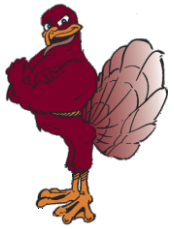
Unsupervised Learning Jointly With Image Clustering

Jianwei Yang

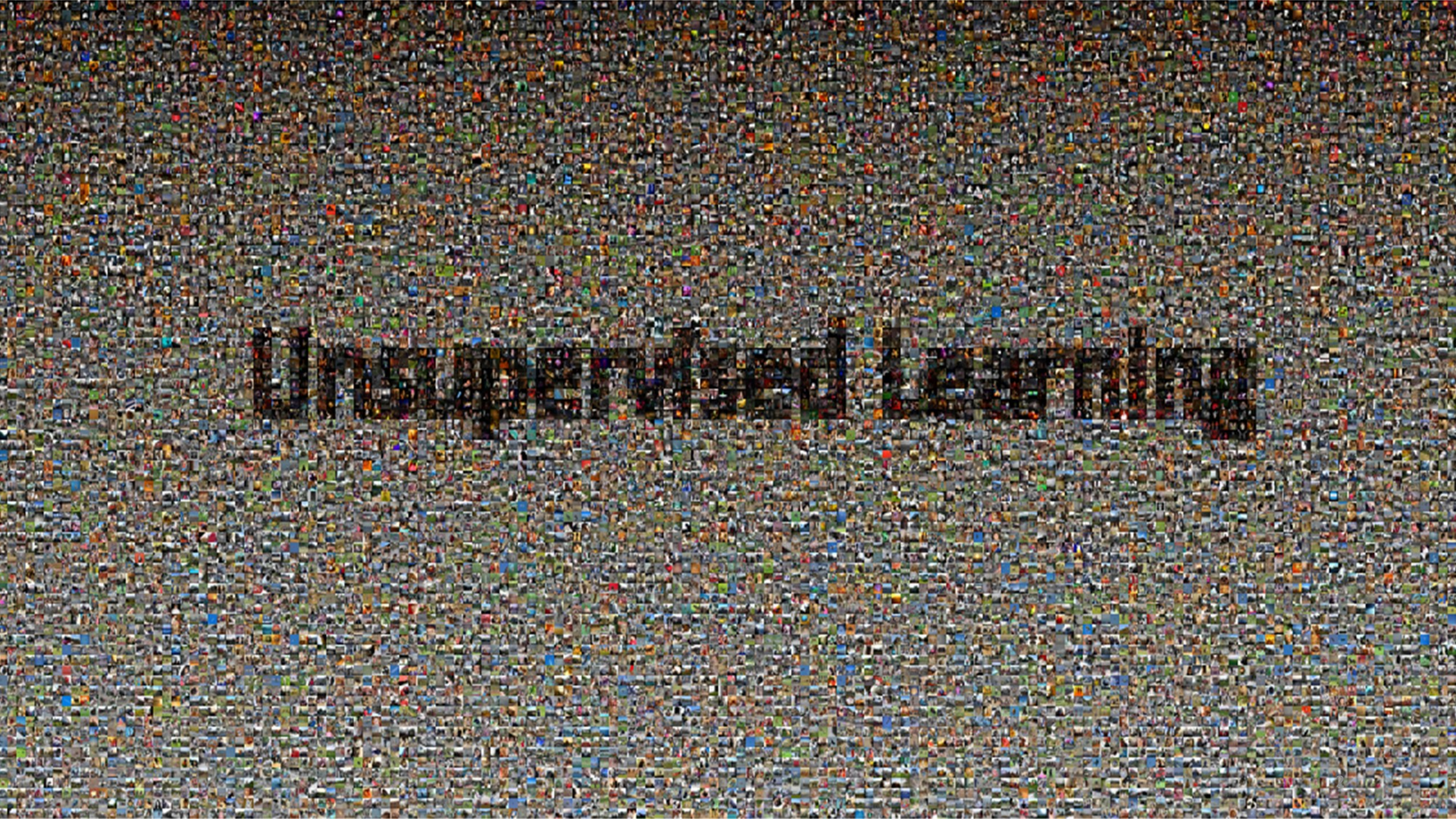
Devi Parikh

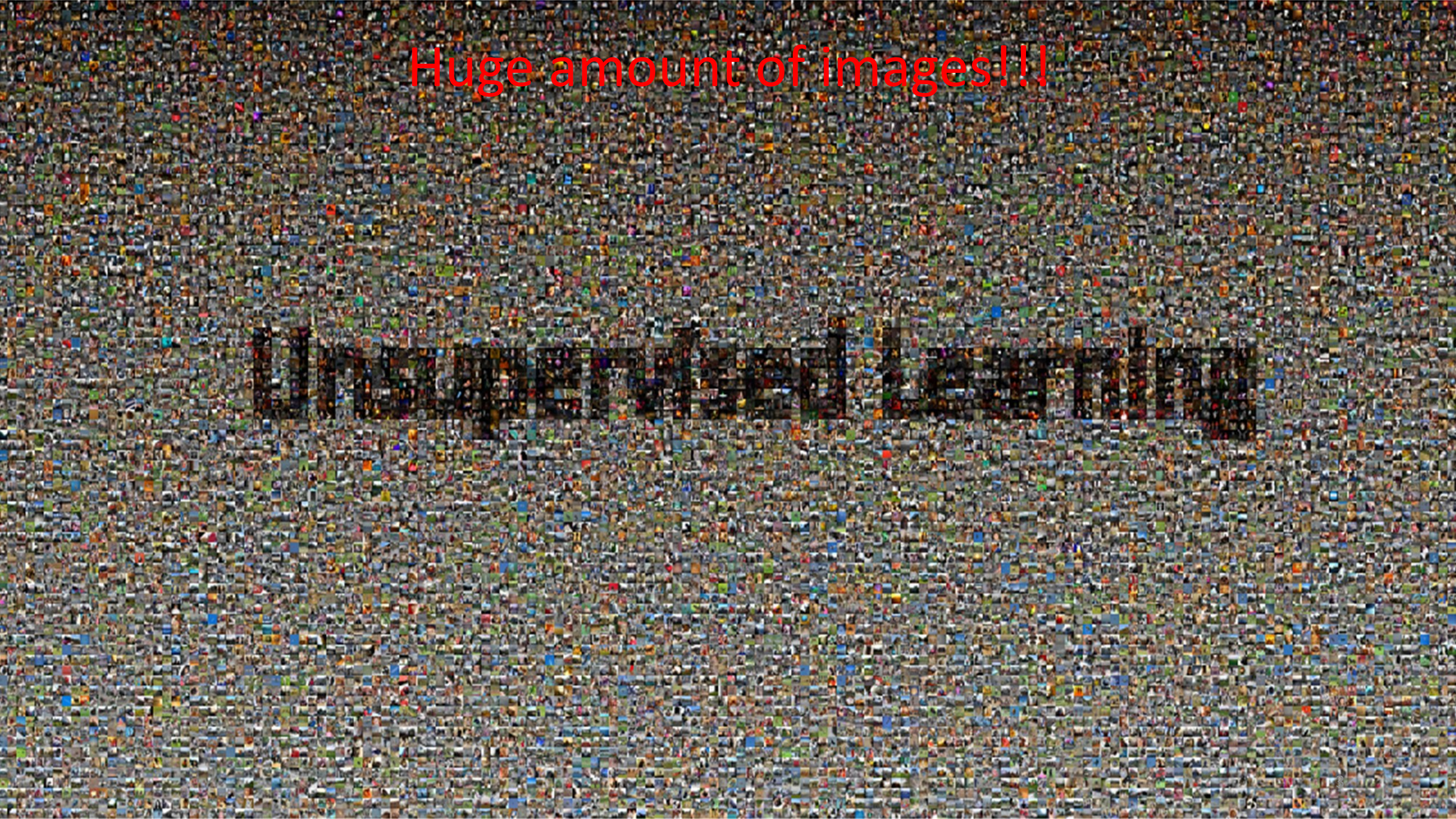
Dhruv Batra

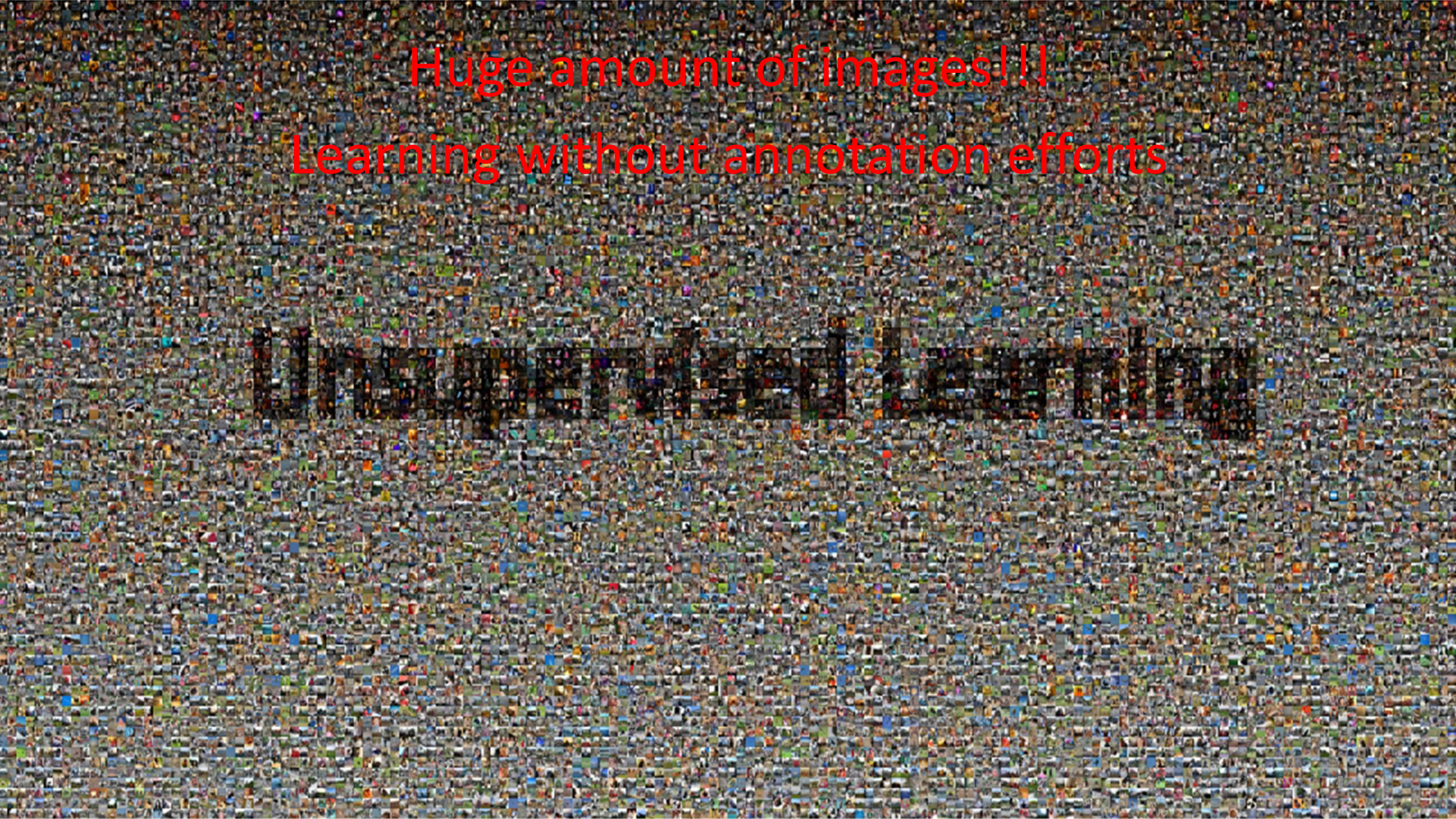


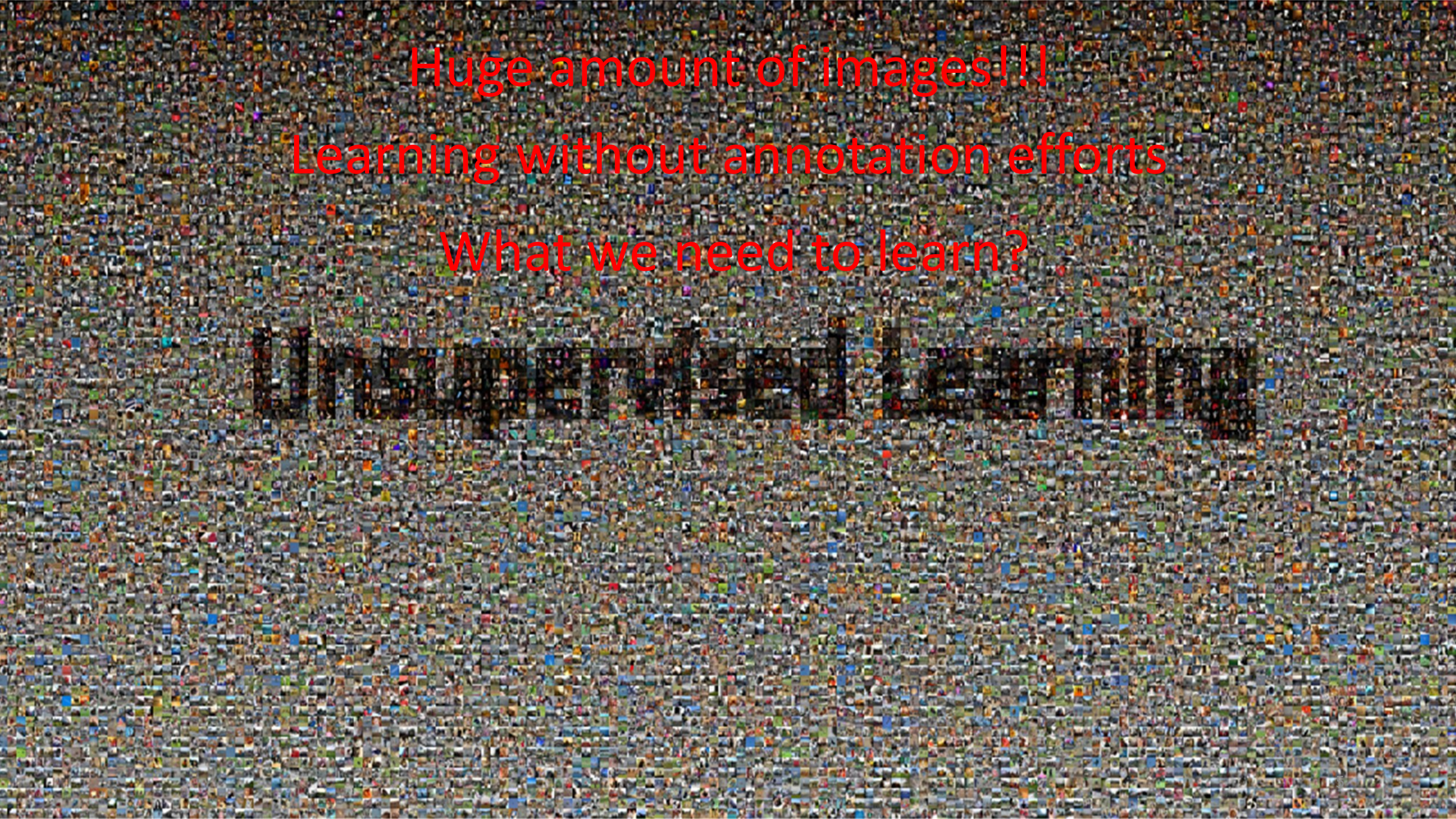
Virginia Tech

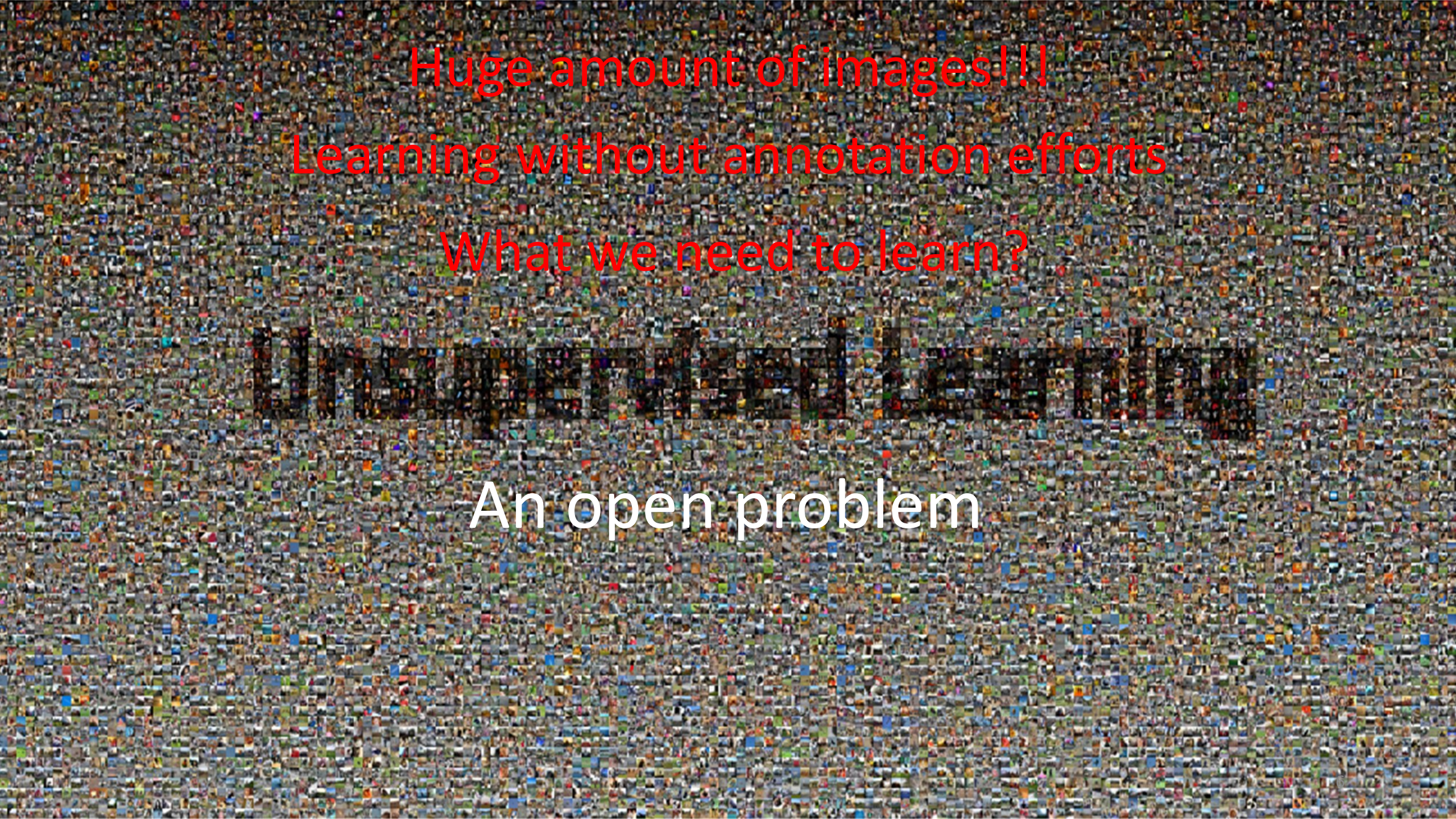
https://filebox.ece.vt.edu/~jw2yang/

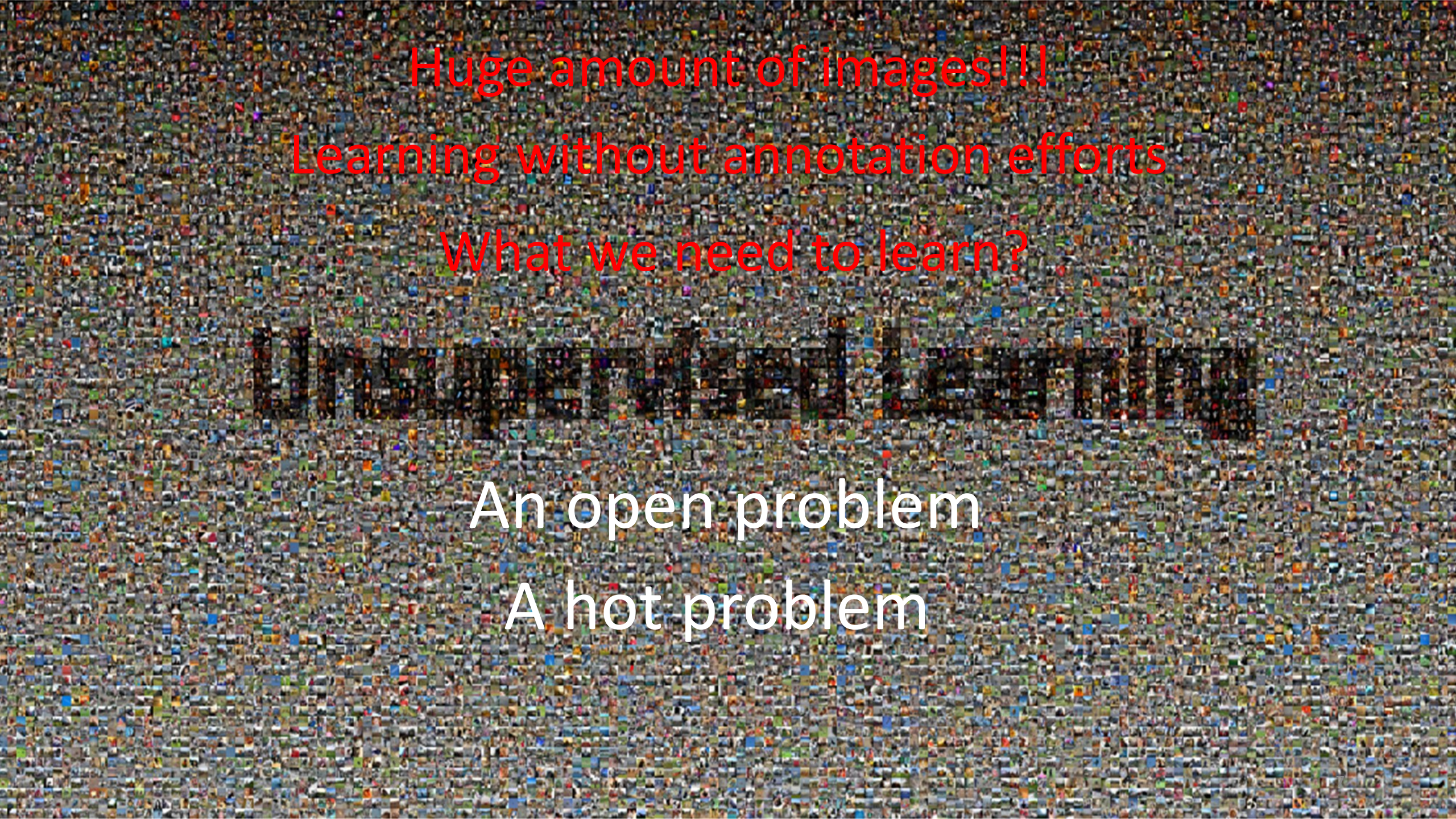

ImageNet

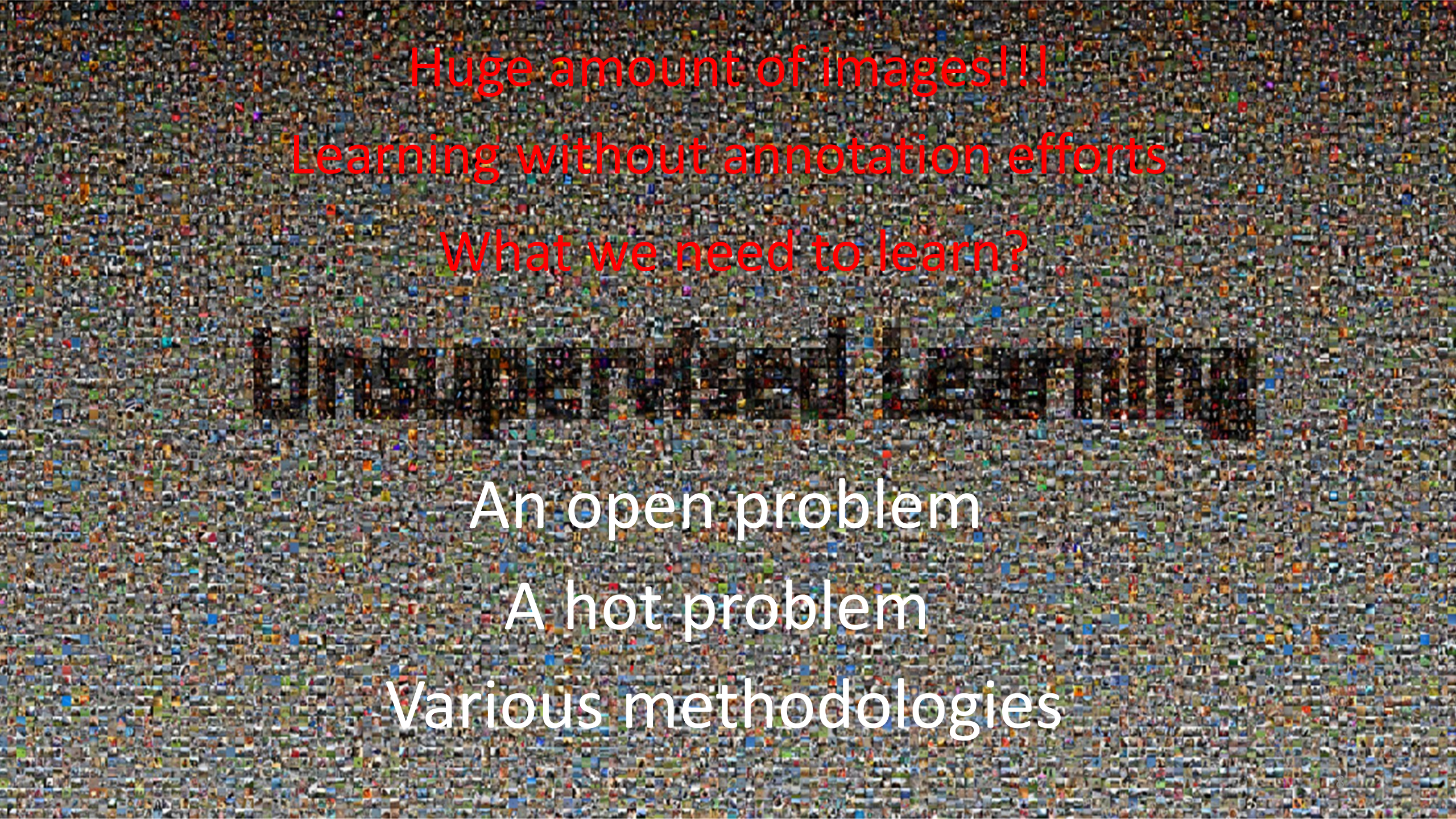

Huge amount of images!!!


Huge amount of images!!!
Learning without annotation efforts


Huge amount of images!!!
Learning without annotation efforts
What we need to learn?
Unsupervised Learning


Huge amount of images!!!
Learning without annotation efforts
What we need to learn?
Unsupervised Learning
An open problem


Huge amount of images!!!
Learning without annotation efforts
What we need to learn?
Unsupervised Learning
An open problem
A hot problem


Huge amount of images!!!
Learning without annotation efforts
What we need to learn?
Unsupervised Learning
An open problem
A hot problem
Various methodologies

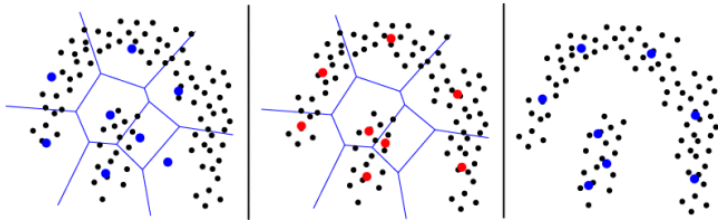
Learning distribution (structure)

Clustering

Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

Clustering

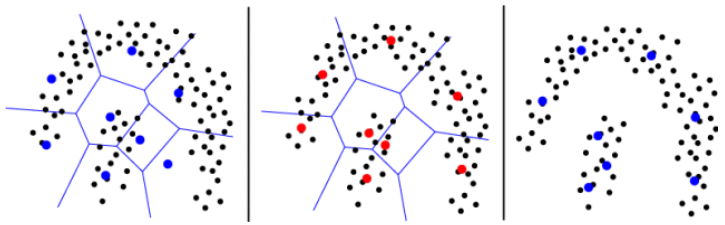


K-means (Image Credit: Jesse Johnson)

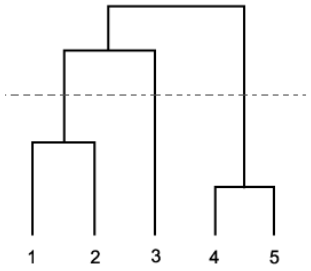
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

Clustering



K-means (Image Credit: Jesse Johnson)

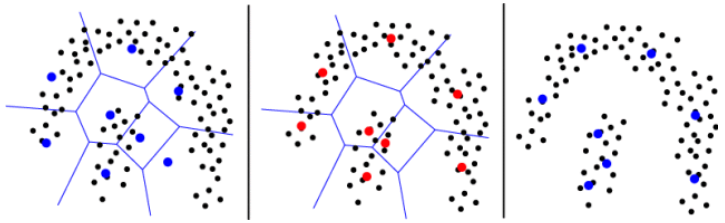


Hierarchical Clustering

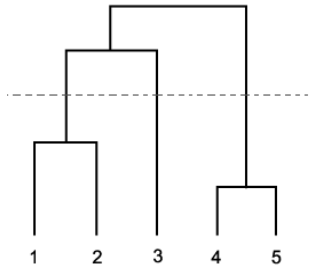
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

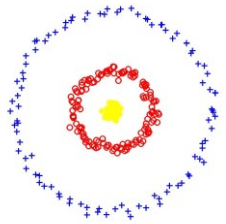
Clustering



K-means (Image Credit: Jesse Johnson)



Hierarchical Clustering

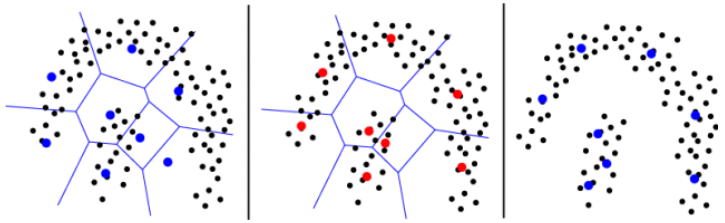


Spectral Clustering
Manor et al, NIPS'04

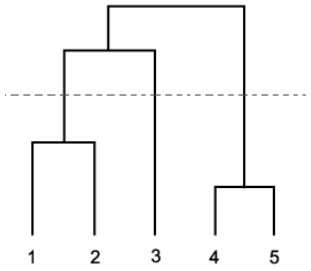
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

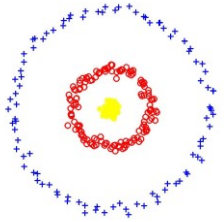
Clustering



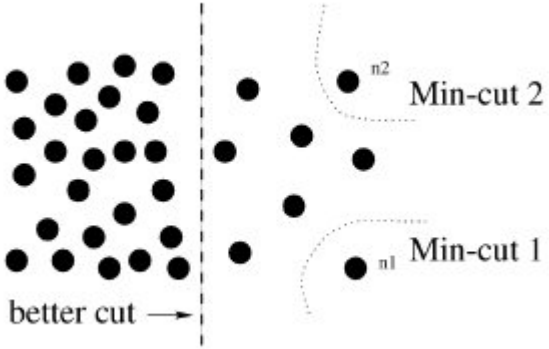
