7-2014

Transformation-based Probabilistic Clustering with Supervision

Siddarth Gopal
Carnegie Mellon University

Yiming Yang
Carnegie Mellon University, yiming@cs.cmu.edu

Published In
Proceedings of the Conference on Uncertainty in Artificial Intelligence, 270-279.

Transformation-based Probabilistic Clustering with Supervision

Siddharth Gopal
Carnegie Mellon University
Pittsburgh, 15213

Yiming Yang
Carnegie Mellon University
Pittsburgh, 15213

Abstract

One of the common problems with clustering is that the generated clusters often do not match user expectations. This paper proposes a novel probabilistic framework that exploits supervised information in a discriminative and transferable manner to generate better clustering of unlabeled data. The supervision is provided by revealing the cluster assignments for *some subset* of the ground truth clusters and is used to learn a transformation of the data such that labeled instances form well-separated clusters with respect to the given clustering objective. This estimated transformation function enables us to fold the remaining unlabeled data into a space where new clusters hopefully match user expectations. While our framework is general, in this paper, we focus on its application to Gaussian and von Mises-Fisher mixture models. Extensive testing on 23 data sets across several application domains revealed substantial improvement in performance over competing methods.

1 INTRODUCTION

Presenting structured and organized views of data to users is crucial for efficient browsing and locating relevant information. Unsupervised clustering techniques have been widely used for discovering latent structures in unlabeled data. Generative clustering models based on Gaussian, von-Mises and Multinomial distributions have emerged as the defacto methods for performing probabilistic clustering.

However, such clustering methods have a fundamental limitation that they do not take into account the user expectations or preferences over different views of data. For example, given a collection of news-stories, one user might prefer to organize them by subject topics such as *Politics, Sports, Business, etc*, while another user might prefer to see the news-stories grouped by regions such as *U.S., India, China, etc*. As another example, in a collection of speech recordings, it is perfectly reasonable to cluster the recordings either by the content of the recording or by the speaker. In both cases, clustering algorithms cannot tell which type of clustering is preferable unless the user's expectation is effectively communicated to the system. This leads to two challenging problems in clustering:

1. How can we effectively inject supervised information to steer the clustering process towards user expectations?
2. If only **some clusters** identified by the user are available as supervision, can the learned user expectations be effectively transferred from the observed clusters to aid in the discovery of **unobserved** (new) clusters in data?

The first problem has been partially addressed by some work in constrained clustering and distance metric learning, although not in sufficient depth (see Section 2). The second problem, to our knowledge, has not been addressed by any work so far. The second problem is particularly important from a practical point of view because realistically users can only label a small set of clusters, whereas the data keeps growing and new clusters are bound to show up. Using a collection of news-stories as a concrete example, if the user identifies three clusters *U.S., India,* and *China* as the supervision, the system should learn that the user is interested in region-based clustering and partition the incoming unlabeled data by regions and discover other new clusters such as *Iraq, Japan, etc*. In other words, we want the system to generalize the learned user expectation by modeling shared properties between the observed and unobserved clusters.

This paper addresses both challenges by proposing a novel probabilistic framework that assumes the existence of an underlying transformation of data (that is common to the observed and unobserved clusters) which reveals the true user expectations on clustering. Using the supervision in the form of labeled instances and true cluster labels for some subset of groundtruth clusters, we estimate such a transformation. Specifically, we define the transformation function $\mathbf{G}$ in a discriminative manner that maximizes the conditional probability of the cluster label given the transformed instance. Once $\mathbf{G}$ is learned, we apply it to un-

labeled data for clustering in the next step. Because the supervised information is propagated by means of a data transformation we call our framework Transformation-based Clustering with Supervision (TCS).

Our framework is flexible and can be applied to any probabilistic clustering model as long as an appropriate transformation function $\mathbf{G}$ can be defined. In this paper, we explore our framework with two widely used probabilistic clustering models which have different clustering objectives, the Gaussian Mixture model (GM) (Bishop, 2006) and the von Mises-Fisher Mixture model (VM) (Banerjee et al., 2006). For GM, we define $\mathbf{G}$ to be a linear transformation whereas for VM, $\mathbf{G}$ is a linear transformation followed by unit normalization. For both the models we develop efficient optimization algorithms to estimate $\mathbf{G}$ and related parameters.

