Image Clustering Based on Deep Sparse Representations

Le Lv*, Dongbin Zhao† and Qingqiong Deng‡
*† The State Key Laboratory of Management and Control for Complex Systems
Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China
‡ College of Information Science and Technology, Beijing Normal University, Beijing, 100875, China
Email: *lvle2012@ia.ac.cn, †dongbin.zhao@ia.ac.cn, ‡qqdeng@bnu.edu.cn

***Abstract*—Currently, the supervised trained deep neural networks (DNNs) have been successfully applied in several image classification tasks. However, how to extract powerful data representations and discover semantic concepts from unlabeled data is a more practical issue. Unsupervised feature learning methods aim at extracting abstract representations from unlabeled data. Large amount of research works illustrate that these representations can be directly used in the supervised tasks. However, due to the high dimensionality of these representations, it is difficult to discover the categorical concepts among them in an unsupervised way. In this paper, we propose combining the winner-take-all autoencoder with the bipartite graph partitioning algorithm to cluster unlabeled image data. The winner-take-all autoencoder can learn the additive sparse representations. By the experiments, we present the properties of the sparse representations. The bipartite graph partitioning can take full advantage of them and generate semantic clusters. We discover that the confident instances in each cluster are well discriminated. Based on the initial clustering result, we further train a support vector machine (SVM) to refine the clusters. Our method can discover the categorical concepts rapidly and the experiment shows that the clustering performance of our method is good.**

I. Introduction

Currently, deep learning is the state of art method in the computer vision field. Especially, the supervised trained DNNs have achieved huge successes in several image classification tasks. However, in many practical applications, it is much easier to access large amount of unlabeled data. Hence, how to extract powerful representations and discover semantic concepts from unlabeled data have received extensive attentions.

Unsupervised feature learning is a key problem in the deep learning. In the early researches, restricted Boltzmann machines (RBM) [1]–[3], regularized autoencoders [4] et al are proposed to capture the salient statistic factors in the probability distribution. Through learning to reconstruct the input signals, the neural networks extract sparse or distributed representations. In those research works, some important theoretical results are obtained. However, in the field of computer vision, the convolutional neural networks (CNNs) [5] may be the most powerful tool. The convolutional architecture reduces the number of parameters and extracts shift invariant features.

In order to improve the performance of DNNs further, many researchers combine the pre-training methods with the convolutional architecture. In [6], Makhzani et al propose the convolutional winner-take-all autoencoder (CONV-WTA-AE). It can extract deep sparse representations to reconstruct the input signals. Compared with the previous sparsity regularization methods [7], the CONV-WTA-AE does not need to infer the sparse representations iteratively. Hence, it can be trained more efficiently. Additionally, the representations extracted by the CONV-WTA-AE can be directly used for the classification task without the fine-tuning operation.

At present, researchers still know little about the properties of the representations extracted by DNNs. In [8], the clustering algorithms are used to analyze the properties of the deep representations. Inspired by these works, we propose applying the bipartite graph partitioning algorithm [9] to analyze the sparsity of the winner-take-all representations. Actually, the input signals are represented as the sparse additive combination of the basis vectors of the decoder. By the clustering analysis, we present two important properties of the extracted features. (1) The components of the deep sparse representations have clear semantic meanings. (2) The visual similar images share a group of visual elements. Intuitively, these properties are the natural results of the winner-take-all constraints. In our work, the effectiveness of the bipartite graph partitioning supports our notion. The co-clustering algorithm associates the instances in different input regions with a private set of parameters [10]. Generally, the spectral clustering can be divided into two steps. The first step is to over-partition the whole dataset and the second step is to merge the clusters according to some criterions. In our experiments, we discover that the similarity metric used by the co-clustering algorithm is not very accurate. Only the confident instances in a cluster are well discriminated. Hence, we propose using linear SVMs to refine the clustering results. Detailedly, we use the confident instances in each cluster as the initial data to train a linear SVM. Then the SVM will provide a better discriminative similarity. The assignments for each cluster will be updated. To avoid overfitting, this process is iterated on many different data batches. In the end, we merge the similar SVM classifiers. We conduct the experiments on the MNIST dataset. The clustering performance is good. In [11] and [12], the deep representations are sparsified without decreasing the classification accuracy. Hence, if only the sparsity assumption is satisfied, our method can also be generalized to other deep architectures.