K-means (Image Credit: Jesse Johnson)



Hierarchical Clustering



Spectral Clustering
Manor et al, NIPS'04

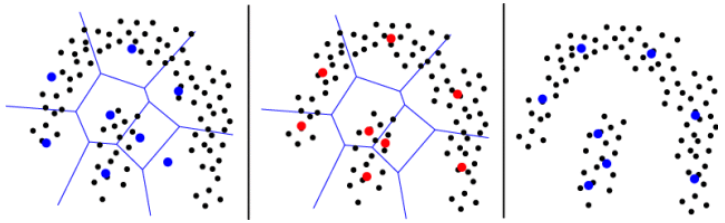


Graph Cut
Shi et al, TPAMI'00

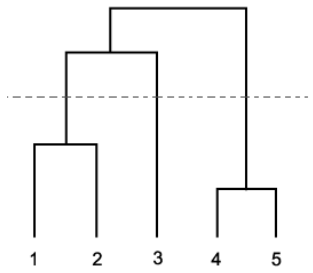
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

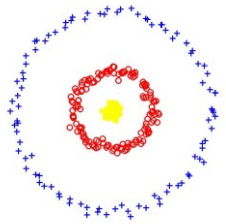
Clustering



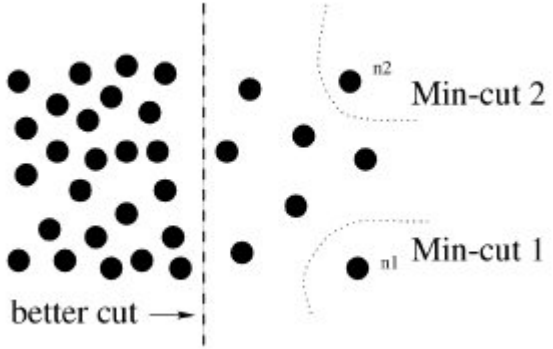
K-means (Image Credit: Jesse Johnson)



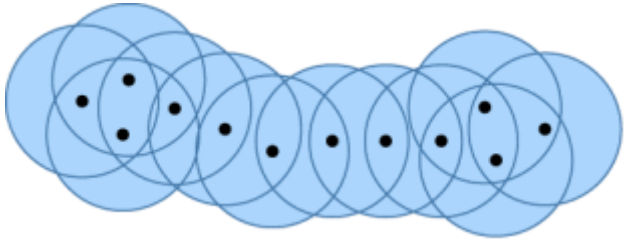
Hierarchical Clustering



Spectral Clustering
Manor et al, NIPS'04



Graph Cut
Shi et al, TPAMI'00

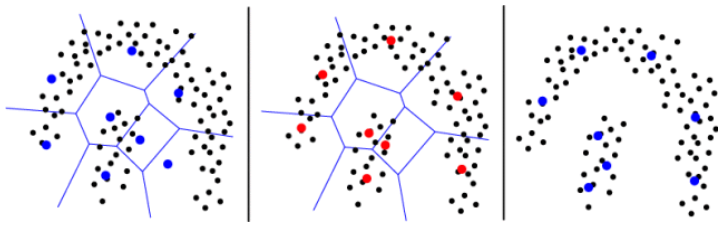


DBSCAN, Ester et al, KDD'96 (Image Credit: Jesse Johnson)

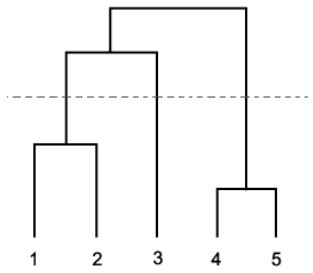
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

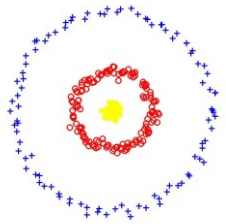
Clustering



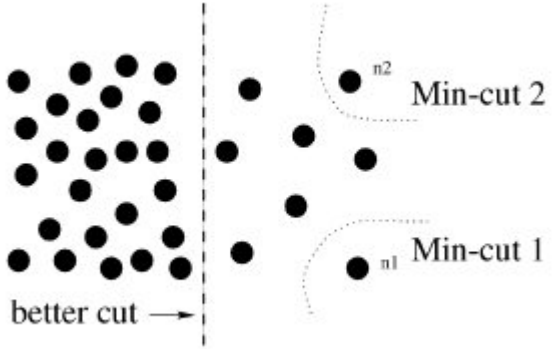
K-means (Image Credit: Jesse Johnson)



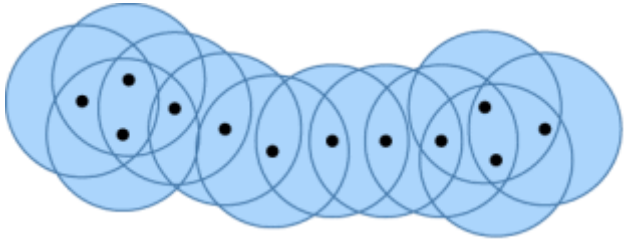
Hierarchical Clustering



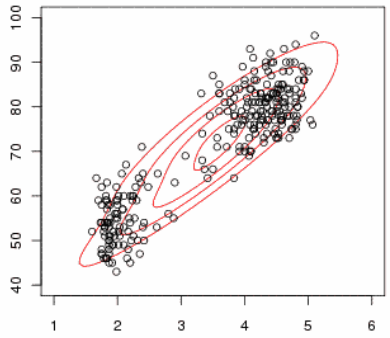
Spectral Clustering
Manor et al, NIPS'04



Graph Cut
Shi et al, TPAMI'00



DBSCAN, Ester et al, KDD'96 (Image Credit: Jesse Johnson)

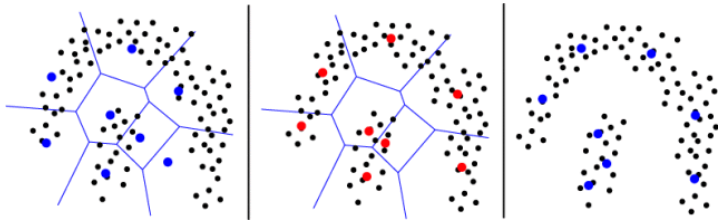


EM Algorithm, Dempster et al, JRSS'77

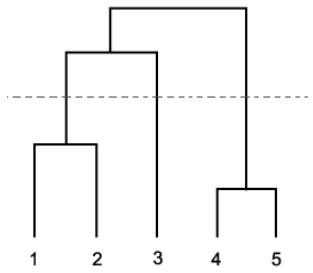
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

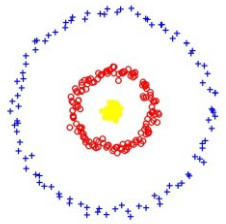
Clustering



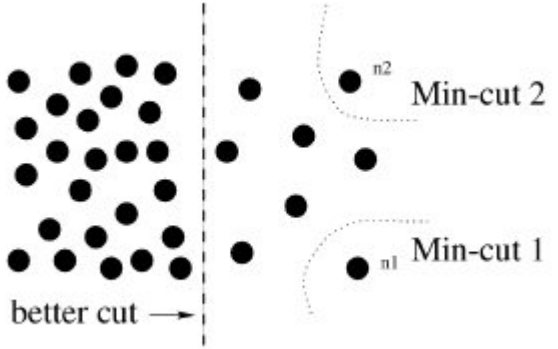
K-means (Image Credit: Jesse Johnson)



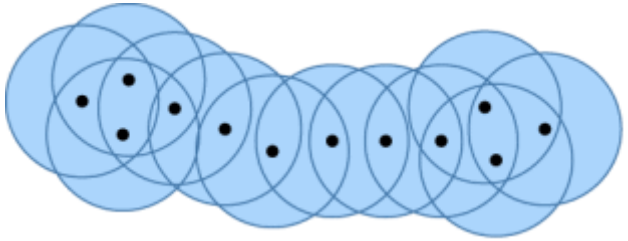
Hierarchical Clustering



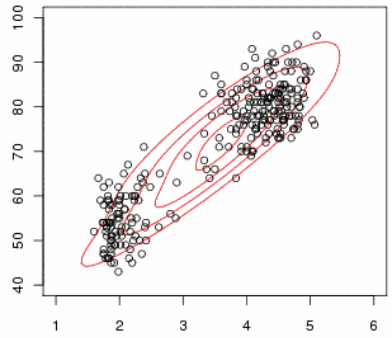
Spectral Clustering
Manor et al, NIPS'04



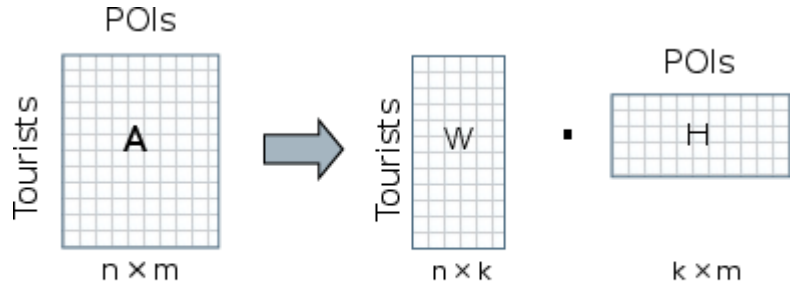
Graph Cut
Shi et al, TPAMI'00



DBSCAN, Ester et al, KDD'96 (Image Credit: Jesse Johnson)



EM Algorithm, Dempster et al, JRSS'77

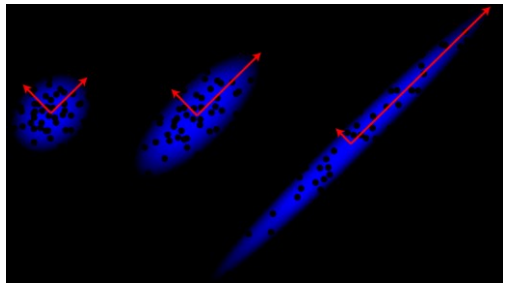


NMF, Xu et al, SIGIR'03 (Image Credit: Conrad Lee)

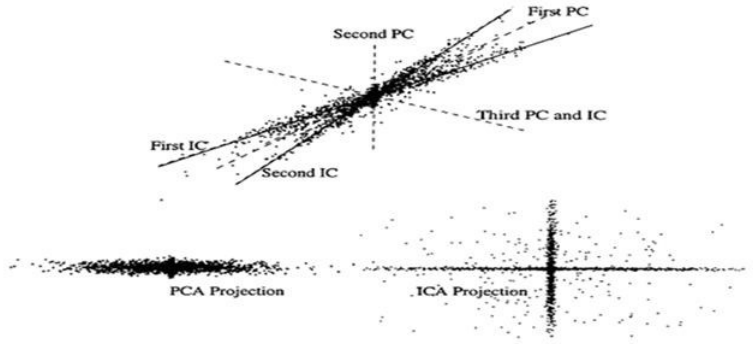
Jain, Anil K., M. Narasimha Murty, and Patrick J. Flynn. "Data clustering: a review." ACM computing surveys (CSUR) 31.3 (1999): 264-323.

