Joint Unsupervised Learning of Deep Representations and Image Clusters

# Jianwei Yang; Devi Parikh; Dhruv Batra

## 2016

A recurrent framework for joint unsupervised learning of deep representations and image clusters by integrating two processes into a single model with a unified weighted triplet loss function and optimizing it end-to-end can obtain not only more powerful representations, but also more precise image clusters.

# Abstract

In this paper, we propose a recurrent framework for Joint Unsupervised LEarning (JULE) of deep representations and image clusters. In our framework, successive operations in a clustering algorithm are expressed as steps in a recurrent process, stacked on top of representations output by a Convolutional Neural Network (CNN). During training, image clusters and representations are updated jointly: image clustering is conducted in the forward pass, while representation learning in the backward pass. Our key idea behind this framework is that good representations are beneficial to image clustering and clustering results provide supervisory signals to representation learning. By integrating two processes into a single model with a unified weighted triplet loss and optimizing it end-to-end, we can obtain not only more powerful representations, but also more precise image clusters. Extensive experiments show that our method outperforms the state-of-the-art on image clustering across a variety of image datasets. Moreover, the learned representations generalize well when transferred to other tasks.

# Study subjects

**20 subjects**

For FRGC-v2.0 and YTF datasets, we first crop faces and then resize them to a constant size. **In FRGCv2.0 dataset, we randomly choose 20 subjects**. As for YTF dataset, we choose the first 41 subjects which are sorted by their names in alphabet order

# Scholarcy Synopsis

This paper proposes a recurrent framework called JULE for joint unsupervised learning of deep representations and image clusters, where image clustering and representation learning are integrated into a single model to obtain more powerful representations and more precise image clusters.  
The method outperforms the state-of-the-art on image clustering and the learned representations generalize well to other tasks.

Yang and colleagues (2016) propose a simple but effective end-to-end learning framework to jointly learn deep representations and image clusters from an unlabeled image set; 2 The team formulate the joint learning in a recurrent framework, where merging operations of agglomerative clustering are expressed as a forward pass, and representation learning of CNN as a backward pass; 3 They derive a single loss function to guide agglomerative clustering and deep representation learning, which makes optimization over the two tasks seamless; 4 The experimental results show that the proposed framework outperforms previous methods on image clustering and learns deep representations that can be transferred to other tasks and datasets.  
They have proposed an approach to jointly learn deep representations and image clusters.  
They are witnessing an explosion in visual content.  
I = {I1, ..., Ins }, the global objective function for learning image representations and clusters can be written as: argmin L(y, θ|I).  
Let the sequence of optimal image cluster labels be given by Yp∗ = {y∗t }, and clusters merged in forward pass are denoted by aim to derive the optimal θ to minimize the losses generated in forward pass.  
  
20 subjects were involved in the study.   
  
Yang and colleagues recommend that the text suggests introducing a loss to confine the within-cluster structure, but this is left as future work due to limited space.

# Findings

Extensive experiments show that our method outperforms the state-of-the-art on image clustering across a variety of image datasets

# Scholarcy Highlights

* We aim to address image clustering and representation learning on unlabeled images in a unified framework
* It is a natural idea to leverage cluster ids of images as supervisory signals to learn representations and in turn the representations would be beneficial to image clustering
* Clustering with representations from a Convolutional Neural Network (CNN) initialized with random weights are not reliable, but nearest neighbors and over-clusterings are often acceptable; 2) These over-clusterings can be merged as better representations are learned; 3) Agglomerative clustering is a recurrent process and can naturally be interpreted in a recurrent framework
* The major contributions of our work are: 1 We propose a simple but effective end-to-end learning framework to jointly learn deep representations and image clusters from an unlabeled image set; 2 We formulate the joint learning in a recurrent framework, where merging operations of agglomerative clustering are expressed as a forward pass, and representation learning of CNN as a backward pass; 3 We derive a single loss function to guide agglomerative clustering and deep representation learning, which makes optimization over the two tasks seamless; 4 Our experimental results show that the proposed framework outperforms previous methods on image clustering and learns deep representations that can be transferred to other tasks and datasets
* Extensive experiments show that our method outperforms the state-of-the-art on image clustering across a variety of image datasets
* We have proposed an approach to jointly learn deep representations and image clusters

# Scholarcy Summary

## Introduction

We are witnessing an explosion in visual content. Significant recent advances in machine learning and computer vision, especially via deep neural networks, have relied on supervised learning and availability of copious annotated data.

