Semi Supervised Clustering

Rahul Baboota

Indraprastha University

1 Introduction to Clustering

Clustering is a classical unsupervised learning problem that seeks insights into the underlying structure of data by naturally grouping similar data instances together. The notion of similarity often depends on the distance measure defined in some feature representational space and is a crucial factor that governs the performance of a clustering algorithm. Therefore, learning a meaningful representational space as well as an appropriate distance metric from the data has shown to significantly improve clustering performance.

While clustering has traditionally been an unsupervised learning problem, many deep learning based clustering approaches have incorporated varying degrees of supervision which can substantially improve the clustering performance. Based on the level of supervision, clustering approaches can be categorized as unsupervised, supervised and semi-supervised.

The approach taken here is a semi-supervised approach in which an autoencoder based architecture is adopted that learns latent representations suitable for clustering. The training works in a semi-supervised setting utilizing the abundant unlabeled data augmented with pairwise constraints generated from a few labeled samples. In addition to the usual reconstruction error, the proposed method uses an objective function that comprises two complementary terms: a k-means style clustering term that penalizes high intra-cluster variance and a pairwise Kullback Leibler (KL) divergence based term that encourages similar pairs to have similar cluster membership probabilities.

2 Background

The proposed framework combines K-means clustering with KL Divergence loss applied to soft pairwise constraints to approach the problem of clustering in a semi-supervised setting.

In the proposed methodology, Constrained K-Means has been used instead of the traditional K-Means. The major difference that arises in the two methods is that instead of randomly initializing the cluster centers (as is done in traditional K-Means), some amount of labelled data is used to initialize the clusters. Also at each iteration of the constrained K-Means, the cluster re-assignment is restricted

only to the unlabeled samples while the membership of the labeled samples is fixed.

Another component of the proposed framework is based on the pairwise Kullback Leibler (KL) Divergence. The framework uses the embeddings from the encoder and weak labels (pairwise constraints) and combine it with a contrastive KL Divergence loss term which minimizes the statistical distance between similar pairs while maximizing it between dissimilar pairs.

3 Proposed Approach

In this section, the proposed approach is explained in detail. The different components of the proposed work can be broadly divided into three major categories:

- Network Architecture
- Loss Function
- Optimization Strategy

Let X^L be the set of labeled data points where the data points belong to K clusters and let X^U be the set of unlabeled data points. The goal is to learn a feature representation that is amenable to forming K clusters in a manner that similar pairs of points tend to belong to the same cluster while the dissimilar pairs do not.

3.1 Network Architecture

The model architecture employed in the framework is a convolutional autoencoder architecture. Let the parameters for the encoder and decoder by represented by θe and θd respectively. The functional form of the encoder representation is given by:

$$Z = f(\theta_e, x) \tag{1}$$

where *f()* is a non-linear mapping from the input space to the latent space.

Similarly, the functional form of the decoder representation is given by:

$$Z = g(\theta_d, Z) \tag{2}$$

where *g()* maps the latent space representation of the input to the reconstructed input space.

The model architecture and the latent space dimension was kept the same throughout all the performed experiments for demonstration of the performance sustainability of the proposed approach.

3.2 Loss Function

The loss function used in the proposed approach has three components:

- Cluster Loss
- Pairwise Loss
- Reconstruction Loss

The pairwise loss and the cluster loss is defined for both the labelled and unlabelled data. The complete loss function is defined as:

$$L = L^L_{pairwise} + L^L_{cluster} + \lambda(L^U_{pairwise} + L^U_{cluster}) + L_{reconstruction} \tag{3}$$

The reconstruction loss is defined for the entire data, however the pairwise loss and the cluster loss are defined in different manners for the labeled and the unlabeled data. The reconstruction loss serves as a regularizer for ensuring the representation has high fidelity, while the other two terms make the latent space more amenable to forming semantically consistent clusters. Here, λ acts as a balancing coefficient and is important for the performance of our algorithm. A high value of λ suppresses the benefits of labeled data, whereas a very small value of λ fails to use the unlabeled data effectively during training. Therefore, a deterministic annealing strategy is used that gradually increases the value of λ with time and tends to avoid poor local minima.