Learning distribution (structure)

Sub-space Analysis

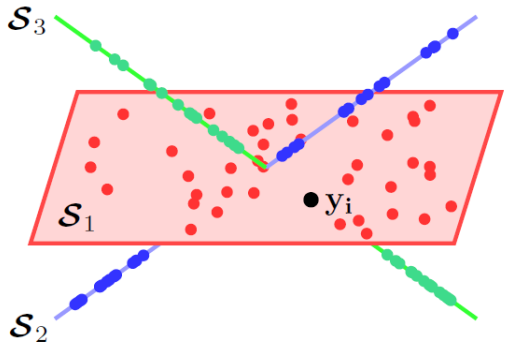


PCA (Image Credit: Jesse Johnson)

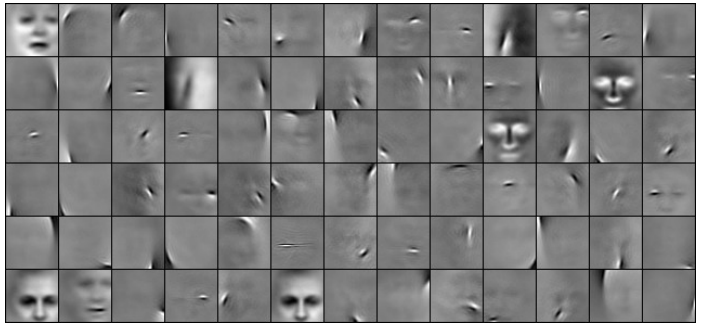


ICA (Image Credit: Shylaja et al)

tSNE, Maaten et al, JMLR'08



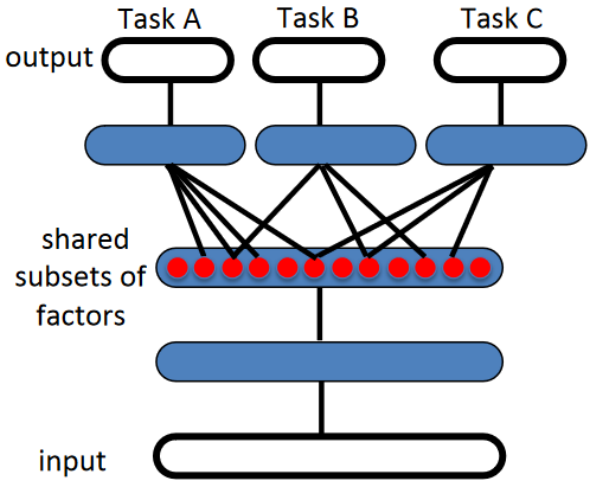
Subspace Clustering, Vidal et al.



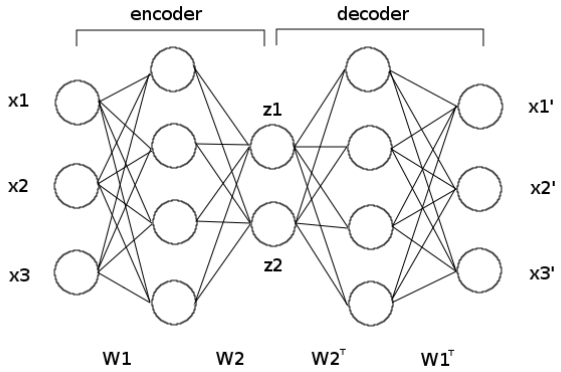
$$\text{minimize}_{a_i^{(j)},\phi_i} \sum_{j=1}^{m} \left\| \mathbf{x}^{(j)} - \sum_{i=1}^{k} a_i^{(j)} \phi_i \right\|^2 + \lambda \sum_{i=1}^{k} S(a_i^{(j)})$$

Sparse coding, Olshausen et al. Vision Research'97

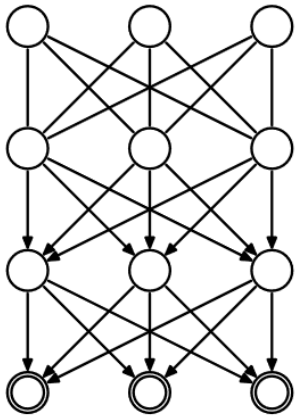
Learning representation (feature)



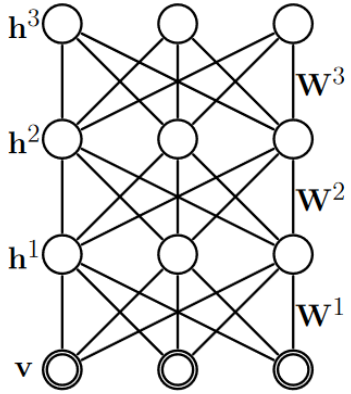
Bengio et al, TPAMI'13



Autoencoder, Hinton et al, Science'06
(Image Credit: Jesse Johnson)



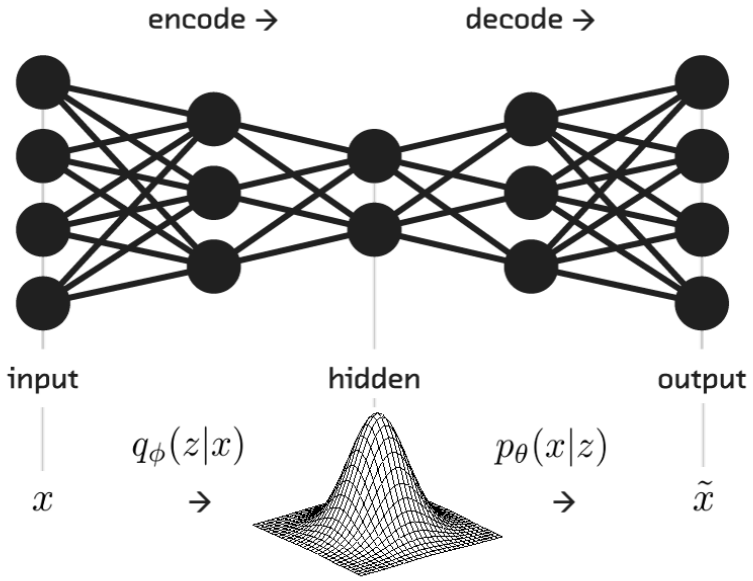
DBN, Hinton et al, Science'06



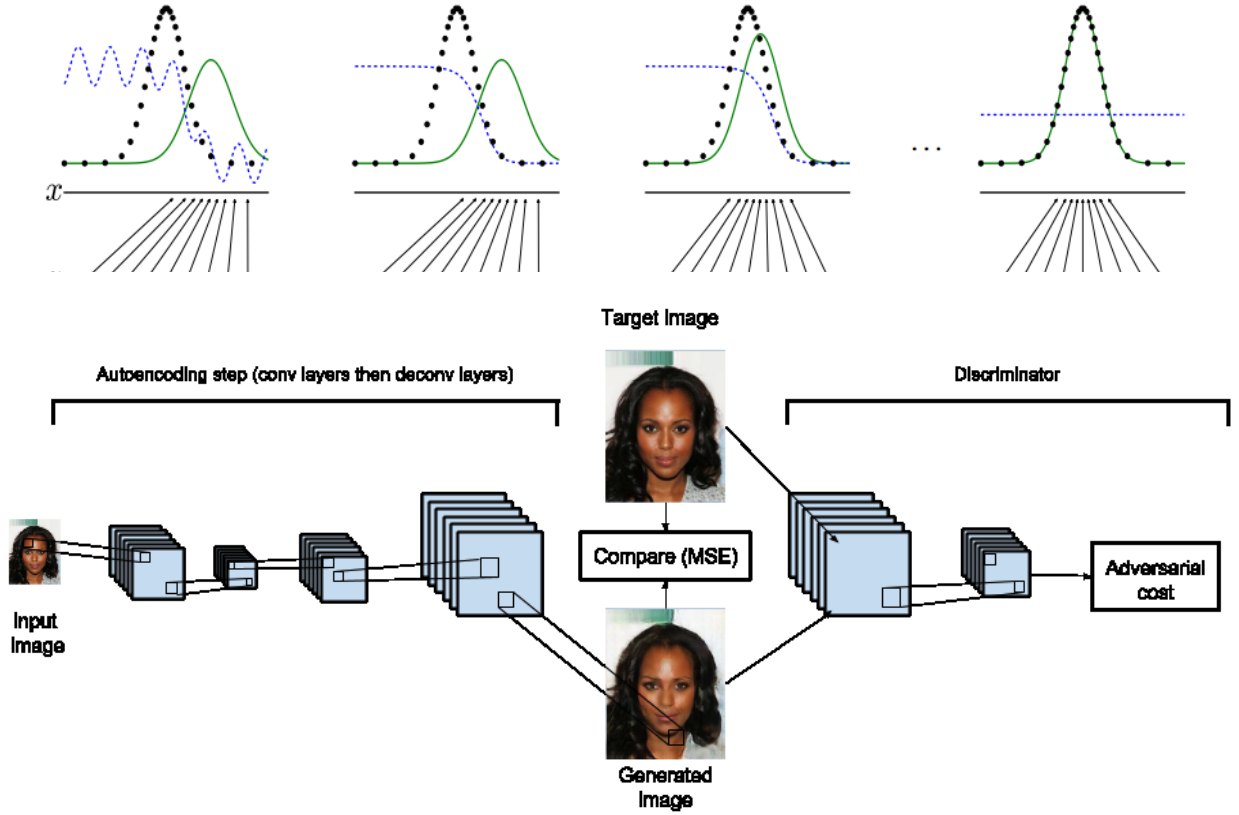
DBM, Salakhutdinov et al, AISTATS'09

Yoshua Bengio, Aaron Courville, and Pierre Vincent. "Representation learning: A review and new perspectives." *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 35.8 (2013): 1798-1828.

Learning representation (feature)

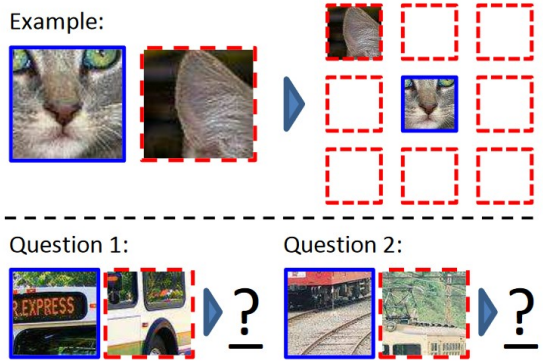


VAE, Kingma et al, arXiv'13
(Image Credit: Fast Forward Labs)

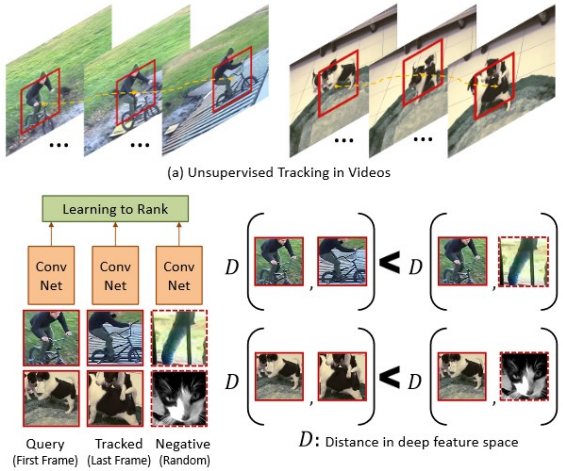


GAN, Goodfellow et al, NIPS'14
DCGAN, Radford et al, arXiv'15
(Image Credit: Mike Swarbrick Jones)