We aim to address image clustering and representation learning on unlabeled images in a unified framework.

It is a natural idea to leverage cluster ids of images as supervisory signals to learn representations and in turn the representations would be beneficial to image clustering.

I = {I1, ..., Ins }, the global objective function for learning image representations and clusters can be written as: argmin L(y, θ|I).

(2a) can be cast as a conventional clustering problem based on fixed representations, while (2b) is a standard supervised representation learning process.

Clustering with representations from a CNN initialized with random weights are not reliable, but nearest neighbors and over-clusterings are often acceptable; 2) These over-clusterings can be merged as better representations are learned; 3) Agglomerative clustering is a recurrent process and can naturally be interpreted in a recurrent framework.

## Related Work

Clustering Clustering algorithms can be broadly categorized into hierarchical and partitional approaches [25].

Deep Representation Learning Many works use raw image intensity or hand-crafted features [55, 10, 20, 19, 46, 24] combined with conventional clustering methods.

Representations learned using deep neural networks have presented significant improvements over hand-designed features on many computer vision tasks, such as image classification [30, 51, 54, 49], object detection [15, 14, 21, 47], etc

These approaches rely on supervised learning with large amounts of labeled data to learn rich representations.

We use Y to denote the sequence {y1, ..., yT } with T timesteps

## Agglomerative Clustering

We first briefly describe conventional agglomerative clustering [17, 31].

The core idea in agglomerative clustering is to merge two clusters at each step until some stopping conditions.

It tries to find two clusters Ca and Cb by.

There are many methods to compute the affinity between two clusters [17, 31, 41, 70, 68].

More details can be found in [25].

We describe how the affinity is measured by A in our approach

## Affinity Measure

We build a directed graph G =< V, E >, where V is the set of vertices corresponding to deep representations X for I, and E is the set of edges connecting vertices.

We define an affinity matrix W ∈ Rns×ns corresponding to the edge set.

The weight from vertex xi to xj is defined by.

0, otherwise where σ2 = nsaKs xi∈X xj∈NiKs ||xi − xj ||22.

0, otherwise where σ2 = nsaKs xi∈X xj∈NiKs ||xi − xj ||22

This way to build up a directed graph can be found in many previous works such as [70, 68].

After constructing a directed graph for samples, we adopt the graph degree linkage in [68] to measure the affinity between cluster Ci and Cj, denoted by A(Ci, Cj)

## A Recurrent Framework

Our key insight is that agglomerative clustering can be interpreted as a recurrent process in the sense that it merges clusters over multiple timesteps.

Based on this insight, we propose a recurrent framework to combine the image clustering and representation learning processes.

That would involve performing agglomerative clustering until we obtain the desired number of clusters, and update the CNN parameters by back-propagation.

The number of timesteps per period is determined by a parameter in our approach.

## Objective Function

We accumulate the losses from all timesteps, which is formulated as.

Yt−1 and θt are not explicitly presented at the right side, but they determine the loss via the image cluster labels and affinities among clusters.

On the right side of the above equation, there are two terms: 1) (7a) measures the affinity between cluster Ci and its nearest neighbour, which follows conventional agglomerative clustering; 2) (7b) measures the difference between affinity of Ci to its nearest neighbour cluster and affinities of Ci to its other neighbour clusters.

This term takes the local structure into account.

