Infinite Ensemble for Image Clustering

Hongfu Liu[1], Ming Shao[1], Sheng Li[1] and Yun Fu[1,2]
[1] Department of Electrical and Computer Engineering, Northeastern University, Boston, USA, 02115.
[2] College of Computer and Information Science, Northeastern University, Boston, USA, 02115.
liu.hongf@husky.neu.edu, {mingshao, shengli, yunfu}@ece.neu.edu.

ABSTRACT

Image clustering has been a critical preprocessing step for vision tasks, e.g., visual concept discovery, content-based image retrieval. Conventional image clustering methods use handcraft visual descriptors as basic features via K-means, or build the graph within spectral clustering. Recently, representation learning with deep structure shows appealing performance in unsupervised feature pre-treatment. However, few studies have discussed how to deploy deep representation learning to image clustering problems, especially the unified framework which integrates both representation learning and ensemble clustering for efficient image clustering still remains void. In addition, even though it is widely recognized that with the increasing number of basic partitions, ensemble clustering gets better performance and lower variances, the best number of basic partitions for a given data set is a pending problem. In light of this, we propose the Infinite Ensemble Clustering (IEC), which incorporates the power of deep representation and ensemble clustering in a one-step framework to fuse infinite basic partitions. Generally speaking, a set of basic partitions is firstly generated from the image data. Then by converting the basic partitions to the 1-of-K codings, we link the marginalized auto-encoder to the infinite ensemble clustering with i.i.d. basic partitions, which can be approached by the closed-form solutions. Finally we follow the layer-wise training procedure and feed the concatenated deep features to K-means for final clustering. Extensive experiments on diverse vision data sets with different levels of visual descriptors demonstrate both the time efficiency and superior performance of IEC compared to the state-of-the-art ensemble clustering and deep clustering methods.

CCS Concepts

•Information systems → Clustering; •Theory of computation → Unsupervised learning and clustering; •Computing methodologies → Ensemble methods;

KDD '16, August 13-17, 2016, San Francisco, CA, USA
© 2016 ACM. ISBN 978-1-4503-4232-2/16/08. . . $15.00
DOI: http://dx.doi.org/10.1145/2939672.2939813

Keywords

Ensemble Clustering, Marginalized Denoising Auto-encoder, K-means

1. INTRODUCTION

Image clustering has been identified as one of the most critical steps for many vision applications, e.g., storyline reconstruction from photo streams [23, 24], automatic visual concept discovery [25], 3D construction from image collections [15], finding iconic images [6, 36] and web-scale fast image clustering [1, 19]. However, most existing image clustering methods use either feature engineering based features [11, 33] or discriminant deep features trained by well-established deep Convolutional Neural Network (CNN) model [8, 14]. As a result, they did not intentionally learn the representation along with the clustering, or in other words, build a joint clustering and representation learning framework. It is especially useful for high-performance image clustering systems, where the learned representation and clustering strategy need substantial interactions for better performance [31, 32, 28].

Recently, representation learning attracts substantial research attention, which has been widely adopted as the unsupervised feature pre-treatment [4]. The layer-wise training and the followed deep structure are able to capture the visual descriptors from coarse to fine [5, 20]. Notably, there are a few deep clustering methods proposed recently, working well with either feature vectors [37] or graph Laplacian [21, 26], towards high-performance generic clustering tasks. There are two typical problems with regard to the deep clustering approaches: (1) how to seamlessly integrate the "deep" concept into the conventional clustering framework, (2) how to solve it efficiently. Few attempts have been made for the first problem [21, 40], however, most of which sacrifice the time efficiency. They follow the conventional training strategy for deep models, whose complexity will be in super-liner with respect to the number of samples. A recent deep linear coding framework attempts to handle the second problem [37], and preliminary results demonstrate its time efficiency with comparable performance on large-scale data sets. However, its performance on vision data has not been thoroughly evaluated yet, given different visual descriptors and tasks.

In order to obtain a robust clustering result for computer vision applications, ensemble clustering is often used due to its high performance and robustness. Tremendous efforts have been devoted to thrive this area in different perspectives [16, 39, 45]. These studies can be roughly generalized into two categories: the methods based on co-association

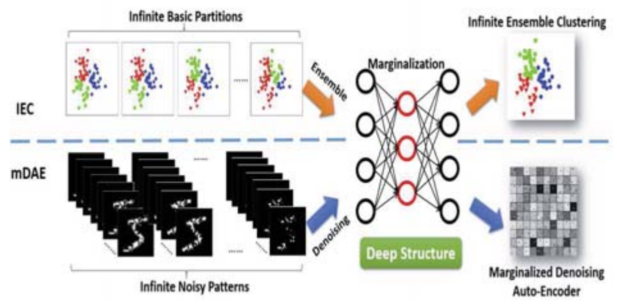

Infinite Basic Partitions
Infinite Ensemble Clustering
Ensemble
Marginalization
IEC
mDAE
Denoising
Deep Structure
Infinite Noisy Patterns
Marginalized Denoising Auto-Encoder

Figure 1: Framework of IEC. We apply marginalized Denoising Auto-Encoder to generate infinite ensemble members by adding drop-out noise and fuse them into the consensus one. The figure shows the equivalent relationship between IEC and mDAE.

matrix or utility function. Although ensemble clustering outperforms the tradition clustering methods, [45] pointed out that different generation strategies for basic partitions have great impacts on the success of ensemble clustering, besides the number of basic partitions determines the robustness [35]. It has been widely recognized that with the increasing number of basic partitions, ensemble clustering achieves better performance and lower variance [45, 35]. However, the best number of basic partitions for a given data sets still remains an open problem. Too few basic partitions cannot exert the capacity of ensemble clustering, while too many basic partitions lead to unnecessary computational resource waste. Here comes the third problem that (3) can we use the infinite ensemble basic partitions to maximize the capacity of ensemble clustering with a low computational cost?

In this work, we simultaneously manage to tackle the three problems mentioned above, and conduct extensive experiments on image data sets with different visual descriptors for demonstration. Our new model links the marginalized denoising auto-encoder to ensemble clustering and leads to a natural integration named "Infinite Ensemble Clustering" (IEC), which is simple yet effective and efficient. To that end, we first generate a moderate number of basic partitions, as the basis for the ensemble clustering. Second, we convert the preliminary clustering results from the basic partitions to 1-of-K codings, which disentangles dependent factors among data samples. Then the codings are expanded infinitely by considering the empirical expectation over the noisy codings through the marginalized auto-encoders with the drop-out noises. Finally, we run K-means on the learned representations to obtain the final clustering. The framework of IEC is demonstrated in Figure 1. The whole process is similar to marginalized Denoising Auto-Encoder (mDAE). Several basic partitions are fed into the deep structure with drop-out noises in order to obtain the expectation of the co-association matrix. It is worthy to note that IEC has roughly linear $O(n)$ time complexity, which can be used for large-scale data sets. Extensive results on diverse vision data sets show that our IEC framework works fairly well with different visual descriptors, in terms of time efficiency and clustering performance, and moreover some key impact factors are thoroughly studied as well. Although we focus on image clustering in this work, IEC is general enough to handle clustering problems of other domains.