We conducted thorough evaluations and tested our framework on 23 data sets from different application domains such as time-series, text, handwriting recognition, face recognition, etc. We have observed substantial and consistent improvement in performance (in five clustering metrics) over other competing methods.

2 RELATED WORK

There are two primary areas of related work that use supervision to improve clustering of unlabeled data - Probabilistic Constrained Clustering (PCC) and Distance Metric Learning (DML).

PCC methods (Wagstaff et al., 2001; Basu et al., 2002) try to inject supervision using a probabilistic generative model for the instances. The instances $\mathbf{X}$ consists of both the labeled part $\mathbf{X}_L$,the unlabeled part $\mathbf{X}_U$ and the cluster assignments $\mathbf{Z} = \{\mathbf{Z}_U \cup \mathbf{Z}_L\}$. The observed cluster assignments $\mathbf{Z}_L \subset \mathbf{Z}$ is used as supervision, i.e., the constraints. The model parameters $\boldsymbol{\theta}$ and the latent cluster assignments $\mathbf{Z}_U$ are estimated by maximizing the likelihood under the constraints as:

$$\max_{\boldsymbol{\theta}, \mathbf{Z}_U} \quad \log P(\mathbf{X}|\mathbf{Z}_L, \mathbf{Z}_U, \boldsymbol{\theta}) \tag{1}$$

However, when the task is to detect previously unobserved cluster, this optimization reduces to plain clustering. This can be seen by rewriting the objective as a sum of the labeled objective (which is constant) and the unlabeled objective (which is just standard clustering) as:

$$\max_{\boldsymbol{\theta}_L} \log P(\mathbf{X}_L|\mathbf{Z}_L, \boldsymbol{\theta}_L) + \max_{\boldsymbol{\theta}_U, \mathbf{Z}_U} \log P(\mathbf{X}_U|\mathbf{Z}_U, \boldsymbol{\theta}_U)$$

Clearly, there is no *learning* nor transfer of information from the observed clusters to the unobserved clusters. There are other works in PCC where supervision is represented in the form of pairwise constraints instead of cluster labels, i.e. the constraints are defined as whether two instances should be put in the same cluster or not. These methods optimize eq (1) with an additional penalty term if the constraints are not obeyed. The penalty is introduced in the form of priors (Lu and Leen, 2005), (Basu et al., 2006) or explicitly modeled using some loss function (Basu et al., 2004), (Bilenko et al., 2004). Despite the different variations in formulating the constraints, PCC methods (Wagstaff et al., 2001; Basu et al., 2002, 2004, 2006; Lu and Leen, 2005) have the same fundamental limitation, i.e., the supervised information from the *observed* clusters is not used to reshape the discovery of *unobserved* clusters. The clustering of the unlabeled part of the data reduces to standard unsupervised clustering.

A natural extension of PCC that addresses some of its limitations is to use a more *Bayesian* approach of sharing parameters across clusters. For example, one could use a Gaussian mixture model where each cluster k has its own mean parameter θ_k, but all clusters share a common covariance matrix $\boldsymbol{\Sigma}$. Here the covariance matrix serves as the bridge to transfer information from the observed clusters to the unobserved clusters. One can envision more sophisticated models with common hyperpriors, e.g. a Gaussian hyperprior for the means and an Inverse Wishart hyperprior for the covariances. To our knowledge, such Bayesian revisions of PCC have not been studied in the context of discovering new clusters in unlabeled data (which is the focus of this paper). As we will show in our experiments (where we implemented such a Bayesian PCC model as a baseline), this way of sharing information is not as effective as directly fitting for the cluster labels in the transformed space, which we propose in the TCS framework.

In DML (Blitzer et al., 2005; Xing et al., 2002; Goldberger et al., 2004) the objective is to learn a distance metric that respects the supervision which is provided in the form pairwise constraints. More specifically, given a set of labeled clusters, the distance metric is estimated such that it pulls the within-cluster pairs towards each other and pushes the cross-cluster pairs away from each other. DML methods differ from each other in how the loss functions are defined over the pairwise constraints, such as the hinge loss (Blitzer et al., 2005), Euclidean-distance loss (Xing et al., 2002), log loss (Goldberger et al., 2004), etc. DML has been typically used in the context of nearest-neighbor classification but not in the context of discovering *unobserved* clusters. We argue that existing DML methods have two problems w.r.t to discovering unobserved clusters: Firstly, DML optimizes for pairwise distances and is therefore 'unaware' of the clustering criterion (the objective function for clustering) used. This can lead to overfitting, for example, even if the data is optimally clustered, DML would still try to increase inter-cluster distances and decrease intra-cluster distances. Secondly, optimizing for different loss functions (e.g., hinge loss or Euclidean loss) do not necessarily yield a metric that is also optimal for clustering. Explicitly fitting for the cluster-labels that also achieves the optimal clustering objective without resorting to surrogate measures like pairwise constraints is the fundamental difference between our TCS models and other DML methods.