## Forward Pass

In forward pass of the p-th (p ∈ {1, ..., P }) partially unrolled period, we update the cluster labels with θ fixed to θp, and the overall loss in period p is tep.

In conventional ab c d e ab c d e (a) agglomerative clustering, it will choose two clusters with largest affinity at each time no mater where the clusters are located

In this specific case, it will choose cluster Cb and its nearest neighbour to merge.

As shown, our algorithm by adding (7b) will find cluster Ce, because it is close to it nearest neighbour, and relatively far away from its other neighbours, i.e., the local structure is considered around one cluster.

Another merit of introducing (7b) is that it will allow us to write the loss in terms of triplets as explained

## Backward Pass

In forward pass of the p-th partially unrolled period, we have merged a number of clusters.

Let the sequence of optimal image cluster labels be given by Yp∗ = {y∗t }, and clusters merged in forward pass are denoted by aim to derive the optimal θ to minimize the losses generated in forward pass.

Because the clustering in current period is conditioned on the clustering results of all previous periods, we accumulate the losses of all p periods, i.e., p. Minimizing (9) w.r.t θ leads to representation learning on supervised by.

(10) is a loss defined on clusters of points, which needs the entire dataset to estimate, making it difficult to use batch-based optimization.

Input: I: = collection of image data; n∗c : = target number of clusters; Output: y∗, θ∗: = final image labels and CNN parameters; 1: t ← 0; p ← 0

## 2: Initialize θ and y

10: until Cluster number reaches n∗c 11: y∗ ← yt; θ∗ ← θp we show that this loss can be approximated by a samplebased loss, enabling us to compute unbiased estimators for the gradients using batch-statistics.

The intuition behind reformulation of the loss is that agglomerative clustering starts with each datapoint as a cluster, and clusters at a higher level in the hierarchy are formed by merging lower level clusters.

We show in the supplement that the loss in (10) can be approximately reformulated as where γ is a weight whose value depends on λ and how clusters are merged during the forward pass.

The batch-wise optimization can be performed using conventional stochastic gradient descent method.

Note that such triplet losses have appeared in other works [59, 50].

Because it is associated with a weight, we call (35) the weighted triplet loss

## Optimization

Given an image dataset with ns samples, we assume the number of desired clusters n∗c is given to us as is standard in clustering.

We can build up a recurrent process with T = ns − n∗c timesteps, starting by regarding each sample as a cluster

Such initialization makes the optimization time-consuming, especially when datasets contain a large number of samples.

To address this problem, we can first run a fast clustering algorithm to get the initial clusters.

Based on the algorithm in [69], we obtain a number of clusters which contain a few samples for each.

Given these initial clusters, our optimization algorithm learns deep representations and clusters.

## Datasets

We evaluate the clustering performance on two handwritten digit image datasets (MNIST [34] and USPS1), two multi-view object image datasets (COIL20 and COIL100 [42]), and four face image datasets (UMist [18], FRGCv2.02, CMU-PIE [53], Youtube-Face (YTF)) [63].

MNIST consists of training set (60,000) and testing set (10,000).

To compare with different approaches, we experiment on the full set (MNIST-full) and testing set (MNIST-test), separately.

For face image datasets such as UMist, CMU-PIE, we use the images provided as is without any changes.

For FRGC-v2.0 and YTF datasets, we first crop faces and resize them to a constant size.

In FRGCv2.0 dataset, we randomly choose 20 subjects.

As for YTF dataset, we choose the first 41 subjects which are sorted by their names in alphabet order

## Experimental Setup

Ks is set to 20, the same value to [68].

We stacked multiple combinations of convolutional layer, batch normalization layer, ReLU layer and pooling layer.

For all the convolutional layers, the number of channels is 50, and filter size is 5 × 5 with stride = 1 and padding = 0.

Its kernel size is 2 and stride is 2.

To deal with varying image sizes across datasets, the number of stacked convolutional layers for each dataset is chosen so that the size of the output feature map is about 10×10.

