Multi-Modal Deep Clustering:
Unsupervised Partitioning of Images

Guy Shiran  
*School of Computer Science and Engineering*  
*Hebrew University of Jerusalem*  
Jerusalem, Israel  
guy.shiran@mail.huji.ac.il

Daphna Weinshall  
*School of Computer Science and Engineering*  
*Hebrew University of Jerusalem*  
Jerusalem, Israel  
daphna@cs.huji.ac.il

Abstract—The clustering of unlabeled raw images is a daunting task, which has recently been approached with some success by deep learning methods. Here we propose an unsupervised clustering framework, which learns a deep neural network in an end-to-end fashion, providing direct cluster assignments of images without additional processing. Multi-Modal Deep Clustering (MMDC), trains a deep network to align its image embeddings with target points sampled from a Gaussian Mixture Model distribution. The cluster assignments are then determined by mixture component association of image embeddings. Simultaneously, the same deep network is trained to solve an additional self-supervised task. This pushes the network to learn more meaningful image representations and stabilizes the training. Experimental results show that MMDC achieves or exceeds state-of-the-art performance on four challenging benchmarks. On natural image datasets we improve on previous results with significant margins of up to 11% absolute accuracy points, yielding an accuracy of 70% on CIFAR-10 and 61% on STL-10.

I. INTRODUCTION

Clustering involves the organization of data in an unsupervised manner, based on the distribution of datapoints and the distances between them. Since these properties are closely tied to the representation of the data, the problems of clustering and data representation are firmly connected and are therefore sometimes solved jointly. In accordance, in this work we start from a recent method for the unsupervised computation of effective data representation (or features discovery), and develop a clustering method whose results significantly improve the state of the art in the clustering of natural images. The method is illustrated in Fig 1.

A large body of work has been devoted to the problem of clustering [20], see Section II for a brief review of some recent related work.

The task of unsupervised image clustering is challenging and interesting, as the algorithm needs to discover patterns in highly entangled data, and produce separated groups without explicitly specifying the grouping features. In recent years, with the emergence of deep learning as the method of choice in visual object recognition and image classification, emphasis has shifted to the computation of effective representations that will support successful clustering [29].

Vice versa, unsupervised clustering loss has been used to drive the computation of image representation and the discovery of enhanced image features by making it possible to use unsupervised data in the training of deep networks, which traditionally require massive amounts of labeled data.

When learning feature representation from unsupervised data by minimizing a clustering-based loss function, one danger is cluster collapse - the representation may collapse to the trivial solution of a single cluster. In [3], a similar problem of representation collapse is managed by mapping the network’s representation to a fixed set of randomly chosen points in some target features space. Here we borrow this mapping idea, and incorporate it into a clustering algorithm.

More specifically, we first sample a fixed set of points in some target space. Since our method is designed to partition the data into $k$ clusters, the target points are chosen from a matched density function - Gaussian Mixture Model (GMM) with $k$ components. Our model trains a randomly initialized neural network to align its image embeddings with the sampled
target points, directly inducing a partition that is based on the mixture components. This is done by simultaneously learning a one-to-one mapping between the output of the network and the target points, and updating the networks parameters to best fit images with their target points as assigned by the mapping.

In the absence of ground truth, the proposed approach is prone to instability as target points are continuously reassigned between images, creating a non-stationary online learning environment. Such instability is often linked with unsupervised learning tasks. To alleviate this problem, unsupervised tasks such as representation learning may be combined with self-supervision tasks to achieve better results [10]. Here we adopt the approach taken by [7] to deal with the notorious instability of training generative adversarial networks. Thus the model is jointly trained on the main clustering task and on a self-supervised auxiliary task as defined in RotNet [14], where all images are subjected to 4 rotation angles. In this auxiliary task the network is trained to recognize the $2D$ rotation of the image.

For computation engine, our method uses off the shelf convnets and standard SGD training with mini-batch sampling in an end-to-end fashion. It is therefore scalable to large datasets. We evaluate our method on several standard benchmarks in image clustering, which is the goal of our method, and achieve or exceed state of the art results. The same set of hyper-parameters is used in all the reported results.

The rest of the paper is organized as follows: In Section II we briefly review recent related work. In Section III we describe our method in detail and elaborate on its various ingredients. Experimental results are reported in Section IV.