Indirectly related to this paper are a few works in discrim-

Table 1: Likelihood at Local optimum reached by EM vs Likelihood of ground-truth. The Local optimum is better !

Algorithm →	GM	VM
Local optimum	**-18115**	**4.09965e+09**
Groundtruth	-18168	4.09906e+09

inative clustering (Krause et al., 2010), (Xu et al., 2004) and constrained spectral clustering (Rangapuram and Hein, 2012), (Wang and Davidson, 2010), (Lu and Carreira-Perpinán, 2008). These former methods use a discriminative objective like that of SVM for clustering, however, cannot handle supervision. The latter methods incorporate supervision in a spectral clustering framework, but however, cannot scale beyond a few thousands of instances, cannot produce probabilistic cluster memberships and do not *directly* optimize for the class-labels. A thorough discussion of the drawbacks of spectral clustering framework is however beyond the scope of this paper (see (Nadler and Galun, 2007) and reference therein). It is also worth mentioning that some other work like (Joulin et al., 2010), (Finley and Joachims, 2005) reformulate the classification problem as a clustering one, but however cannot discover previously unobserved clusters in data.

3 PROPOSED MODEL (TCS)

Any typical clustering task involves making at the least two assumptions - number of clusters (or the prior parameters if using Bayesian non-parameterics) and the distance measure, both of which determine the type of clusters generated. The distance measure in a probabilistic clustering framework is determined by the choice of the distribution for e.g. Euclidean corresponds to Gaussian, cosine-similarity corresponds to von Mises-Fisher etc. Typically, the user's subjective choice of the probability distribution does not match the ground-truth and the generated clusters do not match user expectations.

To demonstrate this, we ran two popular clustering algorithms - the Gaussian Mixture model (GM) and the von Mises-Fisher mixture model (VM), which use different probability distributions and optimize different likelihoods, on the 20newsgroups dataset [1] with 20 clusters using the EM algorithm. We compared the likelihoods (the clustering objective) of the algorithms under two settings,

1. The likelihood at a local optimum reached by EM [2].
2. The likelihood when the true labels are given, i.e. cluster assignments fixed to the ground-truth.

As table 1 shows, the likelihood obtained at a local optimum is better than groundtruth - the groundtruth is suboptimal ! The clustering algorithm is optimizing for something else other than groundtruth. This means that the user expected clusters can never be recovered. This is a clear case of mismatch between what the user expects and the user specified probability distribution.

To address this issue, we propose to transform the data into another space where the instances are indeed distributed according to user expectations. We use the supervised information to estimate such a transformation i.e. we estimate a transformation function G in a discriminative manner that maximizes the likelihood of observing the given labels (not the data) in the transformed space.

More formally, we are provided supervised training data from K clusters, $\mathcal{S} = \{x_i, t_i\}_{i=1}^{N}$ where $x_i \in \mathcal{X}$, $t_i \in \{1, 2...K\}$ and unlabeled examples $\mathcal{U}$. For convenience define $y_{ik} = I(t_i = k)$. Given a probabilistic generative model using a mixture of K distributions $\{f(x|C_k)\}_{k=1}^{K}$ where C_k denotes the parameters of cluster K , we estimate G as,

$$\arg\max_{G} \log P(\mathbf{Y}|\mathbf{C}^{mle}(G), G(\mathbf{X}))$$

where $\mathbf{C}^{mle}(G) = \arg\max_{\mathbf{C}} P(G(\mathbf{X})|\mathbf{C}, \mathbf{Y})$,

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

Here $\mathbf{C}^{mle}$ denotes the maximum likelihood estimates of the cluster parameters in the transformed space i.e the cluster parameters that best explain the transformed data. G on the other hand is estimated by maximizing the conditional likelihood of the cluster-labels given $\mathbf{C}^{mle}$ i.e. the transformation that best explains the cluster-labels. Together this ensures that we learn G such that the optimal cluster parameters $\mathbf{C}^{mle}$ also optimally fit the cluster-labels $\mathbf{Y}$. The transformation G is then applied to $\mathcal{U}$ thereby folding it into a space where hopefully the user specified distribution $f(x|C)$ matches user expectations.

