Supplementary Material for Transformation-Based Probabilistic Clustering using Supervision

Siddharth Gopal
Carnegie Mellon University

Yiming Yang
Carnegie Mellon University

Contents

1 TCS for General distributions

We discuss how to estimate a generic transformation $G(x)$ for $x \in \mathcal{X}$ given a mixture of K densities $f(x|C_k)$ over $\mathcal{X}$, where C_k denotes the parameters of cluster k ($k \in \{1, 2, ..K\}$). We assume that we are given supervised information $\mathcal{S} = \{x_i, t_i\}_{i=1}^N$ where $x_i \in \mathcal{X}$ and $t_i \in \{1, 2..K\}$. For convenience define $y_{ik} = I(t_i = k)$. Note that the transformation $G(x)$ needs to appropriately defined such that the transformation still lies within the support of the distribution.
The conditional distribution for class k given an instance x is defined as, (note that k is also overloaded to represent class k)

$$P(k|\mathbf{C}, x_i) = \frac{f(x|C_k)}{\sum_{j=1}^{K} f(x|C_j)}$$

The transformation G is estimated by maximizing the conditional distribution of the labels given the transformed instances $G(\mathbf{X})$.

$$\log P(\mathbf{Y}|G(\mathbf{X}), \mathbf{C}^{mle}(G)) = \sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\left[\log f(G(x_i)|C_k^{mle}(G)) - \log\left(\sum_{j=1}^{K} f(G(x_i)|C_j^{mle}(G))\right)\right]$$

Here $C_k^{mle}(G)$ denotes the MLE estimate of the k'th cluster parameter under the transformation G.
Since we doing supervised learning, there is a good chance that the transformation will overfit the training data. We therefore add a regularizer $\lambda(G)$ with a regularization constant γ.

Putting everything together, the optimization problem to solve is,

$$\begin{aligned}
\max_{G,\mathbf{C}} \quad & \gamma\lambda(G) + \sum_{i=1}^{N}\left[\sum_{k=1}^{K} y_{ik}\log f(G(x_i)|C_k)\right] - \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K} f(G(x_i)|C_k)\right) \\
\text{s.t.} \quad & C_k = \arg\max_{C_k'} \sum_{i=1}^{N} y_{ik}\log f(G(x_i)|C_k') \qquad \text{(i.e. } C \text{ is the MLE estimate)}
\end{aligned}$$

The argmax constraint can be succintly rewritten as equalitys for each C_k constraint. Rewriting as a minimization problem,

$$\min_{G,\mathbf{C}} \quad \gamma\lambda(G) - \sum_{i=1}^{N}\left[\sum_{k=1}^{K} y_{ik}\log f(G(x_i)|C_k)\right] + \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K} f(G(x_i)|C_k)\right) \tag{1}$$

$$\text{s.t.} \quad \frac{\partial \sum_{i=1}^{N} y_{ik}\log f(G(x_i)|C_k')}{\partial C_k} = 0 \tag{2}$$

For arbitrary distributions, there is no general recipe to solve this problem. However for certain distributions, closed form expressions to the equality constraint are available, which can substituted directly into the maximization problem (1). Sometimes this can lead to a convex optimization problem (for e.g. in the case of Gaussian distribution).

Note at the optimal solution $\mathbf{C}^{mle}$, the gradient w.r.t G satisfies we have

$$\gamma\lambda'(G) + \sum_{i=1}^{N}\sum_{k=1}^{K}\left[y_{ik} - P(k|C_k^{mle}, G(x_i))\right]\frac{\partial \log f(G(x_i)|C_k^{mle})}{\partial G} = 0 \tag{3}$$

$$\tag{4}$$

2 TCS using Gaussian mixtures

For P dimensions, the Gaussian distribution is defined over $\mathcal{X} \equiv \mathcal{R}^P$. We define the transformation function to be a linear transformation given by $G(x) = Lx$, where $L \in \mathcal{R}^{P \times P}$. We consider a simple mixture of K Gaussians with mean parameters $\{\theta_k\}_{k=1}^K$ and unit variance,

$$f(x|\theta_k) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}\|x - \theta_k\|^2)$$

The optimization problem is given by,

$$\begin{aligned} \min_{L,\mathbf{C}} \quad & \gamma\lambda(G) + \frac{1}{2}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} y_{ik}\|Lx_i - \theta_k\|^2\right] + \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\exp(-\frac{1}{2}\|Lx_i - \theta_k\|^2)\right) \\ \text{s.t} \quad & -\frac{1}{2}\frac{\partial \sum_{i=1}^{N} y_{ik}\|Lx_i - \theta_k\|^2}{\partial \theta_k} = 0 \end{aligned}$$

If m_k denotes the mean of all instances which belong to cluster k, the equality constrained is simply $\theta_k = Lm_k$.
Substituting this into the optimization problem,

$$\min_{L} \quad \gamma\lambda(G) + \frac{1}{2}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} y_{ik}\|Lx_i - Lm_k\|^2\right] + \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\exp(-\frac{1}{2}\|Lx_i - Lm_k\|^2)\right)$$

This is a nonconvex function in L. However, it can be rewritten as a convex optimization problem in terms of $A = L^\top L$ with a positive definite constraint on A. Assuming the regularizer is convex, this leads to a convex semidefinite program,

$$\begin{aligned} \min_{L} \quad & f(A) = \gamma\lambda(G) + \frac{1}{2}\sum_{i=1}^{N}\left[\sum_{k=1}^{K} y_{ik} d_A(x_i, m_k)\right] + \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\exp(-\frac{1}{2} d_A(x_i, m_k)\right) \\ \text{s.t} \quad & A \succeq 0 \end{aligned}$$

where $d_A(a, b) = (a - b)^\top A(a - b)$.

To solve this optimization problem we employ a projection gradient algorithm where we take a step along the negative gradient (with line search) and the project the update back into the positive semidefinite cone. The algorithm provably converges to the optimal solution if line search is used. The gradient of the objective is given by,

$$f'(A) = \gamma\lambda'(A) + \sum_{i=1}^{N}\sum_{k=1}^{K}\left(y_{ik} - P(k|Lx_i)\right)(x_i - m_k)(x_i - m_k)^\top$$

Note that the solution in A can recover L only upto a rotation matrix L. This is because $A = L^\top L$ can always be written as $A = L^\top Q^\top Q L$ for any rotation matrix $Q^\top = Q^{-1}$.

We have several choices of regularizers, each of which has a different effect on the estimated transformation,

1. $\lambda(A) = \|A - I\|^2$ (frobenius norm from Identity)
2. $\lambda(A) = \|A\|^2$ (frobenius norm from Zero)
3. $\lambda(A) = trace(A) - log(det(A)) - p$ (log-det divergence)

4. $\lambda(A) = sum(sum(|A - I|))$ (entrywise-L1 norm)

When using the first regularizer, high regularization implies there is no change in the transformation, i.e. it is identity whereas when using the second regularizer high regularization means all points are collapsed to zero. The log-det divergence tries to prefer lower-rank matrices. The L1-norm prefers sparser solutions.

3 TCS using von-Mises Fisher

Unlike the Gaussian mixtures, von Mises-Fisher (vMF) is defined only over $\mathcal{X} \equiv \{x : \|x\| = 1\}$. The transformation is defined as

$$G(x) = \frac{Lx}{\|Lx\|}$$

We consider a simple mixture of K vMF's with mean parameters $\{\mu_k\}_{k=1}^K$ and unit concentration parameter,

$$f(x|\mu_k) = \frac{\exp(\mu_k^\top x)}{(2\pi)^{.5P} I_{.5P-1}(1)}$$

Note that the MLE estimate of μ_k has a closed form solution. This leads to the following optimization problem, The optimization problem is given by,

$$\begin{aligned}
\min_{L,\boldsymbol{\mu}} \quad & \gamma\lambda(L) - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\frac{x_i^\top L^\top \mu_k}{\|Lx_i\|} + \sum_{i=1}^{N}\log\left(\sum_{k=1}^{K}\exp(\frac{x_i^\top L^\top \mu_k}{\|Lx_i\|})\right) \\
\text{s.t} \quad & \mu_k = \frac{\sum_{i=1}^{N} y_{ik}\frac{Lx_i}{\|Lx_i\|}}{\left\|\sum_{i=1}^{N} y_{ik}\frac{Lx_i}{\|Lx_i\|}\right\|}
\end{aligned}$$