II. RELATED WORK

Data clustering. The objective of data clustering is to partition data points into groups such that points in each group are more similar to each other than to data points in the other groups. Traditionally, clustering methods have been divided into density-based methods [23], partition-based methods [12], and hierarchical methods [11]. Partition-based methods, such as the popular k-means [1], [31], minimize a given clustering criterion by iteratively relocating data points between clusters until a (locally) optimal partition is attained. Density-based methods define clusters as areas with high density of points, separated by areas with low density of points [36]. Hierarchical based methods build a hierarchy of clusters in a top-to-bottom [33] or bottom-to-top [16] manner to determine clustering.

Representation Learning. Naively attempting to cluster images with traditional approaches does not produce pleasing partitions of the images, as they work on the raw representations of the images in pixel space, and semantically similar images are not necessarily similar in the high-dimensional pixel space the images reside in. In recent years learning useful image representations in an unsupervised manner has been dominated by deep-learning-based approaches. Autoencoders (AEs) [2] encode images with a deep network and are trained by reconstructing the image using a decoder network. These include several variations such as sparse AEs, denoising AEs [35], and more [28], [41]. Generative models such as Generative Adversarial Networks (GAN) [15] and variational autoencoders (VAE) [22] learn representations as a byproduct of learning to generate images. Tightly connected to our work, Noise-As-Targets (NAT) [3] and DeepCluster [4] adopt a training strategy of iteratively reassigning pseudo-labels to points while training the network to fit them (see Section III).

Self-supervised learning. A family of unsupervised learning algorithms that gained popularity in recent years are self-supervised methods. They learn representations by training a deep network to solve a pretext task, where labels can be produced directly from the data. Such tasks can be jigsaw puzzle solving [30], predicting the relative position of patches in an image [9], generating image regions conditioned on their surroundings [32], or more recently predicting image rotations (RotNet) [14]. In self-supervised GANs [7], predicting image rotations is used as an auxiliary task to stabilize and improve training, by enhancing the discriminators representation capabilities. This is an approach we adopt as well and elaborate on later on.

Deep clustering. The dominant and most successful approach to clustering of images in recent years has been to incorporate the tasks of representation learning and clustering into a single framework. Prominent works in the past years have been Joint Unsupervised Learning (JULE) [40], where the authors adopt an agglomerative clustering approach by iteratively merging clusters of deep representations and updating the networks parameters. Deep Adaptive Clustering (DAC) [5] recasts the clustering problem into a binary pairwise-classification framework, where cosine distances between image features of image pairs are used as a similarity measure to decide if they belong to the same cluster. Associative Deep Clustering (ADC) [17] jointly learns network parameters and embedding centroids with an association loss in order to estimate cluster membership. More recently, Invariant Information Clustering (IIC) [21] adopts an approach that achieves clustering based on maximizing the mutual information between two sets: deep embeddings of images, and instances of the images that underwent random image transformations while keeping the image semantic meaning intact. IIC leverages auxiliary overclustering to increase expressivity in the learned feature representation, improving the representation capabilities of its network. This tactic bears resemblance to our incorporation of rotation prediction as an auxiliary task.

III. METHOD

Our task is to partition a set of images into $k$ clusters, which reflect internal structure in the data. Fig. 2 shows an overview of the proposed approach. The algorithm alternates between solving the main unsupervised clustering task, and an auxiliary self-supervised task that helps the training process. The ingredients of the method are described next. The full method is summarized in Algorithm 1. Code is provided in the supplementary material.
A. Unsupervised learning

The starting point for this work is an unsupervised learning framework for learning image representation from unlabeled data. The method, Noise as Targets [3], learns useful representations of images by training a deep network to align its images’ embeddings with a fixed set of target points. The target points are uniformly scattered on the unit sphere.

More specifically, let \( X = \{x_i\}_{i=1}^n \) denote a set of images, and \( f_\theta : X \to Z \) the parameterized deep network we wish to train. The output of \( f_\theta \) is normalized to have \( \ell_2 \) norm of 1, entailing that \( Z \) is the \( d \)-dimensional unit sphere. NAT starts by uniformly sampling \( n \) targets on this unit sphere. Let \( \{t_i\}_{i=1}^n \) denote the set of target points, which remain fixed throughout the training. Each image \( x_i \) is assigned a unique target \( y_i \), through a permutation \( P : [n] \to [n] \). The optimization objective is formulated as

\[
\min_{\theta, P} \frac{1}{n} \sum_{i} \ell(f_\theta(x_i), y_i) \quad \text{subject to} \quad y_i = t_{P(i)}
\]

where \( \ell \) is the Euclidean distance.

