Generative Adversarial Attack on Ensemble Clustering

Chetan Kumar, Deepak Kumar and Ming Shao
University of Massachusetts Dartmouth, Dartmouth, MA, USA
{ckumar, dkumar2, mshao}@umassd.edu

Abstract

Adversarial attack on learning tasks has attracted substantial attention in recent years; however, most existing works focus on supervised learning. Recently, research has shown that unsupervised learning, such as clustering, tends to be vulnerable due to adversarial attack. In this paper, we focus on a clustering algorithm widely used in the real-world environment, namely, ensemble clustering (EC). EC algorithms usually leverage basic partition (BP) and ensemble techniques to improve the clustering performance collaboratively. Each BP may stem from one trial of clustering, feature segment, or part of data stored on the cloud. We have observed that the attack tends to be less perceivable when only a few BPs are compromised. To explore plausible attack strategies, we propose a novel generative adversarial attack (GA2) model for EC, titled GA2EC. First, we show that not all BPs are equally important, and some of them are more vulnerable under adversarial attack. Second, we develop a generative adversarial model to mimic the attack on EC. In particular, the generative model will simulate behaviors of both clean BPs and perturbed key BPs, and their derived graphs, and thus can launch effective attacks with less attention. We have conducted extensive experiments on eleven clustering benchmarks and have demonstrated that our approach is effective in attacking EC under both transductive and inductive settings.

1. Introduction

Data clustering has been extensively studied in the past few decades [16, 32, 40, 41, 44, 46] and has many potential contributions in the field of computer vision and pattern recognition [5, 17, 30]. Considerable clustering algorithms have been developed for real-world applications, but no single algorithm has proved effective on various datasets. Thus, choosing appropriate clustering algorithm for a given task becomes challenging [15]. While there is limitation presented by a single clustering, the idea of combining results from multiple clusterings on a given data for stability and improved performance becomes prevalent. Multiple clustering may stem from different algorithms, different data sources, or features [12, 35], which is termed as Ensemble Clustering (EC).

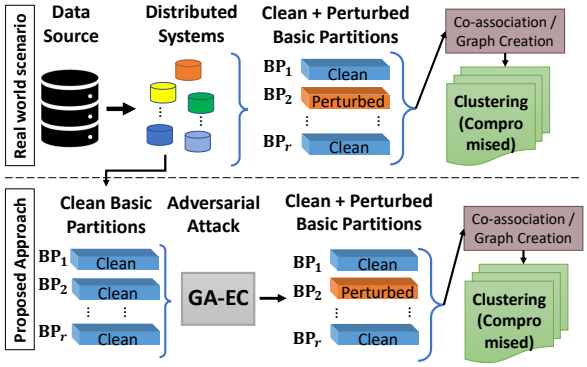


Figure 1: Top part: It shows a real world scenario where BPs that stem from distributed system or different sources may be perturbed and thus compromise the overall ensemble clustering performance. Bottom part: Our proposed Generative Adversarial Attack on Ensemble Clustering (GA2EC) model mimics the above scenario.

Recently, considerable research has undergone on EC due to its promising performance in addressing noisy and bizarre data. Most importantly, it works well in distributed computing that does not require original data or features in clustering and thus benefits privacy [15, 22–24, 26, 38, 42]. In EC, a single clustering result is referred to as basic partition (BP), whereas final clustering assembles all BPs. One of the most popular way to achieve such ensemble is using co-association (CA) matrix, which encodes co-occurrence of data in the same cluster [25].

However, with recent advances in deep learning, the impact of adversarial samples and learning models has become significant, especially for high-dimensional visual data [14,36]. While most adversarial modelings are relevant to supervised learning tasks [1, 20, 21, 29, 48], it has been demonstrated that unsupervised methods including clustering could also be a victim of such attacks. Preliminary adversarial research work has been done recently for conventional clustering tasks [6, 7], and yet the adversarial model-

ing for EC is still underexplored. In fact, EC is more vulnerable as BPs from different sources/features/trials may be easily attacked without notice. In this paper, we hypothesize that ensemble clustering is subject to adversarial attack due to minor changes in BPs, which ultimately compromises the CA matrix used in the final clustering task, as shown in top part of Fig 1. It depicts a real world scenario where some sources in a distributed environment can generate perturbed BPs that can have significant impact on the overall EC. Motivated by this, we developed a Generative Adversarial Attack on Ensemble Clustering (GA2EC) model which perturbs the key BPs while minimizes the change to the original BPs.

Note our adversarial attack is completely unsupervised and capable of simulating attack through generative modeling and thus applicable to even unseen test data. To that end, first, we empirically prove that BPs are vulnerable under supervised adversarial attacks through gradient-based approaches and attacking key (selective) BPs are more effective. Second, we propose a generative model to simulate the gradient-based attack through pseudo labels. The generative model manages to generate both compromised EC features and graph given the clean data. The generator is built with a variational autoencoder (VAE) [18] as the infrastructure, and driven by both conventional VAE loss and perturbed EC feature and graph losses. It balances different losses to minimize the perturbation level while secures the effectiveness of the adversarial attack. Last, extensive experiments on eleven popular visual datasets validate our generative model and adversarial attacks on CA-based EC.

In brief, the contributions of this paper are:

- We examine the feasibility of ensemble clustering (EC) and the possibility of attacking selective basic partitions (BPs) for an effective adversarial attack.
- A generative model is developed to simulate the adversarial attack on EC through a gradient-based approach in a complete unsupervised learning fashion, where both EC features and induced graphs are compromised.
- Quantitative and qualitative results on eleven visual datasets have demonstrated the effectiveness of our model on attacking EC, with extensive discussions on parameters analysis, and model insights.

2. Background and Observations

In this section, we will introduce the ensemble clustering (EC) with a co-association (CA) matrix and then present the preliminary results of using a gradient-based adversarial attack on EC. It will demonstrate the vulnerability of EC and the feasibility of the adversary under this context. The pipeline of this section is shown in Fig. 2. Table 1 summarizes frequently used variables throughout the paper.