Unlike typical clustering algorithms, our TCS framework has a *learning* component as well i.e. we learn how to discover unobserved clusters from supervision. Since any learning algorithm has chances of overfitting, we add an additional regularization term. Together,

$$\text{[OPT1]} \quad \max_{G,\mathbf{C}} \quad \log P(\mathbf{Y}|\mathbf{C}(G), G(\mathbf{X})) - \gamma\lambda(G)$$
$$\text{s.t} \quad \frac{\partial \log P(G(\mathbf{X})|\mathbf{C}, \mathbf{Y})}{\partial \mathbf{C}} = 0$$

where γ is the regularization parameter and λ is the regularization function. Note that the second constraint is another way to say $\mathbf{C}$ is the MLE estimate of $G(\mathbf{X})$. In the following subsections, we discuss how to estimate G with two choices for $f(x|C_k)$ - Gaussian and von Mises-Fisher [3].

[1] qwone.com/ jason/20Newsgroups/

[2] The EM algorithm was intialized with the ground-truth cluster assignments

[3] The supplementary material also discusses the extension to Gamma distributions

3.1 TCS WITH GAUSSIAN MIXTURES

We define a simple mixture of K Gaussians with unit variance and cluster means $\{\theta_k\}_{k=1}^K$ over $\mathcal{R}^P$ where

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

The transformation function is defined as $G(x) = Lx$, a linear transformation using matrix $L \in \mathcal{R}^{P \times P}$. The parameters $\{\theta_k\}_{k=1}^K$ and transformation L is estimated by solving OPT1. Note that the constraint in OPT1 can be written in closed form,

$$\text{[OPT1]} \quad \max_{L,\boldsymbol{\theta}} \quad \log P(\mathbf{Y}|\boldsymbol{\theta}, L\mathbf{X}) - \gamma\lambda(L)$$
$$\text{s.t} \quad \theta_k = \frac{1}{n_k} \sum_{i=1}^{N} y_{ik} L x_i \quad \forall k$$

where n_k denotes the number of instances assigned to cluster k. If m_k denotes the mean of instances assigned to cluster k, then OPT1 can be rewritten as,

$$\text{[OPT1]} \quad \max_{L,\boldsymbol{\theta}} \quad \log P(\mathbf{Y}|\boldsymbol{\theta}, L\mathbf{X}) - \gamma\lambda(L)$$
$$\text{s.t} \quad \theta_k = L m_k \quad \forall k$$

The conditional probability of y_{ik} given the transformed instance Lx_i is,

$$P(y_{ik} = 1|\theta_k, Lx_i) = \frac{e^{-\frac{1}{2}(Lx_i - \theta_k)^\top (Lx_i - \theta_k)}}{\sum_{k'=1}^{K} e^{-\frac{1}{2}(Lx_i - L\theta_{k'})^\top (Lx_i - L\theta_{k'})}}$$

Letting $\theta_k = Lm_k$ and defining $d_{ik} = x_i - m_k$,

$$P(y_{ik} = 1|\theta_k, Lx_i) = \frac{e^{-\frac{1}{2} d_{ik} L^\top L d_{ik}}}{\sum_{k'=1}^{K} e^{-d_{ik'} L^\top L d_{ik'}}}$$

By reformulating OPT1[4] as an optimization problem in terms of $A = L^\top L$, we have a convex semidefinite program,

$$\min_{A} \quad F(A) = \gamma\lambda(A) + \sum_{i=1}^{N}\sum_{k=1}^{K} \frac{y_{ik}}{2} d_{ik}^\top A d_{ik} + \sum_{i=1}^{N} log\left(\sum_{k'} e^{-\frac{1}{2} d_{ik'}^\top A d_{ik'}}\right)$$
$$s.t \;\; A \succeq 0 \qquad \text{where } d_{ik} = (x_i - m_k)$$

We tried different choices for $\lambda(A)$,

1. $\|A - I\|^2$ (Frobenius Norm from Identity)
2. $\|A\|_2^2$ (Frobenius Norm from zero)
3. $\|A\|_*$ (Nuclear Norm)
4. $trace(A) - log(det(A))$ (Log-det divergence)
5. $\|A - I\|_1$ (Entrywise-L1 Norm)

[4]Due to the lack of space, we defer the detailed derivations to the supplementary material.