This optimization problem is solved in a stochastic manner, by iteratively solving it for mini-batches. Given a mini-batch of images \( X_b \), the current representation vectors \( f_\theta(X_b) \) are first computed. Subsequently, Equation (1) is optimized for \( P \) over the points in mini-batch \( X_b \) using the Hungarian method [25], which reassigns the currently assigned targets of the mini-batch to minimize the Euclidean distance (\( \ell_2 \)) between images and their assigned target points. Finally, the gradients of \( f_\theta \) on \( X_b \) with respect to \( \theta \) are computed, and an SGD step is executed.

Intuitively, NAT permutes the assignment of image representation vectors to target points delivered by \( f_\theta \), so that nearby embedding vectors are mapped to nearby target vectors, and then updates \( \theta \) accordingly. This process leads to the grouping of semantically similar images in target space, and to effective representations that perform well in downstream computer vision classification and detection tasks.

B. Multi-modal distribution of target points

The uniform distribution of target points on the unit sphere, as described above, is not well suited for unsupervised clustering, since it is likely to blur the dividing lines between clusters rather than sharpen them. Instead, multi-modal distribution seems like a natural choice for the objective of clustering, as it directly produces separated groups in target space.

In this work, we propose to use the mixture of Gaussians distribution, projected to the unit sphere, for the sampling of target points. Formally, this implies:

\[
p(u) = \sum_{k=1}^{K} \alpha_k \cdot p_k(u) \quad u \in \mathbb{R}^d
\]

\[
p(t_i) = \frac{1}{\int_{u \in \mathbb{R}^d} p(u) du} \int_{\left\{ u \mid \frac{\sqrt{\alpha_k}}{\sqrt{\sum_{k=1}^{K} \alpha_k}} u = t_i \right\}} p(u) du \quad t_i \in Z
\]

where \( K \) denotes the number of Gaussians in the mixture, \( d \) the dimension of the embedding space, \( \alpha_{k=1..K} \) a categorical random variable, and \( p_k(u) \) the multivariate normal distribution \( N(\mu_k, \Sigma_k) \), parameterized by mean vector \( \mu_k \) and covariance matrix \( \Sigma_k \). In the absence of prior knowledge we assume that the mixture components are equally likely, namely \( \alpha_k = \frac{1}{K} \forall k \in [K] \). Finally, since the target points are constrained to lie on the unit sphere, we project the sample in \( \mathbb{R}^d \) to the unit sphere by \( t_i = \frac{u}{\|u\|} \).

We define the cluster assignment \( c_i \) of image \( x_i \) as follows

\[
c_i = \arg \min_k \| f_\theta(x_i) - \mu_k \|_2
\]

Note that if the final network \( f_\theta \) fits that target points exactly, namely \( f_\theta(x_i) = y_i \), and if \( \Sigma_k \) are the same \( \forall k \), then with high probability \( c_i \) is the index of the mixture component from which target point \( y_i \) has been sampled.
Data augmentation is a useful and common technique to improve performance of machine learning algorithms. Usually, random image transformations such as cropping, flipping, rotation, scaling and photometric transformations are applied to images in order to expand the dataset with new and unique images. In our task of unsupervised clustering, these random transformations are essential, because they provide several images. In our task of unsupervised clustering, these random image transformations such as cropping, flipping, improve performance of machine learning algorithms. Usually, we employ RotNet [14], which is a self-supervised learning algorithm that learns image features by training a convnet to predict image rotations. Specifically, images are rotated by r degrees (r ∈ {0°, 90°, 180°, 270°}), and the model is subsequently trained to predict their rotation by optimizing the cross-entropy loss. RotNet produces competitive performance in representation learning benchmarks, and has been shown to benefit training in other tasks, when incorporated into a model as an auxiliary task [7, 13, 27]. We incorporate RotNet into our method, modifying the convnet training procedure to alternate between optimizing the main clustering task and this secondary auxiliary task.