Table 1: Summary of notations.

Variable	Description
$x \in \mathbb{R}^D$	A single sample from dataset X
$X \in \mathbb{R}^{N\times D}$	Input dataset with N samples
$H \in \mathbb{R}^{N\times r}$	Basic partitions (BPs) matrix
$H_i \in \mathbb{R}^N$	A basic partition in H
$\hat{H} \in \mathbb{R}^{N\times r}$	Perturbed H by gradient attack
$H' \in \mathbb{R}^{N\times r}$	Perturbed H using our method
$B \in \mathbb{R}^{N\times d}$	Binary indicator matrix
$S \in \mathbb{R}^{N\times N}$	Co-association matrix
$L \in \mathbb{R}^N$	Ground truth labels
$L' \in \mathbb{R}^N$	Pseudo labels
$N \in \mathbb{R}^1$	Total number of samples in X
$D \in \mathbb{R}^1$	Number of features in x
$r \in \mathbb{R}^1$	Number of basic partitions
$d \in \mathbb{R}^1$	Total number of indicators in B
$m \in \mathbb{R}^1$	Number of selected key BPs

2.1. Ensemble Clustering

Ensemble clustering (EC), also known as consensus clustering, refers to combining the output of multiple basic partitions (BP) or base clusters by mapping the BPs output into a co-association (CA) matrix or similarity matrix, followed by the final clustering [11, 35]. Basic partition refers to a single run of any clustering algorithm, which results in a vector containing the cluster *ids* for each data vector, and in ensemble clustering, each basic partition vector is used as a feature vector [22–24]. The merit of EC over an individual clustering algorithm is it may not require access to the original data features for final clustering, which in turn protects the privacy [43, 47]. In addition, it is more robust and stable than individual clustering as the latter usually generates varied results due to random or different initialization of clusters centers [3, 39, 50].

Given a dataset $X = \{x_1, \cdots, x_N\} \in \mathbb{R}^{N\times D}$ where x_i refers to the *i*-th data sample, N refers to the total number of objects in X, and D is the dimension of feature. In addition, let $H = \{H_1, H_2, ..., H_r\} \in \mathbb{R}^{N\times r}$ be the BP matrix with K_i clusters in H_i, and r the number of BPs. Note in reality, there are several options for generating BPs, including different data sources, feature representations, trials, and clustering algorithms. In this paper, we practice with multiple trials, and the developed algorithms are also applied to other BPs options. Once BPs were generated, they are expected to combine and create a *consensus clustering*. There are two typical ways for this purpose: (1) co-association (CA) matrix [23]; (2) utility function [25, 27, 37]. We stick to the CA matrix based consensus clustering, as recently it was claimed that CA matrix has significant success in ensemble clustering than utility function [23].

We follow a typical way of creating CA matrix through the binary indicator matrix $B = [B_1, B_2, B_3, ..., B_r] \in$

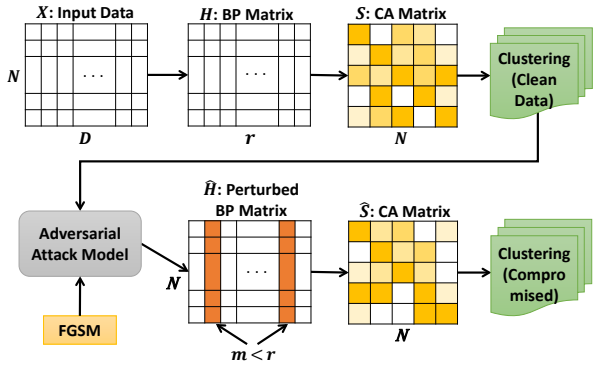


Figure 2: Illustration of adversarial attack on EC.

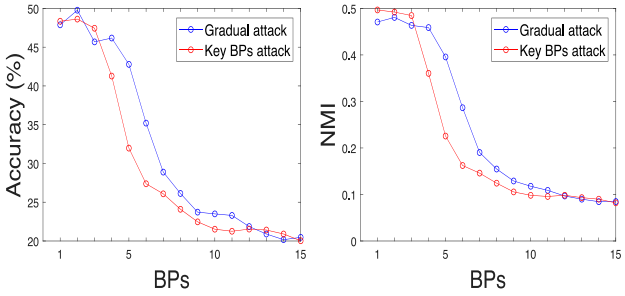


Figure 3: Gradual vs. Key BPs attack on USPS dataset. The selected number of BPs varies from 100 to 1,500.

$\mathbb{R}^{N\times d}$ where $d = \sum_j K_j$, K_j is the number of unique clusters in the j-th trial (BP), and $B_j \in \mathbb{R}^{N\times K_j}, j \in [1, r]$ is a sub-matrix of B for the j-th trial. Specifically, in each row of B_j, the clustering result is presented by 1-of-K_j coding. Thus, the CA matrix S can be computed through $S = BB^T$, which can be further leveraged as an $N \times N$ adjacency matrix or generally a graph. We may either apply Kmeans on the column space, or graph clustering on S for the final clustering task.

2.2. Adversarial Attack in Clustering

Although deep learning methods have offered state-of-the-art performance for visual data, it has been identified vulnerable to adversarial attacks [14, 36], and in classification tasks in particular [1, 48]. One of the popular methods in this line is *gradient based attack*, which adds a small noise to the original data according to the sign of the gradient of the model loss with respect to the inputs, e.g., Fast Gradient Sign Method (FGSM) [14]. One upfront challenge is the lack of label information under the clustering context. To that end, we propose to create pseudo labels first, which could then be used in FGSM model. In our work, Kmeans is used due to its simplicity to generate pseudo labels, but other efficient clustering algorithms are also applicable here. Pseudo labels are used to keep the proposed GA2EC approach fully unsupervised. Once the pseudo label set L' is built, the perturbed BP will be generated by:

$$\hat{H} = H + \eta, \quad \eta = \epsilon sign(\nabla_H J(\psi, H, L')), \qquad (1)$$

where ψ is model parameters, ∇_H is the gradient of the loss function $J(\cdot)$ regarding the input H, and ϵ is a model parameter to define the magnitude of the perturbation applied on the input features. Finally, the perturbation η generated in Eq. 1 is added to the original BPs to create the adversarial samples and fool the clustering models. Note that other recent gradient attack methods could be applied here for similar purposes.