We highlight our contributions as follows.

- We propose a framework called Infinite Ensemble Clustering (IEC) which integrates the deep structure and ensemble clustering. By this means, the complex ensemble clustering problem can be solved with a stacked marginalized Denoising Auto-Encoder structure in an efficient way.
- Within the marginalized Denoising Auto-Encoder, we fuse infinite ensemble members into a consensus one by adding drop-out noises, which maximizes the capacity of ensemble clustering.
- Extensive experimental results on numerous real-world data sets with different levels of features demonstrate IEC has obvious advantages on effectiveness and efficiency compared with the state-of-the-art deep clustering and ensemble clustering methods, and IEC is a promising tool for large-scale image clustering.

2. RELATED WORK

Here we introduce the related work in terms of image clustering, ensemble clustering and auto-encoder, and highlight the difference between existing methods and ours.

2.1 Image Clustering & Ensemble Clustering

Image clustering has been a fundamental problem for many vision applications, in particular with the popularity of photo sharing websites such as Facebook, Instagram and Twitter. Most of existing works focus on either specific vision problems, e.g., automatic visual concept discovery [25], 3D construction from image collections [15], storyline reconstruction from photo streams [23, 24], finding iconic images [6, 36], or on web-scale memory efficient fast clustering [1, 19]. In this work, we develop a generic image clustering method that naturally integrates the "deep" thought with the ensemble clustering. It can be easily deployed to different vision tasks as the basic clustering tool [15, 23, 24, 25], and well balance the performance and running time.

On the other hand, the hashing or binary tricks for memory-efficient web-scale clustering manage to accelerate linear clustering methods [1, 19] and achieve comparable performance on large-scale systems. Our method is a competitive complement to those methods in terms of accuracy, as ours usually performs better than those with K-means by a large-margin (See Table 2 and 3). In addition, our method can be easily deployed in a distributed system as the time consuming basic partitions are generated independently, binarized and hashed with further optimizations [18, 44]. However, these discussions are beyond the scope of this paper.

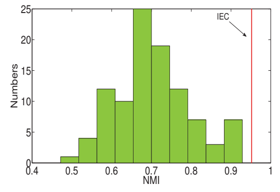
Ensemble clustering, also known as consensus clustering or cluster aggregation, aims to fuse several existing basic partitions into an integrated one. Tremendous efforts have been made to solve ensemble clustering. The existing work can be roughly generalized into two categories according the different levels of similarity. The first category employs the utility function to measure the similarity at the partition-level between the consensus clustering and multiple basic partitions. This kind of methods usually finds the final clustering result by maximizing the utility function value. For instance, [41] proposed a Quadratic Mutual Information based objective function for consensus clustering, and further extended their work to use the EM algorithm with a finite mixture of multinomial distributions for consensus clustering [42]. Along this line, Wu et al. [45] proposed K-means-based Consensus Clustering (KCC) and transferred the consensus clustering into a K-means clustering problem with different KCC utility functions. In addition, there are some other interesting objective functions for the consensus clustering, such as [27, 34]. The second category summarizes the information of basic partitions into a co-association matrix, which counts how many times two instances occur in the same cluster. Therefore, the co-association matrix can be regarded as the similarity matrix based on the instance-level, base on which any graph partition algorithm can be conducted for the final clustering. For example, [16] applied the agglomerative hierarchical clustering, and [39] developed three graph-based algorithms for consensus clustering. Recently, Liu et al. [29] proposed a spectral ensemble clustering method to transform it as a weighted K-means problem and built the connection of these two kinds of ensemble clustering methods. Other methods include Relabeling and Voting [2], Locally Adaptive Cluster based methods [13], genetic algorithm based methods [48], and still many more. Although much efforts have been made to design effective and efficient algorithms, the number of basic partitions is still an open unsolved problem.

2.2 Auto-encoder

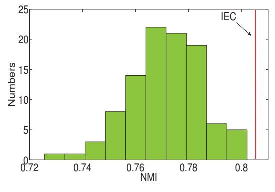
Auto-Encoder (AE) is a building block of deep structure that learns hidden and compressed representations (i.e., codings) from data [3]. The motivation of auto-encoder is to transform inputs into outputs with the least possible amount of deformation. Moreover, stacked Auto-Encoder (SAE) can be developed by inserting multiple hidden layers to auto-encoder, which is one of the most popular deep learning architectures. In reality, learning from the noisy or corrupted data is a challenging problem. Denoising Auto-Encoder (DAE) and stacked DAE learn effective representations by reconstructing input data from artificial corruptions [43]. Marginalized Denoising Auto-Encoder (mDAE) approximately marginalizes out the corruptions during training, taking into account infinitely many corrupted copies of training data [9, 10]. Due to the flexibility and impressive learning capability, auto-encoder and its variants have been successfully applied to many scenarios, such as face recognition [22], domain adaptation [10, 12], and image classification [47].

Most recently, a few auto-encoder based methods have been proposed for graph clustering. [38] augmented the loss function of auto-encoder by incorporating a constraint of the distance between samples and centroids. [21] built a deep embedding network using auto-encoder, and incorporated locality-preserving and group sparsity constraints to the loss function of deep network for clustering-oriented representations. [40] revealed the similarity between auto-encoder and spectral clustering, and presented a GraphEncoder method based on sparse auto-encoder. [37] proposed a deep learning coding approach, which jointly learns feature transforms and discriminative codings for fast graph clustering. However, the connection between auto-encoder and ensemble clustering has not been explored.

In this paper, we aim to build the connection between ensemble clustering and auto-encoder, and apply marginalized Denoising Auto-Encoder to fuse infinite basic partitions for image clustering.



(a) *Caltech101*



(b) *ORL*

Figure 2: Performance distribution of basic partitions. Y-axis represents the number of basic partitions in each bin, and X-axis is the external measurement *Normalized Mutual Information* (NMI). Here Random Parameter Selection strategy is used to obtain 100 sub data sets, on which K-means is conducted with the cluster number varying from K to $2K$, where K is the true cluster number; and finally IEC is employed to fuse these basic partitions into the consensus one.

3. PRELIMINARIES AND PROBLEM DEFINITION

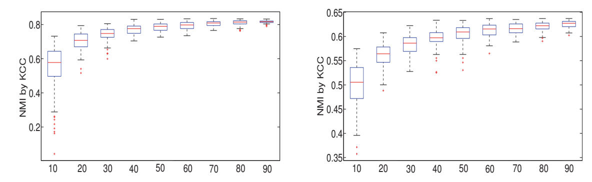
In this section, we introduce the preliminary knowledge in terms of ensemble clustering and marginalized Denoising Auto-Encoder, and then formulate the research problem.