The rest part of this paper is organized as follows. In the section 2, 3, and 4, we will describe the details of our method.

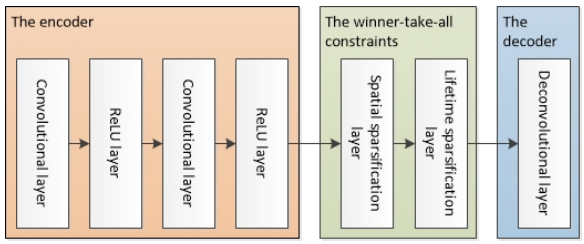

Fig. 1. The architecture of the CONV-WTA-AE.

The experiment results are reported in the section 5. At last, we conclude our works in the section 6.

II. Convolutional Winner-Take-All Autoencoder

Unsupervised feature learning is an important research field in the deep learning. Autoencoders may be the most widely used feature extractors. In order to avoid the neural networks learning the trivial representations, many regularization methods are proposed. Due to the successful applications in the computer vision, CNNs are another important method in deep learning. The convolutional architecture can reduce the redundancy in image information. There are many variants which combine the autoencoder with the convolutional architecture. The CONV-WTA-AE is one of the most outstanding variants. In [6], the representations extracted by the CONV-WTA-AE are used for the classification task. In our work, it is extended into the clustering task. In Fig. 1, We show the architecture of the CONV-WTA-AE. Then we will analyze its features briefly.

A. The Non-Symmetric Autoencoder

The CONV-WTA-AE is a non-symmetric autoencoder. Its encoder is stacked by several rectified linear unit (ReLU) layers and convolutional layers. Due to the lack of pooling layers, the feature maps output by each layer have the same size as the input image. The decoder is only a linear deconvolutional layer. This architecture has two advantages.

1) The learning capacity of the multi-layer encoder is much stronger than the single layer encoder: To extract more expressive features, the regularization methods are becoming more and more complicated. The single layer encoders are too simple to satisfy this requirement. Owing to the stronger nonlinearity, the multi-layer encoder is capable to learn more complex and abstract representations. Hence, the multi-layer architecture draws the attentions of researchers gradually.

2) The linear deconvolutional layer makes the components of representations explainable: The representations can be transformed back into the input space linearly. Hence, the component of representation can be seen as the input signal's projection on the basis vector. Intuitively, the deconvolutional kernels correspond to the visual elements which are statistic significant. In addition, the ReLUs generate non-negative representations. This is also an essential requirement for the clustering part of our method.

B. The Sparsity Regularization

The convolutional architecture learns local features and reduces the redundancy caused by similar feature appearing at different positions. The local features' degree of freedom is much lower than the global features'. But when there are too many adjustable convolutional filters, the convolutional autoencoder is still easy to learn trivial convolutional kernels (such as delta function). Hence, the regularization is indispensable.

The CONV-WTA-AE regularizes the pre-training process by sparsifying the output representations of encoder. But in the classification task, the representations are directly used without any sparsification. The CONV-WTA-AE involves two sparsity constraints: the spatial sparsity and the lifetime sparsity. The spatial sparsity constraint suppresses the non-maximum values in a feature map. The lifetime sparsity considers the statistics of data batch. Given a feature map, we sort the spatial maximum values of each image in a mini-batch data. Then we reserve the top-k values and set others to zero. In other words, these two sparsity constraints reserve the maximum value along the data batch dimension and the spatial dimensions. In the backward step, the gradient information is propagated only through those reserved units.

