Non-Linear Sub-Space Clustering using Mixture of RBMs

Abstract

We propose a completely new framework for non-linear subspace clustering using a mixture of RBMs. Our framework makes no assumptions about the nature of subspaces sought and is based on minimizing the error of reconstruction of data points from projections in the learnt manifolds. Mixture of RBMs learns clusters and projections in the non-linear subspaces simultaneously and hence is a coupled approach to understanding the nature of data. We experimentally compare our clustering results with state of the art subspace clustering methods to show that our framework is both faster and more accurate at learning data distributions. Finally, we devise a general strategy to understand complex data by learning non-linear manifolds in the data followed by clustering these manifolds in a linear fashion.

1. Introduction

Finding structure in data is a challenging and necessary pre-processing step in many machine learning applications. Insight into the nature of the data can help choose or create better features, learn better similarity measures between data points, build better predictive and descriptive models, and ultimately drive better decisions from data. The two most pervasive forms of structures sought in data are *Clusters* or groups of "similar" data points, and *Projections* of high dimensional data into smaller[1] dimensional spaces. Clustering is driven by the hypothesis that data is not randomly distributed across the feature space but has inherent high density regions with few outliers and/or background noise points. The projection methods are driven by the hypothesis that all features in the data are not completely independent of each other and they have some correlations among them, and the real structure in the data could be in one or more linear or non-linear manifold(s) of the raw feature space. Thus "structure" and "dependence" along both data points and feature dimensions is the basis of these paradigms.

There is a vast body of literature in both these areas especially for multi-variate real-valued data. Clustering algorithms range from partitional (e.g. k-means) to hierarchical (e.g. agglomerative, divisive) to graph theoretic (e.g. spectral clustering). They use various types of cost functions, similarity measures, cluster shape and heterogeneity assumptions, have different scaling properties, and have been adaptated to various data types (e.g. multi-variate real valued, bag-of-words text documents, bag-of-objects image representations, etc.). Similarly, projection methods range from simple linear projections such as PCA to more sophisticated non-linear sub-space projections such as Kernel PCA, locally linear embedding (Roweis & Saul, 2000) etc., again using various cost functions and making different underlying assumptions about the nature of the sub-spaces sought.

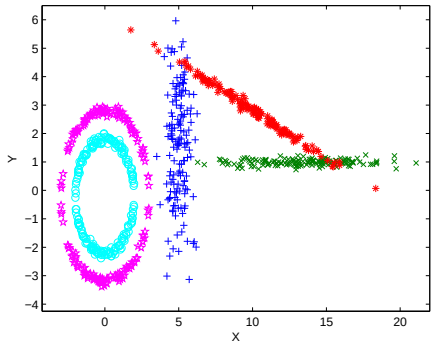

Figure 1. Hypothesis 1: Clustering and projection are two coupled paradigms. Clustering cannot be done in the raw feature space because the data lies in latent manifolds. The right manifolds cannot be discovered without clustering the data.

The first hypothesis of this paper is that *clustering and projection are two "coupled" paradigms for understanding the nature of data*. Clustering cannot be done in the raw feature space because the data really lies in certain other manifolds. Similarly the right mani-

[1]Projections for SVM are typically to a larger space but the goal there is not to find structure in data but to maximize separability among classes.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

folds cannot be discovered without proper groupings of the data. Figure 1 illustrates this point. While most of the work in clustering and projection methods is done independently, attempts have been made to combine the two (Baghshah & Shouraki, 2009; Liu et al., 2010). In this paper, we take this "coupling" a step forward via a framework that learns clusters and projections simultaneously. This is fundamentally different from an approach like Sparse Subspace Clustering (SSC) (Elhamifar & Vidal, 2009) that first learns a sparse representation(SR) of the data and then applies spectral clustering to a similarity matrix built from this SR.

The second hypothesis of this paper is that in general *the data is embedded in multiple non-linear subspaces and and within each manifold there may be further clusters.* Thus the overall solution should first find multiple non-linear sub-spaces within the data and then further cluster the data within each sub-space if necessary. In this paper, we propose a non-linear sub-space clustering framework using a mixture of Restricted Boltzmann Machines (RBMs) (Smolensky, 1987). This framework is motiviated by the two hypotheses above that clustering and subspace projection are a coupled problem and that in general the data can lie in multiple (mixture of) non-linear subspaces.

