrt{\lambda_i}} \left( A_{ij} - \frac{1}{n} \sum_{k=1}^n A_{ik} \right),\]

where $\lambda_i$ is the $i$th dominant eigenvalue of $A$.
iterative optimization (need to perform eigenvalue decomposition at every iteration) and hence are not effective in handling large-scale datasets. In addition, the BDR method of [Lu et al., 2018] has two hyperparameters \((\lambda, \gamma)\) to determine by experience, although it outperformed SSC and LRR on some datasets. A comparison is shown in Figure 1.

**Automated machine learning** Automated model and hyperparameter selection for supervised learning have been extensively studied [Hutter et al., 2019]. In contrast, the study for unsupervised learning is very limited. The reason is that in unsupervised learning there is no ground truth or reliable metric to evaluate the performance of algorithms. Concurrently to our work, Poulakis [2020] also attempted to do automated clustering. Specifically, Poulakis [2020] proposed to use meta-learning to select clustering algorithm and use a heuristic combination of some clustering validity metrics such as Silhouette coefficient [Lu et al., 2010] and S_Dbw [Halkidi and Vazirgiannis, 2001] as an objective to maximize via grid search or Bayesian optimization [Jones et al., 1998]. One problem is that these metrics are mainly based on Euclidean distance or densities and hence may not be suitable to evaluate the clustering performance of non-distance or non-density based clustering algorithms. Another one is that there is no unified metric to compare different clustering algorithms.

### 3 Automated Spectral Clustering (AutoSC)

#### 3.1 Preliminary Knowledge

Let \(A \in \mathbb{R}^{n \times n}\) be an affinity matrix constructed from a given data matrix \(X \in \mathbb{R}^{m \times n}\). The corresponding graph is denoted by \(G = (V, E)\), where \(V = \{v_1, \ldots, v_n\}\) is the vertex set and \(E = \{e_1, \ldots, e_l\}\) is the edge set. The degree matrix of a graph \(G\) is defined as \(D = \text{diag}(A1)\), where \(1 = [1, \ldots, 1]^T\). Our goal is to partition the vertices into \(k\) disjoint nonempty subsets \(C_1, \ldots, C_k\). Let \(C = \{C_1, \ldots, C_k\}\). It is expected to find a partition \(C\) that minimizes the following metric.

**Definition 3.1 (MNCut).** The multiway normalized cut (MNCut) [Meila, 2001] is defined as

\[
\text{MNCut}(C) = \sum_{i=1}^{k} \sum_{j \neq i} \frac{\text{Cut}(C_i, C_j)}{\text{Vol}(C_i)},
\]

where \(\text{Cut}(C_i, C_j) = \sum_{u \in C_i, v \in C_j} A_{uv}\) and \(\text{Vol}(C_i)\) denotes the sum of vertex degrees of \(C_i\).

The normalized graph Laplacian matrix is defined as

\[
L = I - D^{-1/2} A D^{-1/2},
\]

where \(I\) is an identity matrix. The normalized graph Laplacian is often more effective than the unnormalized one in spectral clustering (some theoretical justification was given by [Von Luxburg, 2007]). Let \(\sigma_i(L)\) be the \(i\)-th smallest eigenvalue of \(L\). The following claim shows the connection between MNCut\((C)\) and \(L\).

**Claim 3.2.** The sum of the \(k\) smallest singular values of \(L\) quantifies the potential connectivity among \(C_1, \ldots, C_k\): \(\text{MNCut}(C) \geq \sum_{i=1}^{k} \sigma_i(L)\).

The claim can be easily proved by using Lemma 4 of [Meila, 2001]. We defer all proof of this paper to the appendices. Because the multiplicity \(k\) of the eigenvalue 0 of \(L\) equals the number of connected components in \(G\) [Von Luxburg, 2007], we expect to construct an affinity matrix \(A\) from \(X\) such that \(L\) has \(k\) zero eigenvalues. Thus the optimal partition means \(\text{MNCut}(C) = \sum_{i=1}^{k} \sigma_i(L) = 0\).

#### 3.2 Relative Eigen-Gap Guided Search

In practice, we may construct an \(A\) such that \(\sum_{i=1}^{k} \sigma_i(L)\) is as small as possible because guaranteeing zero eigenvalues is difficult. But this is not enough because \(L\) may have \(k + 1\) or more very small or even zero eigenvalues. The second smallest eigenvalue of the Laplacian matrix of a graph \(G\) is called the algebraic connectivity of \(G\) (denoted by \(ac(G)\)) [Fiedler, 1973]. We have \(ac(G) = 0\) if and only if \(G\) is not connected. When \(G\) has \(k\) disjoined components, there are \(k\) algebraic connectivities, denoted by \(ac(C_1), \ldots, ac(C_k)\). Based on this, we have
Claim 3.3. The $k+1^{th}$ smallest eigenvalue of $L$ quantifies the least potential connectivity of partitions $C_1, \ldots, C_k$ of $C$:

$$\min_{1 \leq i \leq k} \text{MNCut}(C_i) \geq \sigma_{k+1}(L).$$

(3)

In other words, $\sigma_{k+1}(L)$ measures the difficulty in segmenting each of $C_i$ into two subsets. Hence, when $\sigma_{k+1}(L)$ is large, the partitions $C_1, \ldots, C_k$ are stable. Based on Claim 3.2 and Claim 3.3, we may construct an $A$ that has small $\sum_{i=1}^{k} \sigma_i(L)$ and large $\sigma_{k+1}(L)$ simultaneously, by solving

maximize $\theta \sigma_{k+1}(L) - \frac{1}{k} \sum_{i=1}^{k} \sigma_i(L)$, 
subject to $L = I - D^{-1/2} AD^{-1/2}$, $A = f_\theta(X)$. 