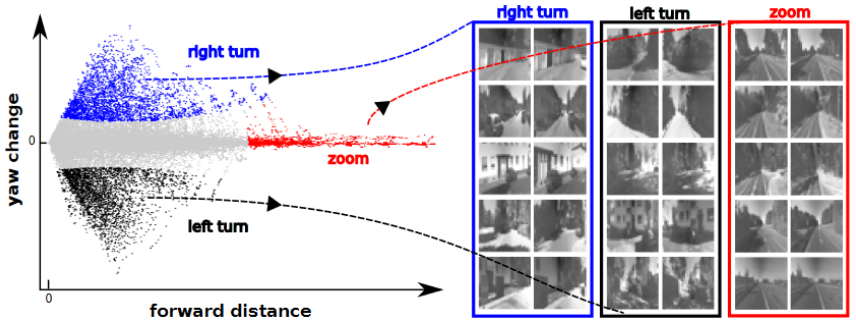
Most Recent CV Works



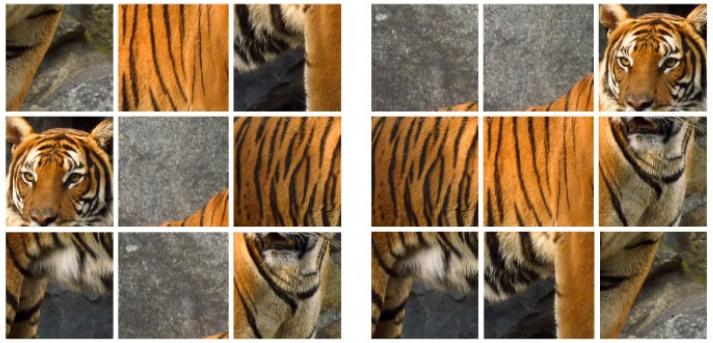
Spatial context, Doersch et al, ICCV'15



Temporal context, Wang et al, ICCV'15



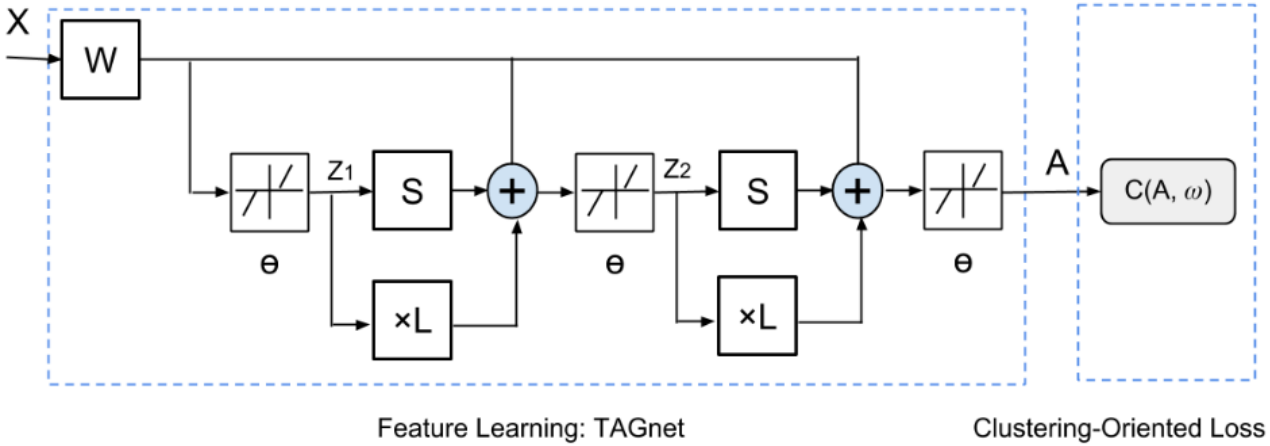
Ego-motion, Jayaraman et al, ICCV'15



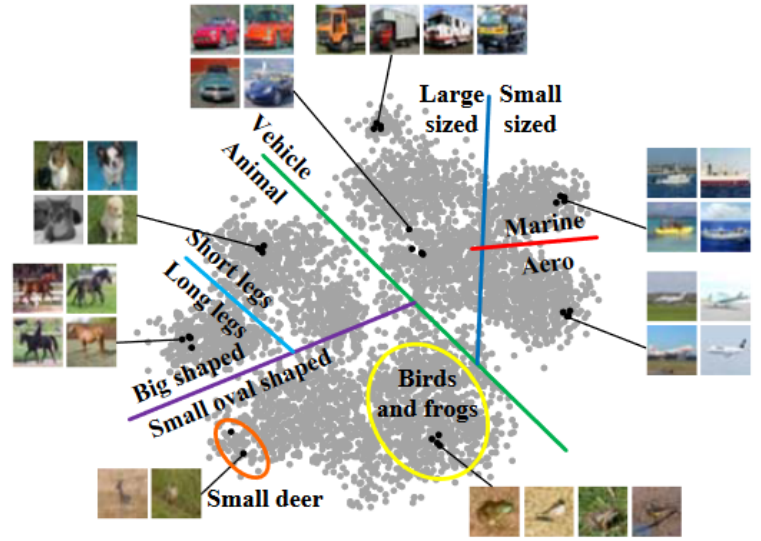
Solving Jigsaw, Noroozi et al, ECCV'16

Context Encoder, Deepak et al, CVPR'16

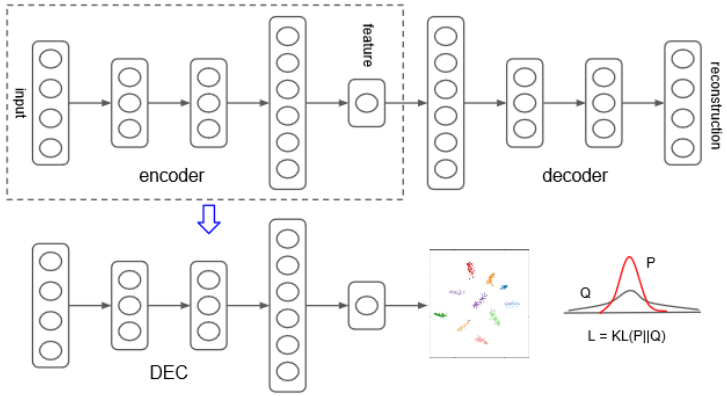
Most Recent CV Works



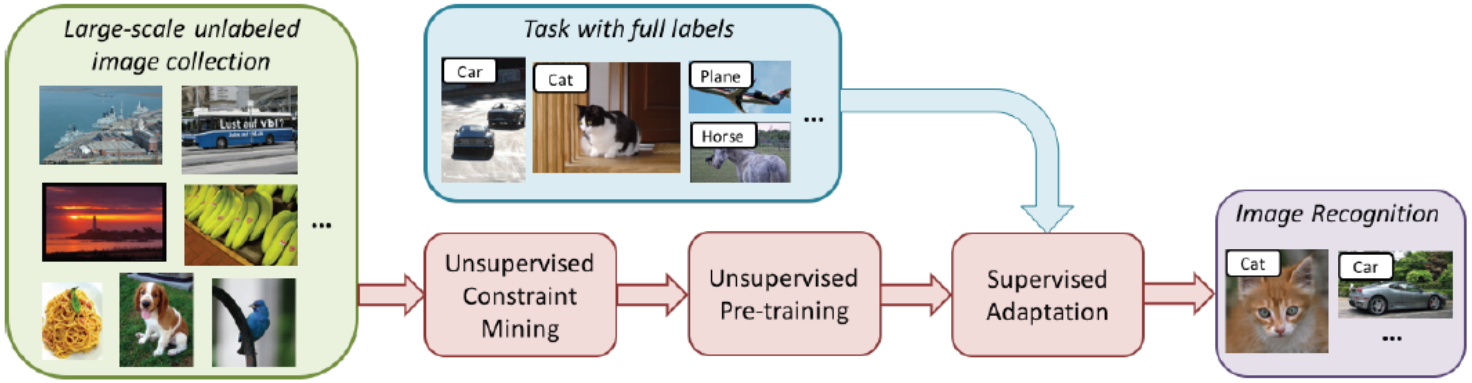
TAGnet, Wang et al, SDM'16



Visual concept clustering, Huang et al, CVPR'16



Deep Embedding, Xie et al, ICML'16



Graph constraint, Li et al, ECCV'16

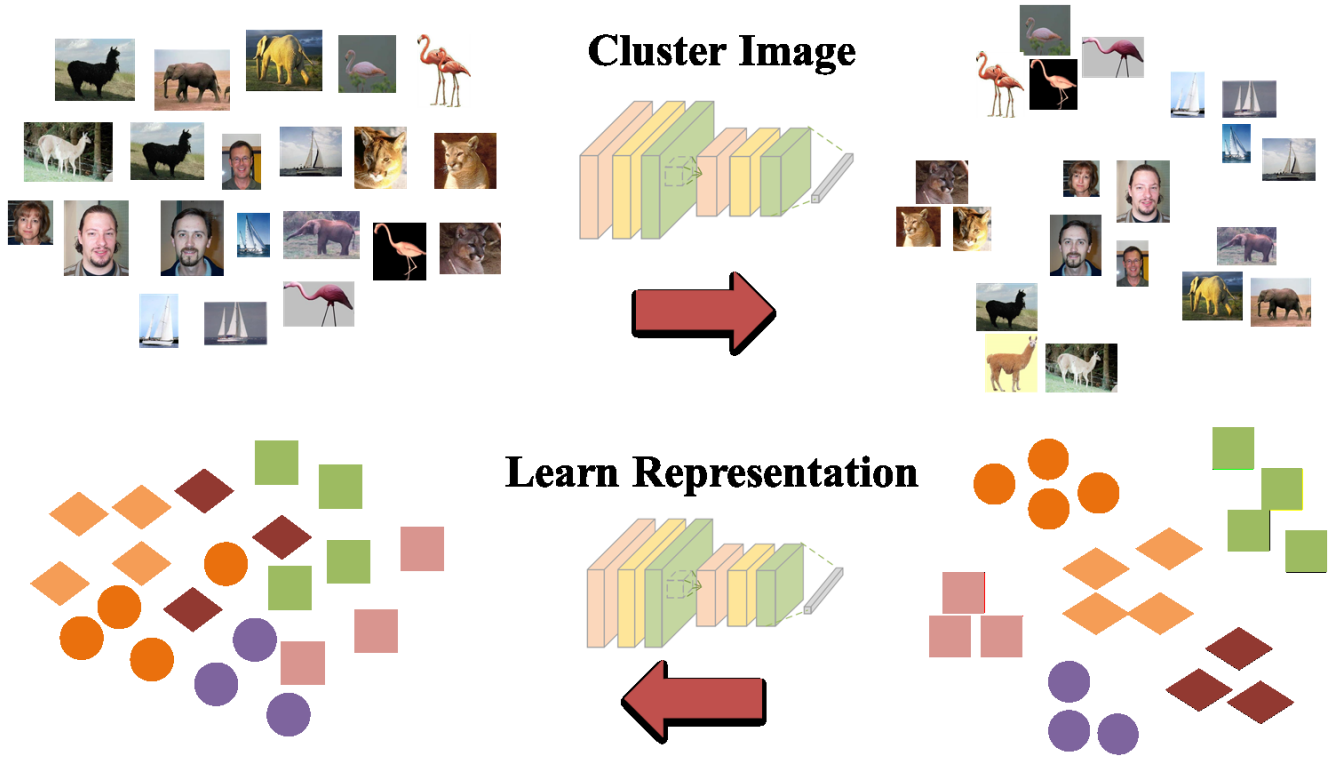
Our Work

Joint Unsupervised Learning (JULE)
of Deep Representations and Image Clusters

Outline

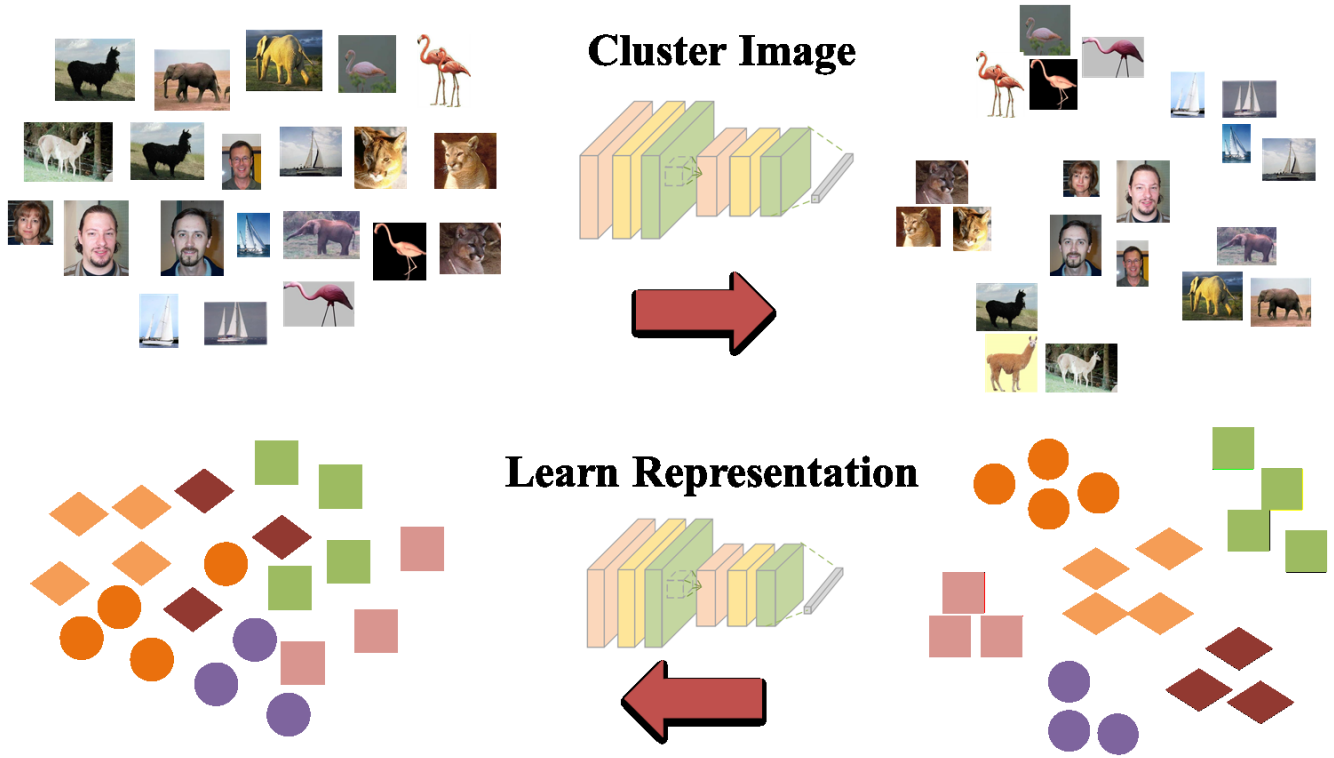
- Intuition
- Approach
- Experiments
- Extensions

Intuition



Meaningful clusters can provide supervisory signals to learn image representations

Intuition



Meaningful clusters can provide supervisory signals to learn image representations

Good representations help to get meaningful clusters

Intuition

Cluster images first, and then learn representations

Intuition

Cluster images first, and then learn representations

Learn representations first, and then cluster images

Intuition

Cluster images first, and then learn representations

Learn representations first, and then cluster images

Cluster images and learn representations progressively

Intuition

Good clusters

Good cluster
Good representations

Poor clusters

Poor representations

Good representations

Intuition

Good clusters

Good cluster
Good representations

Poor clusters

Poor representations

Good representations

Intuition

Good clusters

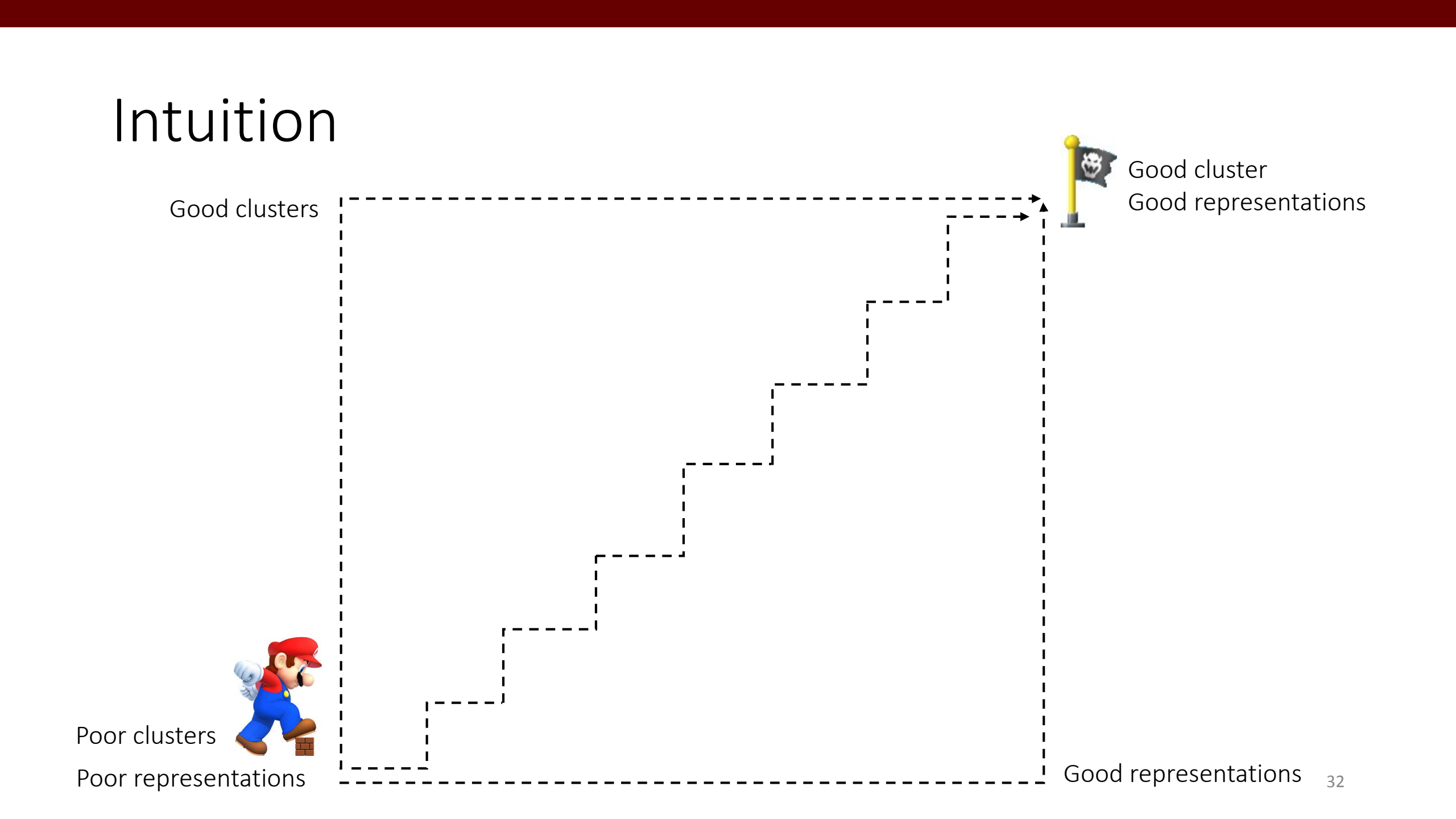
Good cluster
Good representations

Poor clusters

Poor representations

Good representations

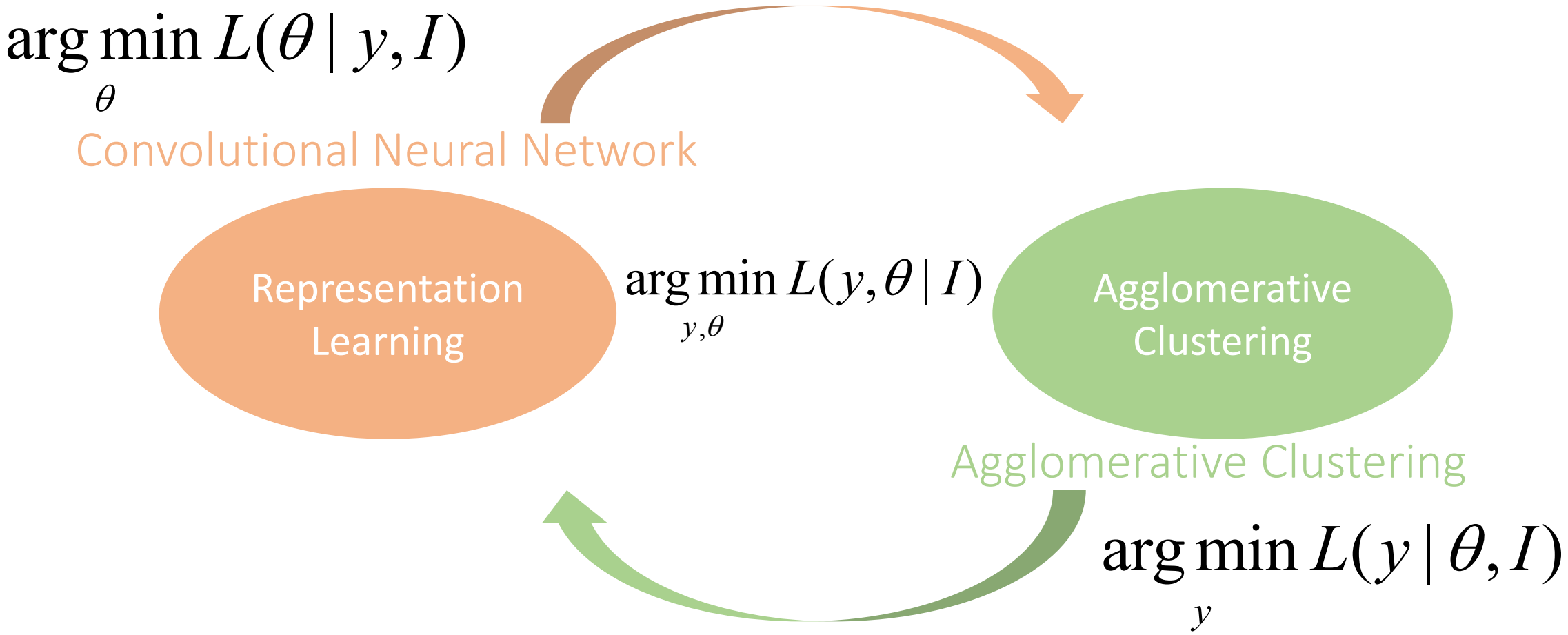
Intuition


Good clusters
Good cluster
Good representations
Poor clusters
Poor representations
Good representations

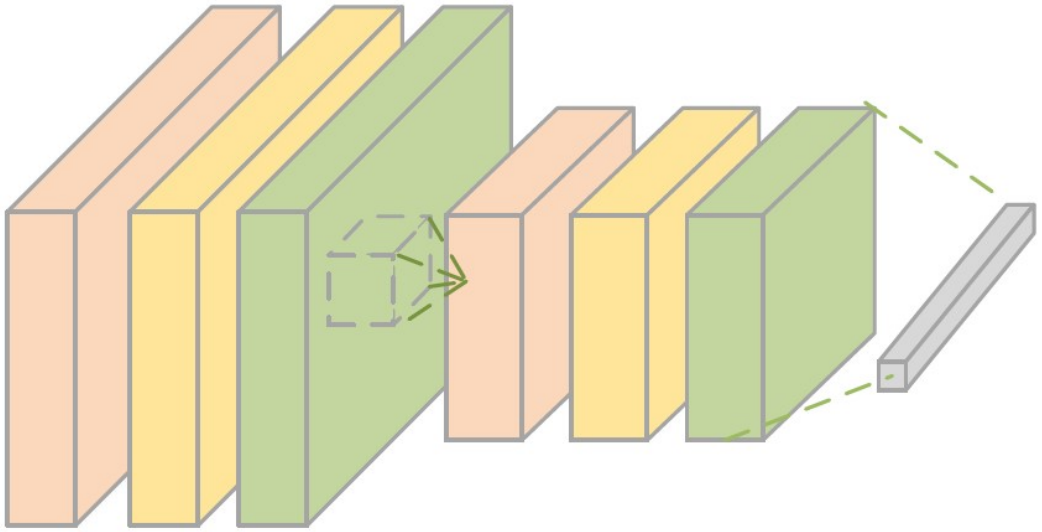
Approach

- Framework
- Objective
- Algorithm & Implementation

Approach: Framework

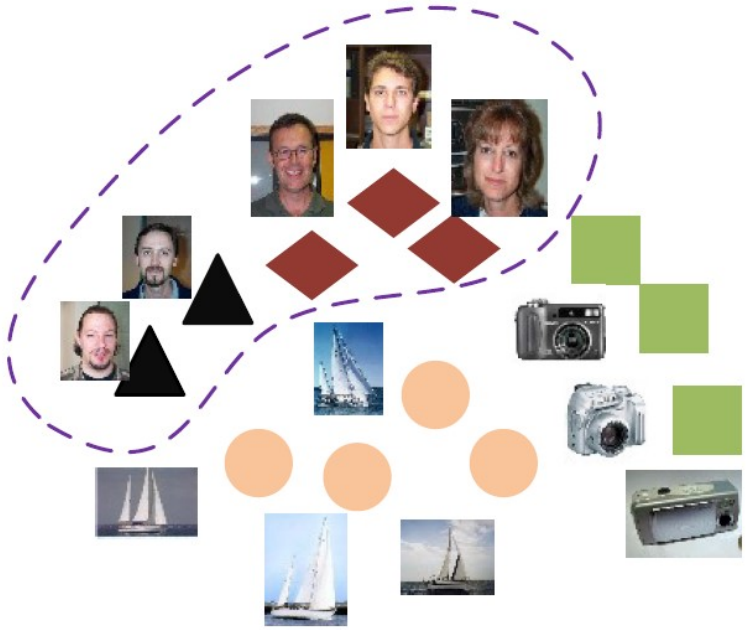


Approach: Framework



Convolutional Neural Network

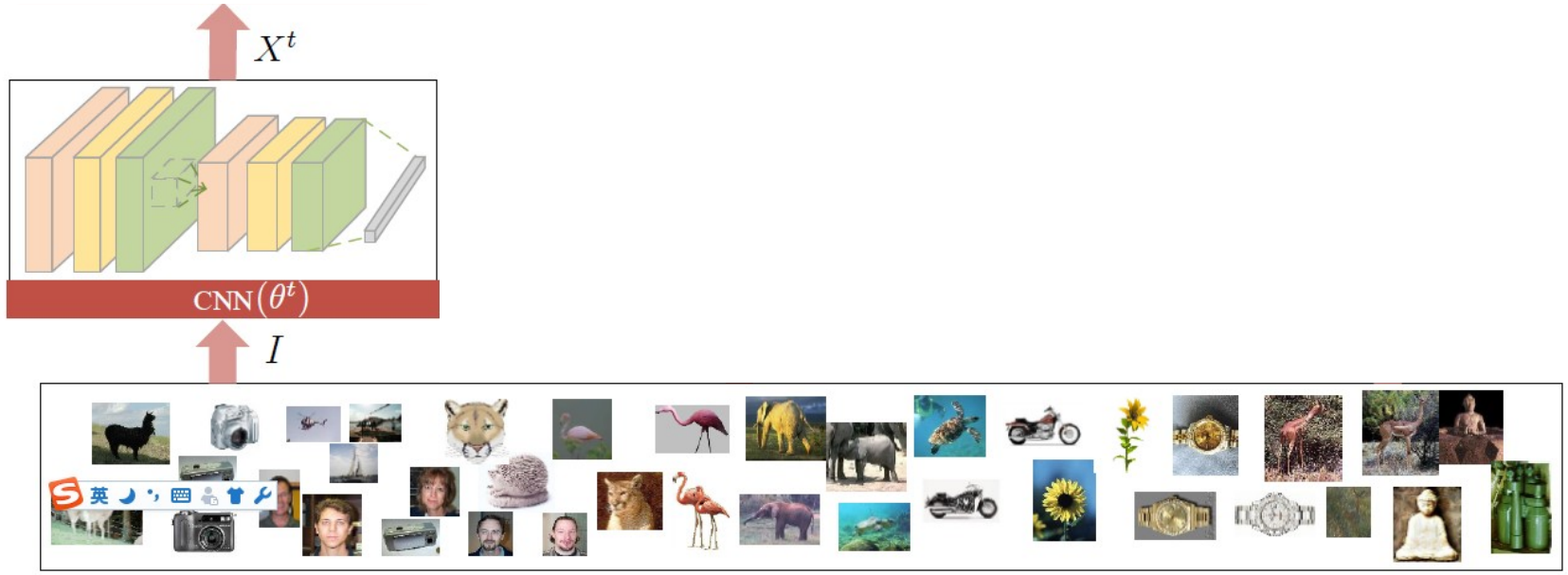
$$\arg\min_{\theta} L(\theta \mid y, I)$$