Reconstruction Loss

While the objective is that the latent space representations forms clusters, the cluster and pairwise objective can lead to a degenerate latent space where points tend to collapse to the cluster centers, in an attempt to minimize the cluster loss. This degeneracy may lead to poor generalization to unseen points. To mitigate this effect, the clustering loss is regularized by adding the reconstruction loss.

The reconstruction loss is defined as:

$$L_{reconstruction} = \sum_{X} ||g(f(x_i)) - x_i||^2 \tag{4}$$

Cluster Loss

The cluster loss aims to minimize the distance between the latent space representations of the data points with their assigned cluster centers to encourage clusterable representations.

The cluster loss for the labelled data is defined as:

$$L^L_{cluster} = \frac{1}{|X^L|} \sum_{X^L} ||f(\theta_e, x_i) - \boldsymbol{C}\alpha^L_i|| \tag{5}$$

Here, $\boldsymbol{C} \in \mathcal{R}^{D*K}$ where K is the total number of clusters and D is the dimensionality of the latent space. Also, α^L refers to the correct label for the particular data point.

Similarly, the cluster loss for the unlabeleed data is defined as:

$$L^U_{cluster} = \frac{1}{|X^U|} \sum_{X^U} ||f(\theta_e, x_i) - \boldsymbol{C}\alpha^U_i|| \tag{6}$$

Here, α^L refers to the predicted label for the particular data point.

Pairwise Loss

Another component of loss function is based on the Kullback Leibler (KL) Divergence, which is a measure of disparity between two probability distributions. The Kullback-Leibler divergence for ground truth distribution p and output distribution q is given by :

$$KL(p||q) = \sum_{i=1}^{K} p_i \log \frac{p_i}{q_i} \tag{7}$$

Traditionally, the KL Divergence is used to measure the distance between an output distribution and a ground truth distribution. However, in the proposed approach, the KL Divergence is used to measure the statistical distance between two output distributions given a pair of images. The approach utilizes pairwise constraints created from similar and dissimilar image pairs. The pairwise constraints exists as tuples with the form:

$$\Psi := (i, j, Relationship) \tag{8}$$

Here, i and j are the respective indices of the images of the sampled image pair and *Relationship* indicates whether the images come from a similar or a dissimilar pair. If the images x_i and x_j belong to the same class label, then they are said to form a similar image pair. If the images x_i and x_j belong to different class labels, then they are said to form a dissimilar image pair.

For labelled data, the pairwise constraints are created using the available label information. The pairwise constraints tuple for the labelled data is defined as:

$$\Psi^L_{sim} := \Big\{(i, j, Similar) : x_i \in X^L_{k1} \ ; \ x_j \in X^L_{k2} \ ; \ k1 = k2\Big\} \tag{9}$$

$$\Psi^L_{dissim} := \Big\{(i, j, Dissimilar) : x_i \in X^L_{k1} \ ; \ x_j \in X^L_{k2} \ ; \ k1 \neq k2\Big\} \tag{10}$$

Similarly, the pairwise constraints are created for the unlabelled data using the predicted label information, The pairwise constraints tuple for the unlabelled data is defined as:

$$\Psi^U_{sim} := \Big\{(i, j, Similar) : x_i \in X^U_{k1} \ ; \ x_j \in X^U_{k2} \ ; \ k1 = k2\Big\} \tag{11}$$

$$\Psi^U_{dissim} := \Big\{(i, j, Dissimilar) : x_i \in X^U_{k1} \ ; \ x_j \in X^U_{k2} \ ; \ k1 \neq k2\Big\} \tag{12}$$

where $k1, \ k2 \ \in \{1, 2, \ldots K\}$.

After generating the pairwise constraints, assignment probabilities are computed for both the labelled and the unlabelled data. The assignment probabilities computes the probability of each image belonging to a particular cluster. Instead of computing the cluster probabilities by applying a softmax layer on the embeddings, the probabilities in this approach are defined on the distances from the cluster centers.

The distance of an image x_i from the k^{th} cluster center is defined as:

$$d_{i,k} = ||f(\theta_e, x_i) - \mu_k||^2 \tag{13}$$

where μ_k is the cluster center for the k^{th} cluster.

Based on this distance metric, the cluster assignment probabilities are computed as:

$$p_{i,k} = \frac{exp(-d_{i,k})}{\sum_K exp(-d_{i,k})} \tag{14}$$

where $p_{i,k}$ is the probability of assigning x_i to the k^{th} cluster.