RBMs are a class of undirected, energy-based graphical models that learn a non-linear subspace by minimizing reconstruction error. RBMs have gained popularity in recent years because of their wide variety of applications. They have been used successfully to learn features for image understanding and classification (Hinton et al., 2006), speech representation (Mohamed et al., 2011), analyze user rating of movies (Salakhutdinov et al.) , and better bag-of-word representation of text data (Salakhutdinov & Hinton). Moreover, RBMs have been stacked together to learn hierarchical representations such as deep belief networks (Hinton et al., 2006; Bengio et al., 2007) and convolutional deep belief networks (Lee et al.) for finding semantically deeper freatures in complex domains such as images.

Most nonlinear subspace learning algorithms (Kanatani; Rao et al., 2008) make various assumptions about the nature of the underlying manifold they are trying to discover and use a variety of objective functions. RBMs, on the other hand, are a generic framework for learning non-linear subspaces, make no assumptions about the sub-spaces, use a standard energy based learning algorithm, and can model subspaces of any degree of complexity via the number of hidden units making them a most suitable choice as general purpose sub-space learning machines.

Our model learns K RBMs simultaneously. Each RBM represents a different subspace manifold in the data. The association of a data point to an RBM depends on the reconstruction error of each RBM for that data point and each RBM updates its weights based on all the data points associated with it. Through various learning tasks on both synthetic and real data, we show the convergence properties, quality of the subspaces learnt, and improvement in the accuracies of both descriptive and predictive tasks based on this framework.

2. Training RBMs

RBMs are two layered, fully connected networks that have a layer of input/visible variables and a layer of hidden random variables. RBMs model a distribution over visible variables by introducing a set of stochastic features. In applications where RBMs are used for image analysis, the visible units correspond to the pixel values and the hidden units correspond to visual features.

There are three kinds of design choices in building an RBM: the objective function used, the frequency of parameter updates, and the type of visible and hidden units. RBMs are usually trained by minimizing the contrastive divergence (Hinton, 2000) objective (CD-1). For an RBM with I visible units $v_i, i = 1, \ldots, I$ ($v_0 = 1$ is the bias terms), J hidden units $h_j, j = 1, \ldots, J$ ($h_0 = 1$ is the bias term) and symmetric weighted connections between the visible and hidden layers denoted by $\mathbf{w} \in \mathbb{R}^{(I+1)\times(J+1)}$ (these include asymmetric forward and backward bias terms), the activation probabilities of units in one layer are computed based on the states of the opposite layer using the following equations:

$$Pr(h_j|\mathbf{v}) = \sigma\left(\sum_{i=0}^{I} \mathbf{w}_{ij} v_i\right) \tag{1}$$

$$Pr(v_i|\mathbf{h}) = \sigma\left(\sum_{j=0}^{J} \mathbf{w}_{ij} h_j\right) \tag{2}$$

$\sigma(\cdot)$ is the sigmoid activation function. In the CD-1 forward pass (visible to hidden), we activate the hidden units ${h_j}^+$ from visible (input) unit activations v_i^+ (Eq.1). In the backward pass (hidden to visible), we recompute visible unit activations v_i^- from h_j^+ (Eq.2). Finally we compute the hidden unit activations h_j^- again from v_i^-. The weights are updated using the following rule:

$$\Delta w_{ij} = \eta(< v_i^+ h_j^+ > - < v_i^- h_j^- >) \tag{3}$$

where η is the learning rate and $<\cdot>$ is defined as the mean over N examples. The reconstruction error for any sample is computed as:

$$\epsilon = \sum_{i=1}^{I} (v_i^+ - v_i^-)^2 \tag{4}$$

Traditionally, the RBM weights are updated once per mini-batch. Other options are once per sample update (fully online) and corpus level update (fully batch). We found that doing a full batch update gives a more reliable gradient and slightly better reconstruction compared to mini batch or online updates.