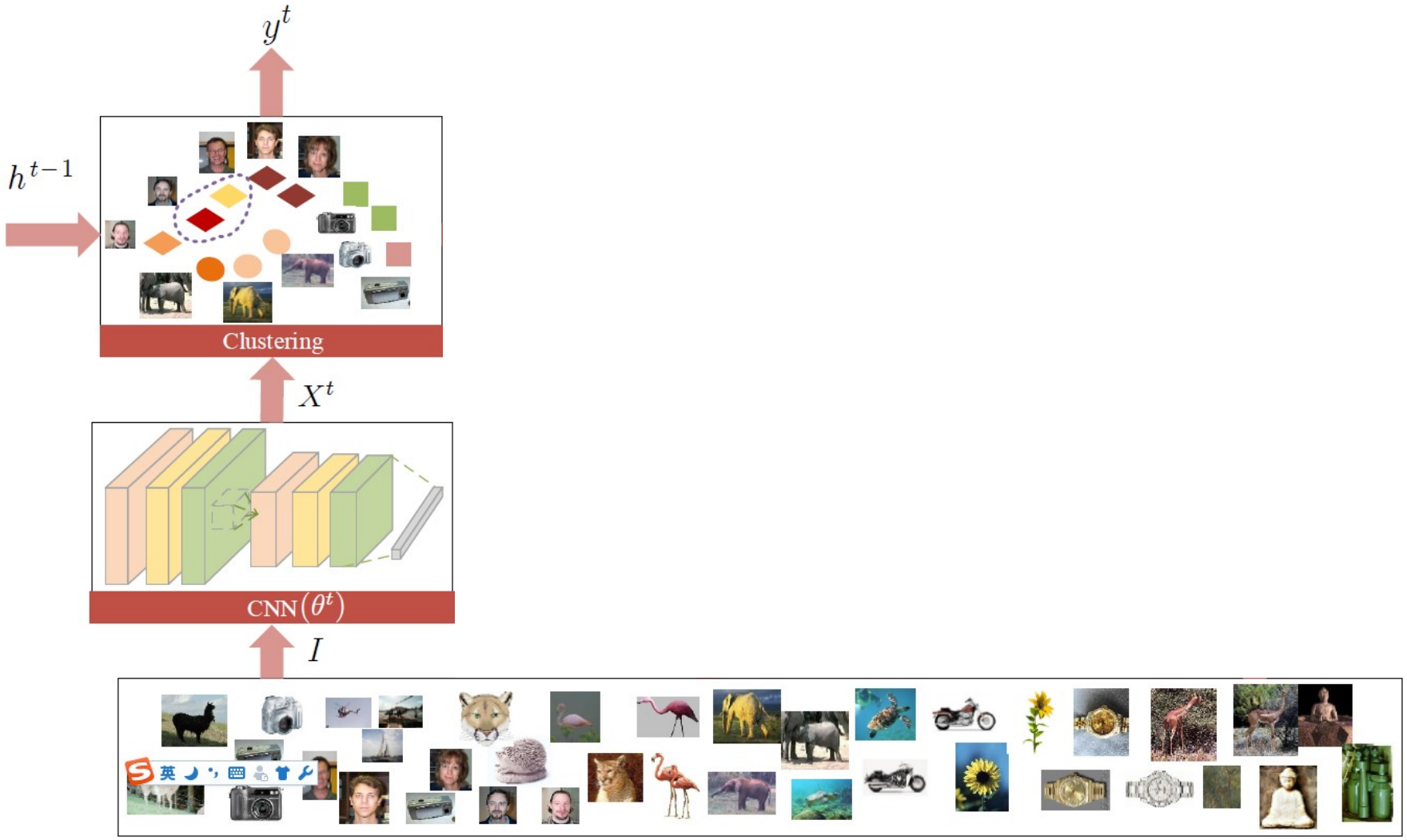
Agglomerative Clustering

$$\arg\min_{y} L(y \mid \theta, I)$$

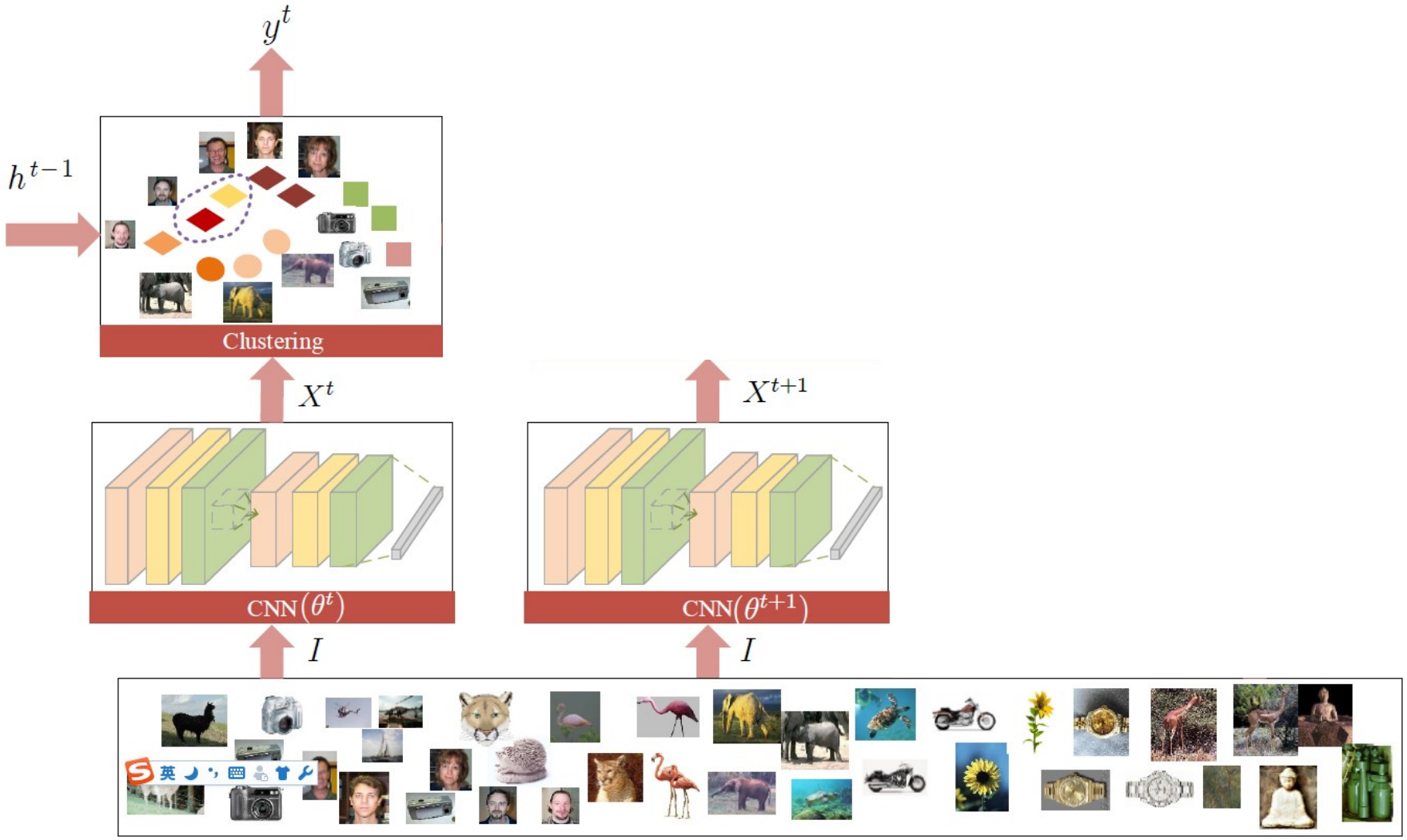
Approach: Recurrent Framework



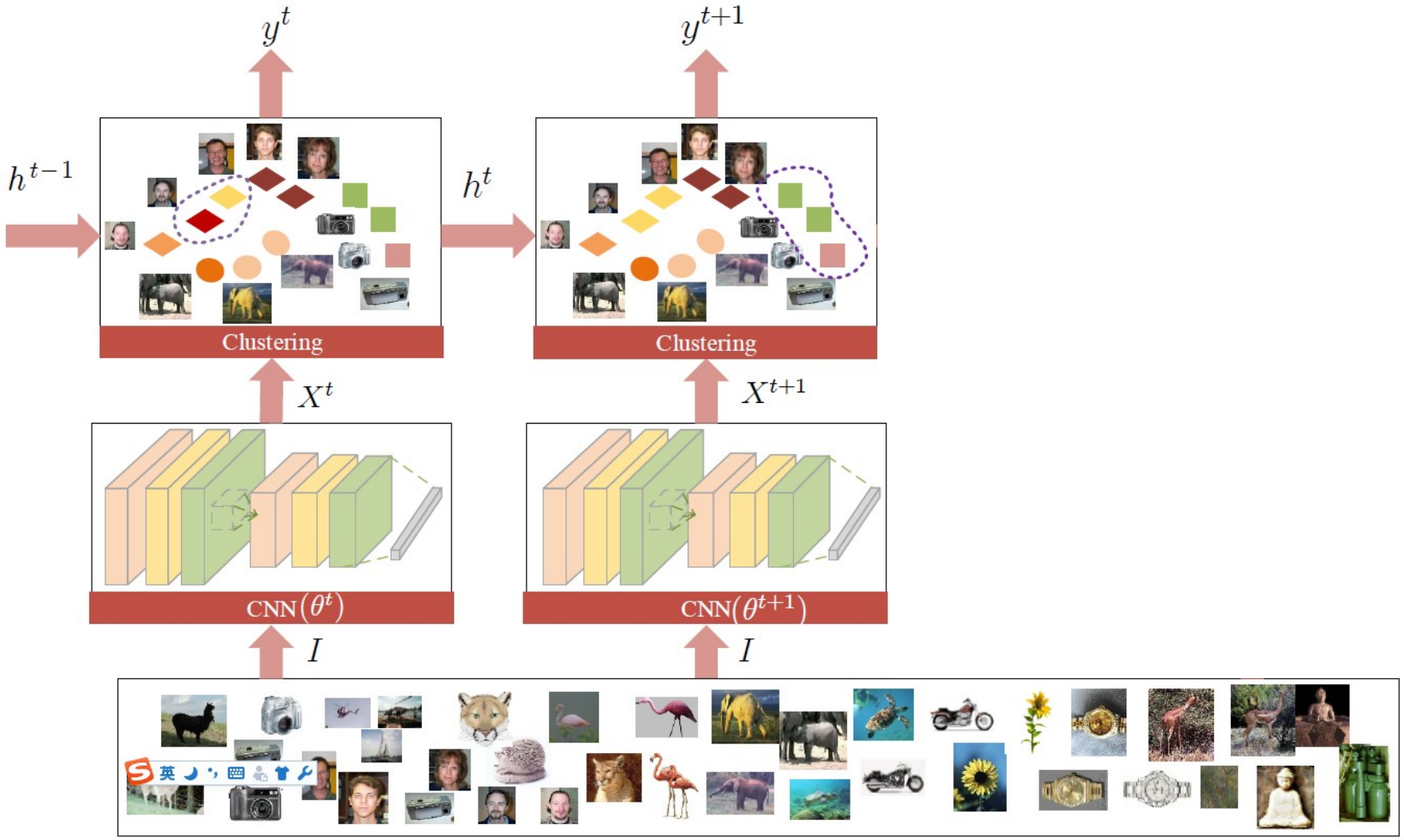
Approach: Recurrent Framework



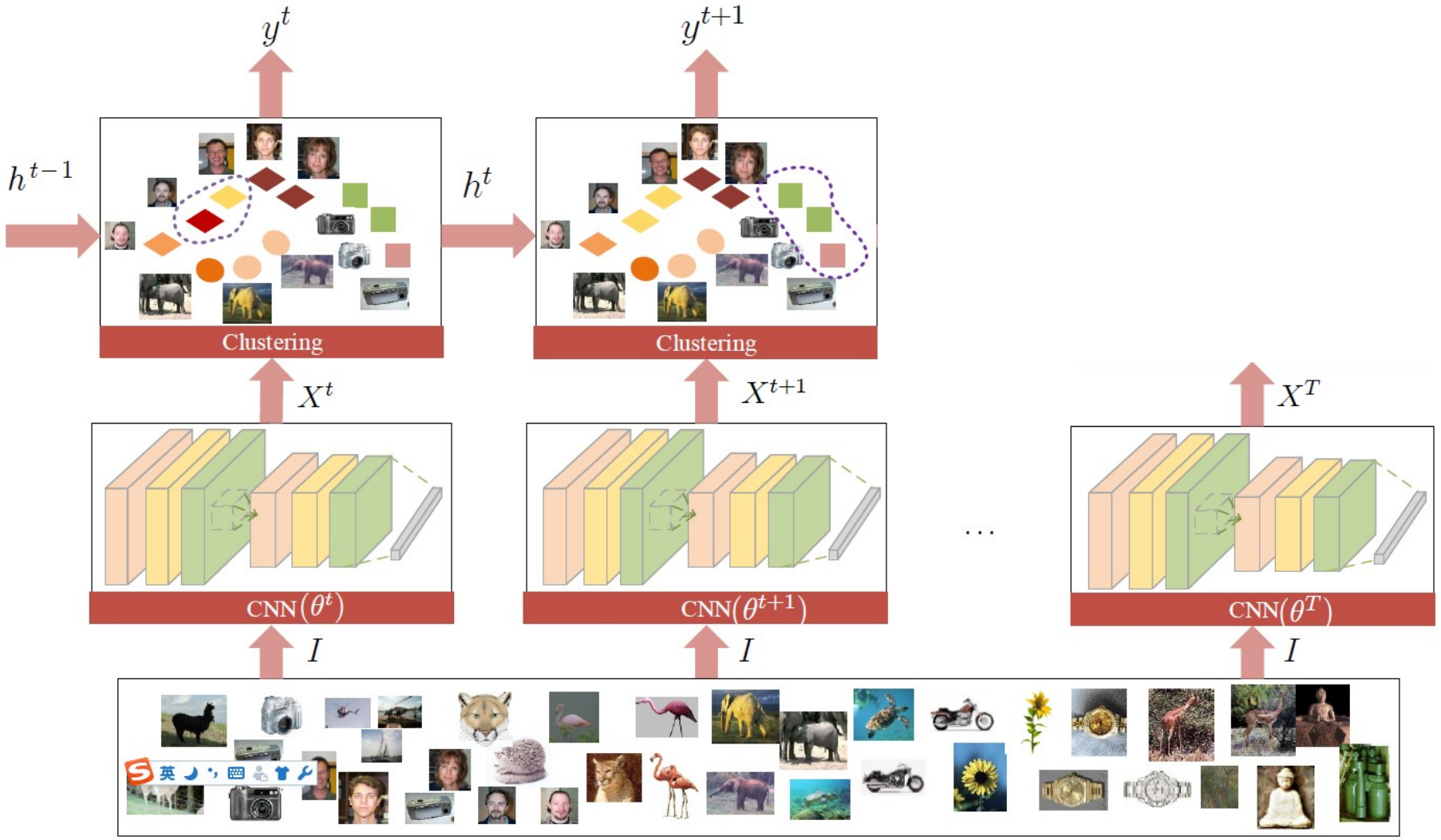
Approach: Recurrent Framework



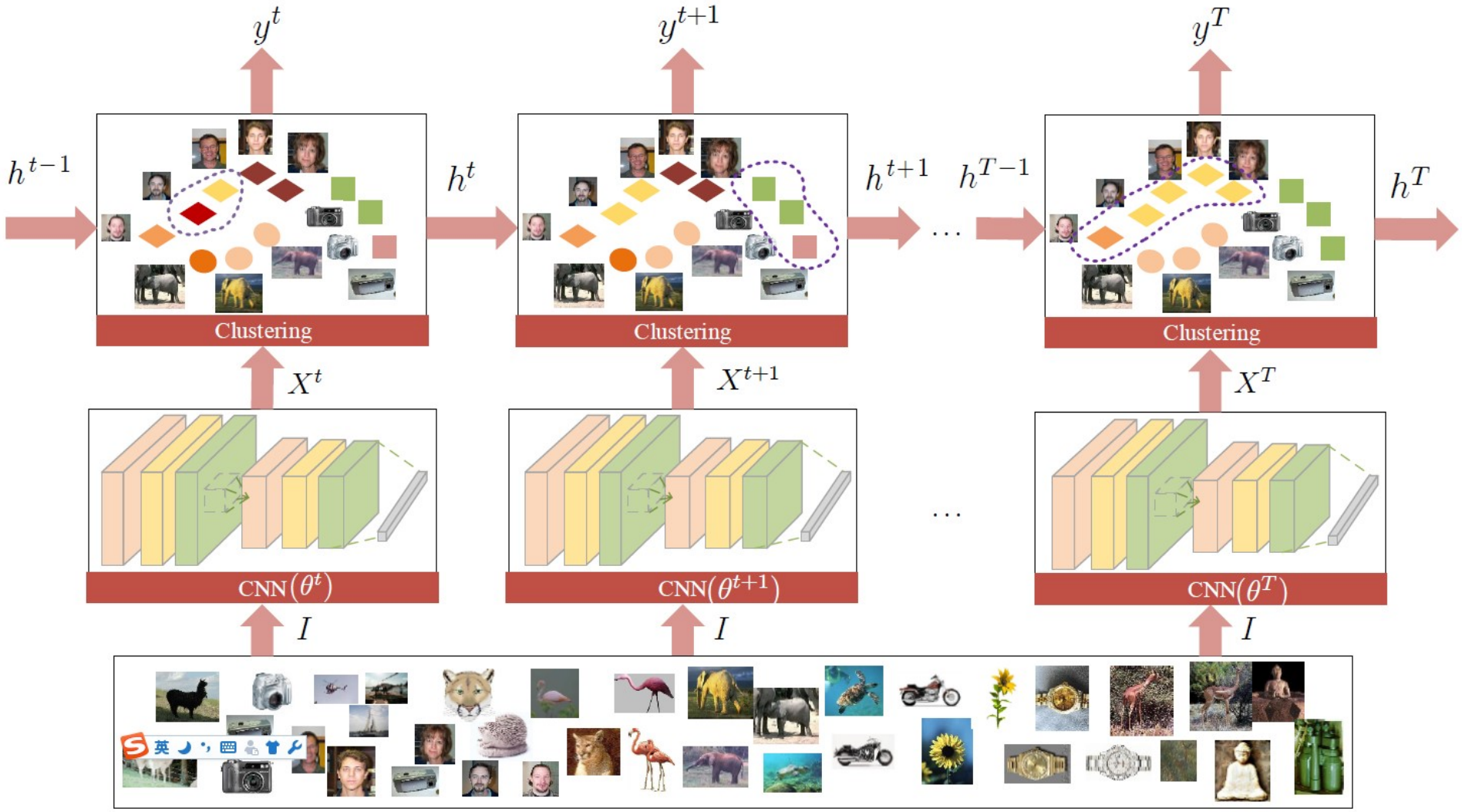
Approach: Recurrent Framework



Approach: Recurrent Framework



Approach: Recurrent Framework



Approach: Recurrent Framework

Backward at each time-step is time-consuming and prone to over-fitting!

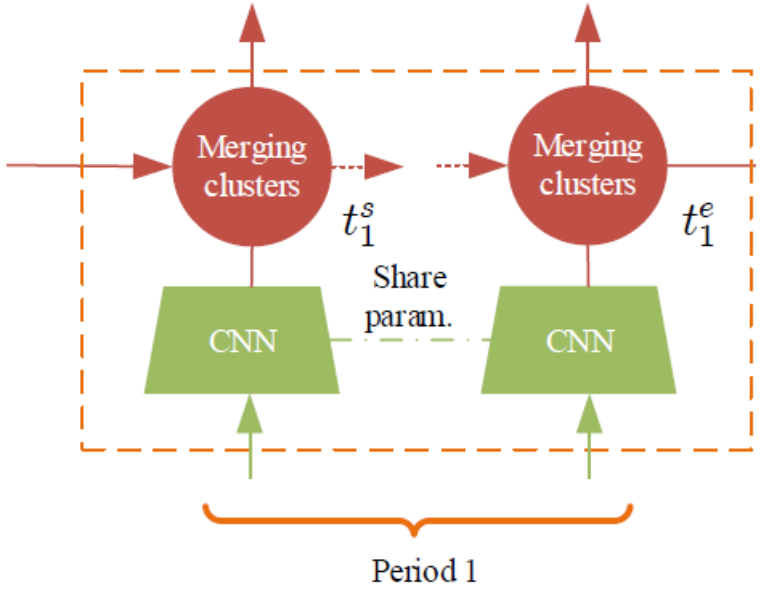
Approach: Recurrent Framework

Backward at each time-step is time-consuming and prone to over-fitting!

How about updating once for multiple time-steps?

Approach: Recurrent Framework

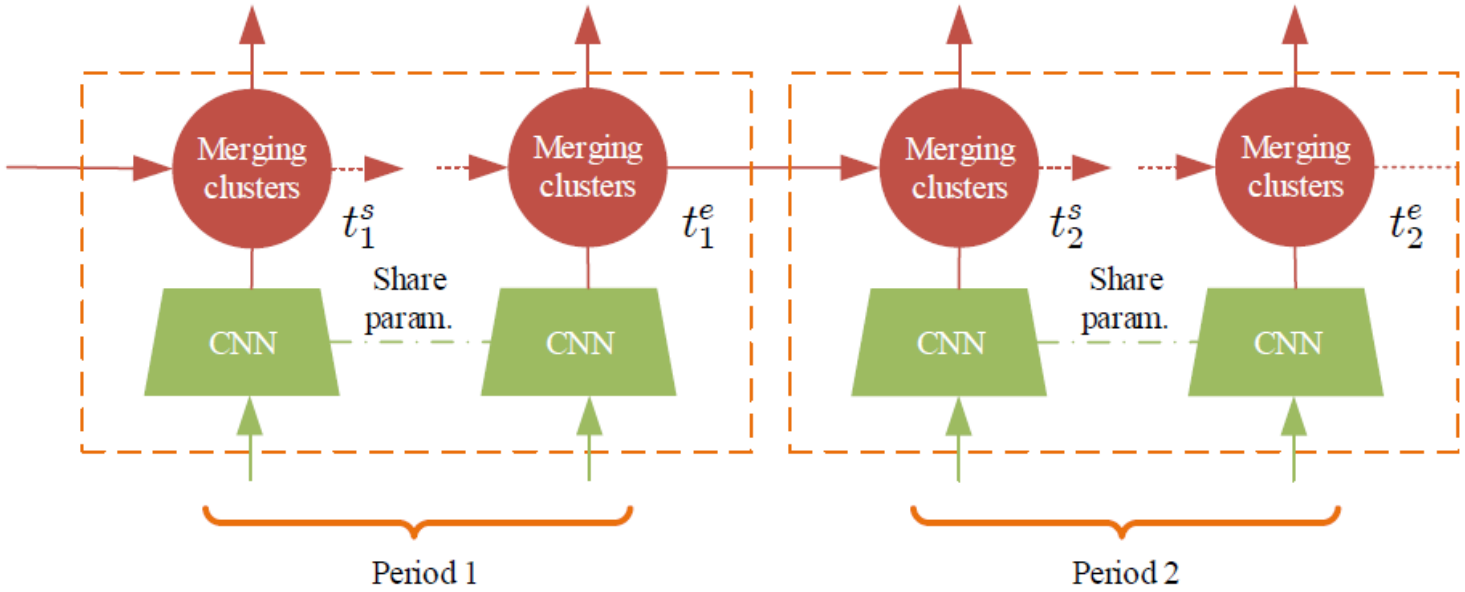
Partially Unrolling: divide all T time-steps into P periods



In each period, we merge clusters for multiple times and update CNN parameters at the end of period

Approach: Recurrent Framework

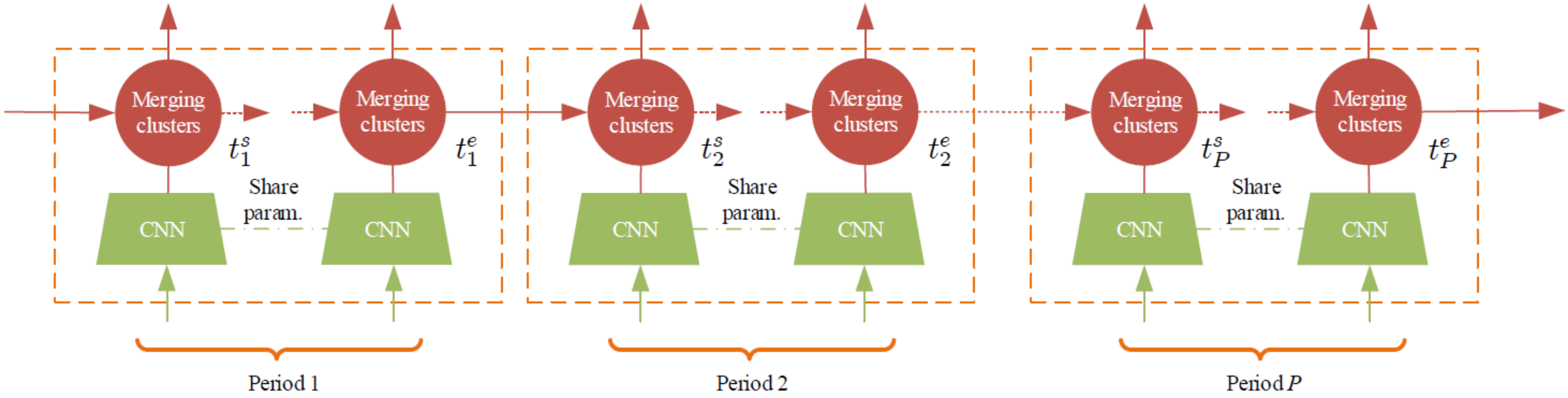
Partially Unrolling: divide all T time-steps into P periods