Substituting the closed form MLE solution into the optimization leads to not-so-nice expressions. We instead recommend an alternating optimization between $\boldsymbol{\mu}$ and L. In one step the μ_k's are updated through the equality constraint, and in the second step, we estimate L by optimizing the objective only over L.
The derivative of the objective w.r.t L is given by,

$$\begin{aligned}
\text{Note that} \quad & \frac{d\left(\frac{x_i^\top L^\top \mu_k}{\sqrt{x_i^\top L^\top L x_i}}\right)}{dL} == \frac{1}{\|Lx_i\|}\mu_k x_i^\top - (x_i^\top L^\top \mu_k)\frac{L(x_i x_i^\top)}{\|Lx_i\|^{3/2}} \\
\text{Define} \quad & n_i = \frac{x_i}{\|Lx_i\|} \\
\text{Gradient} \quad & = \sum_{i=1}^{N}\sum_{k=1}^{K}\left(y_{ik} - p\left(k \mid \frac{Lx_i}{\|Lx_i\|}\right)\right)(\mu_k^\top n_i - (n_i^\top L^\top \mu_k)L(n_i n_i^\top))
\end{aligned}$$

Note that no optimal solution can be guaranteed. Even basic convergence of the solution is not guaranteed since the problem is nonconvex with constraints. However, in practice it worked fine. If convergence is definitely needed, the problem can be formulated using sequential quadratic programming. This will guarantee convergence to a local optimum.
The fundamental reason why nonconvexity arises is because the transformation is not convex w.r.t L (although it is geodesically convex w.r.t x for a fixed L though). A small change in the transformation matrix can cause a dramatic change in the shearing and stretching on the sphere - especially if the transformation is radially outwards/inwards from the sphere. I havent found a *nicer* transformation on spheres that is also flexible enough.

4 TCS for Gamma distributions

Gamma distribution is defined over $\mathcal{X} \equiv \mathcal{R}_+$. We consider a multivariate version of gamma distribution where each dimension is independently drawn. In a P dimensional space, the density for a point $x \in \mathcal{R}_+^P$ is given by,

$$P(x_i|\boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{p=1}^{P} \frac{\beta_p^{\alpha_p}}{\Gamma(\alpha_p)} x_{ip}^{\alpha_p - 1} e^{-\beta_p x_{ip}}$$

We consider a mixture of gamma distributions where each cluster is associated with cluster parameter $\{\alpha_k, \beta_k\}_{k=1}^K$ where $\alpha_k, \beta_k \in \mathcal{R}_+^P$. We define a transformation,

$$G(x_i) = Lx_i \quad , \quad L \in \mathcal{R}_+^{P \times P}$$

The log probability of the transformed point Lx_i from cluster k is given by,

$$\log P(Lx_i|k) = \sum_{p=1}^{P} \alpha_k^p \log(\beta_k^p) - \sum_{p=1}^{P} \log \Gamma(\alpha_k^p) + \sum_{p}^{P} (\alpha_k^p - 1) \log(L_p^T x_i) - \sum_{p=1}^{P} \beta_k^p (L_p^T x_i)$$

where L_p is row p in L

Note that the MLE constraint for α_k, β_k are given by (m_k denotes the mean of all instances assigned to cluster k)

$$\beta_k^p = \frac{\alpha_k^p}{L_p^\top m_k}$$

$$\log(\beta_k^p) - \psi(\beta_k^p) = \log(L_p^\top m_k) - \frac{1}{n_k} \sum_{i=1}^{N} \log(L_p^\top x_i)$$

Since the MLE estimate of β_k^p is relatively simple, we substitute it into the optimization problem (1) with the gamma density defined above. We follow a similar strategy like vMF where alternatively optimize α_k^p and L. In the first step, we find the α_k^p that satisfies the MLE constraint using newton's method, and in the second step we fix the α_k's and optimize for L.

The derivative of above w.r.t row L_p is given by,

$$\sum_{i=1}^{N} \sum_{k=1}^{K} (y_{ik} - p_{ik}) \frac{\partial \log P(Lx_i|\alpha^k, \beta^k)}{\partial L_p}$$

Considering summation with just class K, the derivative w.r.t L_p (the pth row) is given by,

$$\begin{aligned} \text{Gradient} = &\left(\frac{\alpha_k^p m_k}{(L_p^\top m_k)} \sum_{i=1} (y_{ik} - P(k|Lx_i)) \right) + \left((\alpha_k^p - 1) \sum_{i=1}^{N} (y_{ik} - P(k|Lx_i)) \frac{1}{L_p^\top x_i} x_i \right) \\ &- \left(\frac{\alpha_k^p}{L_p^\top m_k} \sum_{i=1}^{N} (y_{ik} - p(k|Lx_i)) x_i \right) + \left(\frac{m_k}{(L_p^\top m_k)^2} \sum_{i=1}^{N} (y_{ik} - P(k|Lx_i)) L_p^\top x_i) \right) \end{aligned}$$

This can be rewritten as,

$$\text{Define} \quad n_k = \sum_{i=1}^{N} y_{ik} \quad , \quad z_k = \sum_{i=1}^{N} P(k|Lx_i) \quad , \quad J_{ikp} = \frac{(y_{ik} - p(k|Lx_i))}{L_p^\top x_i}$$

$$J_{kp} = \sum_{i=1}^{N} J_{ikp} x_i \quad , \quad s_k = \sum_{i=1}^{N} (y_{ik} - p(k|Lx_i)) x_i$$

$$\text{Gradient} = \left(\frac{\alpha_k^p}{L_p^\top m_k} (n_k - z_k) m_k \right) + (\alpha_k^p - 1) J_{kp} - \left(\frac{\alpha_k^p}{L_p^\top m_k} s_k \right) + \left(\frac{L_p^\top s_k}{(L_p^\top m_k)^2} m_k \right)$$

We follow a similar procedure like vMF models. Note that things can simply a lot if we assume that the shape parameter is known or can be estimated from all the data initially.

5 Evaluation Metrics

The five clustering measures we used for ground truth based evaluations are Normalized Mutual Information (**NMI**), Mutual Information (**MI**), Rand Index (**RI**), Adjusted Rand Index (**ARI**), Purity. Following standard notation[1], given a collection of N instances, we let $\Omega = \{\omega_1, \omega_2, ..., \omega_K\}$ denote predicted clusters and $C = \{c_1, c_2, ..c_J\}$ denote the true clusters.

1. **MI**: The standard definition of mutual information is

$$MI(\Omega, C) = I(\Omega, C) = \sum_{k,j} \frac{|\omega_k \cap C_j|}{N} \log\left(\frac{N|\omega_k \cap c_j|}{|\omega_k||c_j|}\right)$$

2. **NMI**: NMI is the normalized variant of MI, where we normalize MI by the sum of the entropies of the two random variables. This quantity always lies between 0 and 1, 0 indicating independence and 1 indicating complete dependence.

$$NMI(\Omega, C) = \frac{I(\Omega, C)}{H(\Omega) + H(C)} \quad \text{where} \quad H(\Omega) = \sum_K \frac{|\omega_k|}{N} log(\frac{|\omega_k|}{N}) \ , \ H(C) = \sum_J \frac{|c_j|}{N} log(\frac{|c_j|}{N})$$

3. **RI**: We can view clustering as predicting a $\{0, 1\}$ decision for all $N(N-1)/2$ pairs of instances, where 1 indicates the pair of instances belong to the same cluster, and 0 indicates different clusters. Given the ground-truth decisions for the pairs, we can define the standard TP, TN, FP and FN for the predicted decisions. The rand index is defined as,

$$RI = \frac{TP + TN}{TP + TN + FP + FN}$$

4. **ARI**: ARI is the corrected-for-chance version of the Rand index. Define $n_{j.} = \sum_k |c_j \cap \omega_k|$ and $n_{.k} = \sum_j |c_k \cap \omega_k|$, and $n_{ij} = |c_k \cap \omega_k|$.

$$ARI = \frac{\sum_{k,j} \binom{n_{ij}}{2} - \left[\sum_j \binom{n_{j.}}{2} \sum_k \binom{n_{.k}}{2}\right] / \binom{N}{2}}{\frac{1}{2}\left[\sum_j \binom{n_{j.}}{2} + \sum_k \binom{n_{.k}}{2}\right] - \left[\sum_j \binom{n_{j.}}{2} \sum_k \binom{n_{.k}}{2}\right] / \binom{N}{2}}$$

5. **Purity**: Each cluster is assigned to the class which is most frequent in the cluster, and purity is the accuracy of this assignment,

$$Purity(\Omega, C) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

[1] http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-clustering-1.html

6 Datasets

We extensively tested our proposed methods on a wide range of datasets from several application domains such as text, time-series, speech, character recognition, face recognition etc.