3.1 Ensemble Clustering

The goal of ensemble clustering is to find a single partition which agrees with existing basic partitions as much as possible. It has been widely recognized that ensemble clustering can help generate robust partitions, find bizarre clusters, handle noise, outliers and sample variations, and integrate solutions from multiple distributed or incomplete sources of data or attributes [39, 41].

Before giving the formulation of ensemble clustering, we demonstrate the power of ensemble clustering on *Caltech101* and *ORL* in Figure 2. As can be seen, the performance of all the basic partitions on *Caltech101* locates in a wide range from 0.5 to 0.9 in terms of *Normalized Mutual Information* (NMI) and most of basic partitions are around 0.7, few of them exceed 0.85; surprisingly, IEC produces a high-quality partition which exceeds the best in all basic partitions. The similar phenomenon also occurs on *ORL*. This indicates that IEC can learn from the low-quality partitions and take use of the rich diversity of basic ones to provide a high-quality partition. In the following, we introduce how ensemble clustering fuses different basic partitions and obtains the final one.

Given a set of r basic partitions of the data matrix $\mathbf{X}$: $\mathcal{H} = \{\mathbf{H}^{(1)}, \mathbf{H}^{(2)}, \cdots, \mathbf{H}^{(r)}\}$ with the cluster number of $\mathbf{H}^{(i)}$ to be K_i, the goal is to fuse all the basic partitions into a consensus partition $\mathbf{H}^*$. Here basic partitions might be generated by the same clustering algorithm with different parameters, or by the same clustering algorithm with different features or even by several different clustering algorithms. Although ensemble clustering can be roughly generalized into two categories, based on co-association matrix or utility function, Liu et al. [29] built a connection between the methods based on co-association matrix and utility functions and pointed out the co-association matrix plays a determinative role in the success of ensemble clustering. Thus, here we focus on the methods based on co-association matrix.



(a) *mm* (b) *reviews*

Figure 3: Performance of different numbers of basic partitions via KCC on *mm* and *reviews* datasets. X-axis is the number of basic partitions. With increasing numbers of basic partitions, the performance goes up and the variance becomes narrow.

For the ensemble clustering with co-association matrix, the representative methods summarize r basic partitions into a co-association matrix as follows:

$$\mathbf{S}(\mathbf{x},\mathbf{y}) = \frac{1}{r}\sum_{i=1}^{r}\delta(\mathbf{H}^{(i)}(\mathbf{x}),\mathbf{H}^{(i)}(\mathbf{y})), \tag{1}$$

where $\delta(\cdot)$ denotes the Kronecker delta function, which returns 1 with two identical input values and returns 0 with different input values. We can regard $\mathbf{S}$ as a similarity matrix between a pair of instances, which simply counts the co-occurrence number in the same cluster in each basic partition. By this means, ensemble clustering problem is redefined as a classical graph partition problem, so that based on the co-association matrix $\mathbf{S}$, some clustering rules or loss functions can be derived in order to obtain the final consensus clustering.

Next, we introduce the impact of the number of basic partitions by the following theorem.

THEOREM 1. *(Stableness [35]) For any $\epsilon > 0$, there exists a matrix $\boldsymbol{S}_0$, such that*

$$\lim_{r\to\infty} P(||\boldsymbol{S}-\boldsymbol{S}_0||_{\mathrm{F}}^2 > \epsilon) = 0,$$

where $||\cdot||_{\mathrm{F}}^2$ denotes the Frobenius norm.

From the above theorem, we have the conclusion that although basic partitions might be greatly different from each other due to different generation strategies, the normalized co-association matrix becomes stable with the increase of the number of basic partitions r. From our previous experimental results [30] in Figure 3, it is easy to observe that with the increasing number of basic partitions, the performance of ensemble clustering goes up and becomes stable. However, the best number of basic partitions for a given data set is difficult to set. Too few basic partitions can not exert the capacity of ensemble clustering, while too many basic partitions lead to unnecessary computational resource waste. Therefore, fusing infinite basic partition is addressed in this paper, instead of answering the best number of basic partitions for a given data set. According to Theorem 1, we expect to fuse infinite basic partitions to maximize the capacity of ensemble clustering. Since we cannot generate infinite basic partitions, how to obtain a stable co-association matrix $\mathbf{S}$ and calculate $\mathbf{H}^*$ in an efficient way is highly needed, which is also one of our motivations. In Section 4, we employ mDAE to equivalently obtain the "infinite" basic partitions and achieve the expectation of co-association matrix.

3.2 Marginalized Denoising Auto-encoder

Denoising Auto-Encoders (DAEs) have been successfully used to learn new representations for a wide range of machine learning tasks [17, 7]. Usually DAE is implemented as a single-hidden-layer neural network where the input is the corrupted data by certain noises and the output is the clean data. The goal of DAE is to make the output to be as close as possible to the clean data $\mathbf{x}$ after learning. Usually a loss function $\ell(\mathbf{x},\mathbf{y})$ is employed to measure the reconstruction error as follows.

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{m}\sum_{j=1}^{m}\ell(\mathbf{x}_i, f(\widetilde{\mathbf{x}}_i^j)), \tag{2}$$

where n is the number of data points, m is the times of corrupted data, $\mathbf{x}_i$ is the i-th clean data point and $\widetilde{\mathbf{x}}_i^j$ is the i-th data point in j-th corruption, and $f(\widetilde{\mathbf{x}}_i^j)) = g \circ h(\widetilde{\mathbf{x}}_i^j))$ is the output of $\mathbf{x}_i^j$, where g and h are the encoder and decoder, respectively.

After getting the one-layer hidden representation $\mathbf{z} = h(\widetilde{\mathbf{x}})$, we can continue to use this strategy by using $\mathbf{z}$ as the input to obtain deep representation for feature generation, which is called Stacked Denoising Auto-encoder.

The disadvantage of DAE is to explicitly corrupt $\mathbf{x}$ by m times to get multiple $\widetilde{\mathbf{x}}$, which enlarges the training samples and increases the computational cost. Recently, Chen et al. [9, 10] proposed the marginalized Denoising Auto-Encoder (mDAE) to overcome this challenge by taking use of the expected average loss as follows,

$$\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{p(\widetilde{\mathbf{x}}_i|\mathbf{x}_i)}[\ell(\mathbf{x}_i, f(\widetilde{\mathbf{x}}_i))]. \tag{3}$$

For a long time, auto-encoder and its variants are regarded as a powerful feature generation tool. Actually it can also be used as an optimization tool. In the following, we will give another interpretation of auto-encoder.