E. Refinement Stage

As we have no prior knowledge regarding the size of the clusters, we begin by assuming that clusters’ sizes are equal. When this assumption cannot be justified, we propose to augment the algorithm with an additional step, performed after the main training is concluded. In this step the assumption is relaxed, while target points are iteratively reassigned based on the outcome of k-means applied to fθ(x1), ..., fθ(xn), and assigning image xi to target μj with label j ∈ [K] derived from the outcome of k-means. This method is similar to DeepCluster [4], proposed by Caron et al. as an approach for representation learning, where they perform the clustering on the latent vectors of the model and not the final output layer. A possible alternative method may start with this stage and discard the first one altogether, as this approach makes no assumption on the size of the clusters. However, we found that starting off with reassigning labels based on k-means is not competitive and produces less accurate clusters, for example, training on MNIST results in an accuracy of 81% (±2.67).

IV. Experiments

We tested our method on several image datasets that are commonly used as benchmark for clustering, see results in Table I. We compare ourselves to state-of-the-art methods such as DeepCluster [4], which is a self-supervised learning algorithm that learns image features by training a convnet to predict image rotations. Specifically, images are rotated by r degrees (r ∈ {0°, 90°, 180°, 270°}), and the model is subsequently trained to predict their rotation by optimizing the cross-entropy loss. RotNet produces competitive performance in representation learning benchmarks, and has been shown to benefit training in other tasks, when incorporated into a model as an auxiliary task [7, 13, 27]. We incorporate RotNet into our method, modifying the convnet training procedure to alternate between optimizing the main clustering task and this secondary auxiliary task.

E. Refinement Stage

As we have no prior knowledge regarding the size of the clusters, we begin by assuming that clusters’ sizes are equal. When this assumption cannot be justified, we propose to augment the algorithm with an additional step, performed after the main training is concluded. In this step the assumption is relaxed, while target points are iteratively reassigned based on the outcome of k-means applied to fθ(x1), ..., fθ(xn), and assigning image xi to target μj with label j ∈ [K] derived from the outcome of k-means. This method is similar to DeepCluster [4], proposed by Caron et al. as an approach for representation learning, where they perform the clustering on the latent vectors of the model and not the final output layer. A possible alternative method may start with this stage and discard the first one altogether, as this approach makes no assumption on the size of the clusters. However, we found that starting off with reassigning labels based on k-means is not competitive and produces less accurate clusters, for example, training on MNIST results in an accuracy of 81% (±2.67).
The architecture is a ResNet model [18] with 34 layers. These bases include a pooling layer and ReLU activation function, and ends with a max normalization [19]. Each block in this neural network consists of two convolutions, each followed by a batch normalization layer and ReLU activation function, and ends with a max pooling layer. Our model has three blocks, except for STL-10, where we use four because images are larger. The second architecture is a ResNet model [18] with 34 layers. These base architectures are followed by a linear prediction layer, that outputs a prediction \( \theta \) for the ResNets. We use batch size of 128 and perform random image augmentations which include cropping, flipping and color jitter. When training on the clustering task, each mini-batch is augmented by color jitter. When training on the auxiliary rotation task, each mini-batch is augmented by random transformations, where \( \theta \) differs.

### A. Implementation details and evaluation scores

#### Datasets.

Five different datasets are used in our empirical study. MNIST [26], Fashion-MNIST [38], CIFAR-10 [24], the 20 superclasses of CIFAR-100 [24], and STL-10 [8], see Table IV-A. We are most interested in the last 3 datasets because they consist of natural images. These datasets are commonly used to evaluate clustering methods.

#### Architectures.

We describe experiments with two convnet architectures. The first is a VGG model [34] with batch normalization [19]. Each block in this neural network consists of two convolutions, each followed by a batch normalization layer and ReLU activation function, and ends with a max pooling layer. Our model has three blocks, except for STL-10, where we use four because images are larger. The second architecture is a ResNet model [18] with 34 layers. These base architectures are followed by a linear prediction layer, that outputs a prediction \( \theta \) for the ResNets. We use batch size of 128 and perform random image augmentations which include cropping, flipping and color jitter. When training on the clustering task, each mini-batch is augmented by random transformations, where \( \theta \) differs.

#### Mixture of Gaussians.

We examined several initialization heuristics to determine the Gaussian means \( \{ \mu_i \} \) in the GMM distribution defined in (2), and concluded that using \( K \) differ-

---

### Table I

**Unsupervised clustering results.** The results of our method are shown below the separation line, based on one of two convnet architectures - VGG or ResNet (see Section IV-A). In each case, we show the average result over multiple runs and standard error (ste). We also show ensemble-based results based on two selection criteria - lowest loss and majority vote (see Section IV-B). Above the separation line we list state-of-the-art results for comparison, see review in Section II. Unreported results are marked with (-). Each experiment was repeated 5 times.