An RBM can have binary or non-binary visible and hidden units. Most RBM implementations use binary visible units. In our applications, we have used Gaussian visible units to model distributions of real valued data. The stochastic output of hidden unit (Eq.1) is always a probability which is thresholded against a random value between 0 and 1 to give a binary activation h_j. In CD-1, it is customary to use binary hidden states when the hidden units are driven by data ($h_j{}^+$) and the probabilities without sampling when the hidden units are driven by reconstructions ($h_j{}^-$). Thresholding introduces sparsity by creating an information bottleneck. We however always use the activation probabilities in place of their binary states for parameter updates. This process, known as Rao-Blackwellization (Blackwell, 1947), gives an estimator with lower variance and better clustering performance (we show this empirically in section 4.3, Table 1). This decision was based on the desire to eliminate unnecessary randomness from our approach[2] and was supported by extensive experimentation.

3. Non-Linear Subspace Clustering using Mixture of RBMs

Our non-linear subspace clustering model uses a mixture of RBMs; each component RBM learns a non-linear subspace. The visible units $v_i, i = 1, .., I$ correspond to an I dimensionsional visible (input) space and the hidden units $h_j, j = 1, .., J$ correspond to a learnt non-linear J-dimensional subspace. For the sake of simplicity, we experiment with RBMs of the same size; all the subspaces our model learns have the same true dimensionality J. However, this restriction is unnecessary and we are free to learn subspaces with different true dimensions.

[2]We use the reconstruction error as a cost function in our clustering; random thresholding introduces randomness in the projections, hence affecting the reconstruction errors.

3.1. Mixture of RBMs

Our mixture of RBMs model has K component RBMs. Each of these maps a set of N sample points $\mathbf{x_n} \in \mathbb{R}^I$ to a projection in $\mathbb{R}^J$. Each component RBM has a set of symmetric weights (and asymmetric biases) $\mathbf{w}^k \in \mathbb{R}^{(I+1)\times(J+1)}$ that learns a non-linear subspace. Note that these weights include the forward and backward *bias* terms. The error of reconstuction for a sample $\mathbf{x}_n$ given by the kth RBM is simply the squared Euclidean distance between the data point $\mathbf{x}_n$ and its reconstruction by the kth RBM, computed using (Eq.4). We denote this error by ϵ_{kn}. The total reconstruction error ϵ_t in any iteration t is given by $\sum_{n=1}^{N} \min_k \{\epsilon_{kn}\}$

Training the mixture model involves training K RBMs simultaneously. During the RBM training, we compute a partition of the data on the basis of the reconstruction errors that each RBM in the mixture gives us for each data point. Each component RBM is trained only on the training sample points associated with it; this association of a data point to a component is a function of how well the component RBM is able to reconstruct the data point. The component RBMs are given random initial weights $\mathbf{w}^k, k = 1, .., K$.

3.2. Hard clustering

Like in traditional K-means clustering, the algorithm alternates between two steps: (1) Computing association of a data point with a cluster and (2) updating the cluster parameters. In mixture of RBMs n^{th} data point is associated with k^{th} RBM (cluster) if its reconstruction error from that RBM is lowest compared to all other RBMs. In other words if this RBM does the best reconstruction of the data point compared to all other RBMs, i.e. if $e_{kn} < e_{k'n} \forall k \neq k', k, k' \in \{1, .., K\}$.

Once all the points are associated with one of the RBMs the weights of the RBMs are learnt in a batch update. In hard clustering the data points are partitioned into the clusters exhaustively (i.e. each data point must be associated with some cluster) and disjointly (i.e. each data point is associated with only one cluster). In contrast with K-means where the update of the cluster center is a closed form solution given the data association with clusters, in mixture of RBMs the weights are learnt iteratively.

3.3. Soft clustering

We extend our model to incorporate soft clustering where instead of assigning a data point to only one

RBM cluster, it can be assigned softly to multiple RBM clusters. The soft association of the n^{th} data point with the k^{th} cluster is computed in terms of the reconstruction error of this data point with the RBM:

$$\alpha_{nk} = \frac{\exp(-\epsilon_{kn}/T)}{\sum_{k'=1}^{K} exp(-\epsilon_{k'n}/T)}$$

where T is the temperature parameter that is reduced over time as in simulated annealing (Kirkpatrick et al., 1983). Each sample $\mathbf{x}_n$ contributes to the training of all RBMs in proportion to its association with the RBMs. In the learning rule (Eq. 3), the association factor is also multiplied with the learning rate.