6.1 Time-series datasets

Dataset Name	#Instances	#Dimension	#Classes
Australian Signs (Aussign)	2565	352	95
Character Trajectory (Char)	2858	192	20
Daily and Sports Activity (DSPA)	152	1440	19
Libras	360	90	15

1. *Australian Signs* [2] The data consists of sample of Australian Sign Language signs. There are 27 examples of each of 95 signs, each of which were captured from a native signer using high-quality position trackers and instrumented gloves. There were 22 features such as x-position, y-position, etc which were measured over time. We used Fast Fourier Transform (FFT) to extract the first 16 important dimensions for each feature. We represented each feature using a 16x22 = 352 dimensional feature space.

2. *Character Trajectory* [3] Each instance represents the pentip trajectory over time to draw a character. We used FFT to extract the first 64 dimensions for each of the 3 directions of the pentip's velocity profile. We used a larger number of frequencies (64) to distinguish between the characters better. The classes refer to the character drawn.

3. *Daily and Sports Activity Recognition* [1] This datasets consists of 19 activities performed by humans in a sports hall, like walking, running etc. Each of the 19 activities is repeated multiple times by different people. Each instance represents a sequence of sensor measurements over time while the activity was performed. The goal is to cluster the sensor measurement profile according to the activity performed. We used FFT on each of the sensor and extracted a 32 dimensional vector, which was concatenated across all sensors leading to 1440 dimensional space.

4. *Libras* [4] The data set contains 15 classes of 24 instances each. Each class references to a hand movement type in the brazilian sign language.

6.2 Text data

Dataset Name	#Instances	#Dimension	#Classes
CNAE	1079	856	9
K9	2340	21839	20
TDT4	622	8895	34
TDT5	6355	20733	125

[2] http://www.cse.unsw.edu.au/ waleed/phd/
[3] http://archive.ics.uci.edu/ml/datasets/Character+Trajectories
[4] http://archive.ics.uci.edu/ml/datasets/Libras+Movement

1. *CNAE* [5] A collection of 1080 documents of free text business descriptions of Brazilian companies categorized into a subset of 9 categories.

2. *K9* [6] This is a collection of 2340 Web pages which appeared in the Yahoo news hompage. The webpages were categorized into one of 20 broad news topics.

3. *TDT-4,TDT-5* [7] These articles appeared on Reuters for a period of several months. We selected only those news articles which had relevance judgements. All news stories were categorized into one of many news 'events' which happened at that time.

Note that all the text datasets were subjected to standard stopword removal, stemming and 'ltc' term weighting.

6.3 Handwriting Recognition

Dataset Name	#Instances	#Dimension	#Classes
Penbased Recognition (Penbased)	10992	16	10
Letter Recognition (Letter)	20000	16	26
USPS	9298	1984	10
Binary Alpha Digits (Binalpha)	1404	2480	36
Optical Recognition (Optrec)	5620	496	10
MNIST	70000	6076	10

1. *Penbased Recognition* Each instance represents a digit from 1 to 10 drawn on a Wiacom tablet. The features are constructed by resampling the positions of the Wiacom pen and normalizing them. Refer to [8] on how exactly the features are constructed.

2. *Letter Recognition* [9] Each instance is a black-and-white rectangular pixel display of one of the 26 capital letters in the English alphabet. The features are some statistical moments of the image such as mean, variance, correlation etc.

3. *USPS* [10] The dataset refers to numeric data obtained from the scanning of handwritten digits from envelopes by the U.S. Postal Service. Each instance is represented by a 16x16 grayscale image. We extracted HOG features from each pixel with a patchsize of 2x2 and concatenated to form a 1984 dimensional space.

4. *Binary Alpha Digits* [11] These are 20x16 grayscale images of one 26 digits + 10 numbers. We extracted HOG features from each pixel with a patchsize of 2 and concatenated them.

5. *Optical Recognition* [12] Each instace represents a number from 0 to 9. The 32x32 bitmaps are divided into nonoverlapping blocks of 4x4 and the number of 'on' pixels are counted in each block. We then used the HOG features with a patchsize of 2 leading to a 496 dimensional feature space.

[5] http://archive.ics.uci.edu/ml/datasets/CNAE-9
[6] http://www-users.cs.umn.edu/boley/ftp/PDDPdata/
[7] http://www.itl.nist.gov/iad/mig/tests/tdt/
[8] http://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits
[9] http://archive.ics.uci.edu/ml/datasets/Letter+Recognition
[10] http://statweb.stanford.edu/ tibs/ElemStatLearn/data.html
[11] http://www.cs.nyu.edu/ roweis/data.html
[12] http://archive.ics.uci.edu/ml/machine-learning-databases/optdigits/

6. *MNIST* [13] It is one of the most popular handwritten digits recognition datasets. Each image is a number from 0 to 9 and is represented by a 28x28 bitmap with every pixel taking a value from 0 to 255. We extracted HOG features from each bitmap with a patchsize of 2. This leads to a 6076 dimensional feature-space.

6.4 Face Recognition

Dataset Name	#Instances	#Dimension	#Classes
AT&T faces	400	19964	40
UMIST	575	10304	20
Faces96	3016	19375	151
Labeled Faces in Wild (LFW)	29791	4324	158

1. *AT & T Faces* [14] A set of face images taken between April 1992 and April 1994 of 40 distinct subject, with ten different images of each of subject. The size of each image is 112x92 pixels, with 256 grey levels per pixel. We extracted HOG features from each pixel with a patchsize of 4 and concatenated them.

2. *UMIST Faces* [15] Consists of 564 , 112x92 cropped grayscale images of 20 people. Pictures covers a range of poses from profile to frontal views, and a range of subjects from different race/sex/appearance. We extracted HOG features from each pixel with a patchsize of 4 and concatenated them.

3. *Faces96 dataset* [16] This is another face recognition dataset with 20, 196x196 grayscale images with different variations in the subject profile. We extracted HOG features with a patchsize of 4.

4. *Labeled Faces in Wild* [17] This is one of the largest collection of faces of popular people as it appears on the web with 13233 images of 5749 people. We did some filtering and removed all subjects with less than 10 images, this leaves with 29791 images of 158 people. We then used HOG features to represent each image.

6.5 Speech/Speaker Recognition

Dataset Name	#Instances	#Dimension	#Classes
Isolet	7797	617	26
Wall Street Journal (WSJ)	34942	25	131

1. *Isolet* [18] This is a collection of speech recordings from 152 people who spoke the name of each alphabet twice. The classes here denote the alphabet that was spoken. The features include spectral coefficients; contour features, sonorant features, pre-sonorant features, and post-sonorant features.

[13] http://www.cs.nyu.edu/ roweis/data.html
[14] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html
[15] http://www.cs.nyu.edu/ roweis/data.html
[16] http://cswww.essex.ac.uk/mv/allfaces/faces96.html
[17] http://vis-www.cs.umass.edu/lfw/
[18] http://archive.ics.uci.edu/ml/datasets/ISOLET

2. *Wall Street Journal* [19] This is a database of oral narration by 131 people of 34942 passages from the wall street journal. Each recording is sampled and 25 MFCC's extracted and averaged over time. Due to the averaging, most of the speaker specific information is lost. This is a ideal dataset to show how supervision can recover the most interesting directions in the featurespace for clustering.

6.6 Other datasets

Dataset Name	#Instances	#Dimension	#Classes
Image	2310	18	7
Vowel	990	10	11
Leaves	1599	192	100

1. *Image* [20] Each instance is a 3x3 image patch denoting one of the 7 outdoor scenes in the image database. The features represent properties of the patch such region centroid, mean color, saturation etc.

2. *Vowel* [21] is highly popular dataset which contain speech utterances from 48 speakers. The 11 classes denote different vowel sounds. The reference states that speech signals were low pass filtered at 4.7kHz and then digitised to 12 bits with a 10kHz sampling rate. Twelfth order linear predictive analysis was carried out on six 512 sample Hamming windowed segments from the steady part of the vowel. The reflection coefficients were used to calculate 10 log area parameters, giving a 10 dimensional input space.

3. *Leaves* [3] The instances are pictures of 100 different types of leaves described by three sets of 64 dimensional vectors - shape, texture, margin, which we concatenated to form a single 192 dimensional feature space.