The pairwise loss function is a contrastive KL Divergence loss. For an image pair *(x_p, x_q)*, the contrastive loss term is defined as:

- If x_p and x_q come from a similar pair, then the loss function is a simple Kullback Leibler Divergence.
- If x_p and x_q come from a dissimilar pair, then the loss function is hinge loss using Kullback Leibler Divergence.

The contrastive KL Divergence loss is defined as:

$$Loss(p||q) = \boldsymbol{I_s} KL(p||q) + \boldsymbol{I_{ds}} max(0, margin - KL(p||q)) \tag{15}$$

where $\boldsymbol{I_s}$ is equal to one when x_p and x_q come from a similar pair and $\boldsymbol{I_{ds}}$ is equal to one when x_p and x_q come from a dissimilar pair. Here p and q are the cluster assignment probabilities defined in equation (14).

The pairwise loss for the labelled data is given by:

$$L^L_{pairwise} = \frac{1}{|\Psi^L|} \sum_{p,q \in \Psi^L} \Big\{ Loss(p||q) + Loss(q||p) \Big\} \tag{16}$$

where $\Psi^L = (\Psi^L_{sim} \cup \Psi^L_{dissim})$

Similarly, the pairwise loss for the unlabelled data is given by:

$$L^U_{pairwise} = \frac{1}{|\Psi^U|} \sum_{p,q \in \Psi^U} \Big\{ Loss(p||q) + Loss(q||p) \Big\} \tag{17}$$

where $\Psi^U = (\Psi^U_{sim} \cup \Psi^U_{dissim})$

3.3 Optimization Strategy

The proposed approach alternatively optimizes the network parameters θ_e and θ_d as well as the cluster centers μ_k for the different clusters. For optimizing the network parameters, the standard backpropagation algorithm is used for finding the optimal set of weights for the auto-encoder.

Cluster Center Initialization

The cluster centers are initialized in accordance to the constrained K-Means algorithm. The cluster centers are explicitly initialized by taking the mean of the labeled samples. The initialization of the cluster centers is done as:

$$\mu_k = \frac{1}{|X^L|} \sum_{X^L} f(\theta_e, x_i) \tag{18}$$

where $k \in \{1, 2, \dots K\}$. and μ_k is the cluster center for the k^{th} cluster.

Cluster Assignment

The cluster assignment scheme is different for labelled and unlabelled data. For the labelled data, the image is assigned to it's true cluster label. For the unlabelled data, the cluster assignment scheme is the same as used in Traditional K-Means clustering where the data point is labelled to that cluster where the distance from the cluster center is minimum.

The cluster assignment for labelled data is defined as:

$$a^L_{i,j} = \begin{cases} 1 & \text{for } j = k \ ; x^L_i \in X_k \\ 0 & otherwise \end{cases}$$

where the image x_i is assigned to the cluster k where it's true cluster assignment was k itself.

The cluster assignment for unlabelled data is defined as:

$$a_{i,j}^U = \begin{cases} 1 & \text{for } j = \text{argmin}_k \, ||f(\theta_e, x_i^U) - \mu_k|| \; where \; k = (1, 2 \cdots K) \\ 0 & otherwise \end{cases}$$

where the image x_i is assigned to the cluster k where the distance of the cluster center μ_k is the minimum from the latent space embedding of x_i.

Cluster Center Updates

The cluster update scheme is not the same as what is done in traditional K-Means. For the cluster center updates, gradient updating with an adaptive learning rate is used and both the labeled as well as the unlabeled data is used to make the required update.

The labelled data is used to update the cluster centers as:

$$\mu_k = \mu_k - \frac{1}{N_k^L}(\mu_k - f(\theta_e, x_i^L)) \quad where \; x_i^L \in X_k \tag{19}$$

Here N_k^L is the number of labelled samples that have been assigned to cluster k.

Similarly, the unlabelled data is used to update the cluster centers as:

$$\mu_k = \mu_k - \frac{1}{N_k^L}(\mu_k - f(\theta_e, x_i^U))a_i^U \tag{20}$$

Here N_k^U is the number of unlabelled samples that have been assigned to cluster k and a_i^U is the cluster assignment for the unlabelled data point x_i.