For each partially unrolled period, the base learning rate is set to 0.01, momentum 0.9, and weight decay 5 × 10−5.

We use the inverse learning rate decay policy, with Gamma=0.0001 and Power=0.75.

## Quantitative Comparison

We report NMI for different methods on various datasets. Results are averaged from 3 runs.

We report the results by re-running the code released by original papers.

We find the results we obtain are somewhat different from the one reported in original papers.

We conduct L2-normalization on the image intensities since it empirically improves the clustering performance.

The value before ’/’ is obtained by re-running code while the value after ’/’ is that reported in previous papers.

The possible reason is that MNISTfull has 70k samples, and these methods cannot cope with such large-scale dataset when using image intensity as representation.

This problem is addressed by our learned representation.

We believe the reason is that OURS-RC uses the final learned representation over the entire clustering process, while OURS-SF starts with image intensity, which indicates that the learned representation is more discriminative than image intensity. 3

## Generalization Across Clustering Algorithms

We evaluate if the representations learned by our joint agglomerative clustering and representation learning approach generalize to other clustering techniques.

Some algorithms like K-means, AC-Link that performed very poorly with raw intensities perform much better with our learned representations, and the variance in performance across all clustering algorithms is much lower.

These results clearly demonstrate that our learned representation is not over-fitting to a single clustering algorithm, but generalizes well across various algorithms.

Using our learned representation, some of the clustering algorithms perform even better than AC-GDL we build on in our approach

## Cross-Dataset Clustering

We study whether our learned representations generalize across datasets.

We experiment on two dataset pairs: 1) multi-view object datasets (COIL20 and COIL100); 2) hand-written digit datasets (USPS and MNIST-test).

We use the representation learned from one dataset to represent another dataset, followed by agglomerative clustering.

We use the representations from top ip layer and the convolutional or pooling layers close to top layer for image clustering.

0.770 0.767 directly using raw image from the data layer, the clustering performance based on learned representations from all layers improve, which indicates that the learned representations can be transferred across these datasets.

The performance on target datasets is worse compared to learning on the target dataset directly.

As for MNIST and USPS, the performance beats OURS-SF, but worse than OURS-RC.

## Face Verification

We evaluate the performance of our approach by applying it to face verification.

The representation is learned on Youtube-Face dataset and evaluated on LFW dataset [23] under the restricted protocol.

We randomly choose about 10k, 20k, 30k, 50k, 100k samples from YTF dataset.

We implement our approach to train CNN model and cluster images on the training set.

Using the same training samples and CNN architecture, we train a CNN model with a softmax loss supervised by the groundtruth labels of the training set.

7. As shown, though no groundtruth labels are used for representation learning in our approach, we obtain analogous performance to the supervised learning approach.

Our approach even beats the supervised learning method in one case

## Image Classification

Unsupervised representation learning methods are starting to achieve promising results for a variety of recognition tasks [5, 4, 26, 36].

The only difference is that we learn a new representation from 6 × 6 patches, and use these new representations to build the codebook with 1,600 codes.

The CNN architecture we use contains two convolutional layers, each of which is combined with a ReLu and a pooling layer, followed by an inner product layer.

Both convolutional layers have 50 3 × 3 filters with pad = 1.

We use the representation output by inner product layer to learn the codebook.

When using 400k randomly extracted patches to learn the codebook, [5] achieved 77.9%.

This performance beats several other methods listed in [4, 16, 26, 36]

## Conclusion

We have proposed an approach to jointly learn deep representations and image clusters.

We combined agglomerative clustering with CNNs and formulate them as a recurrent process.

We used a partially unrolling strategy to divide the timesteps into multiple periods.

We merged clusters step by step during the forward pass and learned representation in the backward pass, which are guided by a single weighted triplet-loss function.

The extensive experiments on image clustering, deep representation transfer learning and image classification demonstrate that our approach can obtain more precise image clusters and discriminative representations that generalize well across many datasets and tasks

## Approximated Affinity Measure

We need to re-compute the affinity between the merged cluster to all other clusters based on 13 and 14 repeatedly.