In each period, we merge clusters for multiple times and update CNN parameters at the end of period

Approach: Recurrent Framework

Partially Unrolling: divide all T time-steps into P periods

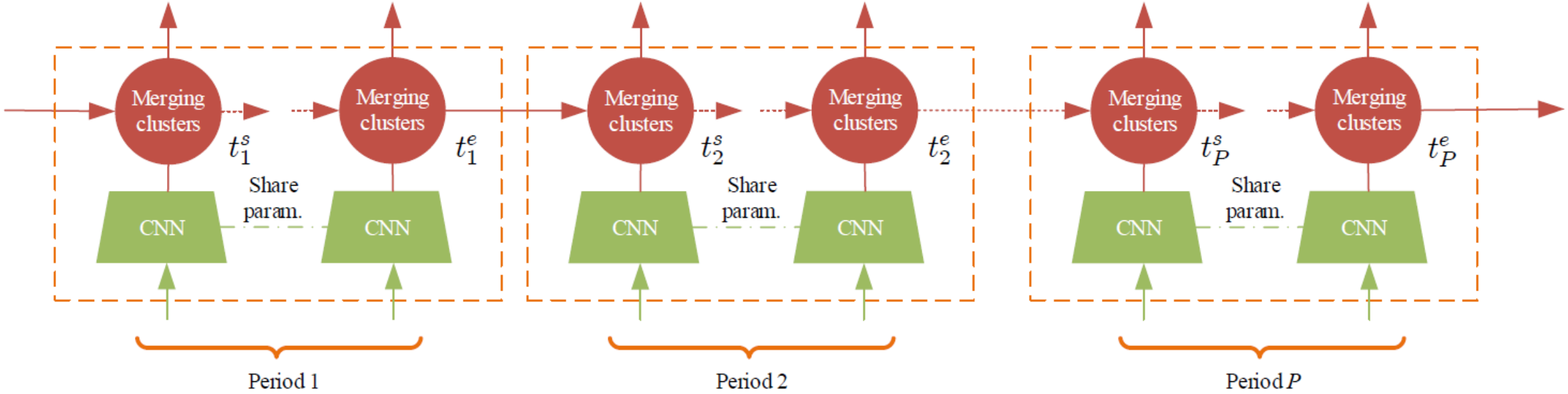


In each period, we merge clusters for multiple times and update CNN parameters at the end of period

P is determined by a hyper-parameter will be introduced later

Approach: Objective Function

$$\arg\min_{y,\theta} L(y,\theta \mid I) \qquad \arg\min_{y} L(y \mid \theta, I) \qquad \arg\min_{\theta} L(\theta \mid y, I)$$



Overall loss: $L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

Affinity measure

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

i-th cluster

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

K_c nearest neighbor clusters of i-th cluster

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

Affinity between i-th cluster and its NN

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

Affinity between i-th cluster and its NN

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Differences between two cluster affinities

Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

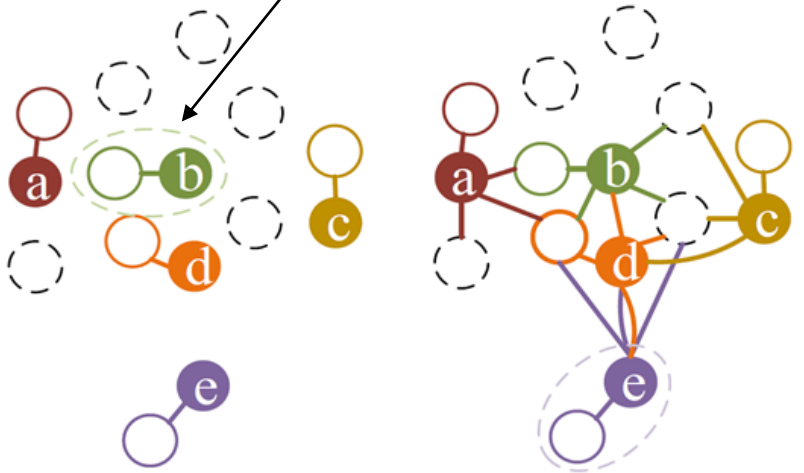
Affinity between i-th cluster and its NN

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Differences between two cluster affinities

Merge these two clusters



Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss at time-step t:

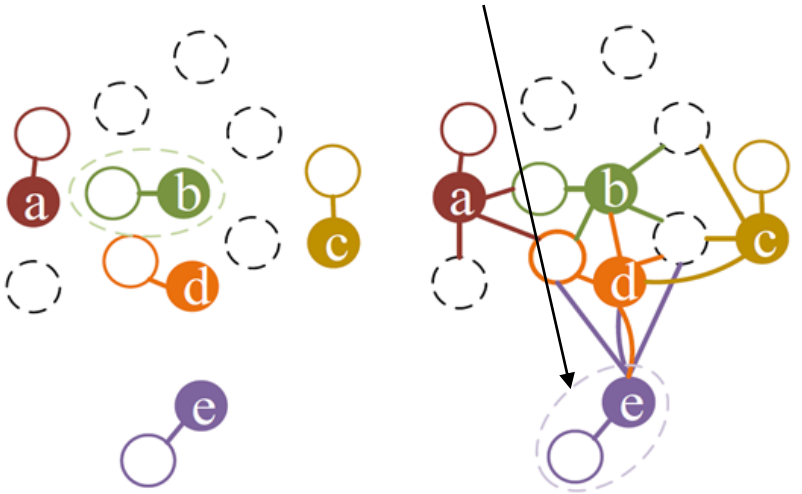
Affinity between i-th cluster and its NN

$$\mathcal{L}^t(\boldsymbol{y}^t,\boldsymbol{\theta}^t|\boldsymbol{y}^{t-1},\boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c-1)}\sum_{k=2}^{K_c}\left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t,\mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k])\right)$$

Differences between two cluster affinities

Merge these two clusters



Conventional Agg. Clustering Strategy

Proposed Agg. Clustering Strategy

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss in forward pass in period *p* (merge clusters):

Loss in forward pass in period *p* (merge clusters):

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss in forward pass in period *p* (merge clusters):

$$L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t\in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$$

Loss in forward pass in period *p* (merge clusters):

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss in forward pass in period *p* (merge clusters):

CNN parameters are fixed

$$L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t\in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$$

Loss in forward pass in period *p* (merge clusters):

Approach: Objective Function

$$L(\underbrace{\{y^1,\ldots,y^T\}}_{\text{cluster labels}},\underbrace{\{\theta^1,\ldots,\theta^T\}}_{\text{CNN parameters}} \mid I) = \underbrace{\sum_{t=1}^{T} L^t(y^t,\theta^t \mid y^{t-1},I)}_{\text{sum over all T timesteps}}$$

Loss in forward pass in period *p* (merge clusters):

CNN parameters are fixed

$$L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t\in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$$

Loss in forward pass in period *p* (merge clusters):

Cluster labels are fixed

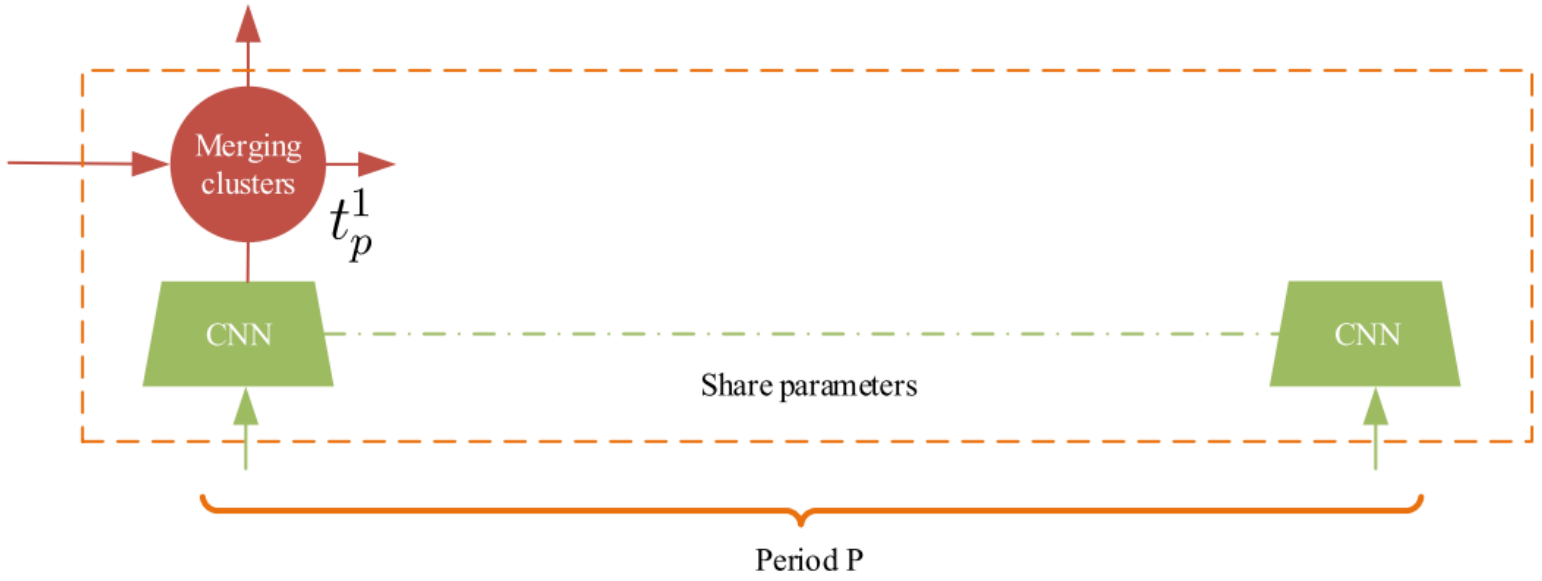
$$L(\theta \mid \underbrace{\{y\}_*^1,\ldots,\{y\}_*^p}_{\text{optimal solutions}}, I) = \underbrace{\sum_{k=1}^{p} L^k(\theta \mid \{y\}_*^k, I)}_{\text{sum over all previous periods}}$$

Approach: Objective Function

Forward Pass: $L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t \in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$

Simple Greedy Algorithm

Merge two clusters which minimize the loss at each time step

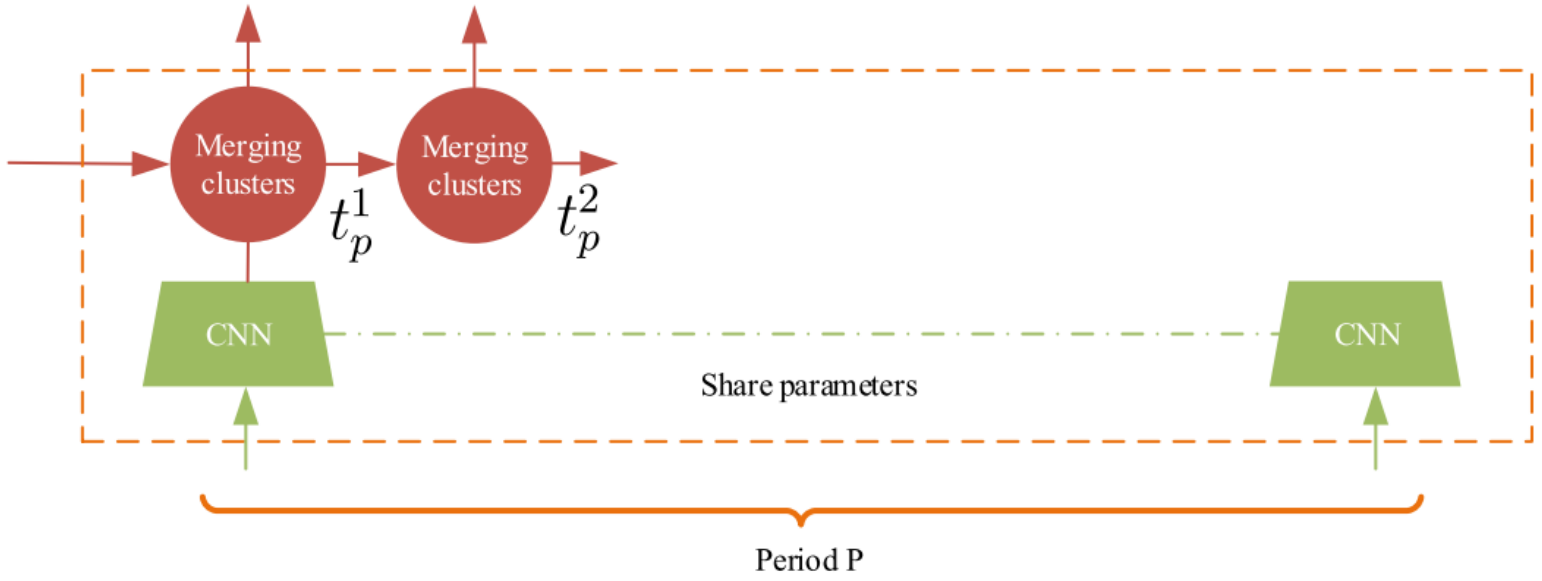


Approach: Objective Function

Forward Pass: $L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t \in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$

Simple Greedy Algorithm

Merge two clusters which minimize the loss at each time step

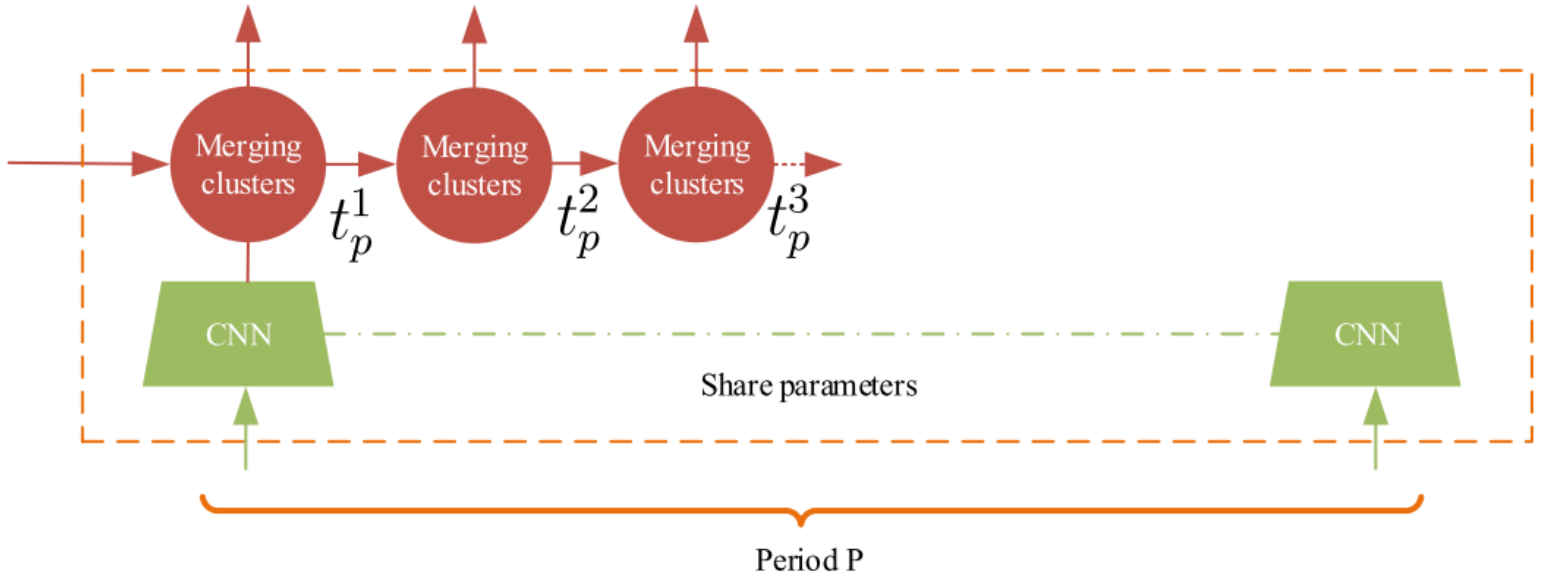


Approach: Objective Function

Forward Pass: $L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t \in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$

Simple Greedy Algorithm

Merge two clusters which minimize the loss at each time step

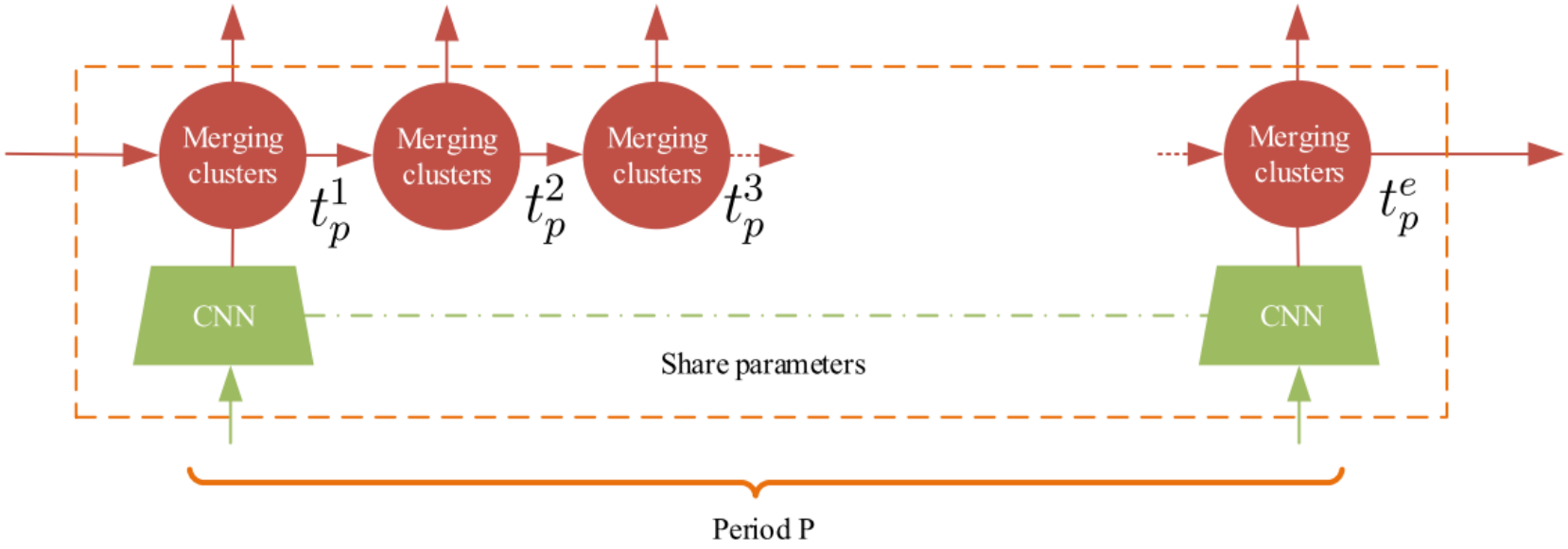


Approach: Objective Function

Forward Pass: $L(\{y\}^p \mid \theta^p, I) = \underbrace{\sum_{t \in p} L^t(y^t \mid \theta^p, y^{t-1}, I)}_{\text{sum over timesteps in unrolling period } p}$

Simple Greedy Algorithm

Merge two clusters which minimize the loss at each time step



Approach: Objective

Backward Pass:

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = \underbrace{\sum_{k=1}^{p} L^k(\theta \mid \{y\}_*^k, I)}_{\text{sum over all previous periods}}$$

Approach: Objective

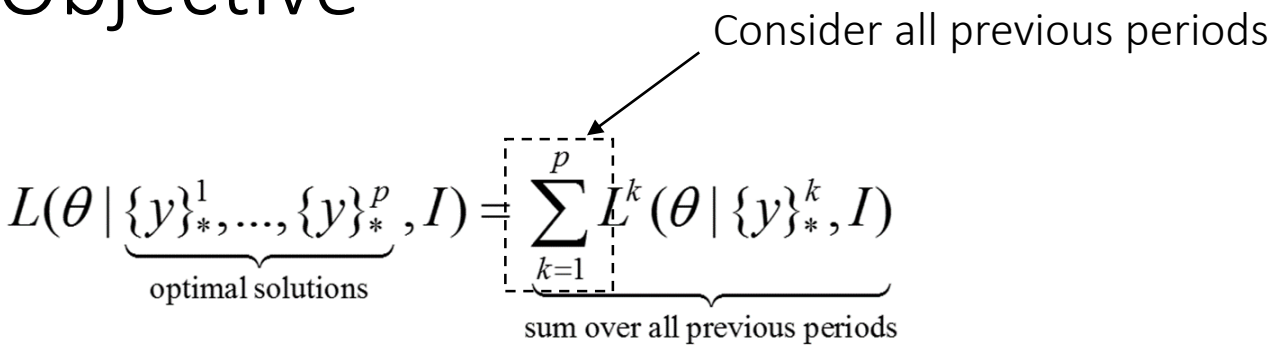
Backward Pass:

Consider all previous periods

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = \underbrace{\sum_{k=1}^{p} L^k(\theta \mid \{y\}_*^k, I)}_{\text{sum over all previous periods}}$$

Approach: Objective

Backward Pass:



Cluster based loss is not proper for batch optimization!!!