[19] http://catalog.ldc.upenn.edu/LDC95S23
[20] http://archive.ics.uci.edu/ml/datasets/Image+Segmentation
[21] http://archive.ics.uci.edu/ml/machine-learning-databases/undocumented/connectionist-bench/vowel/vowel.names

7 Results of Supervised Clustering on Unnormalized Data

In this section we detail the results of using our supervised cluster discovery framework using Gaussian mixtures i.e. using GM. The details of the datasets and metrics can be found in the previous sections.

7.1 Methods for comparison

1. **GM** A mixture of K Gaussian distributions with individual means and a single common variance parameter. All parameters are estimated using EM algorithm. Note that this model is unsupervised and cannot use supervision.

2. **TCS-GM** : Our proposed TCS technique using Gaussian mixture model as discussed in Table 2. The regularizer was $\|A - I\|^2$.

3. **TCS-GM-L2** : Our proposed TCS technique using Gaussian mixture model as discussed in Table 2. The regularizer was plain L2 regularization - $\|A\|^2$. Note this regularizer favors lower-dimensions.

4. **LMNN** [4] A distance metric learning technique which aims to learn a local metric such that each instance's neighborhood contains only instances from the same class. If not, the closest same class neighbors are pulled and imposter classes are pushed out. The distances are penalized based on a hinge-loss.

5. **PCC** [2] use a metric learning approach where they learn a metric which pulls within cluster elements towards each other and pushes out of cluster elements against each other. To enable sharing of information, we uparameterize the metric by a single A which is learnt by,

$$\min_A f(A) = \sum_{i,k} y_{ik} \|x_i - \theta_k\|_A + \sum_{i,j,t_i=t_j} \|x_i - x_j\|_A - \sum_{i,j,t_i \neq t_j} \|x_i - x_j\|_A$$
$$A \succeq 0$$

   As formulated, it is a non-convex minimization with a positive-semi-definite constraint on A. Note that without the positive-definite constraint, this problem has a closed-form expression. We used the authors proposed strategy of starting from the closed-form solution and adding the smallest ϵI to make it positive-definite.

6. **BP**: This is our proposed Bayesian model for TCS where there is sharing of information between the parameters. We assume the following generative process for the data,

$$\theta_k \sim \mathcal{N}(\theta_0, \sigma_0^2 I)$$
$$z_i \sim \text{Categorical}(\frac{1}{K}, \frac{1}{K}, .., \frac{1}{K})$$
$$x_i \sim \mathcal{N}(\theta_{z_i}, \Sigma)$$

   Here $\theta_0, \sigma_0^2, \Sigma$ are shared between all the clusters. We assume that the hidden cluster assignments for some of the data is given as supervision. We used the familiar EM framework (in this case constrained EM) to estimate the parameters.

7.2 Experimental Settings

The following experimental settings were observed,

1. For all the datasets, the class labels are considered as the groundtruth clusters.

2. The classes were randomly divided into three groups - training, validation and testing. The instances from the training classes were provided as supervision. The validation set was used to choose the regularization parameter and the clustering performance was measured on the testing set. We assume that all the instances from the training clusters are provided, if not, we assume we can train appropriate classifiers for these classes.

3. For all the datasets, if the number of dimensions was greater than 200, we used SVD to reduce the dimension to 30, except for faces where we used 50 dimensions. This did not cause much degradation in performance (refer section 9 for detailed results).

4. **For TCS-GM** : The optimization solver in TCS-GM is MATLAB-based and can therefore harness the fast multicore parallel matrix multiplication libraries. The convergence criteria was set as follows - a limit of 2000 iterations or until the gradient was less than 1e^{-3}.

5. **For LMNN** : We used the authors code available online [22]. Since the code requires all instances to have the same number of neighbors, we set the target number of neighbors to a relatively high value like 50. We considered a range of regularization parameters from $\mu = 1$ to $\mu = 0$.

6. **Clustering Algorithm** After the unlabeled instances are folded into the new space, a Gaussian mixtures with soft assignments is used for clustering. All clustering results are averaged over **50** runs. Each run is initialized with a different Kmeans++ initialization. The maximum number of iterations for each clustering was set to 100, or if the difference in improvement was less than 0.001. This seemed to be a very strict criteria- most runs converged within 30 iterations.

7.3 Results

7.3.1 Time-series data

On time-series data, TCS-GM-L2 shows very strong performance. This might be because time-series inherently lie in a very low dimension that the L2 regularization is able to capture.

[22] http://www.cse.wustl.edu/ kilian/code/page21/page21.html

Table 1: Improvement of the various TCS techniques over unsupervised clustering on time-series datasets.

Dataset	*Metrics*	Supervised Learning Method					Unsupervised
		TCS-GM	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**
Aussign	*NMI*	0.890	0.925	**0.937**	0.775	0.838	0.817
	MI	3.048	3.173	**3.229**	2.634	2.856	2.780
	ARI	0.694	0.764	**0.809**	0.473	0.588	0.546
	RI	0.980	0.985	**0.988**	0.963	0.972	0.968
	Purity	0.771	0.814	**0.852**	0.609	0.694	0.668
Char	*NMI*	0.752	**0.754**	0.671	0.690	0.737	0.687
	MI	1.532	**1.540**	1.375	1.409	1.507	1.405
	ARI	0.648	**0.652**	0.572	0.588	0.622	0.568
	RI	0.917	**0.919**	0.902	0.904	0.913	0.900
	Purity	0.761	**0.773**	0.728	0.729	0.750	0.721
DSPA	*NMI*	0.685	**0.734**	0.597	0.668	0.642	0.678
	MI	1.231	**1.367**	0.968	1.204	1.124	1.219
	ARI	0.406	**0.518**	0.253	0.392	0.340	0.398
	RI	0.825	**0.873**	0.668	0.822	0.785	0.823
	Purity	0.660	**0.712**	0.519	0.654	0.605	0.659
Libras	*NMI*	0.608	**0.642**	0.498	0.599	0.592	0.512
	MI	0.945	**1.010**	0.769	0.939	0.918	0.803
	ARI	0.474	**0.540**	0.375	0.474	0.454	0.376
	RI	0.821	**0.849**	0.781	0.825	0.814	0.794
	Purity	0.693	**0.736**	0.623	0.687	0.681	0.631

7.3.2 Text data

On text data, TCS-GM works best. Most of the other methods had lower performance than unsupervised clustering, i.e. could not learn anything meaningful from the supervision. Note that in general, the improvements in text-data seemed to be milder, this might be because the topics are too general and the training classes do not given much information to cluster the test data.

Table 2: Improvement of the various TCS techniques over unsupervised clustering on text data.

Dataset	*Metrics*	Supervised Learning Method					Unsupervised
		TCS-GM	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**
CNAE	*NMI*	**0.608**	0.239	0.314	0.431	0.481	0.485
	MI	**0.638**	0.246	0.309	0.418	0.476	0.480
	ARI	**0.574**	0.202	0.194	0.285	0.355	0.359
	RI	**0.800**	0.626	0.598	0.630	0.674	0.676
	Purity	**0.807**	0.566	0.611	0.649	0.684	0.684
K9	*NMI*	**0.581**	0.405	0.375	0.562	0.510	0.561
	MI	**0.952**	0.654	0.605	0.901	0.780	0.889
	ARI	**0.615**	0.456	0.178	0.367	0.259	0.365
	RI	**0.853**	0.788	0.693	0.756	0.696	0.753
	Purity	**0.768**	0.670	0.625	0.731	0.699	0.733
TDT4	*NMI*	**0.905**	0.689	0.900	0.891	0.894	0.898
	MI	**1.948**	1.498	1.951	1.921	1.939	1.937
	ARI	**0.826**	0.441	0.807	0.794	0.798	0.809
	RI	**0.959**	0.869	0.954	0.949	0.952	0.954
	Purity	**0.926**	0.735	0.923	0.907	0.915	0.914
TDT5	*NMI*	**0.696**	0.695	0.675	0.692	0.694	0.692
	MI	**2.150**	2.146	2.085	2.135	2.127	2.135
	ARI	**0.200**	0.199	0.192	0.193	0.196	0.194
	RI	**0.856**	0.855	0.854	0.854	0.853	0.854
	Purity	**0.859**	0.857	0.837	0.856	0.857	0.858

7.3.3 Handwriting Recognition

TCS-GM and TCS-GM-L2 seemed to be the choice methods for supervised clustering of handwriting recognition data. On the datasets were the features where positions - *Penbased* and *Letter*, TCS-GM-L2 performed better. On the rest of the datasets where the features were based on HOG, TCS-GM performed best.

Table 3: Improvement of the various TCS techniques over unsupervised clustering on the task of Handwriting Recognition.