(4)

where $f_\theta : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \times n}$ is a function with parameter $\theta$, e.g. $A = [\exp(-\|x_i-x_j\|^2/(2\epsilon^2))] - I$. It is difficult to solve (4) because of the composition of $f_\theta$, symmetric normalized Laplacian, and eigenvalue decomposition. On the other hand, in (4), we have to choose $f$ in advance, which requires domain expertise or strong experience because different dataset usually needs different $f$.

Note that different $f$ can result in very different distributions of eigenvalues and the small eigenvalues are sensitive to $f, \theta$, and noise. Hence the objective in (4) is not effective to compared different $f$ and $\theta$. In this paper, we define a new metric relative-eigen-gap as follows

$$\text{reg}(L) := \frac{\sigma_{k+1}(L) - \frac{1}{k} \sum_{i=1}^{k} \sigma_i(L)}{\frac{1}{k} \sum_{i=1}^{k} \sigma_i(L) + \epsilon},$$

(5)

where $\epsilon$ is a small constant (e.g. $10^{-6}$) to avoid zero denominator. $\text{reg}(L)$ is not sensitive to the scale of the small eigenvalues. Therefore, instead of (4), we propose to solve

maximize $\text{reg}(L)$,
subject to $L = I - D^{-1/2} AD^{-1/2}$, $A = f_\theta(X)$,

(6)

where $F$ is a set of pre-defined functions and $\Theta$ is a set of hyperparameters. In fact, (6) is equivalent to choosing one $A$ (or $L$) from a set of candidates constructed by different $f$ with different $\theta$, of which the relative-eigen-gap is largest. The best $\theta$ can be found using grid search (or even random search). For convenience, we call the method AutoSC-GD. Table 1 shows a few examples of $f$ and its parameters. One may use a weighted sum of affinity matrices given by different $f$, like [Huang et al. 2012], which however will introduce more hyperparameters.

Table 1: A few examples of $f$ and its $\theta$ for AMC (AASC: [Huang et al. 2012])

<table>
<thead>
<tr>
<th>$f$</th>
<th>K-NN</th>
<th>$\epsilon$-neighborhood</th>
<th>Gaussian kernel</th>
<th>SSC</th>
<th>LRR</th>
<th>LSR</th>
<th>KSSC</th>
<th>AASC</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\theta$</td>
<td>$K$</td>
<td>$\epsilon$</td>
<td>$\sigma$</td>
<td>$\lambda$</td>
<td>$\lambda$</td>
<td>$\lambda$</td>
<td>$\lambda$</td>
<td>$\sigma_1, \sigma_2, \ldots$</td>
</tr>
</tbody>
</table>

The following theorem shows the connection between $\text{reg}(L)$ and the stability of the clustering $C$.

Theorem 3.4. Let $C$ and $C'$ be two partitions of the vertices of $G$, where $|C| = |C'| = k$. Define the distance between $C$ and $C'$ as $\text{dist}(C, C') = 1 - \frac{1}{k} \sum_{i \in C} \sum_{j \in C'} \frac{(\text{Vol}(C_i \cap C'_j))^2}{\text{Vol}(C_i) \text{Vol}(C'_j)}$. Suppose $\eta \epsilon \geq \sum_{i=1}^{k} \sigma_i(L) \geq k \epsilon$ and $\text{reg}(L) > (k-1)\eta/2$. Let $\delta = \max(MNCut(C) - \sum_{i=1}^{k} \sigma_i(L), MNCut(C') - \sum_{i=1}^{k} \sigma_i(L))$. Then

$$\text{dist}(C, C') < \frac{1.5 \delta^{-1}}{\text{reg}(L)/(1-k)\eta/2}.$$ 

(7)

It indicates that when $\text{reg}(L)$ is large and $\delta$ is small, the partitions $C$ and $C'$ are close to each other. Thus, the clustering has high stability. When $\sum_{i=1}^{k} \sigma_i(L) = k \epsilon$, we have $\text{dist}(C, C') < \frac{6 \delta}{\sigma_{k+1}(L) - k \epsilon}$, which means the larger $\sigma_{k+1}(L)$ the more stable clustering.

This theorem is a modified version of Theorem 1 in [Meila et al. 2005], which is for the eigen-gap $\sigma_{k+1} - \sigma_k$ of $L$. Here we consider $\text{reg}(L)$ instead.
Compare reg($L$) with [Meila and Shortreed, 2006] It is worth noting that [Meila and Shortreed, 2006] proposed to minimize $J(L) := \frac{1}{L} \sum_{i=1}^{L} \sigma_i(L) + \alpha (\sigma_k(L) - \sigma_{k+1}(L))^2$ to find a good affinity matrix for spectral clustering. Although reg($L$) and $J(L)$ seem similar, they are essentially different. As mentioned before, different AMC methods may lead to different scales for the small eigenvalues, which makes it difficult to compare different AMC methods using $J(L)$. In addition, $J(L)$ has a hyperparameter $\alpha$ to determine beforehand, which violates our goal of searching models and hyperparameters. A comparative study ($\alpha \leq 0$) is in Section 4.1 (Table 3).