4 Experiments

The proposed approach is evaluated on five benchmark datasets with two different scoring metrics. The experimental section discusses the following:

- Datasets Description
- Network Details
- Efficient implementation to utilize pairwise constraints
- Experimental Results
- Analysis

4.1 Datasets Description

The proposed approach was evaluated on five different datasets. Out of the five datasets used, two of them are handwritten datasets: *MNIST* and *USPS*, two of them are facial recognition datasets: *FRGC* and *YTF* and a generic dataset: *CIFAR10*. For the *YTF* dataset, the first 41 subjects sorted according to theri names in alphabetical order. For the *FRGC* dataset, 20 randomly selected subjects for the experiments. The details of the dataset are given below.

	MNIST	USPS	FRGC	YTF
Number of Samples	70,000	11,000	2462	10,000
Number of Classes	10	10	20	41
Image Dimensions	32 ×32	16 ×16	32 ×32 × 3	55 ×55 × 3

4.2 Network Details and Training

The network architecture used in the proposed approach is a convolutional autoencoder. The same model architecture is used for all the datasets to depict it's generalization. The encoder consists of three convolutional layers with 32, 64 and 128 filters respectively followed by a fully connected layer of size 32. Each convolutional layer is followed by an Instance Normalization layer and a Leaky ReLU activation. The fully connected layers in both encoder and decoder use a tanh activation function. The stride and padding values are chosen appropriately for the dataset.

Before training the model with the proposed approach, the autoencoder is first pretrained end-to-end for each dataset for minimizing the reconstruction error. The pretraining is done for 100 epochs. This pretrained network is then fine tuned by training it with the proposed approach for 60 epochs. The autoencoder is optimized by training the network with Adam optimizer with a learning rate of 0.0001 and $\beta = (0.9, 0.999)$. The value of the balancing coefficient λ is determined based on the number of epochs into the training of the network.

$$\lambda = \begin{cases} 0 & \text{for } t < T_1 \\ \frac{t-T_1}{T_2-T_1} & \text{for } T_1 \leq t \leq T_2 \\ 1 & \text{for } T_2 < t \end{cases}$$

Here, t denotes the current epoch. The values of T_1 and T_2 were found to be 4 and 40 respectively for optimal performance.

4.3 Experimental Results

The proposed approach was evaluated on two metrics: Normalized Mutual Information (NMI) and Purity. Normalized Mutual Information computes the normalized similarity measure between the true and predicted labels for the data

and is defined as:

$$NMI(y^T, y^P) = \frac{I(y^T, y^P)}{max(H(y^T), H(y^P))} \tag{21}$$

where y^T and y^P represent the true and predicted class labels. $I(y^T, y^P)$ represents the mutual information between y^T and y^P and $H(.)$ represents the cross-entropy.

Purity is an external evaluation criterion of cluster quality. It is defined as:

$$Purity = \frac{1}{N} \sum_{i=1}^{K} max_j |c_i \cap t_j| \tag{22}$$

where N is the total number of data points and k being the total number of clusters. c_i is a cluster in K and t_j is the class label assigned which has the maximum count for the cluster c_i.

The NMI and Purity scores with varying levels of labelled data on the different datasets is depicted in Tables 1-4.

Table 1: NMI and Purity Results for MNIST

Labelled Data Percentage	MNIST	
	NMI	Purity
1	0.939	0.9813
2	0.958	0.9838
5	0.963	0.9865
10	0.970	0.9896

4.4 Efficient implementation to utilize pairwise constraints

The contrastive loss function proposed is suitable to be trained with Siamese Networks. However, when the amount of pairwise constraints increases, it is not efficient to enumerate all pairs of data instances and feed-forward them into the Siamese Networks. As the number of pairwise constraints increases, the probability of data instances being feed-forwarded multiple times in the network also increases which leads to computational redundancy. To avoid this redundancy

Table 2: NMI and Purity Results for USPS

Labelled Data Percentage	USPS	
	NMI	Purity
1	0.933	0.9633
2	0.959	0.9802
5	0.965	0.9843
10	0.971	0.9864