3.3 Problem Definition

Deep structure and clustering techniques are powerful tools for computer vision and data mining applications. Especially, ensemble clustering attracts a lot of attention due to its appealing performance. However, these two powerful tools are usually used separately. Notice that the performance of ensemble clustering heavily depends on the basic partitions. As mentioned before, co-association matrix $\mathbf{S}$ is the key factor for the ensemble clustering and with the increase of basic partitions, the co-association matrix becomes stable. According to Theorem 1, the capability of ensemble clustering goes to the upper bound with the number of basic partitions $r \to \infty$, Then we aim to seamlessly integrate deep concept and ensemble clustering in a one-step framework: *Can we fuse infinite basic partitions for ensemble clustering in a deep structure?*

The problem is very straightforward, but it is quite difficult. The challenges of the problem lie in three folds:

- How to generate infinite basic partitions;
- How to seamlessly integrate the deep concept within ensemble clustering framework;
- How to solve it in a highly efficient way.

4. INFINITE ENSEMBLE CLUSTERING

Here we first uncover the connection between ensemble clustering and auto-encoder. Next, marginalized Denoising Auto-Encoder is applied for the expectation of co-association matrix, and finally we propose our method and give the corresponding analysis.

4.1 From Ensemble Clustering to Auto-encoder

It seems that there exists no explicit relationship between ensemble clustering and auto-encoder due to their respective tasks. The aim of ensemble clustering is to find a cluster structure based on basic partitions, while auto-encoder is usually used for better feature generation. However, by taking a close look at the objective function in Eq. 2 and Eq. 3, we find that auto-encoder can be regarded as an optimization method for minimizing the loss function.

Recalling that the goal of ensemble clustering is to find a single partition which agrees the basic ones as much as possible, we can understand it in the opposite way that the consensus partition has the minimum loss to present all the basic ones. After we summarize all the basic partitions into the co-association matrix $\mathbf{S}$, spectral clustering or some other graph partition algorithms can be conducted on the co-association matrix to obtain the final consensus result. Taking spectral clustering as an example, we aim to find a $n \times K$ low-dimensional space to represent the original input. Each column of low-dimensional matrix is a base for spanning the space. Then K-means can be run on that for the final partition. Similarly, the function of auto-encoder is also to learn a hidden representation with d dimensions with "carrying" as much as possible information with the input, where d is a user pre-defined parameter. Therefore, to some extent spectral clustering and auto-encoder have the similar function to learn new representations according to minimizing certain objective function; the difference is that in spectral clustering, the dimension of new representation is K, while auto-encoder produces d dimensions. From this view, auto-encoder is more flexible than spectral clustering.

Therefore, we have another interpretation of auto-encoder, which not only can generate robust features, but also can be regarded as an optimization method for minimizing the loss function. By this means, we can feed the co-association matrix into auto-encoder to get the new representation, which has the similar function with spectral clustering, and run K-means on that to obtain the consensus clustering. For the efficiency issue, it is not a good choice to use auto-encoder on the ensemble clustering task due to the large space complexity of co-association matrix $O(n^2)$. We will address this issue in the next subsection.

4.2 The Expectation of Co-Association Matrix

According to Theorem 1, with the number of basic partitions going to infinity, the co-association matrix becomes stable. Before answering how to generate infinite ensemble members, we first solve how to increase the number of basic partitions given the limited ones. The naive way is to apply some generation strategy on the original data to produce more ensemble members. The disadvantages lie in two folds: (1) time consuming, (2) sometimes we only have the basic partitions, and the original data are not accessible. Therefore, without the original data, producing more basic partitions with the limited one is like a clone problem. However, simply duplicating the ensemble members does not

Algorithm 1 The algorithm of Infinite Ensemble Clustering

```
Input: $\mathbf{H}^{(1)}, \cdots, \mathbf{H}^{(r)}$,: $r$ basic partitions;
       $l$: number of layers for mDAE;
       $p$: noise level;
       $K$: number of clusters.
Output: optimal $\mathbf{H}^*$;
1: Build the binary matrix $\mathbf{B}$;
2: Apply $l$ layers stacked mDAE with $p$ noise level to get
   the mapping matrix $\mathbf{W}$ by Eq. 4;
3: Run K-means on $\mathbf{B}\mathbf{W}^{\mathrm{T}}$ to get $\mathbf{H}^*$.
```

work. Here we make several copies of basic partitions and corrupt them with erasing some labels in basic partitions to get new ones. By this means, we have extra incomplete basic partitions and Theorem 1 also holds for incomplete basic partitions.

By this strategy, we just amply the size of ensemble members, which is still far from the infinity. To solve this challenge we use the expectation of co-association matrix instead. Actually, $\mathbf{S}_0$ is just the expectation of $\mathbf{S}$, which means if we obtain the expectation of co-association matrix as an input for auto-encoder, our goal can be achieved. Since the expectation of co-association matrix cannot be obtained in advance, we intend to calculate it during the optimization.

Inspired by the marginalized Denoising Auto-Encoder [10], which involves the expectation of certain noises during the training, we corrupt the basic partitions and marginalize it for the expectation. By adding drop-out noise to basic partitions, some elements are set to be zero, which means some instances are not involved during the basic partition generation. By this means, we can use marginalized Denoising Auto-Encoder to finish the infinite ensemble clustering task. If we take a look at Eq. 3, the function f can be linear or non-linear. In this paper, for efficiency issue we use the linear version for mDAE [10] since it has a close-form formulation.

4.3 The Proposed Method

So far, we solve the infinite ensemble clustering problem with marginalized Denoising Auto-Encoder. Before conducting experiments, we notice that the input of mDAE should be the instances with independent and identically distribution; however, the co-association matrix can be regarded as a graph, which disobeys this assumption. To solve this problem, we introduce a binary matrix $\mathbf{B}$.

Let $\mathbf{B} = \{b(x)\}$ be a binary data set derived from the set of r basic partitions $\mathcal{H}$ as follows:

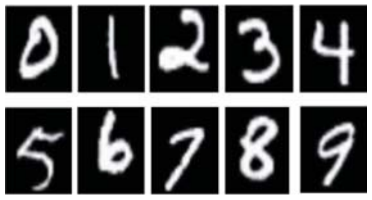
$$b(x) = \langle b(x)_1, \cdots, b(x)_r \rangle, b(x)_i = \langle b(x)_{i1}, \cdots, b(x)_{iK_i} \rangle,$$

$$b(x)_{ij} = \begin{cases} 1, & \text{if } \mathbf{H}^{(i)}(x) = j \\ 0, & \text{otherwise} \end{cases}.$$

We can see that the binary matrix $\mathbf{B}$ is a $n \times d$ matrix, where d equals $\sum_{i=1}^{r} K_i$. It concatenates all the basic partitions with 1-of-K_i coding, where K_i is the cluster number in the basic partition $\mathbf{H}^{(i)}$. With the binary matrix $\mathbf{B}$, we have $\mathbf{B}\mathbf{B}^{\mathrm{T}} = \mathbf{S}$. It indicates that the binary matrix $\mathbf{B}$ has the same information with the co-association matrix $\mathbf{S}$. Since $\mathbf{B}$ obeys the independent and identically distribution, we can put the binary matrix as input for marginalized Denoising Auto-Encoder.