<table>
<thead>
<tr>
<th></th>
<th>MNIST</th>
<th>Fashion</th>
<th>CIFAR-10</th>
<th>CIFAR-100</th>
<th>STL-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-means</td>
<td>0.490</td>
<td>0.572</td>
<td>0.511</td>
<td>0.471</td>
<td></td>
</tr>
<tr>
<td>AE+k-means</td>
<td>0.725</td>
<td>0.812</td>
<td>0.526</td>
<td>0.555</td>
<td></td>
</tr>
<tr>
<td>Spectral Clustering</td>
<td>0.663</td>
<td>0.696</td>
<td>0.630</td>
<td>0.551</td>
<td></td>
</tr>
<tr>
<td>DEC (2016)</td>
<td>0.772</td>
<td>0.843</td>
<td>0.546</td>
<td>0.518</td>
<td></td>
</tr>
<tr>
<td>JULE (2016)</td>
<td>0.913</td>
<td>0.964</td>
<td>0.608</td>
<td>0.563</td>
<td></td>
</tr>
<tr>
<td>DAC (2017)</td>
<td>0.935</td>
<td>0.978</td>
<td>0.632</td>
<td>0.615</td>
<td></td>
</tr>
<tr>
<td>ADC (2018)</td>
<td>-</td>
<td>0.987</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>IIC (2019)</td>
<td>-</td>
<td>0.992</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DCCM (2019)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

**Table II**

The image datasets used in our experiments.

<table>
<thead>
<tr>
<th>Name</th>
<th>Classes</th>
<th>Samples</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>10</td>
<td>70,000</td>
<td>28 x 28</td>
</tr>
<tr>
<td>Fashion-MNIST</td>
<td>10</td>
<td>70,000</td>
<td>28 x 28</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>10</td>
<td>60,000</td>
<td>32 x 32 x 3</td>
</tr>
<tr>
<td>CIFAR-100</td>
<td>20</td>
<td>60,000</td>
<td>32 x 32 x 3</td>
</tr>
<tr>
<td>STL-10</td>
<td>10</td>
<td>13,000</td>
<td>96 x 96 x 3</td>
</tr>
</tbody>
</table>
ent one-hot vectors in $\mathbb{R}^K$ is most effective. A comparison of different initialization schemes is provided in Table III, all vectors lie on the $d$-dimensional unit sphere. Initialization schemes 2 and 3 are sampled from a multi-variate uniform distribution within the range $[-0.1, 0.1]$ and projected onto the unit sphere. Other than varying the mean vectors, the training procedure is the same, and does not include the refinement stage. We set $\Sigma_k = 0.05 \cdot I_{K \times K} \ \forall k \in [K]$.

**Evaluation scores.** To evaluate clustering performance we adopt two commonly used scores: Normalized Mutual Information (NMI), and Clustering Accuracy (ACC). Clustering accuracy measures the accuracy of the hard-assignment to clusters, with respect to the best permutation of the dataset’s ground-truth labels. Normalized Mutual Information measures the mutual information between the ground-truth labels and the predicted labels based on the clustering method. The range of both scores is [0, 1], where a larger value indicates more precise clustering results. We use center crops of images for evaluation.

**B. Ensemble of networks**

Different executions of the algorithm using the same dataset lead to non-negligible variations in clustering quality as evaluated by the two clustering scores - ACC and NMI. This happens due to the stochastic nature of the method, with the random initialization of parameters, mini-batch sampling and random image transformations. A clustering ensemble can be beneficial (cf. [6]), and we propose two methods to improve the clustering results based on an ensemble of networks. We report results with both methods in Table I.

**Lowest loss.** When comparing the train loss of the network with its clustering scores in multiple runs, we noticed a significant positive correlation between them as shown in Figure 5. A possible explanation for this empirical result is that if the mapping $P : [n] \rightarrow [n]$ assigns an image to a bad target point, its neighbors are likely to be points from another class, and the convnet’s attempt to generalize will not produce a good fit for that point resulting in higher loss.