Dataset	*Metrics*	Supervised Learning Method					Unsupervised
		TCS-GM	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**
Penbased	*NMI*	0.556	**0.604**	0.571	0.401	0.487	0.522
	MI	0.753	**0.796**	0.769	0.550	0.658	0.707
	ARI	0.509	**0.504**	0.531	0.362	0.417	0.454
	RI	0.809	**0.807**	0.814	0.757	0.771	0.786
	Purity	0.736	**0.736**	0.758	0.649	0.663	0.686
Letter	*NMI*	0.478	**0.504**	0.434	0.265	0.373	0.351
	MI	1.078	**1.135**	0.976	0.596	0.841	0.792
	ARI	0.301	**0.329**	0.245	0.116	0.193	0.175
	RI	0.867	**0.867**	0.854	0.828	0.843	0.841
	Purity	0.501	**0.525**	0.459	0.307	0.390	0.369
USPS	*NMI*	**0.845**	0.455	0.785	0.807	0.786	0.815
	MI	**1.129**	0.594	1.050	1.075	1.043	1.088
	ARI	**0.829**	0.427	0.757	0.776	0.725	0.784
	RI	**0.932**	0.763	0.904	0.909	0.885	0.913
	Purity	**0.916**	0.682	0.882	0.888	0.863	0.893
Binalpha	*NMI*	**0.794**	0.703	0.703	0.725	0.680	0.719
	MI	**1.948**	1.723	1.732	1.778	1.672	1.765
	ARI	**0.669**	0.550	0.567	0.559	0.509	0.545
	RI	**0.947**	0.928	0.931	0.930	0.923	0.928
	Purity	**0.795**	0.715	0.731	0.682	0.666	0.677
Optrec	*NMI*	**0.936**	0.727	0.905	0.864	0.914	0.912
	MI	**1.028**	0.799	0.995	0.947	1.002	0.998
	ARI	**0.956**	0.791	0.939	0.880	0.929	0.924
	RI	**0.981**	0.907	0.973	0.946	0.968	0.965
	Purity	**0.985**	0.926	0.979	0.955	0.973	0.968
MNIST	*NMI*	**0.842**	0.701	0.717	0.553	0.741	0.830
	MI	**1.165**	0.968	0.983	0.761	1.016	1.149
	ARI	**0.885**	0.719	0.724	0.508	0.740	0.875
	RI	**0.957**	0.894	0.893	0.813	0.899	0.953
	Purity	**0.955**	0.883	0.860	0.731	0.877	0.950

7.3.4 Face recognition

On face recognition, the results are a little mixed. Both TCS-GM-L2 and LMNN seem to show competitive performance. However, we believe all the datasets except LFW are a little contrived - all the images were captured under ideal lighting and posing conditions. This make techniques like LMNN which are nearest neighbor based, work better. On LFW, the images represent the distribution of images on the web and are therefore much more *realistic*. On LFW, TCS-GM-L2 works the best.

Table 4: Improvement of the various TCS techniques over unsupervised clustering on the task of Face recognition.

Dataset	*Metrics*	Supervised Learning Method					Unsupervised
		TCS-GM	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**
AT & T	*NMI*	0.843	**0.879**	0.842	0.822	0.852	0.834
	MI	2.173	**2.266**	2.182	2.119	2.197	2.151
	ARI	0.627	**0.692**	0.652	0.590	0.640	0.613
	RI	0.948	**0.957**	0.953	0.943	0.950	0.946
	Purity	0.755	**0.794**	0.784	0.726	0.772	0.746
Umist	*NMI*	0.588	0.739	**0.792**	0.569	0.563	0.554
	MI	1.180	1.491	**1.574**	1.153	1.129	1.112
	ARI	0.355	0.548	**0.600**	0.355	0.332	0.322
	RI	0.843	0.894	**0.895**	0.849	0.838	0.836
	Purity	0.579	0.712	**0.746**	0.560	0.557	0.552
Faces96	*NMI*	0.922	0.929	**0.939**	0.887	0.894	0.886
	MI	3.568	3.600	**3.642**	3.438	3.465	3.438
	ARI	0.728	0.752	**0.771**	0.662	0.673	0.663
	RI	0.989	0.990	**0.991**	0.986	0.987	0.986
	Purity	0.794	0.810	**0.834**	0.747	0.756	0.745
LFW	*NMI*	0.388	**0.415**	0.331	0.312	0.297	0.285
	MI	1.456	**1.557**	1.239	1.168	1.115	1.070
	ARI	0.081	**0.095**	0.040	0.030	0.026	0.021
	RI	0.939	**0.940**	0.936	0.936	0.936	0.935
	Purity	0.341	**0.367**	0.271	0.253	0.246	0.233

7.3.5 Speech/speaker recognition

For speech (*Isolet*) as well as speaker recognition (*WSJ*), TCS-GM model works best.

Table 5: Improvement of the various TCS techniques over unsupervised clustering on the task of Speech and Speaker recognition.

Dataset	*Metrics*	Supervised Learning Method					Unsupervised
		TCS-GM	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**
Isolet	*NMI*	**0.834**	0.796	0.812	0.747	0.829	0.816
	MI	**1.878**	1.803	1.831	1.700	1.874	1.846
	ARI	**0.707**	0.669	0.667	0.632	0.693	0.682
	RI	**0.942**	0.937	0.934	0.931	0.940	0.938
	Purity	**0.813**	0.798	0.761	0.748	0.786	0.767
WSJ	*NMI*	**0.813**	0.811	0.810	0.521	0.707	0.555
	MI	**2.678**	2.674	2.672	1.742	2.356	1.856
	ARI	**0.369**	0.362	0.361	0.177	0.278	0.200
	RI	**0.914**	0.913	0.913	0.891	0.904	0.894
	Purity	**0.854**	0.852	0.854	0.610	0.785	0.647

7.3.6 Other datasets

One two of the datasets, the TCS models perform better. On one of them - LMNN performs better.

Table 6: Improvement of the various TCS techniques over unsupervised clustering on other datasets.

Dataset	*Metrics*	Supervised Learning Method					Unsupervised
		TCS-GM	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**
Image	*NMI*	**0.836**	0.399	0.495	0.601	0.699	0.780
	MI	**0.915**	0.427	0.529	0.640	0.768	0.851
	ARI	**0.839**	0.344	0.421	0.533	0.697	0.794
	RI	**0.928**	0.700	0.735	0.785	0.865	0.906
	Purity	**0.939**	0.715	0.727	0.720	0.878	0.908
Vowel	*NMI*	**0.414**	0.394	0.346	0.135	0.235	0.253
	MI	**0.662**	0.624	0.546	0.212	0.370	0.400
	ARI	**0.273**	0.274	0.244	0.066	0.159	0.149
	RI	**0.766**	0.760	0.749	0.686	0.718	0.720
	Purity	**0.521**	0.522	0.479	0.325	0.450	0.396
Leaves	*NMI*	0.824	0.822	**0.852**	0.767	0.815	0.778
	MI	2.855	2.850	**2.951**	2.650	2.802	2.687
	ARI	0.592	0.590	**0.637**	0.491	0.559	0.512
	RI	0.975	0.975	**0.978**	0.968	0.972	0.970
	Purity	0.725	0.725	**0.751**	0.650	0.702	0.665

8 Results of Supervised Clustering on Normalized Data

In this section we detail the results of using our supervised cluster discovery framework using von Mises-Fisher mixtures on normalized data.
We used similar preprocessing of data as outlined before i.e. SVD-based dimension reduction on data. We then normalized all the instances to a unit sphere since VM-based models can be applied only on unit normalized data.

8.1 Methods for comparison

1. **VM** A mixture of K vMF distributions with individual means and a single common concentration parameter. All parameters are estimated using EM algorithm. Note that this model like GM is unsupervised and cannot use supervision.

2. **TCS-VM** Our proposed TCS using vMF mixture model and $\|A - I\|^2$ regularization, followed by VM on the transformed unlabeled data.

For an informative comparison we also include the results of TCS-GM, TCS-GM-L2, LMNN, BP and PCC.
We used the same experimental settings as described earlier.

8.1.1 Time-series data

Table 7: Improvement of the various TCS techniques over unsupervised clustering on time-series datasets.