A mixture of RBMs trained using the soft approach can be seen as a set of RBMs, each of which learns a distribution of all the data but using more information from those it can represent most accurately. In other words, each RBM can reconstruct all the points, some more accurately than the others. This is fundamentally different from the hard clustering where each component RBM learns the distribution of a subset of the data and tries to distort samples from other clusters to look like the samples that it has learnt from. This artefact can be seen in Figure 4 where data points associated with one RBM cluster are distorted when reconstructed by other RBMs (cluters).

3.4. Discussion

As mentioned earlier, our clustering framework seeks to learn both the associations (clusters) and the parameters (non-linear subspaces) simultaneously. Thus, here we come across two kinds of convergences: the clustering convergence and the RBM learning (subspace learning) convergence. In our experiments the clustering process is said to have converged when more than 99% of the samples stop changing cluster associations. Usually we require only the cluster associations. We can stop the algorithm once the clustering converges. However, the convergence of clustering just means that the points in each cluster belong to the same non-linear manifold, it does not guarantee the accuracy of the learnt manifolds. In case we require the data projections in these non-linear subspaces too which are useful, for instance, if we intend to further partition the learnt non-linear subspaces (section 4.3), we continue to train the RBMs until the total reconstruction error stabilizes. Figure 2 shows a plot of the two kinds of convergences as training progresses. While the clustering converged within 20 iterations (the data points stopped changing their cluster associations), the reconstruction error contined to drop beyond 100 iterations. We empirically decide the number of epochs our algorithm iterates for and we call this number *maxepoch*.

Mixture of RBMs learns the non-linear subspaces and projections simultaneously. This is therefore, as we mentioned before, an approach that preserves the coupling between the clustering and projection paradigms which is crucial for gaining insights into the true nature of data. The first part of our second hypothesis *(data is embedded in multiple non-linear subspaces)* is supported by the fact that a mixture of RBM learns the data distribution better than a single RBM. Using mean reconstruction error per sample as a measure of the degree of learning, we show that a mixture of RBM learns the data distribution better than a Single RBM. Figure 3 illustrates our point: The single RBM reconstruction error is an order of magnitute higher than mixture of RBMs. The number of RBMs as well as the number of hidden units in each RBM together define the overall complexity of the model; in Figure 3 we show how the number of RBMs affect the reconstruction error for a fixed number of hidden units.

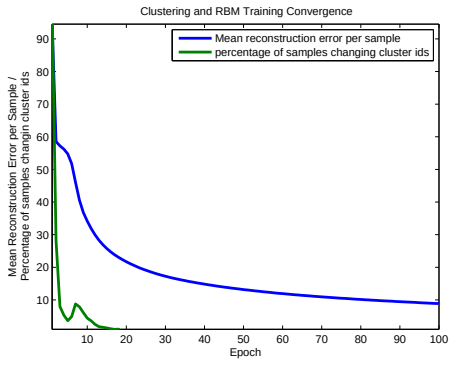

Figure 2. Clustering convergence and rbm training convergence over epochs of the algorithm while clustering the mnist dataset. Clustering converges long before the rbm reconstruction errors stabilize.

4. Applications

In this section we describe applications of mixture of RBMs to clustering on MNIST data, a set of synthetic datasets, and image classification on Pascal VOC dataset.

4.1. Clustering MNIST Data

The goal of the first set of experiments is to investigate the first hypothesis i.e. *clustering* and *projection* are better done in a coupled manner than in a sequential manner. For these experiments we use MNIST(LeCun et al., 1998) data as it is commonly used in literature, is reasonably high dimensional, and has lots of labeled examples. The MNIST data has

Algorithm 1 Hard clustering using mixture of RBMs

Input: data $\mathbf{x_n}, n = 1,..,N$, size of subspaces J and number of mixture Components K
Initialize: $\mathbf{w^k}, k = 1,..,K$ randomly, *maxepoch* empirically.
$epoch \leftarrow 1$
repeat
  for $n = 1$ **to** N **do**
    for $k = 1$ **to** K **do**
      Compute ϵ_{nk} using (Equation 4)
    end for
    Compute Cluster Index $k_n = \underset{k=1,..,K}{\text{argmin}} \epsilon_{nk}$
    Use $\mathbf{x_n}$ to train k_n^{th} RBM using (Eq. 3)
  end for
  $epoch \leftarrow epoch + 1$
until $epoch = maxepoch$