2.3. Vulnerability of BPs

In this section, we will implement an FGSM + pseudo labels based adversarial attack to demonstrate the feasibility, as shown in Fig. 2. First, we use an off-the-shelf clustering method Kmeans to generate pseudo labels for the dataset. Second, we follow the conventional adversarial attack pipeline and use the BPs matrix H as the basic input. Third, upon the efficacy, selected perturbed BPs and clean BPs are blended for EC tasks.

In this experiment, we choose $r = 1500$ BPs generated from different clustering trials. The reasons are two-fold: first, it can provide enough EC features for better representation; second, both empirical observations and theoretical results show that more trials can secure stable performance. We follow the attack model shown in Eq. (1) to obtain the perturbed BPs. It should be noted that not all BPs were equally attacked by FGSM, which enlightens us that fewer BPs could be selected for a less noticeable but effective attack. To that end, we rank the residual γ_j between the j-th original BP and j-th perturbed BP, namely:

$$\gamma_j = \|H_j - \hat{H}_j\|_2, \qquad (2)$$

and choose the first $m (m < r)$ BPs based on their residual values. We will combine the selected m BPs with the rest $(r - m)$ clean BPs to formulate the perturbed BPs $\hat{H}$. To demonstrate the effectiveness of $\hat{H}$, we conduct two experiments with different ways of selecting BPs in building $\hat{H}$. In the first one, we gradually choose m perturbed BPs based on their default order in H, while in the second one, we choose m perturbed BPs based on the ranks of their residual γ. In Fig. 3, it can be seen that the selected key BPs attack is more effective than the gradual attack, and a large margin can be identified when $m = 500$ based on the clustering accuracy and NMI.

3. Methodology

In this section, we will describe our Generative Adversarial Attack on Ensemble Clustering (GA2EC) framework,

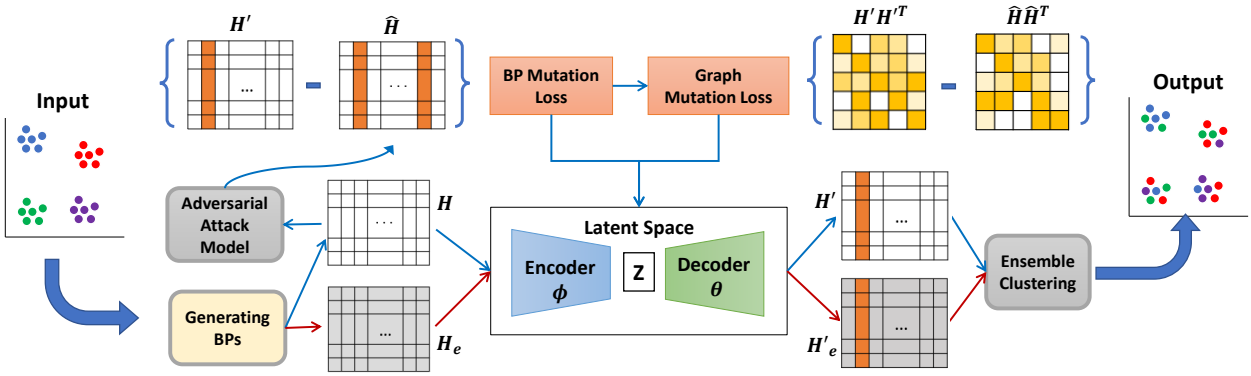


Figure 4: The complete pipeline of generative adversarial attack on ensemble clustering (GA2EC). Note the blue path shows the transductive setting, while the red path shows the inductive setting, with unseen dataset and its BPs matrix H_e.

which is illustrated in Fig. 4. GA2EC is built upon a deep generative model, and in particular variational autoencoder (VAE) [18] in this paper. Dedicated losses consists of (1) BP reconstruction loss, (2) BP mutation loss, and (3) graph mutation loss, are developed to mimic the behaviors of adversarial samples from the adversarial model, including both feature vectors and graphs. The newly generated adversarial samples are expected to render poor clustering performance. These details are presented in the following subsections.

3.1. Generative Model to Mimic Adversary

The BPs's vulnerability has been proved and now we plan to mimic such behaviors through a dedicated generative model with two aims:

- Offer less noticeable adversarial change on BPs
- Extend the generative model to unseen data

The two aims were motivated by the fact that BPs under attack may deviate too much from the clean data, and thus can be easily detected. The generative model may mitigate this issue and ensure attack effectiveness. In addition, it is supposed to extend to unseen data sampled from the same data sources or following similar distributions.

To that end, we leverage VAE as our infrastructure to mimic the adversarial behavior of BPs, while offering a "mild" attack. Essentially, VAE reconstructs clean BPs with two additional constraints. First, we create a BP mutation loss to allow for minor mutations in potential BPs that lead to the malicious effect. Second, we incorporate graph mutation loss to enable further engagement of BP mutations in the final clustering. The two losses together with the conventional loss of VAE secure the performance, while making the attack less noticeable in a measurable way.