Table 1: Experimental Data Sets

Data set	Type	Feature	#Instance	#Feature	#Class	#MinClass	#MaxClass	CV	Density
letter	character	low-level	20000	16	26	734	813	0.0301	0.9738
MNIST	digit	low-level	70000	784	10	6313	7877	0.0570	0.1914
COIL100	object	middle-level	7200	1024	100	72	72	0.0000	1.0000
Amazon	object	middle-level	958	800	10	82	100	0.0592	0.1215
Caltech	object	middle-level	1123	800	10	85	151	0.2087	0.1638
Dslr	object	middle-level	157	800	10	8	24	0.3857	0.1369
Webcam	object	middle-level	295	800	10	21	43	0.1879	0.1289
ORL	face	middle-level	400	1024	40	10	10	0.0000	1.0000
USPS	digit	middle-level	9298	256	10	708	1553	0.2903	1.0000
Caltech101	object	high-level	1415	4096	5	67	870	1.1801	1.0000
ImageNet	object	high-level	7341	4096	5	910	2126	0.3072	1.0000
Sun09	object	high-level	3238	4096	5	20	1264	0.8970	1.0000
VOC2007	object	high-level	3376	4096	5	330	1499	0.7121	1.0000


(a) *Digit*

(b) *Face*

(c) *Car*

Figure 4: Sample Images. (a) *MNIST* is a 0-9 digits data sets in grey level, (b) *ORL* contains faces of 40 people with different poses and (c) *Sun09* is an object data set with different types of cars.

The algorithm is summarized in Algorithm. 1. The first line is to build the binary matrix $\mathbf{B}$. Then we add the constant 1 at the last column of $\mathbf{B}$ and corrupt it with p level drop-out noise. Let $\mathbf{q} = [1-p, \cdots, 1-p, 1] \in \mathbb{R}^{d+1}$ and $\mathbf{\Sigma} = \mathbf{B}^{\mathrm{T}}\mathbf{B}$. The corresponding mapping for $\mathbf{W}$ between input and hidden representations is in closed form as:

$$\mathbf{W} = \mathbb{E}[\mathbf{P}]\mathbb{E}[\mathbf{Q}]^{-1}, \qquad (4)$$

where $\mathbb{E}[\mathbf{P}]_{ij} = \mathbf{\Sigma}_{ij}\mathbf{q}_j$ and $\mathbb{E}[\mathbf{Q}]_{ij} = \mathbf{\Sigma}_{ij}\mathbf{q}_i\tau(i,j,\mathbf{q}_j)$. Here $\tau(i,j,\mathbf{q}_j)$ returns 1 with $i=j$, and returns $\mathbf{q}_j$ with $i \neq j$. After getting the mapping matrix, $\mathbf{B}\mathbf{W}^{\mathrm{T}}$ is used as the new representation. By this means, we can recursively apply marginalized Denoising Auto-Encoder to obtain deep hidden representations. Finally, K-means is called to run on the hidden representations for the consensus partition. Since only r elements are non-zeros in each row of $\mathbf{B}$, it is very efficient to calculate $\mathbf{\Sigma}$. Moreover, $\mathbb{E}[\mathbf{P}]$ and $\mathbb{E}[\mathbf{Q}]$ are both $(d+1)\times(d+1)$ matrixes. Finally, K-means is conducted on all the hidden representations. Therefore, our total time complexity is $O(ld^3 + IKnld)$, where l is the number of layers of mDAE, I is the iteration number in K-means, K is the cluster number, and $d = \sum_{i=1}^{r} K_i \ll n$. This indicates our algorithm is linear to n, which can be applied for large-scale clustering. Since K-means is the core technique in IEC, the convergence is guaranteed.

5. EXPERIMENTAL RESULTS

In this section, we first introduce the experimental settings, then showcase the effectiveness and efficiency of IEC compared with the state-of-the-art deep clustering and ensemble clustering methods. Finally, some impact factors of IEC are thoroughly explored.

5.1 Experimental Settings

Data Sets. 13 real-world image data sets with true cluster labels are used for experiments. Table 1 shows their important characteristics, where #MinClass, #MaxClass, CV and Density denote the instance number of the smallest and biggest clusters, Coefficient of Variation statistic that characterizes the degree of class imbalance, and the ratio of non-zeros elements, respectively. In order to demonstrate the effectiveness of our IEC, we select the data sets with different levels of features, such as pixel, Surf and deep learning features. The first two are characters and digits data sets[1], the middle ones are the objects and digits data sets[2] [3] and the last four data sets are with the deep learning features[4]. In addition, these data sets contain different types of images, such as digits, characters, objects. Figure 4 shows some samples of these data sets.

Comparative algorithms. To validate the effectiveness of the IEC, we compare it with several state-of-the-art methods in terms of deep clustering methods and ensemble clustering methods.

- **K-means** is the baseline method.
- **MAEC** [10] applies mDAE to get new representations and runs K-means on it to get the partition. Here MAEC1 uses the orginal features as the input, MAEC2 uses the Laplace graph as the input.
- **GEncoder** [40] is short for GraphEncoder, which feeds the Laplace graph into the sparse auto-encoder to get new representations.
- **DLC** [37] jointly learns the feature transform function and discriminative codings in a deep mDAE structure.
- **GCC** [39] is a general concept of three benchmark ensemble clustering algorithms based on graph: CSPA, HGPA and MCLA, and returns the best result.

[1]http://archive.ics.uci.edu/ml.

[2]https://www.eecs.berkeley.edu/~jhoffman/domainadapt.

[3]http://www.cad.zju.edu.cn/home/dengcai.

[4]http://www.cs.dartmouth.edu/~chenfang.

Table 2: Clustering Performance of Different Algorithms Measured by Accuracy

Data Sets	Baseline	Deep Clustering Method				Ensemble Clustering Method				
	K-means	MAEC1	MAEC2	GEncoder	DLC	GCC	HCC	KCC	SEC	IEC (Ours)
letter	0.2485	0.1163	N/A	N/A	**0.3087**	0.2598	0.2447	0.2461	0.2137	0.2633
MNIST	0.4493	0.3757	N/A	N/A	0.5498	0.5047	0.4458	0.6026	0.5687	**0.6086**
COIL100	0.5056	0.0124	0.5206	0.0103	0.5348	0.5382	0.5332	0.5032	0.5210	**0.5464**
Amazon	0.3309	**0.4395**	0.2443	0.2004	0.3653	0.3486	0.3069	0.3434	0.3424	0.3904
Caltech	0.2457	0.2787	0.2102	0.2333	0.2840	0.2289	0.2386	0.2618	0.2680	**0.2983**
Dslr	0.3631	0.4140	0.3185	0.2485	0.4267	0.4268	0.3949	0.4395	0.4395	**0.5159**
Webcam	0.3932	0.5085	0.3220	0.3430	**0.5119**	0.4305	0.3932	0.4203	0.4169	0.4983
ORL	0.5475	0.0450	0.3675	0.2050	0.5775	**0.6300**	0.6025	0.5450	0.5850	**0.6300**
USPS	0.6222	0.6290	0.4066	0.1676	0.6457	0.6211	0.6137	0.6857	0.6157	**0.7670**
Caltech101	0.6898	0.4311	0.5060	0.7753	0.7583	0.5152	0.7336	0.7611	0.9025	**0.9866**
ImageNet	0.6675	0.6601	0.3483	0.2892	0.6804	0.5765	0.7054	0.5986	0.6571	**0.7075**
Sun09	0.4360	0.4750	0.3696	0.3854	0.4829	0.4424	0.4235	0.4473	0.4732	**0.4899**
VOC2007	0.4565	0.4138	0.3874	0.4443	0.5130	0.5195	0.5044	**0.5364**	0.5124	0.5178