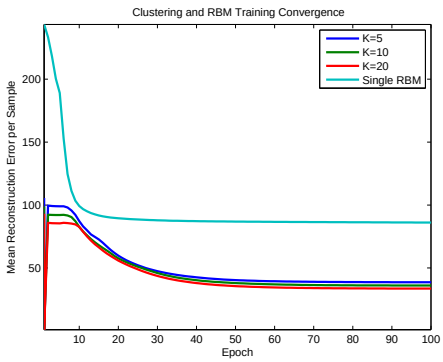

Figure 3. A plot of reconstruction errors vs epochs of training process for our experiments on the VOC Pascal dataset in section 4.3. These clearly show that reconstructions are significantly better when we use a mixture of RBMs as opposed to a single RBM. Note that for plotting purposes, for the Single RBM case, we divide the mean error by 10 to bring it to scale with the others.

$70,000$ data points ($60,000$ training and $10,000$ test) of binary handwritten digits from 0 to 9. Each data point is size-normalized and centered in a fixed-size (28×28) image represented as a $784-$dimensional binary vector. To test our hypothesis we compare three methods: (*i*) *PCA + K-means* where the $784-$dimensional raw data is first linearly projected using PCA into a $100-$dimensional subspace and this projected data is then clustered into 10 clusters using standard K-means clustering. (*ii*) *Single* RBM *+ K-means* where the same $784-$dimensional raw data is first *non-linearly* projected using a single RBM into a *single* $100-$dimensional subspace (just to keep the complexity compatible) and this projected data is then clustered into 10 clusters using K-means. (*iii*) *mixture of* RBMs (also referred to as K-RBM hereafter) where the $784-$dimensional data is clustered using 10 RBMs, each with 100 hidden units. The first two (PCA + K-means and Single RBM + K-means) represent the "decoupled" learning of a projection *followed by* a clustering. The third (mixture of RBMs) represents the *coupled* learning of both a generic non-linear subspace projection *along with* clustering. The mixture of RBMs are learnt in two modes: with *binary* hidden units (K-RBM-b) and with *real valued* hidden units (K-RBM-r). In both cases the visible units (input data) is binary. For a K-RBM-r, instead of using a random threshold on the activations of the hidden units, we simply use their real stochastic activations[3].

Table 1 shows the results comparing the four methods on three (related) metrics: Mutual Information (MI) [4], percentage misclassification error, and weighted mean cluster purity. Both the K-RBM-b and K-RBM-r representing the *coupled* learning of projections and clustering (statistically) significantly outperform the two *decoupled* methods (PCA + K-means and RBM + K-means). Also note that K-RBM-r significantly outperforms K-RBM-b indicating that the loss of information due to binarization of hidden units in pursuit of sparsity may not be the right thing to do in all applications. Figure 4 shows examples of data points in each class (digit) and their cluster centers (obtained by averaging the images of all points associated with cluster), examples of reconstruction of digits from various classes by the single RBM and examples of reconstruction of a digit by all the K-RBM-r here.

Discussion: Figure 4 shows that using linear clustering (K-means) for data that lies in non-linear subspaces is a bad idea. In K-means, the cluster centers may be strikingly different from some of the samples. A single RBM first learns a non-linear subspace that the data lies in and then using K-means on the projections in these subspaces makes sense. However, this is not a very good approach because the data lies in multiple non-linear subspaces. Mixture of RBMs clusters the data, learning the subspace of each cluster. These subspaces are specific to each cluster and when samples from other clusters are reconstructed using these

[3] As mentioned in section 3, this process is called Rao-Blackwellization

[4] Mutual Information between two random variables X (e.g. cluster labels from a clustering method) and Y (e.g. actual class labels) is defined as: $MI(X,Y) = \sum_{y \in Y} \sum_{x \in X} P(x,y) log(\frac{P(x,y)}{P(x)P(y)})$ where $P(x,y)$ is the joint probability of X and Y and $P(x)$ and $P(y)$ are the marginal probability distributions of X and Y respectively.

subspaces, they are distorted to look like the ones in this cluster.