BP Reconstruction Loss. Given the BPs matrix H, we will first normalize the data into [0,1] before feeding it to the deep generative model VAE. The basic VAE structure includes an encoder and a decoder, where the former encodes H into latent space z and the latter decodes the z into H'. Similar to conventional autoencoder modeling, H' and H are expected to be similar under certain measurement, namely, Frobenius norm:

$$\mathcal{L}_R = \|H' - H\|_{\mathrm{F}}^2, \tag{3}$$

where $\mathcal{L}_R$ is referred to as reconstruction loss in the paper. In addition, the latent space is learned from input data and assumed to be parameterized by Gaussian distributions $N(0, I)$. In training, Kulback-Leibler (KL) divergence is used to regularize between learned and standard Gaussian distributions, which is usually defined as follows:

$$\begin{aligned} \mathcal{L}_{KL}(\phi, \theta, H) = \quad & D_{KL}(q_\theta(z|H)\|p_\theta(z)) \\ - \quad & \mathbb{E}_{q_\theta(z|H)}(\log p_\theta(H|z)), \end{aligned} \tag{4}$$

where ϕ and θ are encoder and decoder network parameters and $p_\theta(H|z)$ represents the posterior distribution. Finally, the VAE model is learned by integrating two loss functions above for data generation.

BP Mutation Loss. Original VAE aims to model the underlying distribution of latent vector z and thus faithfully generate new samples under the same distribution. To mimic the adversary imposed on H, we leverage the perturbed BPs from the previous adversarial model. Basically, $m(m < r)$ are selected according to residual defined in Eq. (2). The noisy BPs are then concatenated with the clean BPs to formulate the intended perturbed BPs termed as $\hat{H}$. We define the following loss function to encourage a minor mutation of the reconstructed BPs H', namely,

$$\mathcal{L}_P = \|H' - \hat{H}\|_{\mathrm{F}}^2. \tag{5}$$

Graph Mutation Loss. In EC, CA matrix is used for the final clustering task and we propose to add another loss function regarding the CA matrix. Recall in adversarial attack, the perturbed BPs matrix $\hat{H}$, and the derived binary

Algorithm 1: Generative Adversarial Attack on Ensemble Clustering

Part I: Ensemble Clustering for Pseudo Labels
Input: X
Output: $L^{'}$ (Pseudo Labels)
while $i \leq r$ **do**
  Compute basic partitions: $H_i = \text{Kmeans}(X)$;
end
$H = [H_1, H_2, \cdots H_r]$;
Binary matrix: $B \leftarrow H$ by one-hot coding;
CA matrix: $S \leftarrow BB^T$;
Ensemble clustering: $L' \leftarrow \text{Kmeans}(S)$;

Part II: Data Perturbation
Input: $\{H, L^{'}, m\}$
Output: $\hat{H}$ (Perturbed Data)
Initial $\hat{H}$: Run FGSM $(H, L^{'})$ by Eq. (1);
$\hat{H}$ with key BPs: Select first m BPs by Eq. (2);

Part III: Generative Adversarial Model
Input: $\{H, \hat{H}\}$
Output: $\{H', \phi, \theta\}$
Parameters $\{\phi, \theta\}$: Minimize the loss in Eq. (7);
Reconstructed BPs H': Run H on network $\{\phi, \theta\}$;
Transductive setting: $B' \leftarrow H'$ by one-hot coding, $S' \leftarrow B'B'^T$, $\text{Kmeans}(S')$;
Inductive setting: Run external data BPs H_e on network $\{\phi, \theta\}$ and obtain the output H'_e, $B'_e \leftarrow H'_e$ by one-hot coding, $S'_e \leftarrow B'_e B'^T_e$, $\text{Kmeans}(S'_e)$;

matrix B are used to construct CA matrix S. This inspires us to leverage $\hat{H}\hat{H}^T$ as the measurement for the perturbed CA matrix. The CA matrix based on $H'H'^T$ is assumed to approach $\hat{H}\hat{H}^T$, which defines the following loss function:

$$\mathcal{L}_G = \|H'H'^T - \hat{H}\hat{H}^T\|_{\text{F}}^2. \tag{6}$$

Following this assumption, the VAE based generative model will directly mimic the perturbed CA matrix to secure the adversary effectiveness. On the other hand, the significant mutation of the CA matrix caused by $\mathcal{L}_G$ is less noticeable in the learning process, as it is already different from the original BPs.

3.2. Solution

In brief, our generative model GA2EC can be learned by minimizing four losses which are shown below:

$$\mathcal{L} = \mathcal{L}_{KL} + \lambda_1 \mathcal{L}_R + \lambda_2 \mathcal{L}_P + \lambda_3 \mathcal{L}_G, \tag{7}$$

where $\lambda_1 \sim \lambda_3$ are balancing parameters, and $\mathcal{L}$ is differentiable which can be solved by backpropagation, similar to the existing solution to VAE. Essentially, the balancing parameters are responsible for controlling the mutation of BPs and make sure it is not stretched too much from the clean data, but also provides effective attack. Discussions and empirical discoveries will be provided in the experiments. The overall algorithm for GA2EC can be found in Algorithm 1. Once the GA2EC model is learned, it can be run under both transductive and inductive settings. The unseen data and its BPs matrix H_e can be fed into the proposed VAE to generate H'_e to be used in EC tasks.

Table 2: Details of datasets used in experiments.

Dataset	Type	#Instance	#Attribute	#Class
MNIST	digit	10000	784	10
MFashion	article	10000	784	10
USPS	digit	11000	256	10
CIFAR-10	object	10000	3072	10
STL-10	object	5000	27648	10
COIL-20	object	1440	1024	20
PIE	faces	1407	900	67
YaleB	faces	1440	1024	15
Amazon	object	2817	1000	31
Dslr	object	498	1000	31
Webcam	object	795	1000	31

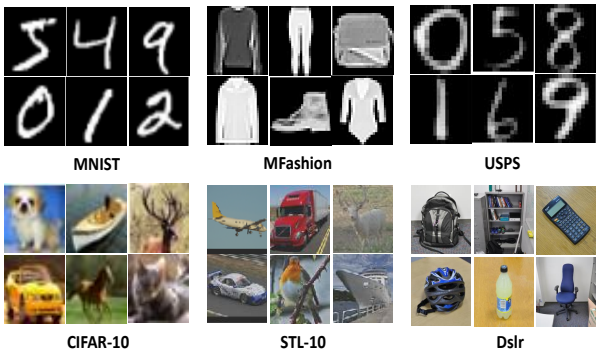


Figure 5: Sample images of benchmark datasets.