Table 3: Clustering Performance of Different Algorithms Measured by NMI

Data Sets	Baseline	Deep Clustering Method				Ensemble Clustering Method				
	K-means	MAEC	MAEC2	GEncoder	DLC	GCC	HCC	KCC	SEC	IEC (Ours)
letter	0.3446	0.1946	N/A	N/A	**0.3977**	0.3444	0.3435	0.3469	0.3090	0.3453
MNIST	0.4542	0.3086	N/A	N/A	0.5195	0.4857	0.5396	0.4651	0.5157	**0.5420**
COIL100	0.7719	0.0769	0.7794	0.0924	0.7764	0.7725	0.7815	0.7761	0.7786	**0.7866**
Amazon	0.3057	**0.3588**	0.1982	0.0911	0.3001	0.2882	0.3062	0.2947	0.2595	0.3198
Caltech	0.2043	0.1862	0.1352	0.1132	0.2104	0.1774	0.2094	0.2031	0.1979	**0.2105**
Dslr	0.3766	0.4599	0.2900	0.1846	0.4614	0.4113	0.4776	0.4393	0.4756	**0.5147**
Webcam	0.4242	0.5269	0.2316	0.3661	**0.5280**	0.4344	0.4565	0.4502	0.4441	0.5201
ORL	0.7651	0.2302	0.6268	0.4431	0.7771	0.7987	0.7970	0.7767	0.7858	**0.8050**
USPS	0.6049	0.4722	0.4408	0.0141	0.5843	0.6219	0.5187	0.6363	0.5895	**0.6409**
Caltech101	0.7188	0.4980	0.5200	0.6922	0.7669	0.6536	0.7747	0.7881	0.8747	**0.9504**
ImageNet	0.4287	0.4827	0.1556	0.0064	0.4117	0.3902	**0.4375**	0.4366	0.4366	0.4358
Sun09	0.2014	**0.2787**	0.0576	0.0481	0.2315	0.2026	0.2091	0.1803	0.1927	0.2197
VOC2007	0.2697	0.2653	0.1118	0.1920	0.2651	0.2588	0.2564	0.2607	0.2511	**0.2719**

- **HCC** [16] is an agglomerative hierarchical clustering algorithm based on the co-association matrix.
- **KCC** [45] is a K-means-based consensus clustering which transfers the ensemble clustering into a K-means optimization problem.
- **SEC** [29] employs spectral clustering on co-association matrix and solves it by weighted K-means.

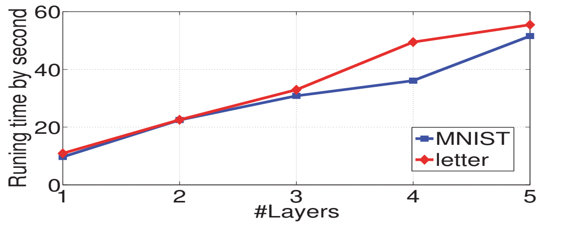
In the ensemble clustering framework, we employ Random Parameter Selection (RPS) strategy to generate basic partitions. Generally speaking, *k-means* is conducted on all features with different numbers of clusters, varying from K to $2K$. To show the best performance of the comparative algorithms, 100 basic partitions via RPS are produced for boosting the comparative methods. Note that we set 5 layers in deep clustering methods and for all clustering methods, we set K to be the true cluster number for fair comparison.

Validation metric. Since the label information is available to these data sets, here we use two external metrics *accuracy* and *Normalized Mutual Information* (NMI) to measure the performance. Note that accuracy and NMI are both positive measurements, which means the larger, the better. The computation details can be found in [46].

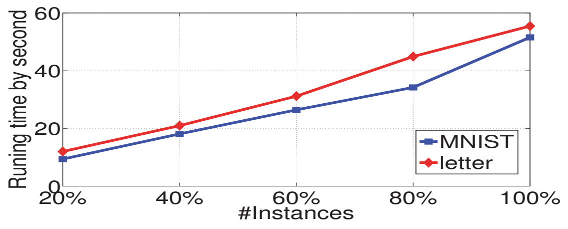
Environment. All the experiments were run on a Windows standard platform of 64-bit edition,which has two Intel Core i7 3.4GHz CPUs and 32GB RAM.

5.2 Clustering Performance

Table 2 and 3 show the clustering performance of different algorithms in terms of accuracy and NMI. The best results are highlighted in bold font. "N/A" denotes there is no result due to out of memory. As can be seen from the tables, three observations are very clear. (1) In the deep clustering method, MAEC1 performs the best and the worst on *Amazon* and *COIL100*, respectively; on the contrary, MAEC2 gets reasonable result on *COIL100* but low quality on *Amazon*. Although we try our best to tune the number of neurons in the hidden layers, GEncoder suffers from the worst performance in all the comparative methods, even worse than K-means. The high computational cost prohibits MAEC2 and GEncoder from handling large-scale data sets. Since clustering belongs to the unsupervised learning, only re-



(a) Different numbers of layers



(b) Different numbers of instances

Figure 5: Running time with different layers and instances.

Table 4: Execution time of different ensemble clustering methods by second

Data sets	GCC	HCC	KCC	SEC	IEC (5 layers)
letter	383.89	1717.88	11.39	8.35	55.46
MNIST	112.44	19937.69	11.98	3.79	51.55
COIL100	21.27	170.02	4.99	3.09	14.93
Amazon	3.93	1.61	0.17	0.08	1.21
Caltech	3.55	2.12	0.23	0.11	1.43
Dslr	2.27	0.09	0.04	0.06	0.70
Webcam	2.09	0.14	0.04	0.05	0.90
ORL	6.81	0.04	0.21	0.21	14.11
USPS	7.66	160.41	1.73	0.53	5.48
Caltech101	1.21	1.68	0.15	0.09	0.53
ImageNet	3.83	52.47	1.40	0.32	1.76
Sun09	2.36	10.01	0.33	0.13	0.82
VOC2007	2.05	10.97	0.32	0.16	0.82

lying on deep structure makes little effect to improve the performance. Instead DLC jointly learns the feature transform function and discriminative codings in a deep structure, which has the satisfactory results. (2) In most cases, ensemble clustering is superior to the baseline method, even better than deep clustering methods. The improvement is obvious when applying ensemble clustering methods on the data sets with high-level features, since high-level features have more structural information. However, ensemble methods do not work well on *SUN09*. One of the reasons might be the unbalanced class structure, which prevents the basic clustering algorithm K-means from uncovering the true structure and further harms the performance of ensemble methods. (3) Our method IEC gets the best results on most of 13 data sets. It is worthy to note that the improvements are over nearly 8%, 8% or 22% on *Dslr*, *USPS* and *Caltech101*, respectively, which are rare in clustering field. Usually the performance of ensemble clustering goes up with the increase the number of basic partitions. In order to show the best performance of the comparative ensemble clustering methods, we use 100 basic partitions. Here we can see that there still exists large space to improve via infinite ensemble members.