Table 1. Comparision of coupled vs. de-coupled projection + clustering learning algorithms on MNIST data.

METHOD	PURITY	ERROR	M.I.
K-MEANS	59.43%	45.23%	1.6651
PCA + K-MEANS	59.36%	45.24%	1.6627
RBM + K-MEANS	59.63%	44.94%	1.6828
K-RBM-B	63.83%	42.79%	1.9127
K-RBM-R	65.16%	38.90%	2.0878

4.2. Experiments on Synthetic Datasets

The goal of this second set of experiments is again to further investigate the first hypothesis with sythetic data instead of real data (e.g. MNIST) so that we can control the nature of the sub-spaces in the data. Additionally, we will compare mixture of RBMs with two state of the art algorithms for non-linear subpace clustering, Random Sample Consensus (RANSAC)(Fischler & Bolles, 1981) and Sparse Subspace Clustering (SSC)(Elhamifar & Vidal, 2009) in addition to PCA + K-means and RBM + K-means. RANSAC works by iteratively sampling a number of points randomly from the data, fitting a model to those points and rejecting outliers. SSC computes a sparse representation (SR) of the data and applies spectral clustering to a matrix obtained from the SR. Both these algorithms also represent *decoupled* learning of projection and clustering. As shown in Table 2 the running time of both RANSAC and SSC algorithms is exponentially higher than training a mixture of RBMs. In these experiments we used 500 data points in the three synthetic datasets described below. Due to the time complexity of RANSAC and SSC it is impractical to train these models on a MNIST like dataset with $70,000$ samples without serious down sampling. Hence we did not compare RANSAC and SSC with mixture of RBMs on the MNIST dataset in the previous experiments.

For these experiments, we used three synthetic datasets. The first two datasets, named $D1$ and $D2$ hereafter comprise of 500 points drawn from 5 subspaces constructed using orthogonal basis functions, 100 points from each subspace. For all the points, the dimension of the raw feature space is 144 while the true intrinsic dimensionality is 36. The dataset $D2$ also contains added Gaussian noise. The third dataset, named $D3$, hereafter also consists of 500 points drawn from 5 subspaces. However, these subspaces differ from the ones in $D1$ and $D2$ in the sense that they don't have orthonormal basis vectors, but oblique projections making it harder than $D1$ and $D2$.

We cluster these datasets into 5 clusters using the five aforementioned methods. The results are reported in Table 2 in terms of misclassification error and mutual information of cluster labels and the known class label. We also quote the running time of these algorithms. Note that SSC reports three misclassification errors for each problem because it uses three different spectral clustering methods. In these tables, we have reported the smallest of these three values. For these experiments, we chose 36 principal components for (PCA + K-means) and both (Single RBM + K-means) as well as mixture of RBMs. The input to the RBM is 144-dimensional Gaussian visible units and 36 binary hidden units. Note that not only mixture of RBMs outperforms these two state of the art methods in terms of quality metrics, it also learns the coupled projection and clustering orders of magnitude faster than these methods both of which are quadratic in the number of data points and hence not very practical to use without serious sampling.

4.3. Non-Linear Subspace Clustering for Visual Bag-of-Words

The goal of these experiments is two-fold: First, to investigate the second hypothesis that in general multivariate real-valued data typically lies in multiple non-linear subspace manifolds (e.g. as learnt by K-RBMS) and there are further potential clusters within each of the sub-spaces. This hypothesis points to a two stage clustering of data: first clustering "coupled" with non-linear projection (e.g. K-RBM) followed by further sub-clustering within each first level cluster. The second goal of these experiments is to propose an alternative to the traditional bag-of-words representations used ubiquitously in computer vision applications (e.g. image classification). We first describe the traditional image representation and our approach, and then highlight the role of the second hypothesis in this application.