Most of the sparsity regularization methods need to infer the optimal sparse coding iteratively. Hence, their training processes are very inefficient. For example, in [7], the single layer neural network is trained to fit the mapping from the input signals to their sparse codes. During each parameters update, a coordinate descent algorithm should be executed to infer the sparse code. However, it is easy to implement the winner-take-all sparsity constraints. The sparsification only involves a sorting operation.

III. Bipartite Graph Partitioning

Generally speaking, for the sparse representations, there exists a duality relation between the input signals and the basis vectors of decoder. A category of objects are often described by some co-occurred image patterns. By considering the statistic of data batch, the lifetime sparsity allows a part of instances to share a basis vector. The sharing is built on the magnitude of activations. If the image contains a pattern which maximizes the activation of the hidden unit, it will be associated with the basis vector. Based on this notion, we pose the clustering task as the bipartite graph partitioning problem. Similar with the practice in [6], the final representations used in our clustering method are the max-pooled outputs of the encoder. The duality of the sparse representations is the most important foundation for our proposed method. Actually, in [11], [12], researchers find that the sparsified representations only sacrifices little classification performance. Hence, if the assumption that the discriminative information is mostly stored in the dimension indices of representations is satisfied, then our method are still applicable.

A. Graph Partitioning

First, we review some relevant terminologies. A graph is a set of vertices and edges. A vertex is denoted by v and the set of vertex is denoted by V. Each edge e connects two vertices v_i and v_j. Hence, we can also denote the edge e as its vertices pair (v_i, v_j). The set of edges is indicated by E. Hence, the graph can be denoted by $\{V, E\}$. Each edge in the graph is associated with a weight W_{ij}. A bipartitioning of the vertex set V is two subsets V_1 and V_2 which satisfy $V_1 \bigcup V_2 = V$ and $V_1 \bigcap V_2 = \emptyset$. The cut of a bipartitioning is the sum of edge weights which connect vertices belong to different subset. The cut can be formulated as:

$$cut(V_1, V_2) = \sum_{v_i \in V_1, v_j \in V_2} W_{ij}$$

In many realistic problems, it is desirable to separate the vertex set into two nearly equally-sized subset meanwhile the cut between them should be as least as possible. This goal can be formulated as minimizing the objective function N.

$$N(V_1, V_2) = \frac{cut(V_1, V_2)}{weight(V_1)} + \frac{cut(V_1, V_2)}{weight(V_2)}$$

where $weight(V_l) = \sum_{v_i \in V_l} \sum_{v_k \in V} W_{ik}$ for $l = 1, 2$. This objective function is often referred as the normalized cut criterion. It can be reformulated as:

$$N(V_1, V_2) = 2 - S(V_1, V_2)$$
$$S(V_1, V_2) = \frac{within(V_1)}{weight(V_1)} + \frac{within(V_2)}{weight(V_2)}$$

where $within(V_l) = \sum_{v_i \in V_l} \sum_{v_k \in V_l} W_{ik}$ for $l = 1, 2$. $S(V_1, V_2)$ measures the strength of associations within all subsets. Minimizing the normalized cut is equivalent to maximizing the intra-class associations. Based on the equivalence, for the multipartitioning task, we should maximize the following objective function:

$$S(V_1, V_2, ..., V_k) = \sum_{i=1}^{k} \frac{within(V_i)}{weight(V_i)}$$

B. Spectral Graph Partitioning

The graph partitioning is a NP-complete problem. The spectral partitioning is an effective heuristic method [13]. In order to explain the principle of spectral partitioning, we introduce some algebraic description methods for graph. For the convenience of discussion, we suppose that the graph $G = \{V, E\}$ has n vertices and m edges. Then we have the following definitions.

Definition 1. The Laplacian matrix L is a $n \times n$ matrix. The matrix element L_{ij} obeys that:

$$L_{ij} = \begin{cases} \sum_k W_{ik}, & i = j \\ -W_{ij}, & i \neq j \wedge \exists (v_i, v_j) \in E \\ 0, & otherwise \end{cases}$$

First, we will talk about the graph bipartitioning problem. Then we will generalize it to the multipartitioning problem.