3.3. Insights and discussions

Recent research revealed that EC may be interpreted and parameterized from a Bayesian perspective [42], in a way similar to the topic modeling and Latent Dirichlet Allocation (LDA) [4]. Essentially, [42] explains the BPs and ensemble clustering process through a multinomial mixture model, which is then solved through variational inference for approximation since the posterior distribution cannot be computed in closed form. In our study, the attack is initialized through a discriminant model and then simulated via a generative model. It would be more interesting to see whether EC can be compromised or attacked from a Bayesian perspective. Relevant works for LDA on security and privacy have been discussed in [9, 28, 49], which may

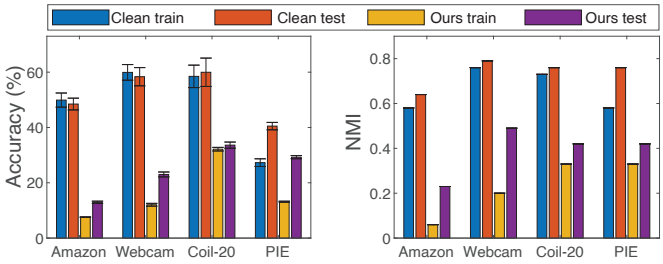


Figure 6: Clustering results under inductive settings.

inspire our future work.

4. Experiments

4.1. Database Introduction

For evaluation of our proposed study, we have conducted experiments on 11 datasets of varied sizes, and their information can be found in Table 2. Sample images from some datasets are shown in Fig. 5. **MNIST and USPS** consist of handwritten digit images from 0 to 9. **MNIST Fashion** (MFashion) [45] is a set of article images, e.g, clothes, shoes, handbags, etc. **CIFAR-10** [19], **STL-10** [8] and **COIL-20** [31] include color images of different objects such as ship, bird, cat, etc. **PIE and YaleB** [13,34] are popular face image datasets. We use raw data as features for above datasets. Finally, **Amazon**, **Dslr** and **Webcam** [33] are popular object image datasets and decaf [10] deep features are used.

4.2. Experiment Setup

We will discuss the evaluation metric, features, preprocessing, the number of BPs attacked, and other hyperparameters used in the experiments.

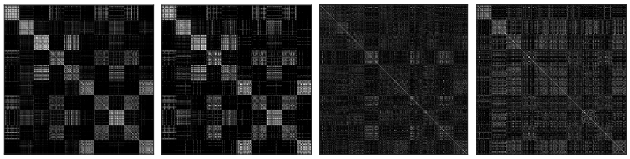
Evaluation metrics. For clustering evaluation, we have incorporated two commonly used evaluation metrics: clustering accuracy and Normalized Mutual Information (NMI) [44]. These metrics are evaluated against Kmeans output and ground truth labels (L).

Pre-processing and features. In our experiments, we have performed EC after Part I $\sim$ III as described in Algorithm 1. When generating BPs on given data as described in Part I in the Algorithm 1, we also normalize the feature vector so that each vector has a length of 1.

Number of BPs. We set the number of basic partitions to 1500 throughout the experiments when generating H, as explained in Sec. 2.3. When applying adversarial attack to H, 500 BPs will be selected based on the residual defined in Eq. (2).

Experiment platform. All experiments are run on a workstation with an Intel i7-7700K CPU, 32GB memory, and a NVIDIA 1080ti GPU. The code was implemented in Python 3.7.4 and PyTorch 1.2.

Other parameters. For the perturbation in Eq. (1), we have set $\epsilon = 0.75$, which empirically works well, as shown later in Fig. 9. We have also explored the balancing parameters $\lambda_1 \sim \lambda_3$, and set the values as: $\lambda_1 = 0.001$, $\lambda_2 = 1$ and $\lambda_3 = 0.2$. These parameters work well in most cases, as demonstrated later in Fig 10.



(a) (b) (c) (d)

Figure 7: Graph visualization on USPS dataset. (a) Clean BPs (b) VAE reconstructed BPs (c) Gradient attacked BPs (d) Ours.

4.3. Experimental Results

Transductive setting results. In a transductive setting, we learn GA2EC model and apply attacks on the same dataset. To evaluate the effectiveness, four experiments are conducted. First, clean data and its BPs go through EC to obtain the clustering results. Second, we apply plain VAE and then use the output, i.e., reconstructed BPs, for EC tasks. This is to verify whether the generated data by plain VAE has similar distribution as the clean data, and provides similar ensemble clustering results as the input data. Third, gradient attack with pseudo labels is applied to the clean data to get perturbed BPs, which is then fed to EC tasks. This demonstrates the effectiveness of the original adversarial attack, with excessive mutations on clean data. Lastly, we show our EC performance via GA2EC. As expected, in Table 3, the performance based on clean and VAE reconstructed BPs are very close. As our method offers minor mutation to make attack less noticeable, the performance is slightly inferior to the gradient attack based approach, but very close in most cases.

Inductive setting results. Another merit of GA2EC is the extension of unseen data under the inductive setting. Basically, the generative model will be trained as shown in the blue path in Fig. 4, and then tested through the red path. For this purpose, we split each dataset into training and testing with a ratio of 80:20. After the generative model is built, we let both *training* and *testing* data go through model for EC tasks. In addition, we compare the EC performance of clean training and testing data, as the baseline. The results are shown in Fig. 6 and we can see that our model performs consistently on unseen data as compared to the training data, showing that the model can extend well to data sampled from the same source.