In our experiments, we use the PASCAL VOC 2007 data with a total of 9963 images in 20 classes. From these, we first pick the following eight classes:*aeroplane, bicycle, bus, tvmonitor, pottedplant, bird, bottle, chair*. These classes were chosen to be classes with large number of images and also capture the diversity in the data. From each class, 35 random images were used for training and another 35 images were used for testing. With the scale of 12 and a shift of 6, we get about 2500 SIFT vectors per im-

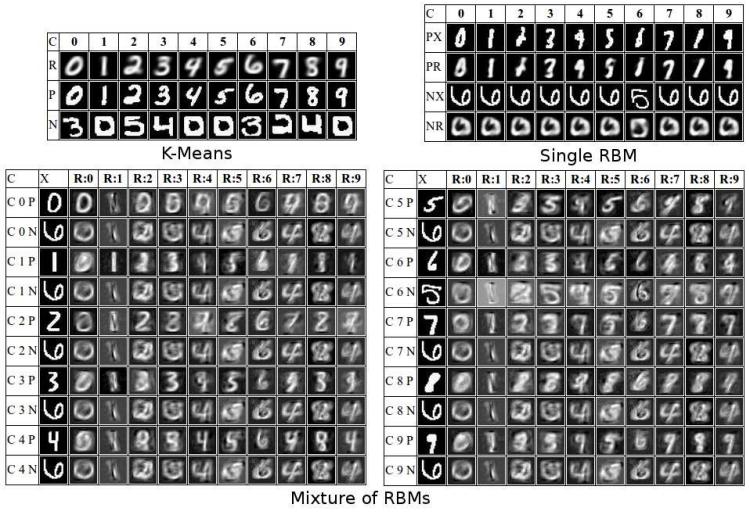

Figure 4. Clustering results of K-means, Single RBM + K-means, mixture of RBMs. C denotes the cluster labels, R is the reconstruction in case of RBMs (R:0 is the reconstruction from the RBM corresponding to cluster 0), mean in case of K-means. P is a positive example (correctly classified) from the cluster, N is a negative example (incorrectly classified) from the cluster. X is the data sample.

Table 2. Running Time, Misclassification Errors and Mutual Information between cluster and class labels of various methods on synthetic $D1$, $D2$ and $D3$ datasets.

Method	Dataset D1			Dataset D2			Dataset D3		
	Runtime	Error	M.I.	Runtime	Error	M.I.	Runtime	Error	M.I.
K-means	$0.73s$	28%	1.9219	$0.68s$	27.4%	1.9219	$2.76s$	29.6%	1.9219
PCA + K-means	$0.41s$	0%	2.3219	$0.37s$	27.4%	1.9219	$0.42s$	29.8%	1.9219
RBM + K-means	$3.07s$	0.6%	2.2708	$3.43s$	2.2%	2.1605	$3.51s$	4.4%	2.0946
RANSAC	$134.76s$	54.6%	0.1994	$134.80s$	66.6%	0.1529	$474.72s$	69.6%	0.1499
SSC	$418.21s$	0%	2.3219	$365.29s$	0%	2.3219	$690.93s$	15.6%	2.0732
Mixture of RBMs	$0.45s$	0%	2.3219	$0.46s$	0%	2.3219	$3.62s$	0%	2.3219

age. All SIFT vectors in the 35 *training images* in each of the eight classes gives a SIFT data of 700,000 SIFT points ($8\times35\times2500$) each being 128-dimensional. This dataset is then clustered using the standard K-means clustering into 1000 clusters to give the baseline codebook. Eight 1-vs-rest classifiers, one for each of the eight classes, were learnt using pegasos SVM and their Average Precisions were averaged over the test set.

Inspired by the second hypothesis, we created the 1000 clusters in a different way. We first trained a mixture-of-RBMs with K_1 mixtures as described in Section 3.2 over the $700,000$ SIFT points. The RBMs use $128-$dimensional Gaussan visible units. These are reduced to $32-$dimensional real valued hidden units. Thus the model here is that the 700,000 points in the original 128-dimensional SIFT space reside in K_1 non-linear 32-dimensional subspaces. Once trained, the mixture of RBMs partitions the 700,000 data points into K_1 exhaustive and non-overlapping (we used hard clustering) subsets. We further clustered each of the K_1 subsets *in the trasnformed 32-dimensional space* into K_2 clusters using simple K-means clustering. This is in-line with our hypothesis that within each subspace there might be multiple clusters. To keep the total number of clusters compatible with the baseline $K = 1000$, we chose the following combinations of K_1 and K_2: $K_1 = 5, 10, 20$ and corresponding $K_2 = 200, 100, 50$. Now each SIFT descriptor in each image is first mapped to one of the K_1 RBM cluster and subspace and then its transformed representation is further mapped to one of the K_2 clusters giving a $K = 1000$ final cluster bag-of-words representation for the images. The same SVM classifier and evaluation methodology was then applied to this new image representation.