Table 3: NMI and Purity Results for FRGC

Labelled Data Percentage	FRGC	
	NMI	Purity
2	0.884	0.8332
5	0.951	0.9457
10	0.971	0.9701

Table 4: NMI and Purity Results for YTF

Labelled Data Percentage	YTF	
	NMI	Purity
2	0.938	0.9054
5	0.989	0.9854
10	0.997	0.9970

of computation, a strategy of enumerating the pairwise relationships only in the cost layer instead of instantiating the Siamese Architecture is adopted. The pairwise constraints only need to be presented to the cost layer in the format of tuples *T : (i, j, relationship)* where i and j are the indices of the data points in the mini-batch *relationship* indicates whether the data points came from a similar or a dissimilar pair. Thus with this strategy, each data instance needs to be feed-forwarded only a single time and the pairwise relationships are computed seperately and used only at the cost layer.

The pairwise constraints are uniformly sampled from the total number of constraints which is *n(n -1)/2* where n is the size of the dataset. The pairwise relationship is converted from the class labels. If a pair has the same class label, then it is a similar pair other it is a dissimilar pair. It is quite observable that the ration between the similar and dissimilar pairs will be roughly *1:(C-1)* where C is the total number of classes. Also it is to be noted that when we use partial pairwise constraints, we sample equal number of data points from each class to insure uniformity.

4.5 Analysis

The autoencoder network was initially pretrained for 100 epochs after which it was fine tuned for another 60 epochs. Figure 1 shows that pretraining the network not only helps in the faster convergence but it helps the network to converge better as well as it reaches a lower loss.

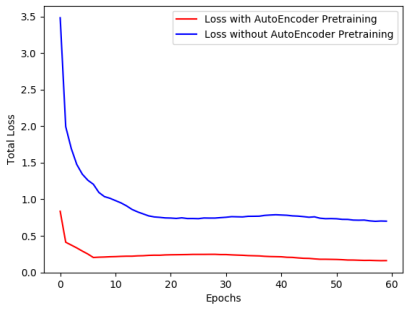

Fig. 1: Comparison of Loss with and without AutoEncoder Pretraining

The network converges smoothly which is depicted by the loss plot shown in Figure 2. Figure 3 shows the loss curves for the different components of the custom loss function.

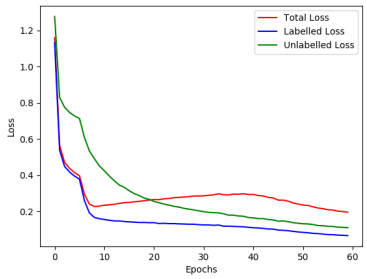

Fig. 2: A figure

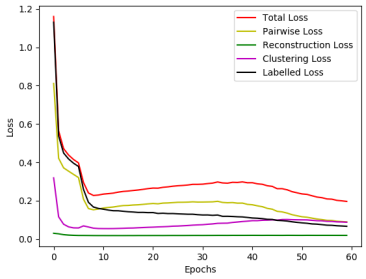

Fig. 3: Another figure

The effect of balancing coefficient on the NMI and Purity scores during training of the network are depicted in Figure 4. Since the amount of labeled data used is typically much smaller than the unlabeled data, it is important to balance the loss contributed by unlabeled data for achieving good representations and clustering performance. Too small a value of λ underutilizes the unlabeled data, while too large a value may lead the training loss to be too high. The performance is better when an annealing strategy is used to adapt λ throughout the training process. Figure 5 depicts the NMI and Purity score curves during training for unlabelled and labelled data.

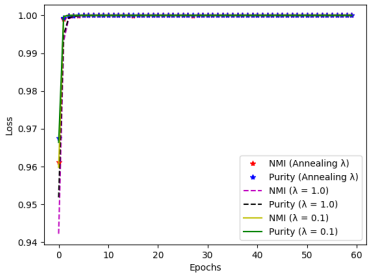

Fig. 4: NMI and Purity curve

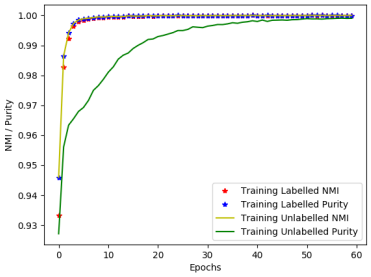

Fig. 5: NMI and Purity curve