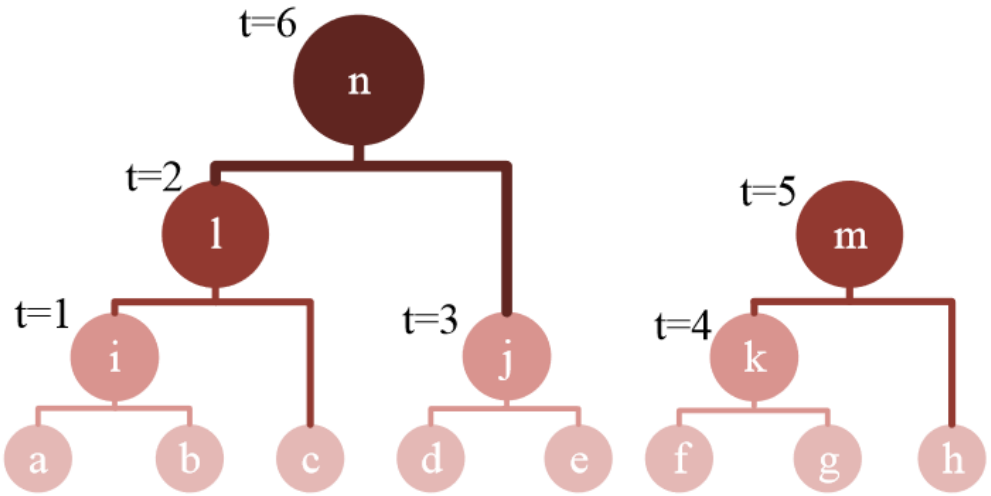
Approach: Objective

Backward Pass:

Consider all previous periods

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = \underbrace{\sum_{k=1}^{p} L^k(\theta \mid \{y\}_*^k, I)}_{\text{sum over all previous periods}}$$

Cluster based loss is not proper for batch optimization!!!



Approximation:

$$\begin{aligned}\mathcal{A}(\mathcal{C}_m \cup \mathcal{C}_n, \mathcal{C}_i) = \mathcal{A}(\mathcal{C}_m \to \mathcal{C}_i) + \mathcal{A}(\mathcal{C}_n \to \mathcal{C}_i) \\ + \frac{|\mathcal{C}_m|}{|\mathcal{C}_m| + |\mathcal{C}_n|}\mathcal{A}(\mathcal{C}_i \to \mathcal{C}_m) \\ + \frac{|\mathcal{C}_n|}{|\mathcal{C}_m| + |\mathcal{C}_n|}\mathcal{A}(\mathcal{C}_i \to \mathcal{C}_n)\end{aligned}$$

Approach: Objective

Backward Pass:

Consider all previous periods

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = \underbrace{\sum_{k=1}^{p} L^k(\theta \mid \{y\}_*^k, I)}_{\text{sum over all previous periods}}$$

Recall cluster-based loss:

$$\mathcal{L}^t(\boldsymbol{y}^t, \boldsymbol{\theta}^t | \boldsymbol{y}^{t-1}, \boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t, \mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$$

$$-\frac{\lambda}{(K_c - 1)} \sum_{k=2}^{K_c} \left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t, \mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t, \mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k]) \right)$$

Convert to sample-based loss:

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = -\sum_{i,j,k} (\overbrace{\gamma}^{\text{weight}} \quad \underbrace{A(x_i, x_j)}_{\text{Intra-sample affinity}} - \underbrace{A(x_i, x_k)}_{\text{Inter-sample affinity}})$$

Approach: Objective

Backward Pass:

Consider all previous periods

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = \underbrace{\sum_{k=1}^{p} L^k(\theta \mid \{y\}_*^k, I)}_{\text{sum over all previous periods}}$$

Recall cluster-based loss: $\mathcal{L}^t(\boldsymbol{y}^t, \boldsymbol{\theta}^t | \boldsymbol{y}^{t-1}, \boldsymbol{I}) = -\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t, \mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1])$

$$-\frac{\lambda}{(K_c - 1)} \sum_{k=2}^{K_c} \left(\boldsymbol{\mathcal{A}}(\mathcal{C}_i^t, \mathcal{N}_{\mathcal{C}_i^t}^{K_c}[1]) - \boldsymbol{\mathcal{A}}(\mathcal{C}_i^t, \mathcal{N}_{\mathcal{C}_i^t}^{K_c}[k]) \right)$$

Convert to sample-based loss:

Weighted triplet loss

$$L(\theta \mid \underbrace{\{y\}_*^1, \ldots, \{y\}_*^p}_{\text{optimal solutions}}, I) = -\sum_{i,j,k} (\overbrace{\gamma}^{\text{weight}} \underbrace{A(x_i, x_j)}_{\text{Intra-sample affinity}} - \underbrace{A(x_i, x_k)}_{\text{Inter-sample affinity}})$$

Approach: Algorithm & Implementation

Algorithm 1 Joint Optimization on $\boldsymbol{y}$ and $\boldsymbol{\theta}$

Input:

$\boldsymbol{I}$: = collection of image data;

n_c^*: = target number of clusters;

Output:

$\boldsymbol{y}^*, \boldsymbol{\theta}^*$: = final image labels and CNN parameters;

$t \leftarrow 0; p \leftarrow 0$

Initialize $\boldsymbol{\theta}$ and $\boldsymbol{y}$

repeat

  Update $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by merging two clusters

  if $t = t_p^e$ **then**

    Update $\boldsymbol{\theta}^p$ to $\boldsymbol{\theta}^{p+1}$ by training CNN

    $p \leftarrow (p+1)$

  end if

  $t \leftarrow t+1$

until Cluster number reaches n_c^*

$\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t; \boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^p$

Approach: Algorithm & Implementation

Algorithm 1 Joint Optimization on $\boldsymbol{y}$ and $\boldsymbol{\theta}$

Input:

$\boldsymbol{I}$: = collection of image data;

n_c^*: = target number of clusters;

Output:

$\boldsymbol{y}^*, \boldsymbol{\theta}^*$: = final image labels and CNN parameters;

$t \leftarrow 0; p \leftarrow 0$
Initialize $\boldsymbol{\theta}$ and $\boldsymbol{y}$
repeat
  Update $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by merging two clusters
  if $t = t_p^e$ **then**
    Update $\boldsymbol{\theta}^p$ to $\boldsymbol{\theta}^{p+1}$ by training CNN
    $p \leftarrow (p+1)$
  end if
  $t \leftarrow t+1$
until Cluster number reaches n_c^*
$\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t; \boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^p$

Raw image data

Approach: Algorithm & Implementation

Algorithm 1 Joint Optimization on $\boldsymbol{y}$ and $\boldsymbol{\theta}$

Input:

$\boldsymbol{I}$: = collection of image data;

n_c^*: = target number of clusters;

Output:

$\boldsymbol{y}^*, \boldsymbol{\theta}^*$: = final image labels and CNN parameters;

$t \leftarrow 0; p \leftarrow 0$
Initialize $\boldsymbol{\theta}$ and $\boldsymbol{y}$
repeat
  Update $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by merging two clusters
  if $t = t_p^e$ **then**
    Update $\boldsymbol{\theta}^p$ to $\boldsymbol{\theta}^{p+1}$ by training CNN
    $p \leftarrow (p+1)$
  end if
  $t \leftarrow t+1$
until Cluster number reaches n_c^*
$\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t; \boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^p$

Raw image data

Assume it is known

Approach: Algorithm & Implementation

Algorithm 1 Joint Optimization on $\boldsymbol{y}$ and $\boldsymbol{\theta}$

Input:

$\boldsymbol{I}$: = collection of image data;

n_c^*: = target number of clusters;

Output:

$\boldsymbol{y}^*, \boldsymbol{\theta}^*$: = final image labels and CNN parameters;

1: $t \leftarrow 0; p \leftarrow 0$
2: Initialize $\boldsymbol{\theta}$ and $\boldsymbol{y}$
3: **repeat**
4: Update $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by merging two clusters
5: **if** $t = t_p^e$ **then**
6: Update $\boldsymbol{\theta}^p$ to $\boldsymbol{\theta}^{p+1}$ by training CNN
7: $p \leftarrow (p+1)$
8: **end if**
9: $t \leftarrow t+1$
10: **until** Cluster number reaches n_c^*
11: $\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t; \boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^p$

Raw image data

Assume it is known

Randomly initialize CNN parameters
4 samples in each cluster in average

Approach: Algorithm & Implementation

Algorithm 1 Joint Optimization on $\boldsymbol{y}$ and $\boldsymbol{\theta}$

Input:

$\boldsymbol{I}$: = collection of image data;

n_c^*: = target number of clusters;

Output:

$\boldsymbol{y}^*, \boldsymbol{\theta}^*$: = final image labels and CNN parameters;

1: $t \leftarrow 0; p \leftarrow 0$
2: Initialize $\boldsymbol{\theta}$ and $\boldsymbol{y}$
3: **repeat**
4: Update $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by merging two clusters
5: **if** $t = t_p^e$ **then**
6: Update $\boldsymbol{\theta}^p$ to $\boldsymbol{\theta}^{p+1}$ by training CNN
7: $p \leftarrow (p+1)$
8: **end if**
9: $t \leftarrow t+1$
10: **until** Cluster number reaches n_c^*
11: $\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t; \boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^p$

Raw image data

Assume it is known

Randomly initialize CNN parameters
4 samples in each cluster in average

Train CNN for about 20 epochs

Approach: Algorithm & Implementation

Algorithm 1 Joint Optimization on $\boldsymbol{y}$ and $\boldsymbol{\theta}$

Input:

$\boldsymbol{I}$: = collection of image data;

n_c^*: = target number of clusters;

Output:

$\boldsymbol{y}^*, \boldsymbol{\theta}^*$: = final image labels and CNN parameters;

1: $t \leftarrow 0; p \leftarrow 0$
2: Initialize $\boldsymbol{\theta}$ and $\boldsymbol{y}$
3: **repeat**
4: Update $\boldsymbol{y}^t$ to $\boldsymbol{y}^{t+1}$ by merging two clusters
5: **if** $t = t_p^e$ **then**
6: Update $\boldsymbol{\theta}^p$ to $\boldsymbol{\theta}^{p+1}$ by training CNN
7: $p \leftarrow (p+1)$
8: **end if**
9: $t \leftarrow t+1$
10: **until** Cluster number reaches n_c^*
11: $\boldsymbol{y}^* \leftarrow \boldsymbol{y}^t; \boldsymbol{\theta}^* \leftarrow \boldsymbol{\theta}^p$

Raw image data

Assume it is known

Randomly initialize CNN parameters
4 samples in each cluster in average

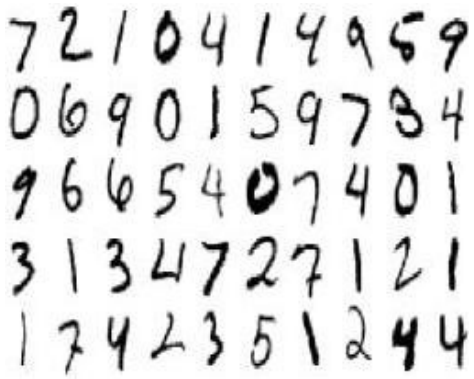
Train CNN for about 20 epochs

We can go back and retrain the model, but it improve slightly

Experiments

- Datasets
- Network Architecture
- Image Clustering
- Representation Learning

Experiments: Datasets



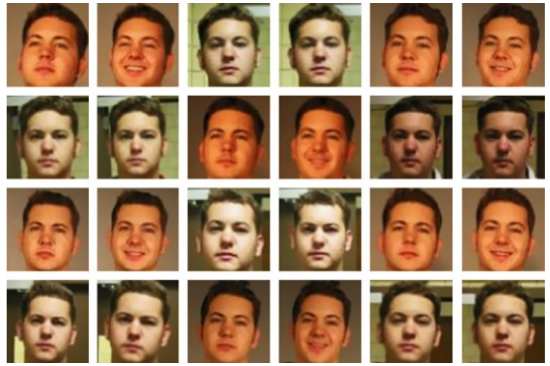
MNIST (70000, 10, 28x28)

USPS (11000, 10, 16x16)

COIL20 (1440, 20, 128x128)

COIL100 (7200, 100, 128x128)

UMist (575, 20, 112x92)



FRGC (2462, 20, 32x32)

CMU-PIE (2856, 68, 32x32)

Youtube Face (1000, 41, 55x55)

Experiments: Settings

Two important parameters

Hyper-parameter	K_s	a	K_c	λ	γ	η
Value	20	1.0	5	1.0	2.0	0.9 or 0.2

Dataset	*COIL20*	*COIL100*	*USPS*	*MNIST-test*	*MNIST-full*	UMist	*FRGC*	*CMU-PIE*	*YTF*
conv1	✓	✓	✓	✓	✓	✓	✓	✓	✓
bn1	✓	✓	✓	✓	✓	✓	✓	✓	✓
relu1	✓	✓	✓	✓	✓	✓	✓	✓	✓
pool1	✓	✓	✓	✓	✓	✓	✓	✓	✓
conv2	✓	✓	–	✓	✓	✓	✓	✓	✓
bn2	✓	✓	–	✓	✓	✓	✓	✓	✓
relu2	✓	✓	–	✓	✓	✓	✓	✓	✓
pool2	✓	✓	–	–	–	✓	✓	✓	✓
conv3	✓	✓	–	–	–	✓	–	–	–
bn3	✓	✓	–	–	–	✓	–	–	–
relu3	✓	✓	–	–	–	✓	–	–	–
pool3	✓	✓	–	–	–	✓	–	–	–
conv4	✓	✓	–	–	–	–	–	–	–
bn4	✓	✓	–	–	–	–	–	–	–
relu4	✓	✓	–	–	–	–	–	–	–
pool4	✓	✓	–	–	–	–	–	–	–
ip1	✓	✓	✓	✓	✓	✓	✓	✓	✓
l2-norm	✓	✓	✓	✓	✓	✓	✓	✓	✓
wt-loss	✓	✓	✓	✓	✓	✓	✓	✓	✓

Set the layer numbers so that the Output feature map is about 10x10

Experiments: Clustering : Performance

+6.43% on NMI to best performance of existing approaches averaged over all datasets

Table 3: Quantitative clustering performance (NMI) for different algorithms using image intensities as input.

Dataset	*COIL20*	*COIL100*	*USPS*	*MNIST-test*	*MNIST-full*	*UMist*	*FRGC*	*CMU-PIE*	*YTF*
K-means [39]	0.775	0.822	0.447	0.528	0.500	0.609	0.389	0.549	0.761
SC-NJW [43]	0.860/0.889	0.872/0.854	0.409/0.690	0.528/0.755	0.476	0.727	0.186	0.543	0.752
SC-ST [67]	0.673/0.895	0.706/0.858	0.342/0.726	0.445/0.756	0.416	0.611	0.431	0.581	0.620
SC-LS [3]	0.877	0.833	0.681	0.756	0.706	0.810	0.550	0.788	0.759
N-Cuts [52]	0.768/0.884	0.861/0.823	0.382/0.675	0.386/0.753	0.411	0.782	0.285	0.411	0.742
AC-Link [25]	0.512	0.711	0.579	0.662	0.686	0.643	0.168	0.545	0.738
AC-Zell [70]	0.954/0.911	0.963/0.913	0.774/0.799	0.810/0.768	0.017	0.755	0.351	0.910	0.733
AC-GDL [68]	0.945/0.937	0.954/0.929	0.854/0.824	0.864/0.844	0.017	0.755	0.351	0.934	0.622
AC-PIC [69]	0.950	0.964	0.840	0.853	0.017	0.750	0.415	0.902	0.697
NMF-LP [1]	0.720	0.783	0.435	0.467	0.452	0.560	0.346	0.491	0.720
NMF-D [57]	0.692	0.719	0.286	0.243	0.148	0.500	0.258	0.983/0.910	0.569
TSC-D [61]	-/0.928	-	-	-	-/0.651	-	-	-	-
OURS-SF	**1.000**	0.978	0.858	0.876	0.906	**0.880**	0.566	0.984	**0.848**
OURS-RC	**1.000**	**0.985**	**0.913**	**0.915**	**0.913**	0.877	**0.574**	**1.00**	**0.848**

Experiments: Clustering : Performance

+12.76% on AC to best performance of existing approaches averaged over all datasets

Table 10: Quantitative clustering performance (AC) for different algorithms using image intensities as input.

Dataset	*COIL20*	*COIL100*	*USPS*	*MNIST-test*	*MNIST-full*	*UMist*	*FRGC*	*CMU-PIE*	*YTF*
K-means [39]	0.665	0.580	0.467	0.560	0.564	0.419	0.327	0.246	0.548
SC-NJW [43]	0.641	0.544	0.413	0.220	0.502	0.551	0.178	0.255	0.551
SC-ST [67]	0.417	0.300	0.308	0.454	0.311	0.411	0.358	0.293	0.290
SC-LS [3]	0.717	0.609	0.659	0.740	0.714	0.568	0.407	0.549	0.544
N-Cuts [52]	0.544	0.577	0.314	0.304	0.327	0.550	0.235	0.155	0.536
AC-Link [25]	0.251	0.269	0.421	0.693	0.657	0.398	0.175	0.201	0.547
AC-Zell [70]	0.867	0.811	0.575	0.693	0.112	0.517	0.266	0.765	0.519
AC-GDL [68]	0.865	0.797	0.867	0.933	0.113	0.563	0.266	0.842	0.430
AC-PIC [69]	0.855	0.840	0.855	0.920	0.115	0.576	0.320	0.797	0.472
NMF-LP [1]	0.621	0.553	0.522	0.479	0.471	0.365	0.259	0.229	0.546
OURS-SF	**1.000**	0.894	0.922	0.940	0.959	**0.809**	**0.461**	0.980	**0.684**
OURS-RC	**1.000**	**0.916**	**0.950**	**0.961**	**0.964**	**0.809**	**0.461**	**1.000**	**0.684**

Experiments: Clustering : Performance

Average +21.5% on NMI

Table 4: Quantitative clustering performance (NMI) for different algorithms using our learned representations as inputs.

Dataset	*COIL20*	*COIL100*	*USPS*	*MNIST-test*	*MNIST-full*	*UMist*	*FRGC*	*CMU-PIE*	*YTF*
K-means [39]	0.926	0.919	0.758	0.908	0.927	0.871	0.636	0.956	0.835
SC-NJW [43]	0.915	0.898	0.753	0.878	0.931	0.833	0.625	0.957	0.789
SC-ST [67]	0.959	0.922	0.741	0.911	0.906	0.847	**0.651**	0.938	0.741
SC-LS [3]	0.950	0.905	0.780	0.912	**0.932**	**0.879**	0.639	0.950	0.802
N-Cuts [52]	0.963	0.900	0.705	0.910	0.930	0.877	0.640	0.995	0.823
AC-Link [25]	0.896	0.884	0.783	0.901	0.918	0.872	0.621	0.990	0.803
AC-Zell [70]	**1.000**	0.989	0.910	0.893	0.919	0.870	0.551	**1.000**	0.821
AC-GDL [68]	**1.000**	0.985	0.913	**0.915**	0.913	0.870	0.574	**1.000**	**0.842**
AC-PIC [69]	**1.000**	**0.990**	**0.914**	0.909	0.907	0.870	0.553	**1.000**	0.829
NMF-LP [1]	0.855	0.834	0.729	0.905	0.926	0.854	0.575	0.690	0.788

Experiments: Clustering : Performance

Average +25.7% on NMI

Table 11: Quantitative clustering performance (AC) for different algorithms using our learned representations as inputs.

Dataset	*COIL20*	*COIL100*	*USPS*	*MNIST-test*	*MNIST-full*	*UMist*	*FRGC*	*CMU-PIE*	*YTF*
K-means [39]	0.821	0.751	0.776	0.957	0.969	0.761	0.476	0.834	0.660
SC-NJW [43]	0.738	0.659	0.716	0.868	0.972	0.707	0.485	0.776	0.521
SC-ST [67]	0.851	0.705	0.661	0.960	0.958	0.697	0.496	0.896	0.575
SC-LS [3]	0.867	0.735	0.792	0.960	**0.973**	0.733	0.502	0.802	0.571
N-Cuts [52]	0.888	0.626	0.634	0.959	0.971	**0.798**	**0.504**	0.981	0.441
AC-Link [25]	0.678	0.539	0.773	0.955	0.964	0.795	0.495	0.947	0.602
AC-Zell [70]	**1.000**	0.931	0.879	0.879	0.969	0.790	0.449	**1.000**	0.644
AC-GDL [68]	**1.000**	0.920	0.949	**0.961**	0.878	0.790	0.461	**1.000**	**0.677**
AC-PIC [69]	**1.000**	**0.950**	**0.955**	0.958	0.882	0.790	0.438	**1.000**	0.652
NMF-LP [1]	0.769	0.603	0.778	0.955	0.970	0.725	0.481	0.504	0.575

Experiments: Clustering : Performance

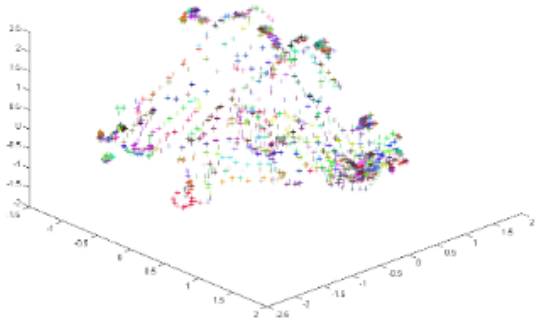
Our clustering performance vs. that of existing clustering approaches using raw image data.