Overall mean classification average precision (AP) on the various code-books is shown in Table 3. For $K_1 = 10$, $K_2 = 100$, the mean AP is highest, significantly higher than the traditional baseline codebook representation. This shows that learning clusters in a two-stage process - first by learning a mixture-of-RBMs followed by clustering within each improves the quality of the clustering. Also, the results show that for a given dataset, the right balance has to be struck on how the complexity is distributed between the first mixture-of-RBMs and the second clustering stage. Moreover, the size of projected RBM spaces (in our case 32-dimensional) is also a factor in the overall complexity of the representation. These need to be empirically determined for an arbitrary dataset. While this was applied to a large problem in image classification, in general this process of feature learning may be applied to other domains as well.

Table 3. Mean Classification AP on the VOC Pascal 2007 Subset

Method	K1	K2	Mean AP
Baseline BOW	-	1000	53.74%
Mixtures of RBMs	5	200	55.90%
Mixtures of RBMs	10	100	57.59%
Mixtures of RBMs	20	50	55.43%

5. Conclusions

We proposed the use of a mixture of RBMs to solve the coupled problem of clustering and non-linear subspace learning. Through a number of experiments on real and synthetic datasets we showed the quality of learning obtained by a mixture of RBMs. Compared to the state of the art methods like SSC and RANSAC mixture of RBMs does a better job in terms of improved clustering quality as well as faster learning. Wherever there is complexity in the feature space and data distribution mixture of RBMs can be used as a robust framework for learning structure in data. So far we have worked with an unsupervised version of mixture of RBMs but this can be extended to supervised version where a separate mixture of RBMs can be learnt for each class.

References

Baghshah, Mahdieh Soleymani and Shouraki, Saeed Bagheri. Semi-supervised metric learning using pairwise constraints. In *IJCAI*, 2009.

Bengio, Yoshua, Lamblin, Pascal, Popovici, Dan, Larochelle, Hugo, Montral, Universit De, and Qubec, Montral. Greedy layer-wise training of deep networks. In *In NIPS*. MIT Press, 2007.

Blackwell, Davi. Conditional expectation and unbiased sequential estimation. *Ann. Math. Statistics*, 18:105–110, 1947.

Elhamifar, Ehsan and Vidal, René. Sparse subspace clustering. In *CVPR*, 2009.

Fischler, Martin A. and Bolles, Robert C. Random sample consensus. *Commun. ACM*, 1981.

Hinton, Geoffrey. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:2002, 2000.

Hinton, Geoffrey E., Osindero, Simon, and Teh, Yee-Whye. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18:1527–1554, July 2006.

Kanatani, K. Motion segmentation by subspace separation and model selection. In *ICCV 2001.*

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science*, 1983.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Lee, Honglak, Grosse, Roger, Ranganath, Rajesh, and Ng, Andrew Y. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. ICML '09.

Liu, Guangcan, Lin, Zhouchen, and Yu, Yong. Robust subspace segmentation by low-rank representation. In *ICDMW*, 2010.

Mohamed, Abdel, Dahl, George, and Hinton, Geoffrey. Deep belief networks for phone recognition. In *ICASSP*, 2011.

Rao, S.R., Tron, R., Vidal, R., and Ma, Y. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. In *CVPR*, 2008.

Roweis, Sam T. and Saul, Lawrence K. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.

Salakhutdinov, Ruslan and Hinton, Geoffrey. Replicated softmax: an undirected topic model. In *In NIPS 2010.*

Salakhutdinov, Ruslan, Mnih, Andriy, and Hinton, Geoffrey. Restricted boltzmann machines for collaborative filtering. In *ICML 2004.*

Smolensky, P. In *Parallel Distributed Processing: Volume 1: Foundations*. 1987.