Comparisons with existing method. To the best of our knowledge, none of the existing work explores the adversarial learning on EC tasks. Very recently, research works,

Table 3: Clustering performance on benchmark datasets by different models. Clean: clustering on the original dataset; RC: reconstructed BPs from plain VAE; GA: perturbed BPs by FGSM; Ours: perturbed BPs by GA2EC. Note standard deviation of NMI in the table is usually very small and rounding to 0 in some cases.

Dataset	Accuracy (%)				NMI			
	Clean	RC	GA	Ours	Clean	RC	GA	Ours
MNIST	56.45±5.59	54.40±3.45	22.45±0.29	28.26±1.33	0.53±0.03	0.48±0.01	0.09±0.00	0.18±0.01
MFashion	52.61±2.30	50.01±2.46	25.01±0.55	38.03±2.20	0.56±0.01	0.52 ±0.01	0.12±0.00	0.32±0.02
USPS	45.58±3.11	45.08±2.11	21.36±0.46	23.58±1.07	0.45±0.02	0.42±0.00	0.10±0.00	0.12±0.01
CIFAR-10	23.15±0.70	17.47±0.25	15.23±0.26	15.59±0.53	0.10±0.00	0.04±0.00	0.02±0.00	0.03±0.00
STL-10	24.85±0.45	22.54±0.46	17.04±0.34	17.23±1.05	0.15±0.00	0.10±0.00	0.05±0.00	0.06±0.01
COIL-20	56.36±3.75	50.61±4.99	16.32±0.43	22.29±0.62	0.72±0.01	0.70±0.02	0.14±0.00	0.24±0.00
PIE	26.35±1.32	23.16±0.87	9.99±0.21	11.03±0.14	0.58±0.01	0.54±0.00	0.31±0.00	0.32±0.01
YaleB	38.00±3.33	40.96±3.17	23.39±0.58	25.39±0.92	0.43±0.01	0.45±0.01	0.29±0.00	0.31±0.01
Amazon	53.08±1.63	50.44±2.29	10.35±0.22	8.32±0.10	0.60±0.00	0.58±0.01	0.10±0.00	0.07±0.01
Dslr	58.15±3.31	61.00±2.81	18.05±0.60	18.82±0.58	0.76±0.01	0.77±0.01	0.30±0.00	0.32±0.01
Webcam	60.31±2.66	56.02±3.39	16.57±0.46	16.69±0.31	0.76±0.00	0.75±0.01	0.26±0.00	0.27±0.00

Table 4: Comparison with existing adversarial clustering method. "# of error" means the number of mis-clustered samples (the larger the better), and "ℓ_2 norm" indicates the Frobenius norm between perturbed and clean BPs (the smaller the better).

# of error	Spill-over	Poisoning	Ours
MNIST 1&4	11	12±4.5	13
MNIST 2 & 3	2	12.2±1.0	20
Digits 1 & 4	24	24±0.0	13
Digits 8 & 9	21	21±0.0	37
ℓ_2 norm	Spill-over	Poisoning	Ours
MNIST 1 & 4	585.38	782.7±124.20	567.41±10.19
MNIST 2 & 3	872.84	497.9±92.50	513.85±9.07
Digits 1 & 4	23.93	19.84±1.96	7.62±0.40
Digits 8 & 9	15.70	13.86±2.96	8.51±0.51

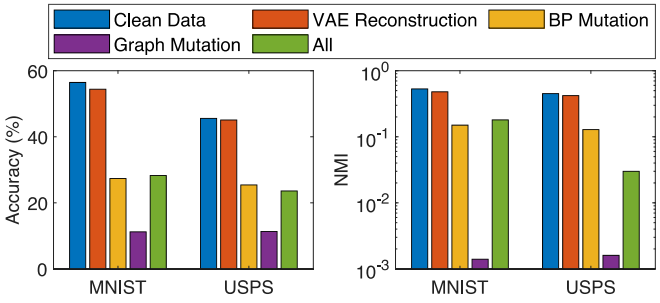


Figure 8: Ablation study: different loss functions and their results on MNIST and USPS datasets.

i.e., spill-over [6] and poisoning clustering [7] discussed the adversarial learning on standard clustering. Moreover, they consider only two classes for attack at a time while we consider all the classes for attack. For fair comparisons, we restrict our model to two classes on the same dataset as [6] and compare based on their provided code and data. In particular, [6] and [7] have evaluated their model on MNIST (digits 1&4, 2&3) and Digits [2] (digits 1&4, 8&9). For further details on database selection, readers can refer to [6]. Note in [6] and [7], their models attacked one sample at a time, whereas our model has the option to attack different samples with minor noises. Note that we have compared our results to poisoning clustering [7] where they have set noise threshold to the maximum. As we can see from Table 4, in most cases, our model is able to achieve higher mis-clustered rates, compared to state-of-the-art methods, while keeping the perturbation level low measured by ℓ_2 norm.

Graph loss and visualization. It is also important to understand how graph constraint and mutation loss contribute to the EC tasks. To that end, we visualize graph generated by HH^T on USPS dataset in Fig. 7. As expected, the graph generated by plain VAE is very close to the clean data, while the one by gradient attack is very different, which explains why it can compromise performance badly. On the other hand, our model achieves a good balance between (b) and (d) in Fig. 7.

Empirical time complexity. The time taken to train the generative model is proportional to the number of data points in a given dataset. For instance, large datasets such as MNIST and MFashion take around 151 and 265 minutes respectively while small datasets such as COIL-20 and Amazon take around 11 and 34 minutes respectively. Comparatively testing time for EC is very less and it takes around 1.59, 1.63, 0.20 and 0.45 minutes for MNIST, MFashion, COIL-20 and Amazon datasets respectively.

4.4. Ablation Study

Our model includes four losses: (1) VAE reconstruction loss, (2) VAE KL divergence loss, (3) BP mutation loss, and (4) graph mutation loss. The four losses work syner-

Table 5: Attack sensitivity analysis via Frobenius norm between the original H and attacked H. RC: reconstructed BPs from the plain VAE; GA: perturbed BPs by FGSM; Ours: perturbed BPs by GA2EC. For norms in the table, the smaller the better.