Definition 2. Given a bipartitioning of V into V_1 and V_2, the generalized partition vector q is defined as:

$$q_i = \begin{cases} +\sqrt{\frac{\eta_2}{\eta_1}} & v_i \in V_1 \\ -\sqrt{\frac{\eta_1}{\eta_2}} & v_i \in V_2 \end{cases}$$

where $\eta_l = weight(V_l)$ for $l = 1, 2$.

Theorem 1.

$$\frac{q^T L q}{q^T D q} = \frac{cut(V_1, V_2)}{weight V_1} + \frac{cut(V_1, V_2)}{weight V_2}$$

where D is the diagonal degree matrix meeting that $D_{ii} = \sum_k W_{ik}$.

Inspired by Theorem 1, although it is a NP-complete problem to find the optimal generalized partition vector, the real relaxation to optimal solution can be solved efficiently.

Theorem 2. *The problem*

$$\min_{q \neq 0} \frac{q^T L q}{q^T D q}, \quad s.t. \quad q^T W e = 0$$

is solved when q is the eigenvector corresponding to the second smallest eigenvalue λ_2 of the generalized eigenvalue problem

$$Lx = \lambda D x.$$

These theorems are the classical conclusions from the algebraic graph theory. According to the eigenvector q, we can partition the graph approximately.

There are two ways to generalize the bipartitioning method to the multipartitioning problem. The first one is to apply the bipartitioning method recursively to generate k subsets and then merge them to k' subsets ($k > k'$). The second is a more efficient approach. Instead of solving the eigenvalues and eigenvectors of $Lx = \lambda Dx$, the singular value decomposition (SVD) of $H = D^{-1/2} L D^{-1/2}$ is computed. The k eigenvectors corresponding to the k smallest eigenvalues form the matrix $X = [x_1, x_2, ..., x_k]$. Then the row vectors of X are normalized to unit length. The row vectors of X can be seen as points in R^k. We can cluster them. Each row vector correspond to a vertex in the graph. In the end, the clusters are merged.

C. Bipartite Graph partitioning

We model the associations between the input image and the visual elements of decoder as a bipartite graph. Each input image corresponds to a vertex and the image vertices' set is denoted by $U = \{u_1, u_2, ..., u_m\}$. Each visual element of decoder also corresponds to a vertex and the visual element vertices' set is denoted by $V = \{v_1, v_2, ..., v_n\}$. The components of representations extracted by the CONV-WTA-AE describe the association strength between the input images u_i and the visual elements v_j. Hence, they are used as the weight value of the edge (u_i, v_j). Obviously, there are no edges connecting the vertices in the same set. Hence, the set of edge E can be denoted by $\{(u_i, v_j) | u_i \in U, v_j \in V\}$. The bipartite graph is denoted by a triple $G = \{U, V, E\}$. The

bipartite graph is only a particular case of the ordinary graph. Hence, the method described above can be directly applied to it. However, due to the adjacency feature of bipartite graph, we can further improve the efficiency of the algorithm.

Because there only exist connections between vertices in different sets, we can define the association matrix A. It is a $m \times n$ matrix which the rows correspond to vertices in U and the columns correspond to vertices in V. The element A_{ij} meets that:

$$A_{ij} = \begin{cases} W_{ij} & \exists(u_i, v_j) \in E \\ 0 & otherwise \end{cases}$$

Hence, the Laplacian matrix L and the degree matrix D can be rewritten as:

$$L = \begin{pmatrix} D_1 & -A \\ -A^T & D_2 \end{pmatrix}, \quad D = \begin{pmatrix} D_1 & 0 \\ 0 & D_2 \end{pmatrix}.$$