For efficiency, to make fair comparisons here we only report the execution time of ensemble clustering methods. Although additional time is needed for generating basic partitions, *k-means* and parallel computation make it quite efficient. Table 4 shows the average time of ten runs via these methods. GCC runs three methods on small data sets but runs two methods on large data sets, and HCC runs fast on data sets containing few instances but struggles as the number of instances increases due to its $O(n^3)$ time complexity. KCC, SEC and IEC are all K-means-based methods, which are much faster than other ensemble methods. Since our method only applies mDAE on basic partitions which has the closed-form solution and then runs K-means on the new representations, therefore IEC is suitable for large-scale image clustering. Moreover, Figure 5 shows the running time on *MNIST* and *letter* with different number of layers and instances. We can see that the running time is linear to the layer number and instant number, which verifies the high efficiency of IEC. Therefore, if we only use one layer in IEC, the execution time is similar to KCC and SEC.

5.3 Inside IEC: Factor Exploration

Next we thoroughly explore the impact factors of IEC in terms of the number of layers, the generation strategy of basic partitions, the number of basic partitions, and the noise level, respectively.

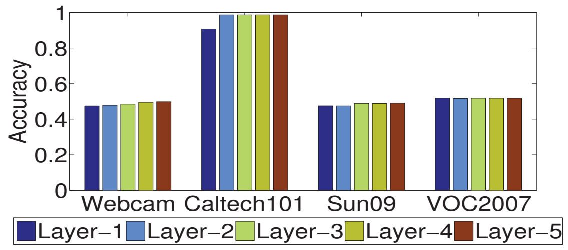


Figure 6: Performance of IEC with different layers.

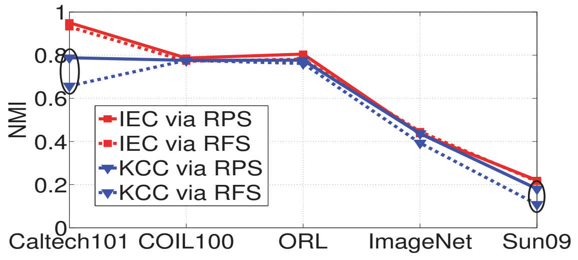


Figure 7: Impact of basic partition generation strategies.

Number of layers. Since stacked marginalized Denoising Auto-Encoder is used to fuse infinite ensemble members, here we explore the impact of the number of layers. As can be seen in Figure 6, the performance of IEC goes slightly up with the increase of layers. Except that the second layer has large improvements over the first layer on *Caltech101*, IEC demonstrates the stable results on different layers, because only one-layer marginalized Denoising Auto-Encoder calculates the expectation of co-association matrix. Usually the deep representation is successful in many applications on computer vision, here the default value of the number of layers is set to be 5.

Generation strategy of basic partitions. So far we rely solely on Random Parameter Selection (RPS) to generate basic partitions, with the number of clusters varying in $[K, 2K]$. In the following, we demonstrate whether the generation strategy will impact the performance of IEC.

Here Random Feature Selection (RFS) is proposed as a comparison, which still uses *k-means* as the basic clustering algorithm with random selecting 50% original features to obtain 100 basic partitions. Figure 7 demonstrates the performance of KCC and IEC via RPS and RFS on 5 data sets. As we can see that, IEC exceeds KCC in most cases of RPS and RFS. When we take a close look, the performance of IEC via RPS and RFS is almost the same, while KCC produces large gaps between RPS and RFS on *Caltech101* and *Sun09* (See the ellipses). This indicates that although the generation of basic partitions is of high importance to the success of ensemble clustering, we can take use of infinite ensemble clustering to alleviate the impact.

Number of basic partitions. The key problem of this paper is to use limited basic partitions to achieve the goal of fusing infinite ensemble members. Here we discuss the impact of the number of basic partitions to ensemble clustering. Figure 8 shows the performance of 4 ensemble clustering methods on *USPS*. Generally speaking, the performance of HCC, KCC and GCC goes up with the increase

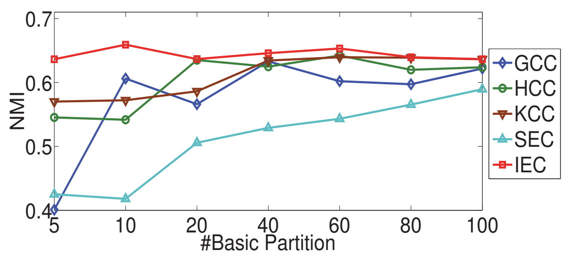


Figure 8: Impact of the number of basic partitions via different ensemble methods on *USPS*.

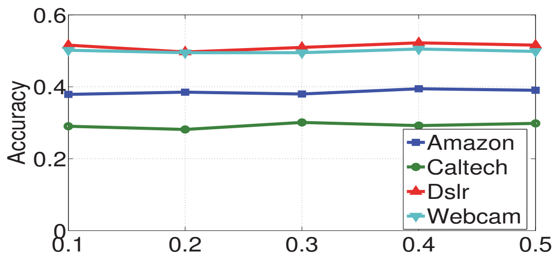


Figure 9: Performance of IEC with different noise levels.

of the number of basic partitions and becomes stable when enough basic partitions are given, which is consistent with Theorem 1. It is worthy to note that IEC enjoys the high performance even with 5 ensemble members. It is worthy to note that for large-scale data sets, generating basic partition suffers from high time complexity even with ensemble process. Thus, it is appealing that IEC uses limited basic partitions and achieves the high performance, which is suitable for tons of image clustering.

Noise level. The core idea of this paper is to obtain the expectation of co-association matrix via adding the drop-out noise. Figure 9 shows the results of IEC with different noise level on four data sets. As can be seen that, the performance of IEC is quite stable even to 0.5 noise level. Note that if we set the noise level to zero, IEC will equivalently degrade into KCC.

6. CONCLUSION

In this paper, we proposed a novel ensemble clustering algorithm Infinite Ensemble Clustering (IEC) for image clustering. Generally speaking, we built a connection between ensemble clustering and auto-encoder, and applied marginalized Denoising Auto-Encoder to fuse infinite incomplete basic partitions. Extensive experiments on 13 data sets with different levels of features demonstrated our method IEC had promising performance over the state-of-the-art deep clustering and ensemble clusterings methods; besides, we thoroughly explored the impact factors of IEC in terms of the number of layers, the generation strategy of basic partitions, the number of basic partitions, and the noise level to show the robustness of our method.