Dataset	RC	GA	Ours
MNIST	0.48±0.01	0.97±0.00	0.74±0.00
MFashion	0.54±0.16	1.41±0.03	0.84±0.08
USPS	0.51±0.13	0.97±0.00	0.77±0.00
CIFAR-10	0.58±0.19	0.92±0.00	0.87±0.00
STL-10	0.57±0.27	1.18±0.00	0.79±0.00
COIL-20	0.82±0.74	1.54±0.18	1.36±0.04
PIE	0.94±0.02	0.97±0.06	0.93±0.00
YaleB	0.77±0.74	0.89±0.05	1.14±0.23
Amazon	0.94±0.71	1.79±0.14	1.59±0.05
Dslr	0.86±0.67	1.53±0.26	1.64±0.06
Webcam	0.46±0.15	1.11±0.50	1.09±0.63

gistically to drive data towards adversarial in nature but still preserving the features similar to original data. To demonstrate the necessity and impact of each loss, we add and remove the loss one at a time in this ablation study. As shown in Fig. 8, we first use the VAE reconstruction and KL divergence loss which gives us the reconstruction data and results similar to clean data. Second, we add BP mutation loss and we can see there is a drop in results. Third, we remove the BP mutation loss but add the graph mutation loss instead, and performance is further reduced. The downside of BP mutation is that it cannot achieve a competitive adversary. On the other hand, graph mutation offers an aggressive attack and make it noticeable. Our model maintains the right balance among all losses.

4.5. Parameters Analysis

Attack sensitivity. BPs after attack will deviate from the original values and quantitative measurement of the deviation, namely, mutation should be provided for each step. We compare the ℓ_2 norms between perturbed BPs and original BPs in Table 5. It can be seen that reconstructed BPs by the plain VAE offer the lowest norm difference, while gradient attack the highest one. Our method lowers the reconstruction error on most datasets, while secures an effective attack. This is desired in real-world adversarial learning.

Value of ϵ. The magnitude (ϵ) of gradient attack on MNIST and USPS datasets is shown in Fig. 9. It can be seen that a larger ϵ value usually offers lower accuracy and NMI, and the same trend can be found for both datasets. Therefore, we use $\epsilon = 0.75$ throughout all experiments.

Balancing parameter λ. The four losses in Eq. (7) shall be balanced through parameters $\lambda_1 \sim \lambda_3$. We visualize their impacts on YaleB dataset, and show both accuracy and NMI in Fig. 10. Note x, y axis indicate λ_2 and λ_3, while

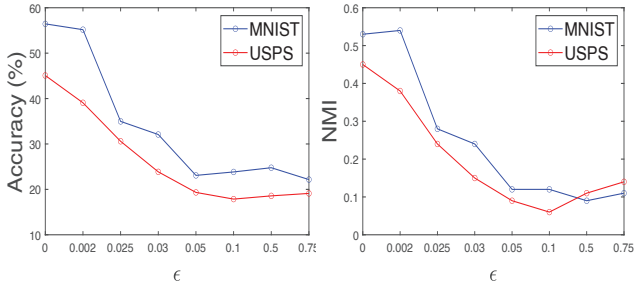


Figure 9: Impacts of ϵ on MNIST and USPS datasets.

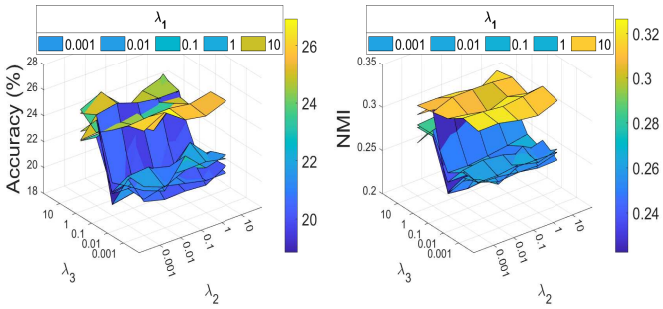


Figure 10: Impacts of λ on Yale dataset.

meshes show different λ_1 values. This is consistent with our setting on these balancing parameters, which also empirically works well on other datasets.

5. Conclusion

In this paper, we proposed a novel generative adversarial ensemble clustering method, which demonstrated that ensemble clustering tends to be vulnerable due to gradient-based adversarial attack, a common strategy in adversarial machine learning. To that end, we first examined the feasibility of conventional adversarial attack on ensemble clustering through pseudo labels. Second, a dedicated generative model was developed to mimic the attacks, which could be extended to unseen data under an inductive setting. Extensive experiments on eleven clustering tasks and thorough analysis showed that our method is effective, yet less noticeable compared to the existing methods. In the future, adversarial attack from a Bayesian perspective could be explored to offer probabilistic insights and interpretable solutions.

Acknowledgement

This work is supported in part by the UMass Dartmouth College of Engineering faculty start-up fund, UMass Dartmouth's Marine and Undersea Technology (MUST) Research Program funded by the Office of Naval Research (ONR) under Grant No. N00014-20-1-2170.