Algorithm 1 Accelerated gradient descent for TCS-GM.

Input: $\{\mathbf{X}, \mathbf{Y}\}$, step-length sequence S_t
Initialize: Define $H_t = I, \beta_t = 1$
while not converged **do**
  $A_t = \mathcal{PSD}(H_t - s_t \nabla F(H_t))$
  $\beta_{t+1} = \frac{1+\sqrt{1+4\beta_t^2}}{2}$
  $H_{t+1} = A_t + \frac{\beta_t - 1}{\beta_t}(A_t - A_{t-1})$
end while
Output: A_t
$\mathcal{PSD}$ denotes projection to the positive semidefinite cone

Different regularizers favor different kinds of linear transformations, for e.g., Nuclear norm (Ma et al., 2011) favours lower rank solutions, Entrywise-L1 norm favours sparser transformations, the log-det regularlizer also prefers lower rank solutions but penalizes sum of log of eigenvalues instead (Davis et al., 2007) etc.

For differentiable regularizers, the optimization can be solved using projected gradient descent where we take a step along the direction of the negative gradient (with the stepsize determined by backtracking line search) and then project the update back into the positive semidefinite cone. We observed that we could significantly improve the speed by using accelerated gradient descent (Beck and Teboulle, 2009) instead (Algorithm 1). For the non-differentiable regularizers we can use projected subgradient instead. These algorithms provably converge to the optimal solution if the step size is appropriately set (Boyd and Vandenberghe, 2004). The gradient of the objective can be succintly written as,

$$\nabla F(A) = \gamma\lambda'(A) + \sum_{i=1}^{N}\sum_{k=1}^{K} (y_{ik} - p_{ik}(A)) d_{ik} d_{ik}^\top$$
$$\text{where } p_{ik}(A) = P(y_{ik}|Lx_i, \theta_k)$$

We found that our customized parallel solver using accelerated projected gradient descent worked much faster than existing SDP solvers. The solution in A recovers L upto any rotation matrix. This is because $A = L^\top L$ can always be rewritten using $A = L^\top (Q^\top Q) L$, for any rotation matrix $Q^{-1} = Q^\top$. However rotating the data corresponds to changing the basis and does not affect the clustering algorithm.

Note that A is very different from the seemingly similar covariance matrix of a Gaussian distribution. Firstly, unlike the covariance matrix, A is not parameter of the distribution and does not make the distribution sum to 1. Secondly, covariance matrices are typically estimated by maximizing $P(\mathbf{X}|\mathbf{Y})$, this includes supervised versions such as linear discriminant analysis (LDA) and Fisher LDA (Hastie et al., 2009). The transformation matrix A on the other hand, is optimized to fit the labels $P(\mathbf{Y}|L\mathbf{X}, \theta)$. Unlike LDA, A does not have a closed form solution.

Algorithm 2 Optimizing $L, \{\mu_k\}_{k=1}^K$ for TCS-VM.

1: **Input:** $\{\mathbf{X}, \mathbf{Y}\}, T$ iterations
2: **Initialize:** L
3: **for** $t = 1, ..T$ **do**
4: **update** $\mu_k = \frac{\sum_{i=1}^{N} y_{ik} n_i}{\| \sum_{i=1}^{N} y_{ik} n_i \|}$ where $n_i = \frac{Lx_i}{\|Lx_i\|}$
5: **update** $L = \arg\min_L$
$\gamma\lambda(L) + \sum_{i=1}^{N}\sum_{k}^{K} y_{ik}\left[\frac{x_i^\top L^\top \mu_k}{\|Lx_i\|} - \log\left(\sum_{j=1}^{K}\exp(\frac{x_i^\top L^\top \mu_j}{\|L^\top x_i\|})\right)\right]$
6: **end for**

3.2 TCS WITH VON-MISES FISHER MIXTURES

The von Mises-Fisher (vMF) distribution defines a probability density over points on a unit-sphere. It is parameterized by mean parameter μ and concentration parameter κ, the former defines the direction of the mean and the latter determines the spread of the probability mass around the mean. The density function over $\mathcal{X} \equiv \{x : \|x\| = 1, x \in \mathcal{R}^P\}$, is given by

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

where $I_\nu(a)$ is the modified bessel function of first kind with order ν and argument a.