This assumption is mild because the condition to merge Cm and Cn is that they are similar to each other

In this case, the ratio between WCi,Cm 1|Cm| and WCi,Cn 1|Cn| is analogy to the ratio between the number of samples in two set, i.e., Above approximation provides us a potential way to reduce the computational complexity of agglomerative clustering.

Though we computed A(Ci → (C ∪ Cn)) based on Eq (14) in all our experiments, we found the approximation version achieves analogy performance while costs much less time than the original one.

Because Cl and Cj are regarded as different clusters previously, sample pairs from both of them are with positive weights in the loss function

It will be diminished by positive pairs at current time step.

To omit the case that A(xi, xj) is much larger than A(xi, xk), we add a margin threshold like the triplet loss function used in [59, 50]

## Detailed CNN Architectures in our Paper

The CNN architectures vary from dataset to dataset.

As we mentioned in the main paper, we stacked different number of layers for different datasets so that the size of most top layer response map is about 10×10.

Means the layer is used, while − means the layer is not used

## Performance Evaluated by Accuracy

We evaluate the performance of different algorithms based on clustering accuracy (AC) metric, as a supplement to the NMI metric used in our main paper.

As we can see from table 10, the proposed method outperform other methods on all datasets, which has similar trend as evaluated using NMI.

According to table 11, all other clustering algorithms are boosted after using the learned representation as evaluated on AC.

These results further prove the proposed method is superior to other clustering algorithms and learns powerful deep representations that generalize well across different clustering algorithms

## Robustness Analysis

We choose the two most important parameters: unfolding rate η and Ks for evaluating the robustness of our approach to variations in these parameters.

Increasing Ks result in similar degradation in the agglomerative clustering algorithms we compare to.

This suggests that Ks should not be set to very large value in general

## Reliability Analysis

We evaluate the reliability by measuring the purity of samples at the beginning of our algorithm.

Because we use agglomerative clustering, there are very few samples in each cluster at the beginning.

Most samples in the same cluster tend to belong to the same category.

For each sample in a dataset, we count the number of samples (Km) that belong to the same category within its K nearest neighbours, and compute the precision Km/K for it.

As we can see, based on raw image data, all datasets have high ratios when K is smaller, and the ratios increase further when using our learned deep representations.

When K is small, the pseudo-labels are reliable enough to learn plausible deep representations

## Clustering based on Hand-Crafted Features

We evaluate the performance of clustering based on image features, instead of image intensities.

We choose three different types of datasets for testing: COIL100, MNIST-test and UMist, and three types of clustering algorithms including SC-LS [3], N-Cuts [52] and AC-PIC [69] for comparison since their better performance among all the algorithms.

For these three datasets, we use spatial pyramid descriptor [32]4, histogram of oriented gradient (HOG) [7]5 and local binary pattern (LBP) [44] for representation, respectively.

Directly learning from image intensities is more straightforward and achieves better performance

## Visualizing Data in Low Dimension

Projecting high-dimensional data into low-dimensional space can help people to intuitively understand the data.

Instead of updating the affinities among samples based on the learned representations gradually, we consistently use the affinities among raw image data to perform the agglomerative cluster, which guides representation learning in low-dimensional space.

By this way, we can obtain a lowdimensional space (2D or 3D) which can retain the structure of the original data.

Ponents analysis (NCA) [48], and parametric t-SNE [38]

Though both [38] and our visualization method are based on neural networks, there are two main differences: 1) In [38], a Kullback-Leibler divergence between the joint distributions of original data and the embedded data is considered.

Table 13: 1-nearest neighbor classification error on lowdimensional embedding of MNIST dataset

## Method

0.998 0.971 0.999 0.999 0.975 one cluster.

The algorithm will learn embeddings that discriminate clusters well but possibly disorder the samples in each cluster.

We believe this can be solved by introducing a loss to confine the within-cluster structure.