3.3 AutoSC via Bayesian Optimization

Bayesian optimization (BO) [Jones et al., 1998] has become a promising tool for hyperparameter optimization of supervised machine learning algorithms [Snoek et al., 2012; Klein et al., 2017]. Given a black-box function $g : \mathcal{X} \to \mathbb{R}$, BO aims to find an $x^* \in \mathcal{X}$ that globally minimizes $g$ and usually has three steps. The first step is finding the most promising point $x_{t+1} \in \arg\max_x a_{p(g)}(x)$ by numerical optimization, where $a_{p(g)} : \mathcal{X} \to \mathbb{R}$ is an acquisition function (e.g., Expected Improvement) relying on an prior $p(g)$ (e.g. Gaussian processes [Williams and Rasmussen, 2006]).

The second step is evaluating the expensive and possibly noisy function $y_{t+1} \sim g(x^*) + \mathcal{N}(0, \sigma^2)$ and adding the new sample $(x_{t+1}, y_{t+1})$ to the observation set $D_t = \{(x_1, y_1), \ldots, (x_t, y_t)\}$. The last step is updating $p(g)$ and $a_{p(g)}$ using $D_{t+1}$.

As an alternative to the grid search for (6), we can maximize reg($L$) via BO. Suppose we have a set of different AMC models, i.e., $\mathcal{F} = \{f_1, f_2, \ldots, f_M\}$. For $i = 1, 2, \ldots, M$, let $g_i(\theta^{(i)}):= -\text{reg}(L(f_i(\theta^{(i)}|\mathcal{X})))$,

where $\theta^{(i)}$ denotes the hyperparameters in $f_i$ and $\mathcal{X}$ denotes the dataset. Then we use BO to find

$$\theta_*^{(i)} = \arg\min_{\theta^{(i)} \in S^{(i)}} g_i(\theta^{(i)}) ,$$

where $S^{(i)}$ denotes the set of constraints. Finally we get the best model with its best hyperparameters

$$f_*(\theta_*^{(i)}|\mathcal{X}), \quad \text{where } * = \arg\min_{1 \leq i \leq M} g_i(\theta^{(i)}).$$

For convenience, we denote the method by AutoSC-BO. Note that $\mathcal{F}$ can include any AMC methods such as those in Table 1 and even DSC [Ji et al., 2017] (see Appendix D.5).

In AutoSC-BO, we use Expected Improvement (EI) acquisition function

$$a_{\text{EI}}(s|D_t) = \mathbb{E}_p[\max(g_{\text{min}} - g(s), 0)],$$

where $g_{\text{min}}$ is the best function value known. The closed-form formulation is

$$a_{\text{EI}}(s|D_t) = (g_{\text{min}} - \mu) \Phi \left( \frac{g_{\text{min}} - \mu}{\sigma} \right) + \phi \left( \frac{g_{\text{min}} - \mu}{\sigma} \right),$$

where $\mu = \mu(s|D_t, \theta_K)$ and $\sigma = \sigma(s|D_t, \theta_K)$ are the mean and variance of the Gaussian process, $\phi$ and $\Phi$ are standard Gaussian cumulative density function and probability density function respectively, and $\theta_K$ denotes the hyperparameters of the Gaussian process. For the covariance function, we use the automatic relevance determination (ARD) Matérn $5/2$ kernel [Matérn, 2013]

$$k_{5/2}(s, s') = \theta_0 \left( 1 + \sqrt{5r^2(s, s')} + \frac{5}{3} r^2(s, s') \right) \exp \left( -\sqrt{5r^2(s, s')} \right),$$

where $r^2(s, s') = \sum_{j=1}^{d} (s_j - s'_j)^2 / \theta_j^2$.

3.4 Discussion on AMC Methods for AutoSC and LSR with Thresholding

In AutoSC, the size of searching space is $|\mathcal{F}| \times \prod_j |\theta_j|$, which should be large enough to include effective models and their hyperparameters. A large number of works have shown that the self-expressive models [Elhamifar and Vidal, 2013; Liu et al., 2013; Lu et al., 2013; Ji et al., 2018; Ji et al., 2017] often outperform other AMC models such as Gaussian kernel. However, the self-expressive model based
AMC methods often require iterative optimization and has at least quadratic time complexity per iteration, which leads to huge time cost in AutoSC. Although LSR [Lu et al., 2012] has closed-form solution, the clustering accuracy is not satisfactory [Lu et al., 2018]. In this work, we will show that LSR with a simple post-processing operation can be a good AMC method and can outperform SSC, LRR, and BDR [Lu et al., 2018]. Specifically, the LSR model is given as

\[
\text{minimize } \frac{1}{2} \| X - XC \|_F^2 + \frac{\lambda}{2} \| C \|_F^2,
\]

of which the closed-form solution is \( C = (X^T X + \lambda I)^{-1} X^T X \). Let \( \text{diag}(C) = 0 \) and \( C \leftarrow |C| \), the affinity matrix can be constructed as \( A = (C + C^T)/2 \). One problem is that the off-diagonal elements of \( A \) are dense (leading to a connected graph), which can result in low clustering accuracy. Therefore, we propose to truncate \( C \) by keeping only the largest \( \tau \) elements of each column of \( C \). Nevertheless, it is not easy to determine \( \tau \) beforehand. When \( \tau \) is too small, the corresponding graph will have \( k + 1 \) or more connected components. When \( \tau \) is too large, the corresponding graph will have \( k - 1 \) or less connected components. However, \( \tau \) can be automatically determined by our AutoSC-GD and AutoSC-BO.