As in the Gaussian mixtures case, we consider a mixture of K vMF distributions with mean parameters $\{\mu_k\}_{k=1}^K$ and a single concentration parameter κ. Since the support of vMF is only over the unit-sphere, any transformation function should ensure the transformed space still lies on the unit-sphere. Therefore we define the transformation function G as a linear transformation with matrix $L \in \mathcal{R}^{P \times P}$ with normalization,

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

With this transformation function, the optimization problem OPT1 is a non-convex function in L. Note that the cluster mean parameters $\{\mu_k\}_{k=1}^K$ have a closed form expression for the MLE estimate in terms of L and $\mathbf{X}$. However unlike the GM case, we do not recommend substituting it into the objective of OPT1 as it leads to computationally intensive expressions. We instead resort to an alternating optimization between μ_k's and L as shown in Algorithm 2. The optimization step in line 5 is nonconvex in L and can solved using gradient descent to converge to a locally optimal solution. We found that in practice it worked fine.

4 EXPERIMENTS

4.1 METHODS FOR COMPARISON

We conducted an extensive empirical study of our proposed framework against several competing methods. We tested,

1. **GM** The simple Gaussian Mixture where the data is assumed to be generated from a mixture of K Gaussians 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 using Gaussian mixture model (section 3.1) and $\|A - I\|^2$ regularization, followed by GM on the linearly transformed unlabeled data.
3. **TCS-GM-L2** Our proposed TCS using Gaussian mixture model and plain $\|A\|^2$ regularization, followed by GM on the linearly transformed unlabeled data.
4. **LMNN** (Weinberger and Saul, 2009) One of the most popular local distance metric learning methods that uses hingeloss to push and pull target neighbors. To ensure competitive performance, the number of target neighbors was set to high value - 50. This is followed by GM on the linearly transformed unlabeled data. Note that LMNN is a stronger baseline than other DML methods like Relevant Component Analysis (Shental et al., 2006), Neighborhood Component Analysis (Goldberger et al., 2004), Linear Discriminant Analysis (Fisher, 1936), etc.
5. **PCC** (Bilenko et al., 2004) A popular probabilistic constrained clustering framework. We used a common covariance matrix for all clusters to ensure transfer of information from known to unknown clusters.
6. **Bayesian** We implemented a Bayesian Gaussian mixture model as an additional baseline, where all the cluster means are drawn from a Normal hyperprior and all clusters share a single covariance matrix. We also tried other variants such as cluster-specific covariance matrices with a shared Inverse Wishart hyperprior, but it did not yield any appreciable improvements. We used the familiar constrained EM algorithm (Basu et al., 2002) to derive point estimates for the unknown parameters.
7. **CSP** (Wang and Davidson, 2010) A representative spectral clustering method that can handle supervision in the form of constraints. We use the author recommended method of representing the unlabeled instances with top k (where k is the number of clusters) dimensions generated by the algorithm followed by GM.

We also tested the vMF-based models but due to lack of space we discuss only a part of the results in Section 5.2.1. We defer the complete set of vMF-based results to the supplementary material.

4.2 DATASETS

We tested our framework on 23 datasets (Table 2) from various application domains including timeseries, text, speech, images,etc. A thorough description of the datasets and the details of the feature extraction process is given in the supplementary material.

Table 2: A list of datasets used along with their characteristics.

Domain	*Dataset*	#Instances	#Dimension	#Classes
Time-series	*Australian Signs (Aussign)*	2565	352	95
	Character Trajectory (Char)	2858	192	20
	Digital Sports Activity (DSPA)	152	1440	19
	Libras	360	90	15
Text	*CNAE*	1079	856	9
	K9	2340	21839	20
	TDT4	622	8895	34
	TDT5	6355	20733	125
Handwritten Characters	*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
Faces	*AT&T faces*	400	19964	40
	UMIST	575	10304	20
	Faces96	3016	19375	151
	Labeled Faces in Wild (LFW)	29791	4324	158
Speech	*Isolet*	7797	617	26
	Wall Street Journal (WSJ)	34942	25	131
Other datasets	*Image*	2310	18	7
	Vowel	990	10	11
	Leaves	1599	192	100