Method	K-means	SC-NJW	SC-LS	N-Cuts	AC-Zell	AC-GDL	AC-PIC	NMF-LP	NMF-D	OURS-SF	OURS-RC
Avg. NMI	0.598	0.595	0.751	0.559	0.696	0.699	0.710	0.553	0.489	0.877	**0.892**
Avg. AC	0.486	0.428	0.612	0.394	0.569	0.631	0.639	0.449	-	0.850	**0.861**

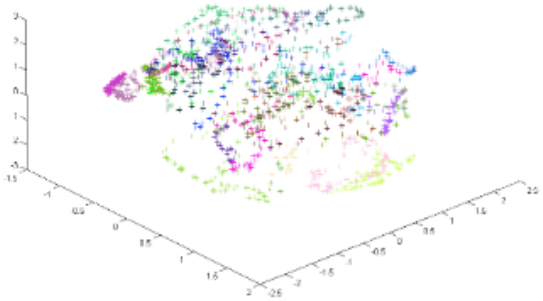
Clustering performance using our representation fed to existing clustering algorithms.

Method	K-means	SC-NJW	SC-LS	N-Cuts	AC-Zell	AC-GDL	AC-PIC	NMF-LP
Avg. NMI	0.860	0.842	0.861	0.860	0.884	**0.890**	0.886	0.795
Avg. AC	0.778	0.716	0.771	0.756	0.838	**0.848**	0.847	0.707

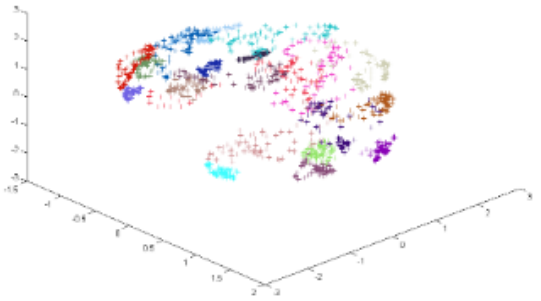
Experiments: Clustering : Visualization



(a) Initial stage (421)

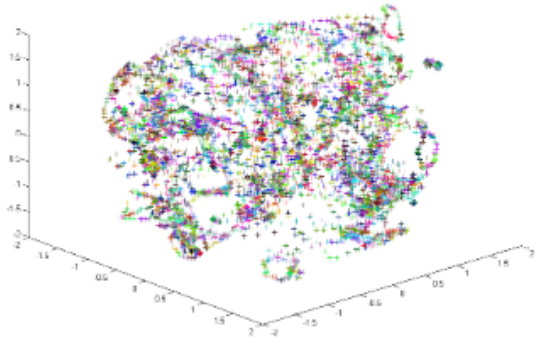


(b) Middle stage (42)

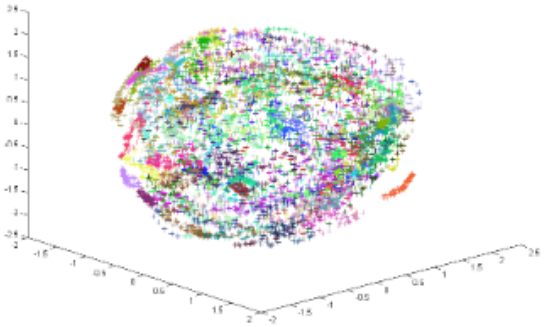


(c) Final stage (20)

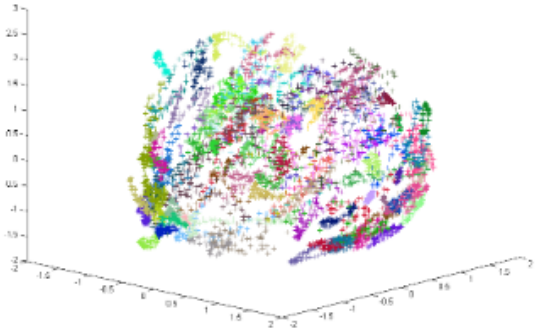
COIL-20



(d) Initial stage (2162)



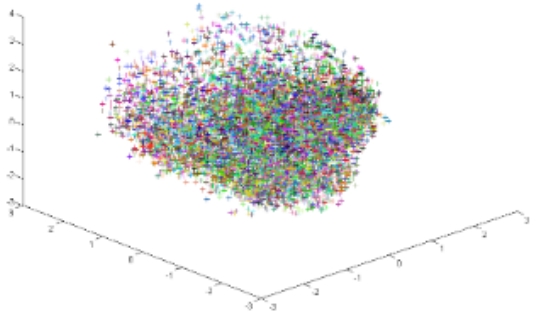
(e) Middle stage (216)



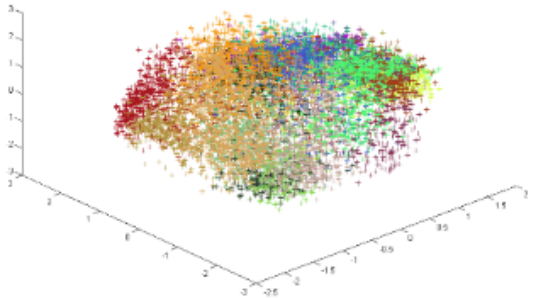
(f) Final stage (100)

COIL-100

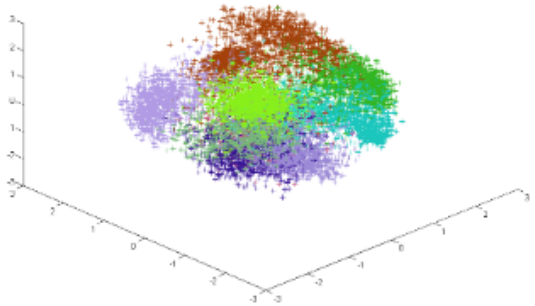
Experiments: Clustering : Visualization



(g) Initial stage (2232)

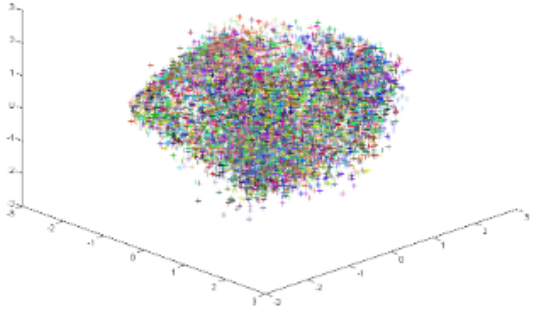


(h) Middle stage (22)

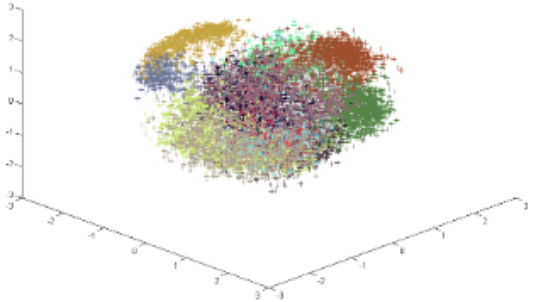


(i) Final stage (10)

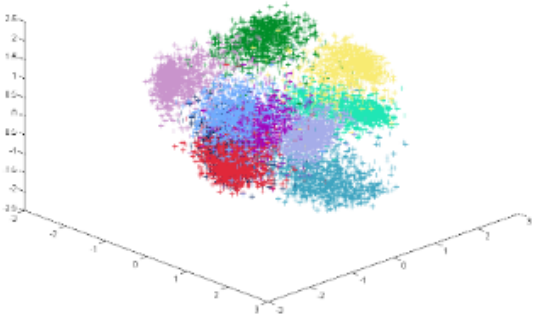
USPS



(j) Initial stage (1762)



(k) Middle stage (22)



(l) Final stage (10)

MNIST-test

Experiments: Clustering : Ablation study

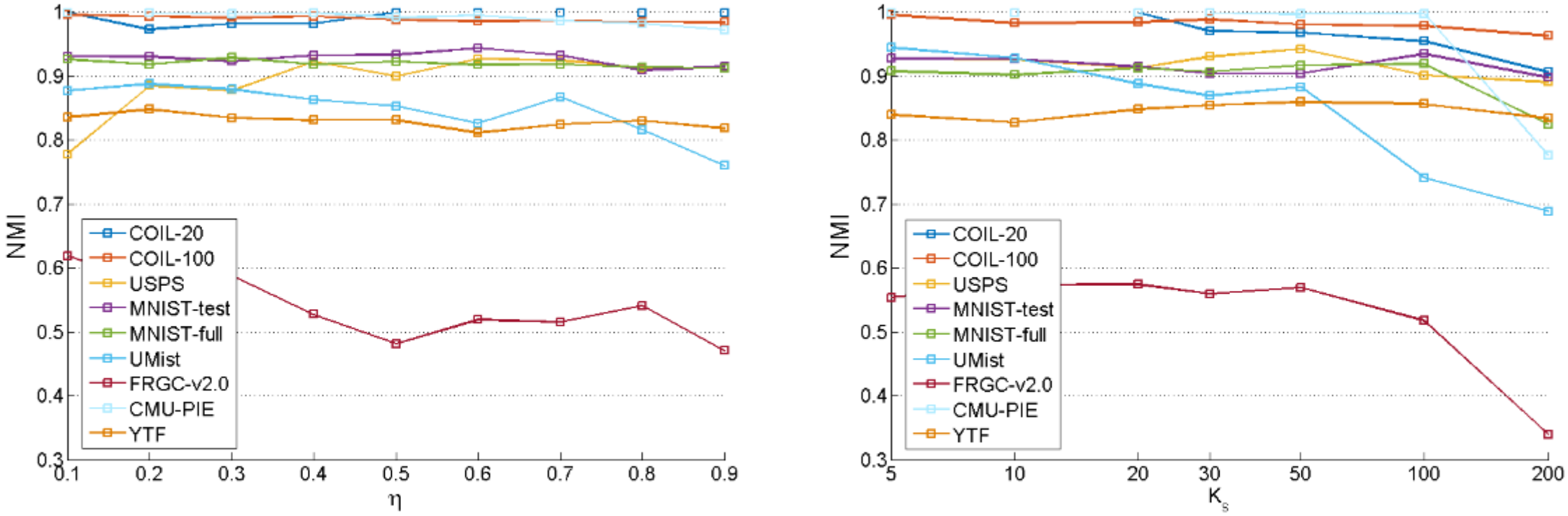


Figure 7: Clustering performance (NMI) with different η (left) and K_s (right).

Experiments: Clustering : Verification

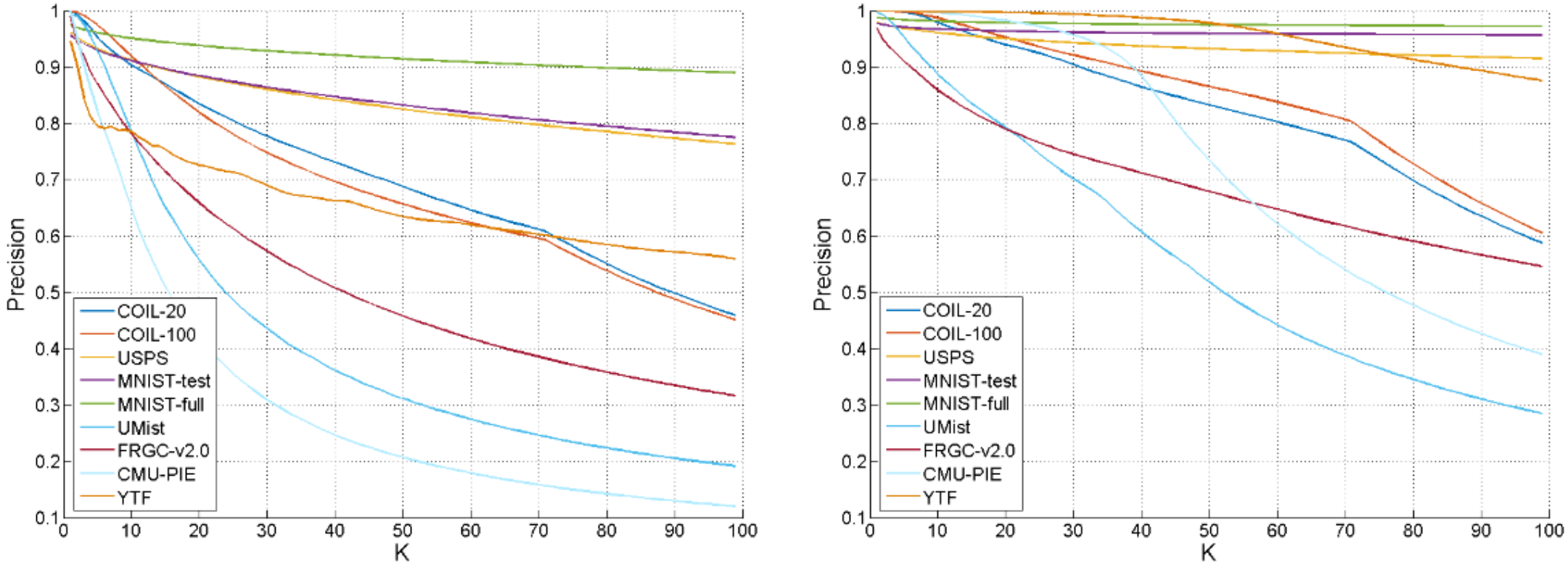
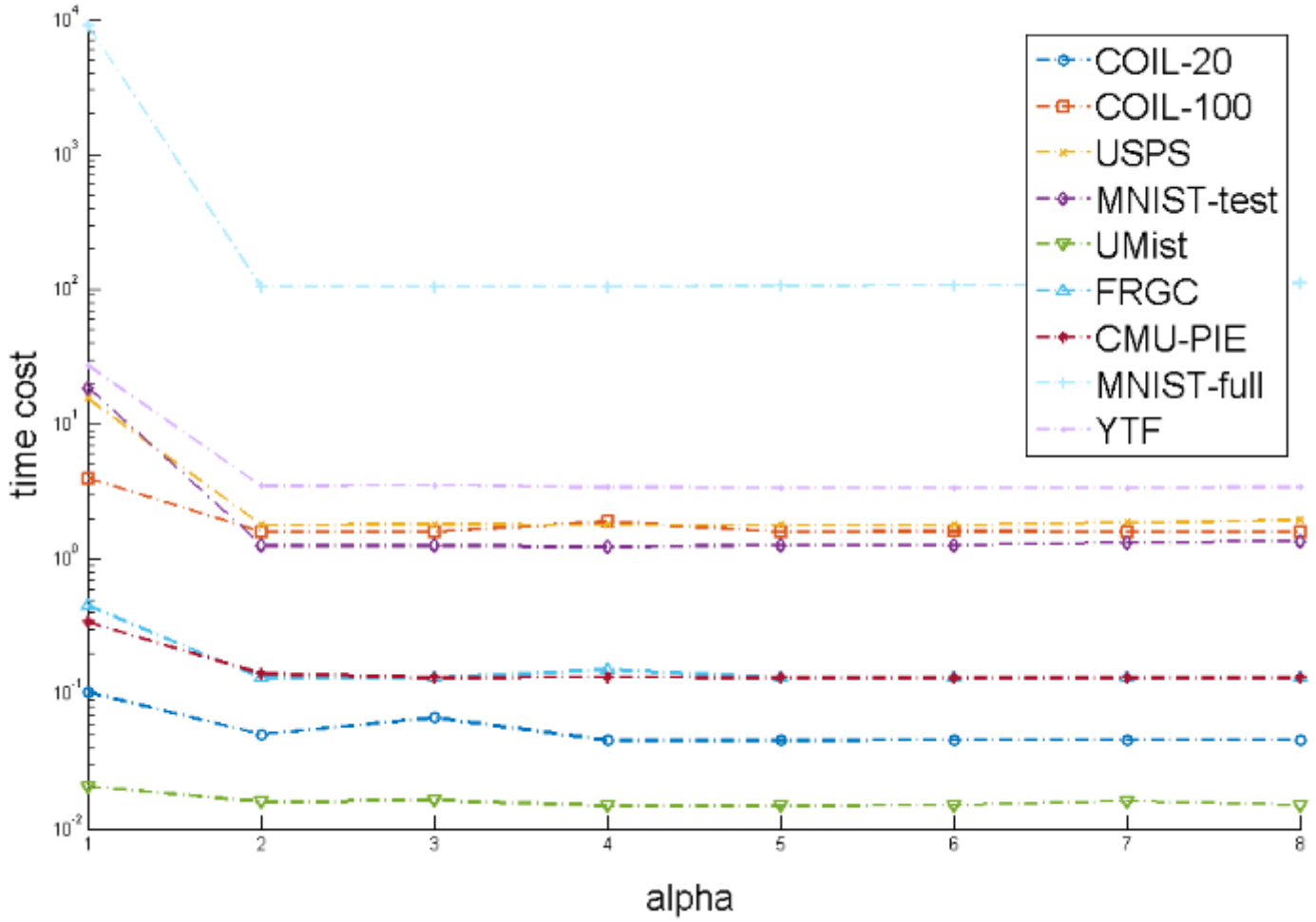


Figure 8: Average purity of K-nearest neighbour for varying values of K. Left is computed using raw image data, while right is computed using our learned representation.

Experiments: Clustering : Time Cost



Experiments: Representation Learning

Representation transfer

Table 5: NMI performance across COIL20 and COIL100.

Layer	*data*	top(*ip*)	top-1	top-2
COIL20 → COIL100	0.924	0.927	**0.939**	0.934
COIL100 → COIL20	0.944	0.949	**0.957**	0.951

Table 6: NMI performance across MNIST-test and USPS.

Layer	*data*	top(*ip*)	top-1	top-2
MNIST-test → USPS	0.874	0.892	0.907	**0.908**
USPS → MNIST-test	0.872	0.873	**0.886**	-

Representation learning

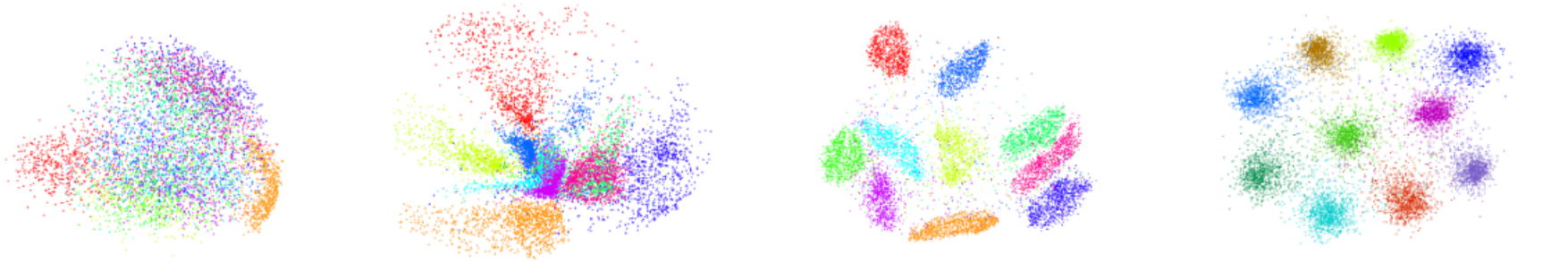
Testing generalization of our learnt (unsupervised) representation to LFW face verification.

#Samples	10k	20k	30k	50k	100k
Supervised	**0.737**	**0.746**	0.748	**0.764**	0.770
OURS	0.728	0.743	**0.750**	0.762	0.767

Evaluation on CIFAR-10 classification

#Samples	K-means	conv1	conv2	conv1&2
5k	62.81%	63.05%	63.10%	**63.50%**
10k	68.01%	68.30%	68.46%	**69.11%**
25k	74.01%	72.83%	72.93%	**75.11%**
50k (full set)	76.59%	74.68%	74.68%	**78.55%**

Extensions: Data Visualization



(a) PCA. (b) Autoencoder. (c) Parametric t-SNE. (d) Visualization by our method.

Figure 9: Visualization of 10,000 MNIST test samples in different embedding spaces.

Table 13: 1-nearest neighbor classification error on low-dimensional embedding of MNIST dataset.

Method	2D	10D	30D
PCA [62]	0.782	0.430	0.108
NCA [48]	0.568	0.088	0.073
Autoencoder [22]	0.668	0.063	**0.027**
Param. t-SNE [38]	0.099	0.046	**0.027**
OURS	**0.067**	**0.019**	**0.027**

Table 14: Trustworthiness T(12) on low-dimensional embedding of MNIST dataset.

Method	2D	10D	30D
PCA [62]	0.744	0.991	0.998
NCA [48]	0.721	0.968	0.971
Autoencoder [22]	0.729	0.996	**0.999**
Param. t-SNE [38]	**0.927**	**0.997**	**0.999**
Ours	0.768	0.936	0.975

Conclusion

- A new unsupervised learning method jointly with image clustering, cast the problem into a recurrent optimization problem;
- In the recurrent framework, clustering is conducted during forward pass, and representation learning is conducted during backward pass;
- A unified loss function in the forward pass and backward pass;
- Performance outperforms the state-of-the-art over a number of datasets;
- It can also learn plausible representations for image recognition.

Thanks!

https://github.com/jwyang/joint-unsupervised-learning