In the case that the data have some low-dimensional nonlinear structures, the similarity between pair-wise columns of \( X \) cannot be well recognized by the linear regression (13). Therefore, we also consider the following nonlinear regression model

\[
\text{minimize } \frac{1}{2} \| \phi(X) - \phi(X)C \|_F^2 + \frac{\lambda}{2} \| C \|_F^2,
\]

where \( \phi \) denotes a nonlinear feature map performed on each column of the matrix, i.e. \( \phi(X) = [\phi(x_1), \ldots, \phi(x_n)] \). In [14], letting \( \phi \) be some feature map induced by a kernel function \( k(\cdot, \cdot) \) (e.g. polynomial kernel \( k(x_i, x_j) = (x_i^T x_j + \beta)^d \) and Gaussian kernel), we get the kernel LSR (KLSR):

\[
\text{minimize } \frac{1}{2} \text{Tr} (K - 2KC + C^T KC) + \frac{\lambda}{2} \| C \|_F^2,
\]

where \( K = \phi(X)^T \phi(X) \) and \( [K]_{ij} = k(x_i, x_j) \). The closed-form solution is \( C = (K + \lambda I)^{-1} K \). The post-processing is the same as that for the solution of LSR.

We introduce the following property, a necessary condition of successful subspace clustering, which is similar to the one used in [Wang and Xu, 2013; Soltanolkotabi et al., 2014].

**Definition 3.5** (Subspace Detection Property). A symmetric affinity matrix \( A \) obtained from \( X \) has subspace detection property if for all \( i \), the nonzero elements of \( A_i \) correspond to the columns of \( X \) in the same subspace as \( x_i \).

For convenience, let \( \pi(i) \) be the index of the subspace \( x_i \) belongs to and \( C_j \) be the index set of the columns of \( X \) in subspace \( j \). We consider the following deterministic model.

**Definition 3.6** (Deterministic Model). The columns of \( X \in \mathbb{R}^{m \times n} \) are drawn from a union of \( k \) different subspaces and are further corrupted by noise, where \( \text{dim}(S_1 \cup \cdots \cup S_k) = d < m \leq n \). Let \( X = US\Sigma V^T \) be the SVD of \( X \), where \( \Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n) \) and \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \). Let \( \gamma = \sigma_{d+1}/\sigma_d \). Denote \( v_i = (v_{i1}, \ldots, v_{im}) \) the \( i \)-th row of \( V \) and let \( \bar{v}_i = (v_{i1}, \ldots, v_{id}) \). Suppose the following conditions hold:

1. For every \( i \in [n] \), the \( \tau \)-th largest element of \( \{|v_i^T \bar{v}_j| : j \in C_{\pi(i)}\} \) is greater than \( \alpha \); 2. \( \max_{i \in [n]} \max_{j \in C_{\pi(i)} \setminus C_{\pi(i)} \setminus \pi(i)} |v_i^T \bar{v}_j| \leq \beta \); 3. \( \max_{i,j,l} |v_{il} \bar{v}_{jl}| \leq \mu \).

Then the following theorem verifies the effectiveness of (13) followed by the truncation (thresholding) operation in subspace detection.

**Theorem 3.7.** Suppose \( X \) is given by Definition 3.6 and \( C \) is given by (13) with

\[
\frac{(\rho - \sqrt{\rho^2 - 4(2\mu d - \Delta)(2\mu m - 2\mu d - \Delta)}) \sigma_d^2}{4\mu d - 2\Delta} < \lambda < \frac{(\rho + \sqrt{\rho^2 - 4(2\mu d - \Delta)(2\mu m - 2\mu d - \Delta)}) \sigma_d^2}{4\mu d - 2\Delta},
\]

where \( \rho = 2\mu \gamma^2 - \Delta(1 + \gamma^2) \) and \( \Delta = \alpha - \beta \). Then the \( C \) truncated by \( \tau \leq \bar{\tau} \) has the subspace detection property.

\footnote{\( \gamma \) measures the noise level, \( \beta \) is dominated by the difference between subspaces, and \( \mu \) quantifies the incoherence in the singular vectors.}
Finally, we perform k-means on the feature matrix proposed by Li et al. Adam [Kingma and Ba, 2014] to solve NSE. It is worth noting that NSE is different from the method where the algorithm 2 (see Appendix B). Note that the time complexity of AutoSC+NSE is $O(dmn + ds^2)$, where $d$ depends on the specific AMC method. When $s \ll n$, the time complexity of AutoSC+NSE is linear with the number of data points $n$. Proposition C.4 in Appendix C.2 shows that a small number of hidden nodes in NSE are sufficient to make the clustering succeed.