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018. 1, 3

[2] Ethem Alpaydın and Cenk Kaynak. Cascading classifiers. *Kybernetika*, 34:369–374, 07 1997. 7

[3] Tahani Alqurashi and Wenjia Wang. Clustering ensemble method. *International Journal of Machine Learning and Cybernetics*, 10(6):1227–1246, 2019. 2

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003. 5

[5] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018. 1

[6] Anshuman Chhabra, Abhishek Roy, and Prasant Mohapatra. Suspicion-free adversarial attacks on clustering algorithms, 2019. 1, 7

[7] Antonio Emanuele Cinà, Alessandro Torcinovich, and Marcello Pelillo. A black-box adversarial attack for poisoning clustering. *arXiv preprint arXiv:2009.05474*, 2020. 1, 7

[8] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011. 6

[9] Christopher DeCarolis, Mukul Ram, Seyed A Esmaeili, Yu-Xiang Wang, and Furong Huang. An end-to-end differentially private latent dirichlet allocation using a spectral algorithm. *arXiv preprint arXiv:1805.10341*, 2018. 5

[10] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014. 6

[11] Ana LN Fred and Anil K Jain. Data clustering using evidence accumulation. In *Object recognition supported by user interaction for service robots*, volume 4, pages 276–280. IEEE, 2002. 2

[12] Ana LN Fred and Anil K Jain. Combining multiple clusterings using evidence accumulation. *IEEE transactions on pattern analysis and machine intelligence*, 27(6):835–850, 2005. 1

[13] Athinodoros S. Georghiades, Peter N. Belhumeur, and David J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE transactions on pattern analysis and machine intelligence*, 23(6):643–660, 2001. 6

[14] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. 1, 3

[15] Dong Huang, Chang-Dong Wang, and Jian-Huang Lai. Locally weighted ensemble clustering. *IEEE transactions on cybernetics*, 48(5):1460–1473, 2017. 1

[16] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010. 1

[17] Jean-Michel Jolion, Peter Meer, and Samira Bataouche. Robust clustering with applications in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(8):791–802, 1991. 1

[18] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2, 4

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[20] Chetan Kumar, Riazat Ryan, and Ming Shao. Adversary for social good: Protecting familial privacy through joint adversarial attacks. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 11304–11311, 2020. 1

[21] Deepak Kumar, Chetan Kumar, Chun Wei Seah, Siyu Xia, and Ming Shao. Finding achilles' heel: Adversarial attack on multi-modal action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 3829–3837, New York, NY, USA, 2020. Association for Computing Machinery. 1

[22] Tao Li and Chris Ding. Weighted consensus clustering. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, pages 798–809. SIAM, 2008. 1, 2

[23] Hongfu Liu, Tongliang Liu, Junjie Wu, Dacheng Tao, and Yun Fu. Spectral ensemble clustering. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 715–724, 2015. 1, 2

[24] Hongfu Liu, Ming Shao, and Yun Fu. Feature selection with unsupervised consensus guidance. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2319–2331, 2018. 1, 2

[25] Hongfu Liu, Ming Shao, Sheng Li, and Yun Fu. Infinite ensemble for image clustering. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1745–1754, 2016. 1, 2

[26] Hongfu Liu, Ming Shao, Sheng Li, and Yun Fu. Infinite ensemble clustering. *Data Mining and Knowledge Discovery*, 32(2):385–416, 2018. 1

[27] Hongfu Liu, Zhiqiang Tao, and Zhengming Ding. Consensus clustering: An embedding perspective, extension and beyond. *arXiv preprint arXiv:1906.00120*, 2019. 2

[28] Shike Mei and Xiaojin Zhu. The security of latent dirichlet allocation. In *Artificial Intelligence and Statistics*, pages 681–689, 2015. 5

[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016. 1

[30] Samina Naz, Hammad Majeed, and Humayun Irshad. Image segmentation using fuzzy clustering: A survey. In *2010 6th international conference on emerging technologies (ICET)*, pages 181–186. IEEE, 2010. 1

[31] Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-100). 1996. 6

[32] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002. 1

[33] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010. 6

[34] Terence Sim, Simon Baker, and Maan Bsat. The cmu pose, illumination, and expression (pie) database. In *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*, pages 53–58. IEEE, 2002. 6

[35] Alexander Strehl and Joydeep Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *Journal of machine learning research*, 3(Dec):583–617, 2002. 1, 2

[36] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1, 3

[37] Zhiqiang Tao and Hongfu Liu. Simultaneous clustering and ensemble. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2017. 2

[38] Zhiqiang Tao, Hongfu Liu, Jun Li, Zhaowen Wang, and Yun Fu. Adversarial graph embedding for ensemble clustering. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3562–3568. AAAI Press, 2019. 1

[39] Sandro Vega-Pons and José Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(03):337–372, 2011. 2

[40] Chang-Dong Wang, Jian-Huang Lai, and Jun-Yong Zhu. Graph-based multiprototype competitive learning and its applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):934–946, 2011. 1

[41] Hongjun Wang, Tao Li, Tianrui Li, and Yan Yang. Constraint neighborhood projections for semi-supervised clustering. *IEEE Transactions on Cybernetics*, 44(5):636–643, 2014. 1

[42] Hongjun Wang, Hanhuai Shan, and Arindam Banerjee. Bayesian cluster ensembles. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 4(1):54–70, 2011. 1, 5

[43] Junjie Wu, Hongfu Liu, Hui Xiong, Jie Cao, and Jian Chen. K-means-based consensus clustering: A unified view. *IEEE transactions on knowledge and data engineering*, 27(1):155–169, 2014. 2

[44] Junjie Wu, Hui Xiong, and Jian Chen. Adapting the right measures for k-means clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 877–886, 2009. 1, 6

[45] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. 6

[46] Lei Xu, Adam Krzyzak, and Erkki Oja. Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. *IEEE Transactions on Neural networks*, 4(4):636–649, 1993. 1

[47] Hye-Sung Yoon, Sun-Young Ahn, Sang-Ho Lee, Sung-Bum Cho, and Ju Han Kim. Heterogeneous clustering ensemble method for combining different cluster results. In *International Workshop on Data Mining for Biomedical Applications*, pages 82–92. Springer, 2006. 2

[48] Xiaoyong Yuan, Pan He, Qile Zhu, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 30(9):2805–2824, 2019. 1, 3

[49] Fangyuan Zhao, Xuebin Ren, Shusen Yang, and Xinyu Yang. On privacy protection of latent dirichlet allocation model training. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4860–4866. AAAI Press, 2019. 5

[50] Zhi-Hua Zhou and Wei Tang. Clusterer ensemble. *Knowledge-Based Systems*, 19(1):77–83, 2006. 2