Dataset	***Metrics***	**Supervised Learning Method**						**Unsupervised**	
		TCS-VM	**TCS-GM**	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**	**VM**
Aussign	*NMI*	0.912	0.913	0.912	**0.927**	0.802	0.843	0.824	0.826
	MI	3.131	3.133	3.129	**3.181**	2.735	2.884	2.817	2.819
	ARI	0.768	0.742	0.740	**0.777**	0.540	0.609	0.576	0.570
	RI	0.985	0.983	0.983	**0.986**	0.969	0.974	0.972	0.971
	Purity	0.818	0.800	0.797	**0.824**	0.649	0.703	0.678	0.680
Char	*NMI*	0.692	**0.714**	0.610	0.673	0.698	0.732	0.710	0.707
	MI	1.421	**1.453**	1.251	1.386	1.422	1.496	1.447	1.442
	ARI	0.579	**0.600**	0.524	0.558	0.578	0.610	0.593	0.591
	RI	0.904	**0.907**	0.892	0.901	0.902	0.910	0.905	0.905
	Purity	0.736	**0.738**	0.705	0.712	0.721	0.745	0.734	0.734
DSPA	*NMI*	**0.823**	0.773	0.776	0.766	0.717	0.697	0.725	0.721
	MI	**1.556**	1.472	1.480	1.440	1.327	1.271	1.336	1.317
	ARI	**0.687**	0.591	0.583	0.561	0.474	0.434	0.480	0.461
	RI	**0.923**	0.902	0.900	0.847	0.859	0.839	0.859	0.850
	Purity	**0.802**	0.764	0.755	0.662	0.700	0.677	0.708	0.699
Libras	*NMI*	0.629	0.680	**0.682**	0.506	0.641	0.616	0.571	0.580
	MI	0.981	1.068	**1.070**	0.816	0.999	0.948	0.889	0.900
	ARI	0.506	0.579	**0.580**	0.385	0.511	0.472	0.441	0.448
	RI	0.834	0.861	**0.861**	0.790	0.834	0.816	0.811	0.812
	Purity	0.700	0.753	**0.756**	0.636	0.705	0.690	0.664	0.668

8.1.2 Text data

On text data, TCS-VM works best.

Table 8: Improvement of the various TCS techniques over unsupervised clustering on time-series datasets.

Dataset	*Metrics*	Supervised Learning Method						Unsupervised	
		TCS-VM	**TCS-GM**	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**	**VM**
CNAE	*NMI*	**0.916**	0.815	0.815	0.692	0.614	0.693	0.792	0.909
	MI	**1.006**	0.890	0.890	0.752	0.647	0.740	0.862	0.997
	ARI	**0.950**	0.826	0.826	0.694	0.539	0.646	0.795	0.932
	RI	**0.978**	0.922	0.922	0.862	0.784	0.838	0.907	0.969
	Purity	**0.983**	0.935	0.935	0.878	0.807	0.856	0.916	0.974
K9	*NMI*	**0.638**	0.621	0.443	0.479	0.617	0.584	0.616	0.615
	MI	**1.132**	1.099	0.801	0.856	1.090	1.017	1.085	1.090
	ARI	**0.514**	0.527	0.468	0.274	0.453	0.357	0.452	0.456
	RI	**0.836**	0.837	0.802	0.756	0.815	0.775	0.814	0.816
	Purity	**0.842**	0.810	0.690	0.698	0.810	0.793	0.808	0.810
TDT4	*NMI*	**0.936**	0.916	0.755	0.923	0.914	0.911	0.915	0.933
	MI	**2.058**	2.024	1.690	2.027	2.017	2.003	2.020	2.052
	ARI	**0.871**	0.827	0.554	0.847	0.825	0.807	0.821	0.857
	RI	**0.971**	0.962	0.904	0.965	0.961	0.957	0.960	0.968
	Purity	**0.954**	0.941	0.812	0.946	0.938	0.939	0.941	0.954
TDT5	*NMI*	**0.781**	0.750	0.746	0.707	0.750	0.756	0.755	0.766
	MI	**2.336**	2.292	2.281	2.165	2.291	2.297	2.302	2.326
	ARI	**0.393**	0.255	0.254	0.241	0.255	0.265	0.260	0.276
	RI	**0.882**	0.864	0.863	0.861	0.864	0.865	0.864	0.867
	Purity	**0.927**	0.905	0.900	0.872	0.905	0.908	0.909	0.920

8.1.3 Handwriting Recognition

Table 9: Improvement of the various TCS techniques over unsupervised clustering on the task of Handwriting Recognition.

Dataset	*Metrics*	Supervised Learning Method						Unsupervised	
		TCS-VM	**TCS-GM**	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**	**VM**
Penbased	*NMI*	0.519	0.603	**0.604**	0.571	0.401	0.487	0.522	0.520
	MI	0.682	0.795	**0.795**	0.769	0.550	0.658	0.707	0.701
	ARI	0.448	0.529	0.504	**0.531**	0.362	0.417	0.454	0.450
	RI	0.780	**0.818**	0.807	0.814	0.757	0.771	0.786	0.783
	Purity	0.655	**0.751**	0.736	0.700	0.649	0.663	0.686	0.684
Letter	*NMI*	0.433	0.504	**0.504**	0.435	0.265	0.373	0.351	0.347
	MI	0.987	1.135	**1.136**	0.980	0.596	0.841	0.792	0.780
	ARI	0.303	0.330	**0.331**	0.246	0.116	0.193	0.175	0.171
	RI	**0.870**	0.867	0.867	0.853	0.828	0.843	0.841	0.838
	Purity	**0.527**	0.525	0.526	0.460	0.307	0.390	0.369	0.367
USPS	*NMI*	0.793	**0.811**	0.517	0.808	0.806	0.799	0.802	0.805
	MI	1.061	**1.084**	0.695	1.079	1.076	1.062	1.072	1.075
	ARI	0.778	**0.827**	0.523	0.773	0.762	0.743	0.757	0.768
	RI	0.912	**0.931**	0.811	0.909	0.905	0.895	0.903	0.907
	Purity	0.894	**0.918**	0.746	0.883	0.875	0.863	0.874	0.880
Binalpha	*NMI*	**0.693**	0.690	0.683	0.664	0.685	0.652	0.682	0.681
	MI	**1.699**	1.690	1.677	1.631	1.676	1.596	1.668	1.668
	ARI	**0.536**	0.528	0.528	0.523	0.510	0.474	0.502	0.503
	RI	**0.926**	0.925	0.925	0.921	0.922	0.916	0.920	0.921
	Purity	0.683	0.700	**0.705**	0.701	0.651	0.628	0.646	0.649
Optrec	*NMI*	0.537	0.685	0.267	**0.695**	0.720	0.672	0.681	0.666
	MI	0.741	0.941	0.367	**0.958**	0.991	0.919	0.932	0.913
	ARI	0.509	0.662	0.250	**0.689**	0.713	0.645	0.649	0.638
	RI	0.814	0.871	0.715	**0.882**	0.890	0.862	0.865	0.861
	Purity	0.708	0.830	0.567	**0.861**	0.863	0.812	0.827	0.813
MNIST	*NMI*	**0.724**	0.698	0.553	0.627	0.591	0.734	0.594	0.590
	MI	**0.997**	0.957	0.765	0.854	0.803	1.000	0.810	0.806
	ARI	**0.757**	0.698	0.551	0.595	0.526	0.694	0.549	0.549
	RI	**0.908**	0.883	0.831	0.842	0.815	0.880	0.826	0.826
	Purity	**0.897**	0.859	0.717	0.794	0.732	0.847	0.763	0.765

8.1.4 Face recognition

Table 10: Improvement of the various TCS techniques over unsupervised clustering on the task of Face Recognition.