3.5 AutoSC+NSE for Large-Scale Data

Since the time and space complexity of AutoSC are quadratic with $n$, it cannot be directly applied to large-scale datasets. To solve the problem, we propose to perform Algorithm 1 on a set of landmarks of the data (denoted by $\hat{X}$) to get a $\hat{Z}$. The landmarks can be generated by k-means or randomly. Then we regard $\hat{Z}$ as a feature matrix and learn a map $g : \mathbb{R}^m \rightarrow \mathbb{R}^k$ from $\hat{X}$ to $\hat{Z}$. According to the universal approximation theorem [Sonoda and Murata, 2017] of neural networks, we approximate $g$ by a two-layer neural network and solve

$$\min w_1, w_2, b_1, b_2 \frac{1}{2s} \| \hat{Z} - W_2 \text{ReLU}(W_1 \hat{X} + b_1 1^n_T) - b_2 1^n_T \|_F^2 + \frac{\gamma}{2} (\|W_1\|_F^2 + \|W_2\|_F^2),$$

where $W_1 \in \mathbb{R}^{d \times m}$, $W_2 \in \mathbb{R}^{k \times d}$, $b_1 \in \mathbb{R}^d$, and $b_2 \in \mathbb{R}^k$. Since $A$ is sparse, $k$ is often less than $m$, and a neural network is used, we call (17) Neural Sparse Embedding (NSE). We use mini-batch Adam [Kingma and Ba, 2014] to solve NSE. It is worth noting that NSE is different from the method proposed by Li et al. [2020]. In Li et al. [2020], the regression is for an affinity matrix, which leads to high computational cost. The network learned from (17) is applied to $\hat{X}$ to extract a $k$-dimensional feature matrix $Z$:

$$Z = \hat{g}(\hat{X}) = W_2 \text{ReLU}(W_1 \hat{X} + b_1 1^n_n) + b_2 1^n_n.$$  

Finally, we perform k-means on $Z$ to get the clusters. The procedures are summarized into Algorithm 2 (see Appendix B). Note that the time complexity of AutoSC+NSE is $O(dmn + ds^2)$, where $d$ depends on the specific AMC method. When $s \ll n$, the time complexity of AutoSC+NSE is linear with the number of data points $n$. Proposition C.4 in Appendix C.2 shows that a small number of hidden nodes in NSE are sufficient to make the clustering succeed.

4 Experiments

We test our AutoSC on Extended Yale B Face [Kuang-Chih et al., 2005], ORL Face [Samaria and Harter, 1994], COIL20 [Nene et al., 1996], AR Face [Martinez and Kak, 2001], MNIST [LeCun et al., 1998], Fashion-MNIST [Xiao et al., 2017], GTSRB [Stallkamp et al., 2012], subsets and extracted features of MNIST and Fashion-MNIST. The descriptions for the datasets are in Appendix D.1. Our MATLAB codes are available at https://github.com/jicongfan/Automated-Spectral-Clustering.

4.1 Intuitive validation of AutoSC

Performance of Relative-Eigen-Gap First, we use LSR and KLSR to show the effectiveness of the proposed reg($L$). Figure 1(i) shows an intuitive example of the performance of LSR and KLSR for Large-Scale Data.
in clustering a subset of the Extended Yale B database, where for (15) we use Gaussian kernel with 
\[ \kappa = \frac{1}{n} \sum_{ij} \| x_i - x_j \|. \] 
We see that: 1) in LSR and KLSR, for a fixed \( \lambda \) (or \( \tau \)), the \( \tau \) (or \( \lambda \)) with larger \( \text{tr}(L) \) provides higher clustering accuracy; 2) for a fixed \( \lambda \) and a fixed \( \tau \), if LSR has a larger \( \text{tr}(L) \), its clustering accuracy is higher than that of KLSR, and vice versa. We conclude that roughly a larger \( \text{tr}(L) \) indeed leads to a higher clustering accuracy, which is consistent with our theoretical analysis in Section 3.2.

(i) Clustering accuracy and \( \text{tr}(L) \) (rescaled between 0 and 1) and clustering accuracy of LSR and KLSR on the first 5 subjects of the Extended Yale Face B database.

Figure 1: Examples about \( \text{tr}(L) \), clustering accuracy, and hyperparameters on Extended Yale B.

Now we show the superiority of LSR and KLSR compared to a few important AMC methods. Figure 1(ii) shows the clustering accuracy of SSC [Elhamifar and Vidal, 2013], LRR [Liu et al., 2013], and BDR-B [Lu et al., 2018] with different hyperparameters and our method Auto-GD with LSR and KLSR (detailed by Algorithm 1 in the supplementary material) on the Extended Yale Face B subset. SSC and BDR-B are sensitive to the value of \( \lambda \), especially for the relatively difficult task, say Figure 1(ii-b). LRR is not sensitive to the value of \( \lambda \) but its accuracy is low. LSR and KLSR are more accurate and efficient than other methods.