We leave this as a future work for limited space

## Findings

Extensive experiments show that our method outperforms the state-of-the-art on image clustering across a variety of image datasets

## A\_10\_ Visualizing Learned Deep Representations

We show the image intensities at the first column.

We use different colors for representing different clusters that we predict during the algorithm.

At the bottom of each plot, we give the number of clusters at the corresponding stage.

The number of cluster is same to the number of categories in the dataset.

After a number of iterations, we can learn more discriminative representations for the datasets, and facilitate more precise clustering results

# Builds on previous work

In this paper, we propose an approach that alternates between the two steps – updating the cluster ids given the current representation parameters and updating the representation parameters given the current clustering result. **Specifically, we cluster images using agglomerative clustering**[17] and represent images via activations of a Convolutional Neural Network (CNN)

Afterwards, a linear SVM [6] is applied for image classification on 6,400-d feature. In **our approach, the only difference is that we learn a new representation from 6 × 6 patches, and then use these new representations to build the codebook with 1,600 codes**

This performance also beats several other methods listed in [4, 16, 26, 36]. **However, it is still lower than what we achieved**.

We also evaluate the performance of clustering based on image features, instead of image intensities. **We choose three different types of datasets for testing**: COIL100, MNIST-test and UMist, and three types of clustering algorithms including SC-LS [3], N-Cuts [52] and AC-PIC [69] for comparison since their better performance among all the algorithms

We choose three different types of datasets for testing: COIL100, MNIST-test and UMist, and three types of clustering algorithms including SC-LS [3], N-Cuts [52] and AC-PIC [69] for comparison since their better performance among all the algorithms. **For these three datasets, we use spatial pyramid descriptor** [32]4, histogram of oriented gradient (HOG) [7]5 and local binary pattern (LBP) [44] for representation, respectively

Though we stop the learning process as such, it should be noted that the stop criterion is not confined. For quantitative analysis, **we compute the nearest-neighbor classification error and trustworthiness as in** [38]

In Table 13, we show the 1-nearest neighbour classification error on MNIST test dataset. **We copy t**he best results of the compared methods from [38]

# Differs from previous work

Another group of techniques learn discriminative representations after fabricating supervisory signals for images, and then finetune them supervisedly for downstream applications [12, 11, 60]. **Unlike our approach, the fabricated supervisory signal in these previous works is not updated during representation learning**

To connect image clustering and representation learning more closely, [64] conducted image clustering and codebook learning iteratively. **Unlike our work, they do not jointly optimize for the representation learning and clustering**.

When using 400k randomly extracted patches to learn the codebook, [5] achieved 77.9%. **However, it is still lower than what we achieved**

ponents analysis (NCA) [48], and parametric t-SNE [38]. Though both [38] and **our visualization method are based on neural networks, there are two main differences**: 1) In [38], a Kullback-Leibler divergence between the joint distributions of original data and the embedded data is considered

Though both [38] and our visualization method are based on neural networks, there are two main differences: 1) In [38], a Kullback-Leibler divergence between the joint distributions of original data and the embedded data is considered. **However, in our method, we employ a weighted triplet loss that directly takes the local structure of embedded data into account**; 2) In [38], the authors need to pretrain a stack of RBMs layer-by-layer, and then fine-tune the neural network

# Contributions

In this paper, we have proposed an approach to jointly learn deep representations and image clusters. In our approach, we combined agglomerative clustering with CNNs and formulate them as a recurrent process. We used a partially unrolling strategy to divide the timesteps into multiple periods. In each period, we merged clusters step by step during the forward pass and learned representation in the backward pass, which are guided by a single weighted triplet-loss function. The extensive experiments on image clustering, deep representation transfer learning and image classification demonstrate that our approach can obtain more precise image clusters and discriminative representations that generalize well across many datasets and tasks.

# Future work

We believe this can be solved by introducing a loss to confine the within-cluster structure. We leave this as a future work for limited space.