Dataset	*Metrics*	Supervised Learning Method						Unsupervised	
		TCS-VM	**TCS-GM**	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**	**VM**
AT&T	*NMI*	**0.951**	0.830	0.764	0.884	0.823	0.854	0.828	0.841
	MI	**2.478**	2.140	1.965	2.283	2.122	2.201	2.136	2.176
	ARI	**0.878**	0.608	0.513	0.707	0.595	0.643	0.603	0.632
	RI	**0.984**	0.946	0.933	0.959	0.944	0.950	0.945	0.949
	Purity	**0.913**	0.737	0.667	0.811	0.725	0.768	0.733	0.757
Umist	*NMI*	**0.702**	0.649	0.653	0.562	0.534	0.538	0.492	0.507
	MI	**1.441**	1.306	1.316	1.136	1.089	1.091	1.001	1.035
	ARI	**0.561**	0.432	0.439	0.435	0.319	0.314	0.268	0.284
	RI	**0.903**	0.864	0.866	0.854	0.844	0.840	0.831	0.837
	Purity	**0.695**	0.621	0.630	0.633	0.539	0.540	0.502	0.515
Faces96	*NMI*	0.893	0.927	0.928	**0.938**	0.888	0.889	0.884	0.887
	MI	3.311	3.589	3.597	**3.638**	3.442	3.443	3.424	3.446
	ARI	0.618	0.741	0.748	**0.781**	0.663	0.660	0.652	0.668
	RI	0.980	0.989	0.990	**0.991**	0.986	0.986	0.986	0.987
	Purity	0.659	0.803	0.806	**0.836**	0.744	0.745	0.739	0.750
LFW	*NMI*	0.393	**0.409**	0.409	0.314	0.296	0.294	0.285	0.285
	MI	1.468	**1.531**	1.531	1.175	1.104	1.098	1.063	1.065
	ARI	0.131	**0.087**	0.087	0.030	0.026	0.026	0.021	0.021
	RI	0.938	**0.939**	0.939	0.935	0.934	0.934	0.934	0.934
	Purity	0.328	**0.364**	0.364	0.253	0.245	0.247	0.236	0.235

8.1.5 Speech/speaker recognition

For speech (*Isolet*) as well as speaker recognition (*WSJ*).

Table 11: Improvement of the various TCS techniques over unsupervised clustering on the task of Face Recognition.

Dataset	*Metrics*	Supervised Learning Method						Unsupervised	
		TCS-VM	**TCS-GM**	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**	**VM**
Isolet	*NMI*	**0.824**	0.820	0.776	0.819	0.756	0.824	0.812	0.819
	MI	**1.874**	1.853	1.761	1.848	1.714	1.853	1.835	1.849
	ARI	**0.751**	0.689	0.669	0.675	0.629	0.669	0.670	0.676
	RI	**0.954**	0.941	0.937	0.936	0.929	0.934	0.935	0.937
	Purity	**0.844**	0.802	0.792	0.768	0.747	0.769	0.758	0.767

8.1.6 Other datasets

One two of the datasets, the TCS models perform better. On one of them - LMNN performs better.

Table 12: Improvement of the various TCS techniques over unsupervised clustering on the task of Face Recognition.

Dataset	*Metrics*	Supervised Learning Method						Unsupervised	
		TCS-VM	**TCS-GM**	**TCS-GM-L2**	**LMNN**	**PCC**	**BP**	**GM**	**VM**
Image	*NMI*	**0.838**	0.836	0.399	0.495	0.601	0.699	0.780	0.814
	MI	**0.916**	0.915	0.427	0.529	0.640	0.768	0.851	0.891
	ARI	**0.832**	0.839	0.344	0.421	0.533	0.697	0.794	0.827
	RI	**0.924**	0.928	0.700	0.735	0.785	0.865	0.906	0.922
	Purity	**0.933**	0.939	0.715	0.727	0.720	0.878	0.908	0.930
Vowel	*NMI*	0.351	0.375	**0.386**	0.316	0.154	0.308	0.340	0.353
	MI	0.554	0.590	**0.611**	0.487	0.238	0.484	0.528	0.552
	ARI	0.214	0.258	**0.276**	0.195	0.075	0.185	0.203	0.210
	RI	0.736	0.752	**0.761**	0.718	0.678	0.725	0.724	0.731
	Purity	0.435	0.532	**0.562**	0.454	0.336	0.424	0.413	0.421
Leaves	*NMI*	**0.860**	0.838	0.837	0.837	0.769	0.820	0.785	0.787
	MI	**2.976**	2.910	2.906	2.906	2.675	2.833	2.727	2.736
	ARI	**0.656**	0.624	0.622	0.619	0.513	0.582	0.537	0.539
	RI	**0.980**	0.978	0.977	0.977	0.971	0.974	0.972	0.973
	Purity	**0.778**	0.752	0.750	0.747	0.672	0.717	0.684	0.686

9 Unsupervised Clustering Results

In this section, we study the effect of different clustering algorithms on different kinds of data representations. We have four different clustering algorithms,

1. **GM-hard** Hard clustering with Gaussian mixtures. This exactly corresponds the llyod's algorithms for Kmeans clustering.
2. **GM** Soft clustering with Gaussian mixtures. Here we use the EM algorithm for optimization. We also added a single 'variance' parameter common to all classes. The difference between hard and soft is that instances are allowed to belong partially to multiple clusters. After completion, all partial memberships are hardened by thresholding.
3. **VM-hard** Same as GM-hard except using von Mises-Fisher distribution.
4. **VM** Same as GM except using von Mises-Fisher distribution. Instead of the variance parameter, we introduced a single 'concentration' parameter common to all classes.

We have three types of data representation,

1. **Full data** Here we represent the data as it is. No processing.
2. **SVD data** The data is reduced to a lower dimensional space using SVD. We reduced the dimension to 30 for all datasets except faces, where it was 50.
3. **Normalized data** We unit normalize the SVD data.

The results try to answer the following questions,

1. Is there loss in performance using SVD ?
2. Is there any gain in performance using Soft instead of Hard clustering ?
3. What is the effect of normalization ?

9.1 Effect of (a) SVD dimension reduction and (b)Hard vs Soft clustering,

We compare the clustering performance of the full dataset against a dimension reduced dataset (using SVD). Table 13 tabulates the results of GM-hard and GM on the full and SVD dimension reduced datasets. Note that all results are averaged over 10 different runs with Kmeans++ initialization.

1. **Full vs SVD:** On most datasets, there is no almost no appreciable change in performance. In fact on some datasets - TDT4, AT & T, LFW there was a gain in performance by using SVD. The datasets on which we lost performance due to SVD was - Faces96 and TDT5 (I did not try increasing number of dimensions).
2. **Hard vs Soft**: On most datasets, soft clustering seemed to perform very mildly better. Although not tabulated, the time taken for soft clustering is about 1.5x times the time for hard clustering. Note that all soft-assignments are finally *hardened* (by taking the maximum) for evaluation purposes.

9.2 Effect of Normalization

Here we report (Table 14) the results of GM, VM on the normalized dataset. We also added the performance of GM on the non-normalized data for comparison (note that VM models are applicable only on normalized data). All clustering algorithms were intialized using Kmeans++ using the appropriate distance metric. All results are averaged over 10 runs.

1. **VM vs GM** On an average, VM seems to be mildly better than GM.
2. **Effect of Normalization** On certain kinds of data, normalization helps - time-series and text. On other kinds of data like Images or handwriting recognition, normalization hurts.

Table 13: Tabulates the results of two comparisons - (a) Full dataset vs SVD based dimension reduced dataset (b) Hard clustering vs Soft clustering (both using Gaussian mixtures).