The performance of AutoSC-BO We show the performance of AutoSC-BO with many AMC methods such as SSC [Elhamifar and Vidal, 2013] and LSR. Taking the KLSR model (15) with polynomial kernel as an example, the parameters are \( \theta = (\lambda, b, q, \tau) \) and the constraints are given by \( S = \{ \lambda \in \mathbb{R} : \lambda_{\text{min}} \leq \lambda \leq \lambda_{\text{max}} ; b \in \mathbb{R} : b_{\text{min}} \leq b \leq b_{\text{max}} ; q \in \mathbb{Z}^+ : q_{\text{min}} \leq q \leq q_{\text{max}} ; \tau \in \mathbb{Z}^+ : \tau_{\text{min}} \leq \tau \leq \tau_{\text{max}} \} \). More details are in Appendix D.5 Shown in Table 2 larger \( \text{tr}(L) \) corresponds to higher clustering accuracy and KLSR with polynomial kernel (the optimal \( q \) is 1) performs best. Figure 2 shows the performance of KSSC [Patel and Vidal, 2014] and KLSR in each iteration of AutoSC-BO.

<table>
<thead>
<tr>
<th>AMC</th>
<th>( \epsilon )-neigh</th>
<th>Polynomial kernel</th>
<th>Gaussian kernel</th>
<th>KSSC (Gauss)</th>
<th>KSSC (Poly)</th>
<th>KLSR (Gauss)</th>
<th>KLSR (Poly)</th>
</tr>
</thead>
<tbody>
<tr>
<td>reg_{max}</td>
<td>0.776</td>
<td>1.294</td>
<td>1.307</td>
<td>0.892</td>
<td>1.388</td>
<td>2.217</td>
<td>2.379</td>
</tr>
<tr>
<td>Accuracy</td>
<td>0.325</td>
<td>0.389</td>
<td>0.393</td>
<td>0.584</td>
<td>0.859</td>
<td>0.963</td>
<td>0.966</td>
</tr>
</tbody>
</table>

Relative-Eigen-Gap versus Eigen-Gap We compare the proposed relative-eigen-gap (\( \text{tr}(L) \)) with eigen-gap (denoted by \( \text{tr}(L) \)), and the regularizer \( J(L, \alpha) \) proposed by [Meila and Shortreed, 2006] with different \( \alpha \). Note that in AutoSC, we need to minimize \( J(L, \alpha) \) instead. The clustering results of AutoSC-GD are reported in Table 3 (details about the datasets are in Section 4). We see that our \( \text{tr}(L) \) outperforms \( \text{tr}(L) \) and \( J(L, \alpha) \) in all cases except AR.
(i) reg$(L)$ and clustering accuracy of KLSR and KSSC in each iteration of BO.

(ii) reg$(L)$ and hyperparameters of KLSR and KSSC in each iteration of BO.

Figure 2: AutoSC-BO with KSSC and KLSR on the first 10 subjects of YaleB Face dataset.

Table 3: The comparison of AutoSC-GD with reg$(L)$, eg$(L)$, and J$(L, \alpha)$

<table>
<thead>
<tr>
<th></th>
<th>YaleB</th>
<th>ORL</th>
<th>COIL20</th>
<th>AR</th>
<th>MNIST</th>
<th>F-MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>reg$(L)$</td>
<td>0.790</td>
<td>0.619</td>
<td>0.805</td>
<td>0.663</td>
<td>0.562</td>
<td></td>
</tr>
<tr>
<td>J$(L, 0)$</td>
<td>0.823</td>
<td>0.750</td>
<td>0.804</td>
<td>0.726</td>
<td>0.516</td>
<td></td>
</tr>
<tr>
<td>J$(L, -0.1)$</td>
<td>0.818</td>
<td>0.768</td>
<td>0.826</td>
<td>0.718</td>
<td>0.518</td>
<td></td>
</tr>
<tr>
<td>J$(L, -1)$</td>
<td>0.812</td>
<td>0.769</td>
<td>0.817</td>
<td>0.722</td>
<td>0.523</td>
<td></td>
</tr>
<tr>
<td>J$(L, -10)$</td>
<td>0.804</td>
<td>0.768</td>
<td>0.832</td>
<td>0.731</td>
<td>0.539</td>
<td></td>
</tr>
<tr>
<td>J$(L, -100)$</td>
<td>0.788</td>
<td>0.769</td>
<td>0.817</td>
<td>0.735</td>
<td>0.545</td>
<td></td>
</tr>
<tr>
<td>reg$(L)$</td>
<td>0.897</td>
<td>0.795</td>
<td>0.782</td>
<td>0.786</td>
<td>0.755</td>
<td></td>
</tr>
</tbody>
</table>

4.2 Comparative studies of AutoSC and baselines

First we compare AutoSC with SSC, LRR [Liu et al., 2013], LSR [Lu et al., 2012], EDSC [Pan Ji et al., 2014], KSSC, SSC-OMP [You et al., 2016b], BDR-Z [Lu et al., 2018], and BDR-B [Lu et al., 2018] on six smaller datasets. The clustering accuracy and time cost are reported in Table 4. AutoSC-GD and AutoSC-BO outperformed other methods significantly in almost all cases. SSC-OMP and AutoSC-GD are more efficient than SSC, LRR, EDSC, and KSSC. The time cost of AutoSC-BO is much higher than that of AutoSC-GD because the former optimizes all hyperparameters including $\lambda$, $\tau$, and kernel parameters (e.g. $b, q$).