7. ACKNOWLEDGEMENT

This research is supported in part by the NSF CNS award 1314484, ONR award N00014-12-1-1028, ONR Young Investigator Award N00014-14-1-0484, and U.S. Army Research Office Young Investigator Award W911NF-14-1-0218. We thank KDD anonymous reviewers and SPC for their constructive comments, which help to improve this work to a new level in the final revision.

8. REFERENCES

[1] Y. Avrithis, Y. Kalantidis, E. Anagnostopoulos, and I. Emiris. Web-scale image clustering revisited. In *Proceedings of International Conference on Computer Vision*, 2015.

[2] H. Ayad and M. Kamel. Cumulative voting consensus method for partitions with variable number of clusters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(1):160–173, 2008.

[3] Y. Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[4] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[5] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Proceedings of Advances in Neural Information Processing Systems*, 2007.

[6] T. Berg and A. C. Berg. Finding iconic images. In *Proceedings of Computer Vision and Pattern Recognition Workshops*, 2009.

[7] M. Carreira-PerpinÃạn and R. Raziperchikolaei. Hashing with binary autoencoders. In *Proceedings of Computer Vision and Pattern Recognition*, 2015.

[8] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[9] M. Chen, K. Weinberger, F. Sha, and Y. Bengio. Marginalized denoising autoencoders for nonlinear representation. In *Proceedings of International Conference on Machine Learning*, 2014.

[10] M. Chen, Z. Xu, K. Weinberger, and F. Sha. Marginalized stacked denoising autoencoders for domain adaptation. In *Proceedings of International Conference on Machine Learning*, 2012.

[11] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of International Conference on Computer Vision*, 2005.

[12] Z. Ding, M. Shao, and Y. Fu. Deep low-rank coding for transfer learning. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2015.

[13] C. Domeniconi and M. Al-Razgan. Weighted cluster ensembles: Methods and analysis. *ACM Transactions on Knowledge Discovery from Data*, 2(4):17, 2009.

[14] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013.

[15] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, et al. Building rome on a cloudless day. In *Proceedings of Euroupean Conference on Computer Vision*, 2010.

[16] A. Fred and A. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, 27(6):835–850, 2005.

[17] M. Ghifary, W. Kleijn, M. Zhang, and D. Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of International Conference on Computer Vision*, 2015.

[18] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12):2916–2929, 2013.

[19] Y. Gong, M. Pawlowski, F. Yang, L. Brandy, L. Boundev, and R. Fergus. Web scale photo hash clustering on a single machine. In *Proceedings of Computer Vision and Pattern Recognition*, 2015.

[20] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[21] P. Huang, Y. Huang, W. Wang, and L. Wang. Deep embedding network for clustering. In *Proceedings of International Conference on Pattern Recognition*, 2014.

[22] M. Kan, S. Shan, H. Chang, and X. Chen. Stacked progressive auto-encoders (spae) for face recognition across poses. In *Proceedings of Computer Vision and Pattern Recognition*, 2014.

[23] G. Kim, L. Sigal, and E. P. Xing. Joint summarization of large-scale collections of web images and videos for storyline reconstruction. In *Proceedings of Computer Vision and Pattern Recognition*, 2014.

[24] G. Kim and E. P. Xing. Reconstructing storyline graphs for image recommendation from web community photos. In *Proceedings of Computer Vision and Pattern Recognition*, 2014.

[25] Y. J. Lee and K. Grauman. Shape discovery from unlabeled image collections. In *Proceedings of Computer Vision and Pattern Recognition*, 2009.

[26] S. Li, Y. Jiang, and Z. Zhou. Partial multi-view clustering. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2014.

[27] T. Li, D. Chris, and M. Jordan. Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In *Proceedings of International Conference on Data Mining*, 2007.

[28] H. Liu and Y. Fu. Clustering with partition level side information. In *Proceedings of International Conference on Data Mining*, 2015.

[29] H. Liu, T. Liu, J. Wu, D. Tao, and Y. Fu. Spectral ensemble clustering. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015.

[30] H. Liu, J. Wu, D. Tao, Y. Zhang, and Y. Fu. Dias: A disassemble-assemble framework for highly sparse text clustering. In *Proceedings of SIAM International Conference on Data Mining*, 2015.

[31] T. Liu and D. Tao. On the performance of manhattan nonnegative matrix factorization. *IEEE Transactions on Neural Networks and Learning Systems*, 2015.

[32] T. Liu, D. Tao, and D. Xu. Dimensionality-dependent generalization bounds for k-dimensional coding schemes. *arXiv preprint arXiv:1601.00238*, 2016.

[33] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[34] Z. Lu, Y. Peng, and J. Xiao. From comparing clusterings to combining clusterings. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2008.

[35] D. Luo, C. Ding, H. Huang, and F. Nie. Consensus spectral clustering in near-linear time. In *Proceedings of International Conference on Data Engineering*, 2011.

[36] R. Raguram and S. Lazebnik. Computing iconic summaries of general visual concepts. In *Proceedings of Computer Vision and Pattern Recognition*, 2008.

[37] M. Shao, S. Li, Z. Ding, and Y. Fu. Deep linear coding for fast graph clustering. In *Proceedings of International Joint Conference on Artificial Intelligence*, 2015.

[38] C. Song, F. Liu, Y. Huang, L. Wang, and T. Tan. Auto-encoder based data clustering. *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 117–124, 2013.

[39] A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining partitions. *Journal of Machine Learning Research*, 3:583–617, 2003.

[40] F. Tian, B. Gao, Q. Cui, E. Chen, and T. Liu. Learning deep representations for graph clustering. In *Proceedings of AAAI Conference on Artificial Intelligence*, 2014.

[41] A. Topchy, A. Jain, and W. Punch. Combining multiple weak clusterings. In *Proceedings of International Conference on Data Mining*, 2003.

[42] A. Topchy, A. Jain, and W. Punch. A mixture model for clustering ensembles. In *Proceedings of SIAM International Conference on Data Mining*, 2004.

[43] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedingse of International Conference on Machine Learning*, 2008.

[44] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proceedings of Advances in Neural Information Processing Systems*, 2009.

[45] J. Wu, H. Liu, H. Xiong, J. Cao, and J. Chen. K-means-based consensus clustering: A unified view. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):155–169, 2015.

[46] J. Wu, H. Xiong, and J. Chen. Adapting the right measures for k-means clustering. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009.

[47] G.-S. Xie, X.-Y. Zhang, and C.-L. Liu. Efficient feature coding based on auto-encoder network for image classification. In *Proceedings of Asian Conference on Computer Vision*, 2015.

[48] H. Yoon, S. Ahn, S. Lee, S. Cho, and J. Kim. Heterogeneous clustering ensemble method for combining different cluster results. *Data Mining for Biomedical Applications*, 2006.