Dataset	***Metric***	Full GM-hard	Full GM	SVD GM-hard	SVD GM	**Dataset**	***Metric***	Full GM-hard	Full GM	SVD GM-hard	SVD GM
Leaves	*NMI*	0.784	0.787	0.784	0.787	**Penbased**	*NMI*	0.680	0.684	0.680	0.684
	MI	3.543	3.558	3.543	3.558		*MI*	1.535	1.544	1.535	1.544
	ARI	0.430	0.438	0.430	0.438		*ARI*	0.547	0.550	0.547	0.550
	RI	0.987	0.988	0.987	0.988		*RI*	0.911	0.912	0.911	0.912
	Purity	0.589	0.592	0.589	0.592		*Purity*	0.724	0.728	0.724	0.728
Aussign	*NMI*	0.771	0.774	0.772	0.774	**Letter**	*NMI*	0.364	0.364	0.365	0.364
	MI	3.425	3.436	3.428	3.444		*MI*	1.168	1.168	1.168	1.168
	ARI	0.395	0.400	0.397	0.408		*ARI*	0.134	0.133	0.134	0.133
	RI	0.984	0.985	0.984	0.985		*RI*	0.930	0.930	0.930	0.930
	Purity	0.542	0.547	0.540	0.545		*Purity*	0.284	0.281	0.284	0.281
Char	*NMI*	0.763	0.760	0.754	0.759	**USPS**	*NMI*	0.800	0.800	0.792	0.793
	MI	2.246	2.239	2.223	2.239		*MI*	1.812	1.812	1.798	1.800
	ARI	0.597	0.593	0.587	0.600		*ARI*	0.740	0.740	0.732	0.732
	RI	0.958	0.958	0.957	0.959		*RI*	0.950	0.950	0.949	0.949
	Purity	0.724	0.720	0.714	0.724		*Purity*	0.858	0.858	0.853	0.853
DSPA	*NMI*	0.678	0.678	0.672	0.695	**Binalpha**	*NMI*	0.669	0.668	0.684	0.690
	MI	1.860	1.861	1.838	1.912		*MI*	2.369	2.366	2.433	2.459
	ARI	0.297	0.298	0.282	0.320		*ARI*	0.378	0.378	0.410	0.422
	RI	0.903	0.903	0.898	0.907		*RI*	0.965	0.964	0.967	0.968
	Purity	0.543	0.543	0.529	0.556		*Purity*	0.531	0.528	0.562	0.579
Libras	*NMI*	0.576	0.578	0.573	0.581	**Optrec**	*NMI*	0.776	0.777	0.746	0.753
	MI	1.528	1.530	1.519	1.537		*MI*	1.771	1.773	1.701	1.718
	ARI	0.293	0.294	0.290	0.297		*ARI*	0.712	0.713	0.676	0.685
	RI	0.906	0.905	0.905	0.906		*RI*	0.946	0.946	0.940	0.941
	Purity	0.459	0.456	0.458	0.458		*Purity*	0.835	0.836	0.805	0.814
CNAE	*NMI*	0.404	0.389	0.404	0.389	**AT & T**	*NMI*	0.724	0.724	0.806	0.810
	MI	0.776	0.752	0.776	0.752		*MI*	2.611	2.612	2.932	2.947
	ARI	0.218	0.206	0.218	0.206		*ARI*	0.357	0.357	0.499	0.504
	RI	0.766	0.763	0.766	0.763		*RI*	0.966	0.966	0.975	0.976
	Purity	0.436	0.429	0.436	0.429		*Purity*	0.564	0.564	0.673	0.678
K9	*NMI*	0.541	0.542	0.544	0.548	**Umist**	*NMI*	0.646	0.646	0.643	0.647
	MI	1.408	1.412	1.353	1.364		*MI*	1.883	1.883	1.874	1.886
	ARI	0.330	0.331	0.265	0.268		*ARI*	0.323	0.323	0.324	0.326
	RI	0.880	0.880	0.842	0.844		*RI*	0.928	0.928	0.928	0.928
	Purity	0.605	0.606	0.589	0.592		*Purity*	0.497	0.497	0.505	0.506
TDT4	*NMI*	0.854	0.854	0.887	0.885	**Faces96**	*NMI*	0.873	0.873	0.840	0.842
	MI	2.713	2.711	2.820	2.815		*MI*	4.300	4.300	4.166	4.173
	ARI	0.647	0.646	0.709	0.704		*ARI*	0.548	0.548	0.492	0.495
	RI	0.966	0.966	0.972	0.972		*RI*	0.993	0.993	0.993	0.993
	Purity	0.867	0.867	0.908	0.907		*Purity*	0.655	0.655	0.616	0.618
TDT5	*NMI*	0.835	0.834	0.796	0.797	**LFW**	*NMI*	0.304	0.304	0.357	0.362
	MI	3.476	3.473	3.376	3.380		*MI*	1.359	1.359	1.698	1.726
	ARI	0.396	0.395	0.296	0.293		*ARI*	0.009	0.009	0.008	0.009
	RI	0.968	0.968	0.965	0.965		*RI*	0.960	0.960	0.968	0.969
	Purity	0.889	0.887	0.856	0.856		*Purity*	0.173	0.173	0.171	0.173

Table 14: Tabulates the results of two comparisons - (a) VM-soft vs GM-soft on unit normalized SVD data and (b) Normalized vs Unnormalized SVD data using GM-soft.

Data type →		SVD with Normalized data		SVD with Unnormalized
Dataset	***Metric***	VM-soft	GM-soft	GM-soft
Image	*NMI*	0.579	0.566	0.566
	MI	1.111	1.078	1.078
	ARI	0.489	0.459	0.459
	RI	0.869	0.859	0.859
	Purity	0.662	0.626	0.626
Vowel	*NMI*	0.343	0.350	0.368
	MI	0.809	0.825	0.871
	ARI	0.144	0.148	0.164
	RI	0.850	0.849	0.856
	Purity	0.324	0.328	0.333
Leaves	*NMI*	0.798	0.795	0.787
	MI	3.633	3.615	3.558
	ARI	0.472	0.464	0.438
	RI	0.989	0.989	0.988
	Purity	0.628	0.618	0.592
Aussign	*NMI*	0.782	0.781	0.774
	MI	3.470	3.472	3.444
	ARI	0.410	0.410	0.408
	RI	0.985	0.985	0.985
	Purity	0.559	0.561	0.545
Char	*NMI*	0.758	0.759	0.759
	MI	2.232	2.237	2.239
	ARI	0.588	0.602	0.600
	RI	0.957	0.959	0.959
	Purity	0.729	0.736	0.724
DSPA	*NMI*	0.709	0.714	0.695
	MI	1.945	1.986	1.912
	ARI	0.320	0.342	0.320
	RI	0.902	0.911	0.907
	Purity	0.574	0.579	0.556
Libras	*NMI*	0.605	0.593	0.581
	MI	1.590	1.562	1.537
	ARI	0.323	0.309	0.297
	RI	0.905	0.905	0.906
	Purity	0.488	0.474	0.458
K9	*NMI*	0.590	0.593	0.548
	MI	1.601	1.609	1.364
	ARI	0.382	0.403	0.268
	RI	0.906	0.911	0.844
	Purity	0.684	0.688	0.592
TDT4	*NMI*	0.907	0.900	0.885
	MI	2.913	2.890	2.815
	ARI	0.719	0.703	0.704
	RI	0.974	0.973	0.972
	Purity	0.935	0.925	0.907
TDT5	*NMI*	0.848	0.840	0.797
	MI	3.554	3.534	3.380
	ARI	0.410	0.386	0.293
	RI	0.970	0.969	0.965
	Purity	0.901	0.896	0.856

Data type →		SVD with Normalized data		SVD with Unnormalized
Dataset	***Metric***	VM-soft	GM-soft	GM-soft
Penbased	*NMI*	0.678	0.684	0.684
	MI	1.526	1.544	1.544
	ARI	0.536	0.550	0.550
	RI	0.907	0.912	0.912
	Purity	0.718	0.728	0.728
Letter	*NMI*	0.360	0.364	0.364
	MI	1.152	1.168	1.168
	ARI	0.130	0.133	0.133
	RI	0.929	0.930	0.930
	Purity	0.283	0.281	0.281
USPS	*NMI*	0.790	0.777	0.793
	MI	1.792	1.758	1.800
	ARI	0.729	0.709	0.732
	RI	0.948	0.944	0.949
	Purity	0.849	0.828	0.853
Binalpha	*NMI*	0.676	0.679	0.690
	MI	2.404	2.416	2.459
	ARI	0.393	0.400	0.422
	RI	0.966	0.967	0.968
	Purity	0.556	0.556	0.579
Optrec	*NMI*	0.754	0.740	0.753
	MI	1.721	1.685	1.718
	ARI	0.689	0.667	0.685
	RI	0.942	0.937	0.941
	Purity	0.819	0.802	0.814
AT & T	*NMI*	0.818	0.807	0.810
	MI	2.976	2.932	2.947
	ARI	0.517	0.495	0.504
	RI	0.976	0.975	0.976
	Purity	0.681	0.668	0.678
Umist	*NMI*	0.604	0.602	0.647
	MI	1.774	1.772	1.886
	ARI	0.290	0.288	0.326
	RI	0.927	0.928	0.928
	Purity	0.473	0.464	0.506
Faces96	*NMI*	0.845	0.844	0.842
	MI	4.188	4.185	4.173
	ARI	0.496	0.500	0.495
	RI	0.993	0.993	0.993
	Purity	0.624	0.619	0.618
LFW	*NMI*	0.360	0.359	0.362
	MI	1.711	1.708	1.726
	ARI	0.010	0.009	0.009
	RI	0.968	0.968	0.969
	Purity	0.175	0.175	0.173
Isolet	*NMI*	0.716	0.716	0.7213
	MI	2.293	2.292	2.3204
	ARI	0.453	0.462	0.4717
	RI	0.955	0.956	0.9578
	Purity	0.579	0.572	0.5871

References

[1] Kerem Altun and Billur Barshan. Human activity recognition using inertial/magnetic sensor units. In *Human Behavior Understanding*, pages 38–51. Springer, 2010. 10

[2] M. Bilenko, S. Basu, and R.J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the twenty-first international conference on Machine learning*, page 11. ACM, 2004. 14

[3] Charles Mallah, James Cope, and James Orwell. Plant leaf classification using probabilistic integration of shape, texture and margin features, 2013. 13

[4] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009. 14