Table 4: Clustering performance on the six small datasets. For the MNIST-1k and Fashion-MNIST-1k, we report the average results of 20 trials because the subset is formed randomly. AutoSC chose LSR for Yale B and AR and chose KLSR for others datasets. The NMI results are in Table 7 (See Appendix D.3).

<table>
<thead>
<tr>
<th></th>
<th>SSC</th>
<th>LRR</th>
<th>LSR</th>
<th>EDSC</th>
<th>KSSC</th>
<th>SSC-OMP</th>
<th>BDR-Z</th>
<th>BDR-B</th>
<th>AutoSC-GD</th>
<th>AutoSC-BO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yale B acc</td>
<td>0.723</td>
<td>0.643</td>
<td>0.592</td>
<td>0.806</td>
<td>0.649</td>
<td>0.768</td>
<td>0.596</td>
<td>0.719</td>
<td>0.897</td>
<td>0.909</td>
</tr>
<tr>
<td></td>
<td>273.8</td>
<td>928.1</td>
<td>7.3</td>
<td>58.6</td>
<td>464.3</td>
<td>8.9</td>
<td>368.8</td>
<td>368.8</td>
<td>19.1</td>
<td>78.3</td>
</tr>
<tr>
<td>ORL acc</td>
<td>0.711</td>
<td>0.762</td>
<td>0.680</td>
<td>0.712</td>
<td>0.707</td>
<td>0.665</td>
<td>0.739</td>
<td>0.735</td>
<td>0.795</td>
<td>0.803</td>
</tr>
<tr>
<td></td>
<td>2.7</td>
<td>8.8</td>
<td>0.5</td>
<td>2.0</td>
<td>2.6</td>
<td>0.4</td>
<td>3.9</td>
<td>3.9</td>
<td>2.3</td>
<td>20.5</td>
</tr>
<tr>
<td>COIL20 acc</td>
<td>0.871</td>
<td>0.729</td>
<td>0.695</td>
<td>0.739</td>
<td>0.912</td>
<td>0.638</td>
<td>0.713</td>
<td>0.791</td>
<td>0.782</td>
<td>0.878</td>
</tr>
<tr>
<td></td>
<td>61.8</td>
<td>221.2</td>
<td>1.4</td>
<td>15.4</td>
<td>100.6</td>
<td>2.5</td>
<td>86.8</td>
<td>86.8</td>
<td>7.6</td>
<td>39.2</td>
</tr>
<tr>
<td>AR acc</td>
<td>0.718</td>
<td>0.709</td>
<td>0.665</td>
<td>0.673</td>
<td>0.726</td>
<td>0.699</td>
<td>0.745</td>
<td>0.731</td>
<td>0.786</td>
<td>0.826</td>
</tr>
<tr>
<td></td>
<td>317.5</td>
<td>1220.6</td>
<td>14.5</td>
<td>69.1</td>
<td>627.4</td>
<td>57.6</td>
<td>578.7</td>
<td>578.7</td>
<td>43.4</td>
<td>130.6</td>
</tr>
<tr>
<td>MNIST-1k acc</td>
<td>0.596</td>
<td>0.513</td>
<td>0.554</td>
<td>0.536</td>
<td>0.577</td>
<td>0.542</td>
<td>0.576</td>
<td>0.578</td>
<td>0.615</td>
<td>0.619</td>
</tr>
<tr>
<td></td>
<td>(0.054)</td>
<td>(0.037)</td>
<td>(0.041)</td>
<td>(0.035)</td>
<td>(0.053)</td>
<td>(0.038)</td>
<td>(0.037)</td>
<td>(0.043)</td>
<td>(0.041)</td>
<td>(0.038)</td>
</tr>
<tr>
<td></td>
<td>24.9</td>
<td>69.1</td>
<td>0.8</td>
<td>5.2</td>
<td>32.8</td>
<td>1.3</td>
<td>26.9</td>
<td>26.9</td>
<td>2.5</td>
<td>21.7</td>
</tr>
<tr>
<td>Fashion-MNIST-1k acc</td>
<td>0.553</td>
<td>0.515</td>
<td>0.563</td>
<td>0.544</td>
<td>0.548</td>
<td>0.566</td>
<td>0.574</td>
<td>0.563</td>
<td>0.581</td>
<td>0.584</td>
</tr>
<tr>
<td></td>
<td>(0.025)</td>
<td>(0.014)</td>
<td>(0.023)</td>
<td>(0.017)</td>
<td>(0.016)</td>
<td>(0.034)</td>
<td>(0.019)</td>
<td>(0.031)</td>
<td>(0.025)</td>
<td>(0.021)</td>
</tr>
<tr>
<td></td>
<td>24.1</td>
<td>68.5</td>
<td>0.7</td>
<td>5.1</td>
<td>35.9</td>
<td>1.2</td>
<td>25.7</td>
<td>25.7</td>
<td>2.6</td>
<td>22.7</td>
</tr>
</tbody>
</table>
We compare our method with LSR, LSC-K [Chen and Cai, 2011], SSC-OMP [You et al., 2016], and S^5C [Matsushima and Brbic, 2019], and S^3COMP-C [Chen et al., 2020] on the larger datasets. The parameter settings are in Appendix D.6. Table 5 shows the clustering accuracy and standard deviation of 10 repeated trials on the raw-pixel data of MNIST and Fashion-MNIST. Table 6 shows the results on MNIST and Fashion-MNIST features and GTSRB. Our methods have the highest clustering accuracy in every case. Note that if NSE is ablated, the accuracy of AutoSC-GD on MNIST(features)-10k, -20k, and -30k are 0.9772, 0.9783 and 0.9864 respectively, higher than those of AutoSC-GD+NSE, though the time costs increased.