$Lx = \lambda Dx$ can be rewritten as:

$$D_1^{-1/2} A D_2^{-1/2} y = (1-\lambda) z \tag{1}$$

$$D_2^{-1/2} A^T D_1^{-1/2} z = (1-\lambda) y \tag{2}$$

where $y = D_2^{1/2} x_2$ and $z = D_1^{1/2} x_1$. y, z are the right and left singular vectors of $A_n = D_1^{-1/2} A D_2^{-1/2}$. The vector $[z^T, y^T]^T$ is the singular vectors of H. Generally, the size of H is much larger than A_n. Hence, the computational cost to solve the SVD of A_n is much less. The modified algorithm is shown in Algorithm 1. Given the bipartite graph, we construct the association matrix A and the degree matrix D_1, D_2. The SVD of A_n is solved. Remarkably, in Algorithm 1, we choose the singular vectors corresponding to the biggest singular values. According to (1), the singular value of A_n is equal to $(1-\lambda)$ where λ is the eigenvalues of H. Hence, the smallest eigenvalues of H correspond to the biggest singular values of A_n. We form the matrix X by stacking the right and left singular vectors. Furthermore, we note that Z, Y describe the multi-modal information of the images and the visual elements respectively. Then we normalize the row vectors of X and cluster them. In (1), given the right singular vector, we can infer the left singular vector and vice versa. This is a explain for that the clusters of images and visual elements can be implied each other.

Many methods have been proposed to merge the overpartitioned clusters. In our work, we discover that the clustering result generated by spectral clustering algorithm is not accurate enough. However, the confident instances are well discriminated. Hence, we propose a complete different method to merge clusters in the next section.

Algorithm 1 The spectral partitioning algorithm for bipartite graph

1: **Input:** The graph $G = \{U, V, E\}$, the number of overpartitioned subsets k, and the number of partitioned subsets k'.
2: Construct the association matrix A and the degree matrix D_1, D_2.
3: Compute the SVD of $A_n = D_1^{-1/2} A D_2^{-1/2}$, find k right and left singular vectors $z_1, z_2, ..., z_k$ and $y_1, y_2, ..., y_k$ of A_n (which correspond to the k biggest singular values), and form the matrix

$$X = \begin{pmatrix} Z \\ Y \end{pmatrix}$$

by stacking the right and left singular vectors in columns.
4: Normalize the row vectors of X to unit length.
5: Apply K-means algorithm to cluster the row vectors of X.
6: The row vectors in Z and Y corresponds to the vertices in U and V respectively. Based on the clusters of row vectors of X, the vertices in the graph G can be clustered.
7: Merge k subsets into k' clusters.
8: **Output:** The k' clusters of vertices.

IV. Unsupervised Discriminative Clustering

The clustering result is not very accurate. We can consider that the spectral clustering contains two steps. The first step is the dimensionality reduction. The second step is the common clustering algorithm. Although the clustering is efficient, the cluster results are not well discriminated. In our experiments, the representations extracted by CONV-WTA-AE are more than 10,000 dimensions. The number of clusters we used is less than 50. Hence, large amount of information may be lost in the first step. However, we discover that the confident instances in each cluster are well discriminated. We should refine the clustering results and merge them. In [14], Singh et al claim that, due to the simplicity of similarity metric, most popular clustering algorithm can't deal with high dimensional data effectively. The supervised trained classifiers usually provide a good similarity metric but it is inferred from the labeled data. Singh et al proposed an important idea whether we can start from an initial clustering and train a classifier to generate a better similarity metric. Then the similarity metric can be used to compute well discriminated clusters. Through iterating this operation, the accuracy of clustering will be refined. However, this method easily causes the overfitting problem. Hence, the dataset should be divided into several subsets. Each greedy improvement on a subset should be generalized to other subsets. Based on this idea, we design an algorithm to refine the clustering results. We show it in Algorithm 2. First, we choose confident instances in the initial clusters. The initial cluster C_i obtained by spectral graph partitioning contains images and visual elements. The confidence of the image is computed by its associations with the visual elements in the same cluster. Then, we divide the representation dataset into m subdatasets. We train a linear SVM classifier for each cluster. Given the previous classifier, we can compute the signed distances of the samples to the hyperplane. According to the confidences, the positive and negative instances are reassigned and the classifier is retrained. The number of positive instances is increased. In Algorithm 2, the size of each cluster are assumed to be equal. Hence, when we have enough positive instances, we can stop the iterations. We compute the predictions of classifier on the training set