**Majority vote.** We implement majority voting as a strategy to combine the solutions of a clustering ensemble. Without ground-truth labels this task is not trivially solved, as the absolute clustering labels are arbitrary and do not necessarily

<table>
<thead>
<tr>
<th>Initialization</th>
<th>ACC</th>
<th>NMI</th>
</tr>
</thead>
<tbody>
<tr>
<td>one-hot $(d=10)$</td>
<td>97.47 (±0.11)</td>
<td>94.16 (±0.19)</td>
</tr>
<tr>
<td>uniform $(d=10)$</td>
<td>96.6 (±0.44)</td>
<td>92.68 (±0.66)</td>
</tr>
<tr>
<td>uniform $(d=128)$</td>
<td>93.04 (±1.59)</td>
<td>88.96 (±1.12)</td>
</tr>
</tbody>
</table>
align between multiple clustering solutions. To circumvent this, we fix one solution, specifically the one with the lowest loss. We then find for each other clustering solution the permutation that most agrees with the fixed one, using once again the Hungarian algorithm.

C. Analysis

The results of our method when applied to five image datasets are reported in Table I. Clearly, our clustering algorithm is able to separate unlabeled images into distinct groups of semantically similar images with high accuracy, improving the state-of-the-art in three of the more challenging datasets. Using an ensemble of clustering networks almost always improves the results over the average. We use the same ResNet architecture and the same image augmentations as [21], so the improvement in clustering quality can only be attributed to the clustering idea.

Figure 4 shows the confusion matrix obtained by the algorithm when partitioning the very challenging CIFAR-10 dataset, without using its labels. The majority of images in each cluster correspond to one of the 10 object classes in a unique manner, where most of the false assignments correspond to images from semantically similarly looking objects such as cats and dogs, deer and horse, or airplanes and ships. It is interesting that the smaller VGG architecture performs better on the STL-10 dataset as compared to ResNet, possibly because the more limited capacity of the smaller VGG model is able to generalize better on the smaller dataset.

The auxiliary task in our method, which is added to the clustering task, is based on image rotation and is designed for natural images. We use it to enhance the clustering of CIFAR-10, CIFAR-100 and STL-10.

D. Ablation study

Next we describe the results of an ablation study examining the critical components of the algorithm. We start with the basic model based on VGG, without contractive regularization (CR), and without optimizing the auxiliary task. We then add CR. Then we add the auxiliary task, training to predict rotation angle. Finally, we replace the simpler VGG architecture with the bigger and more effective ResNet architecture, which requires more resources to train. Even though the NAT and RotNet methods do not produce a clustering of the data, we examine their capabilities by training them and clustering the latent space with k-means. We use the CIFAR-10 dataset as our study case. Results are shown in Table IV-D, illustrating how each component contributes to the performance of the final algorithm. In Figure 6 we show how the training curves change with each added component.

Refinement stage. We compare clustering performance with and without the proposed refinement stage in Table IV-D. MNIST is the only dataset with class imbalance, as its smallest class has 6313 samples while its largest has 7877. The refinement stage helps the algorithm achieve near perfect clustering with accuracy of 99.1%.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.971</td>
<td>0.991</td>
</tr>
<tr>
<td>Fashion</td>
<td>0.721</td>
<td>0.723</td>
</tr>
<tr>
<td>CIFAR-10</td>
<td>0.700</td>
<td>0.697</td>
</tr>
<tr>
<td>CIFAR-100</td>
<td>0.312</td>
<td>0.287</td>
</tr>
<tr>
<td>STL-10</td>
<td>0.611</td>
<td>0.606</td>
</tr>
</tbody>
</table>

TABLE IV

Ablation study. Each row corresponds to the addition of one component in our method, where "ResNet + CR + RotNet" corresponds to the final configuration. We also show clustering results using NAT and RotNet alone.
V. SUMMARY

For the task of unsupervised semantic image clustering, we presented an end-to-end deep clustering framework, that trains a convnet to align image embeddings with targets sampled from a Gaussian Mixture Model. To achieve effective training, we employed contractive regularization and incorporated an additional auxiliary task - the prediction of image rotation. Our ablation study shows that the contribution of each component of the method is essential for the success of the method. Even though the proposed method is quite simple, it yields a significant improvement on previous state-of-the-art methods such as IIC while using the same convnet architecture. The variant of our method, which is based on running the algorithm repeatedly and choosing the solution that achieves the minimal training loss, leads to a very significant improvement over the state of the art on two datasets: for CIFAR-10 accuracy is improved from 62% to 70%, and for STL-10 accuracy is improved from 50% to 61%.

REFERENCES