It is worth mentioning that, to the best of our knowledge, the deep clustering method of [Zhang et al., 2019a] has SOTA performance on Yale B (acc=0.98) and ORL (acc=0.89), the method proposed by [Zhang et al., 2019b] has SOTA performance on Fashion-MNIST (acc=0.72), the method proposed by [Mahon and Lukasiewicz, 2021] has SOTA performance on MNIST (acc=0.99), and our method has SOTA performance on GTSRB. Nevertheless, we focus on automated spectral clustering.

Table 5: Clustering accuracy and time cost (second) on MNIST and Fashion MNIST. “—” means the computation is out of memory.

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST-10k</th>
<th>Fashion-MNIST-10k</th>
<th>GTSRB</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSR</td>
<td>0.583(0.007)</td>
<td>0.561(0.008)</td>
<td>0.871(0.0510)</td>
</tr>
<tr>
<td>LSC-K</td>
<td>0.652(0.037)</td>
<td>0.571(0.025)</td>
<td>0.871(0.0510)</td>
</tr>
<tr>
<td>SSC-OMP</td>
<td>0.433(0.014)</td>
<td>0.509(0.0038)</td>
<td>—</td>
</tr>
<tr>
<td>S^5C</td>
<td>0.646(0.045)</td>
<td>0.565(0.021)</td>
<td>—</td>
</tr>
<tr>
<td>S^3COMP-C</td>
<td>0.623(0.028)</td>
<td>0.569(0.0024)</td>
<td>—</td>
</tr>
<tr>
<td>AutoSC-GD</td>
<td>0.687(0.035)</td>
<td>0.596(0.021)</td>
<td>—</td>
</tr>
<tr>
<td>AutoSC-GD+NSE</td>
<td>0.679(0.034)</td>
<td>0.576(0.011)</td>
<td>—</td>
</tr>
<tr>
<td>AutoSC-BO+NSE</td>
<td>0.679(0.034)</td>
<td>0.572(0.012)</td>
<td>—</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>MNIST acc</th>
<th>Fashion-MNIST acc</th>
<th>GTSRB acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSR</td>
<td>0.862</td>
<td>0.613</td>
<td>0.871</td>
</tr>
<tr>
<td>LSC-K</td>
<td>0.815</td>
<td>0.379</td>
<td>0.871</td>
</tr>
<tr>
<td>SSC-OMP</td>
<td>0.782</td>
<td>0.599</td>
<td>0.871</td>
</tr>
<tr>
<td>S^5C</td>
<td>0.963</td>
<td>0.596</td>
<td>0.871</td>
</tr>
<tr>
<td>S^3COMP-C</td>
<td>0.962</td>
<td>0.569</td>
<td>0.871</td>
</tr>
<tr>
<td>AutoSC-GD</td>
<td>0.977</td>
<td>0.639</td>
<td>0.978</td>
</tr>
<tr>
<td>AutoSC-GD+NSE</td>
<td>0.974</td>
<td>0.646</td>
<td>0.988</td>
</tr>
<tr>
<td>AutoSC-BO+NSE</td>
<td>0.974</td>
<td>0.646</td>
<td>0.988</td>
</tr>
</tbody>
</table>

5 Conclusion

We have proposed an automated spectral clustering method. Extensive experiments showed the effectiveness and superiority of our methods over baseline methods. The efficiency improvement is from the closed-form solutions of the least squares regressions. The accuracy improvement is from the effectiveness of LSR and KLSR with thresholding and the automation of model and hyperparameter selection. One limitation of this work is that we only considered automated spectral clustering while there are many other clustering methods (e.g. [Fan, 2021]) not relying on affinity matrix.

Acknowledgments

The work of Jicong Fan was supported in part by the Youth program 62106211 of the National Natural Science Foundation of China and the research funding T001202100002 of Shenzhen Research Institute of Big Data. The work of Yiheng Tu was supported in part by the National Natural Science Foundation of China under Grant no. 32171078. The work of Zhao Zhang was supported in part by the National Natural Science Foundation of China under Grant no. 62072151 and Anhui Provincial Natural Science Fund for the Distinguished Young Scholars (2008085130). The work of Mingbo Zhao was supported in part by the National Natural Science Foundation of China under Grant no. 61971121. The work of Haijun Zhang was supported in part by the National Natural Science Foundation of China under Grant no. 61972112 and no. 61832004, the Guangdong Basic and Applied Basic Research Foundation under Grant no. 2021B1515020088.

The authors appreciate the reviewers’ comments and time.
References


14
Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default [TODO] to [Yes], [No], or [N/A]. You are strongly encouraged to include a justification to your answer, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [N/A]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]