Algorithm 2 The unsupervised discriminative clustering algorithm

Input: The representation vectors of training data and their initial cluster labels. The number of initial clusters is n_c.
The n_{min} most confident instances in each cluster are chosen.
The set of representation vectors D is divided into m subsets $D_1, D_2, ..., D_m$.
while $i = 1 : n_c$ **do**
  The confident instances in the cluster C_i are chosen as the positive instances and other confident instances are chosen as the negative instances, then a linear SVM clf_i is trained.
  while $j = n_{min} : inc : n_{max}$ **do**
    while $k = 1 : m$ **do**
      Given the classifier clf_i, the confidence (the signed distance of the sample to the hyperplane) of instances in D_k are computed.
      The j most confident instances are chosen as the positive instances and other instances are chosen as the negative instances, then clf_i is re-trained.
    end while
  end while
end while
If the prediction results of the classifiers clf_i and the classifiers clf_j are similar, we will merge these two clusters.
Output: A set of SVM classifiers.

and compare it with previous classifiers. If eighty percent of the predicted results are the same as a previous classifier, we will discard the current classifier.

V. Experiment

We verify our method on the MNIST dataset. This dataset contains 60,000 hand-written digit images as training data and 10,000 images as test data. The size of images is 28×28. The total number of categories is 10.

First, we train the CONV-WTA-AE to extract the representations. We refer the architecture proposed in [6]. There is a drawback in the training process of CONV-WTA-AE. At the beginning of training, the sparsity regularizations act on the representations. Due to the randomness of parameters, the representations extracted by encoder may be meaningless. The winner-take-all constraints discard most of the components. The decoder has to use a limited number of randomly initialized basis vectors to reconstruct the input signal. The reconstruction error is big. However, at the beginning of the training of other sparsity regularization methods such as PSD, the random dictionary matrix leads the representations dense. The reconstruction error is little and many parameters can be trained. Hence, the early training stage of CONV-WTA-AE is not very efficient. Indeed, the big reconstruction error often causes the training divergent. For the ReLUs, the big learning rate often causes that some units are activated a big value and some units are always inactivated. Too many big activation values cause overflow in the calculation. Too many inactivated units cause the decrease of learning capacity. Therefore, the learning rate of stochastic gradient descent (SGD) has to be set at a small value. In [15], leaky ReLU (LReLU) is proposed. It can speed up the optimization. Hence, we train an encoder which is a stack of LReLU convolutional layers firstly. Then we switch the LReLU to ReLU. We show the training curve in Fig. 2. The vertical axis show the reconstruction error. The mean square error is used. We take a iteration on all training data as one epoch and the reconstruction error is averaged over the epoch. The switch happened at the 30-th epoch. We can see that the fluctuation of training process is not obviously. We observe the learning curve in real time. When the curve achieves a smooth stage, we will decrease the learning rate correspondingly. The training is stopped until the reconstruction error is convergent.

Fig. 2. The training curve of the CONV-WTA-AE.

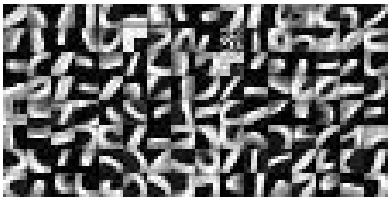

Fig. 3. The visualized deconvolutional kernel.

We also visualized the deconvolutional kernel in Fig. 3. These kernels look like the "strokes" of the digit characters. They have clearly semantic meaning. Hence, intuitively, the sparse representations denote the association between the input image and the visual elements.

The bipartite graph partitioning algorithm is applied to obtain the initial clustering instances. Generally, we need overpartition the graph first. For MNIST dataset, we know its actual class number in advance. Hence, we use a process of cross-validation to determine the number of clusters. We test k in $\{10, 20, 40\}$. On one hand, increasing k can discriminate the instances in different categories better. On the other hand, with the increase of k, Algorithm 2 will also cost more computation time. Hence, we set the number of clusters to be 20. The

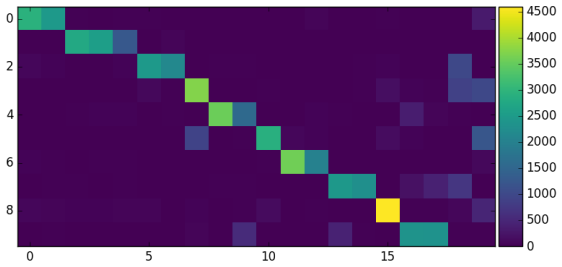

Fig. 4. The visualized confusion matrix between clustering results and real categories.

Fig. 5. The top-10 confident instances in each cluster.

confusion matrix between the clustering results and the real categories is visualized in Fig. 4. The rows index the real categories and the columns index the clusters. The color of each cell corresponds to the number of instances.

The instances can't be clustered well. We can see the last cluster include the instances belong to "3" and "5". If we assign a real category to each cluster, the recognition accuracy is 83.6%.

In Fig. 5, the confident instances in each cluster are plotted in one column. We can see that the confident instances are well discriminated. Particularly, the confident instances in the last cluster are all the character "5". Based on these confident instances, we apply Algorithm 2 to refine and merge the clusters. By cross-validation, we set the parameters $m = 6$, $n_{min} = 30$, $inc = 10$, $n_{max} = 800$. Finally, we obtain 10 clusters and the recognition accuracy on the test set achieves 95%. We use scikit-learn [16] to implement these algorithms. The experiments are run on a computer with Intel i5-2400 at 3.1 GHz CPU. The bipartite graph partitioning algorithm is very efficient. Its running only need 20 minutes. Because a set of linear SVMs should be trained in each iteration, the learning process of Algorithm 2 costs more time. The whole clustering algorithm is completed in 4 hours.

VI. Conclusion

In this paper, we apply the bipartite graph partitioning algorithm to the sparse representations extracted by CONV-WTA-AE. The duality between the visual elements and the input images can be used to cluster the images. The CONV-WTA-AE can't directly deal with large images. It is natural to use CONV-WTA-AE to study the properties of image patches. Hence, in the future, we will apply our method to discover the semantic mid-level image patches.

Acknowledgment

This work is supported by National Natural Science Foundation of China (NSFC) under Grants No. 61273136, No. 61573353 and No. 61533017.

References

[1] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[2] M. Zorzi, A. Testolin, and I. P. Stoianov, "Modeling language and cognition with deep unsupervised learning: a tutorial overview." *Frontiers in Psychology*, vol. 4, pp. 515–515, 2013.

[3] Q. V. Le, "Building high-level features using large scale unsupervised learning," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, 2013, pp. 8595–8598.

[4] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, 2008, pp. 1096–1103.

[5] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[6] A. Makhzani and B. J. Frey, "Winner-take-all autoencoders," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2015, pp. 2791–2799.

[7] K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "Fast inference in sparse coding algorithms with applications to object recognition," *CoRR*, vol. abs/1010.3467, 2010.

[8] L. V. Der Maaten and G. E. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[9] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," *Knowledge Discovery and Data Mining*, 2001.

[10] Y. Bengio, A. C. Courville, and P. Vincent, "Unsupervised feature learning and deep learning: A review and new perspectives," *CoRR*, vol. abs/1206.5538, 2012.

[11] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 2892–2900.

[12] Y. Li, L. Liu, C. Shen, and A. van den Hengel, "Mid-level deep pattern mining," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 971–980.

[13] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, 2001, pp. 849–856.

[14] S. Singh, A. Gupta, and A. A. Efros, "Unsupervised discovery of mid-level discriminative patches," in *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part II*, 2012, pp. 73–86.

[15] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, Georgia, 16-21 June 2013*, 2013.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.