As in any matrix learning method, for high dimensional datasets, learning (or even storing) a full matrix L is computationally intensive. Previous literature have identified three ways to tackle this issue,

1. Learn a diagonal L instead of full matrix L. This drastically improves the scalability, but at the cost of flexibility in the set of transformations (Weinberger and Saul, 2009).
2. Directly learn a low rank transformation, i.e. $L \in \mathcal{R}^{r \times P}$ where $r \ll P$ instead of a full rank matrix. However, the optimization problem now becomes nonconvex in terms of this low rank L (see (Journée et al., 2010)).
3. Reduce the dimension of the data using Singular Value Decomposition (SVD), followed by learning a full rank matrix on the low dimensional data.

In our experiments, we found that solution (3) worked best. Specifically, dimension reduction using SVD actually **improved** the clustering performance on multiple datasets since it removes the intrinsic noise in the data. This is agreement with several observations in practice (Zha et al., 2001), (Drineas et al., 2004) and in theory (Ding and He, 2004),(Kannan et al., 2004). Therefore, on datasets with more than 200 dimensions, we used SVD to fold the data into a 30 dimensional space. Refer supplementary material (sec 9.1) for detailed experiments on how SVD improves clustering.

4.3 EXPERIMENTAL SETTINGS

For all the experiments, we consider the class-labels associated with the data to be the user expected groundtruth clusters. We randomly partitioned the classes into three sets - training, validation and testing. The methods TCS-GM, TCS-GM-L2 and LMNN use the validation set for tuning the regularization parameter. Note that CSP could not scale on datasets with more than 5000 instances. All the results are reported only on the test-set.

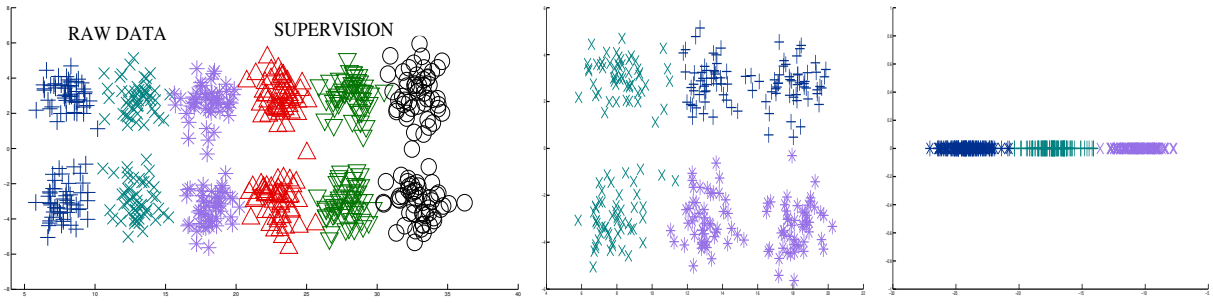
We assume the number of clusters in the unlabeled data is always known, if not, well established techniques like AIC or BIC can be used (finding the right number of clusters is a separate problem that we will not focus on).

We used five clustering metrics for evaluations - Normalized Mutual Information (NMI), Mutual Information (MI), Rand Index (RI), Adjusted Rand Index (ARI) and Purity (refer supplementary material for definitions). All results are averaged over 50 different restarts with Kmeans++ initialization (Arthur and Vassilvitskii, 2007).

5 RESULTS

We present three sets of results that answer the following questions,

1. Does supervision help in clustering better ?
2. By how much do the different methods exploit this supervision ?
3. How does the amount of supervision affect the quality



(a) (b) (c)

Figure 1: (a) Data from of a mixture of six bimodal Gaussians, (b) Clusters generated by GM on raw data. The generated clusters do not match ground-truth. (c) Clusters generated by TCS-GM. The generated clusters match ground-truth

of clusters generated ?

5.1 ANALYSIS ON SYNTHETIC DATA

First, we show how supervision can be helpful using a small synthetic dataset. We generated data from a mixture of six clusters, where each cluster is a bimodal Gaussian distribution. The clusters lie along the x-axis, where as the modes of the Gaussians are stretched out on the Y-axis (Figure 1a). These are the clusters the user expects to see in the data.

Consider the task of clustering the raw data from first 3 clusters - the darkblue, cyan and purple clusters. Without any supervision, using unsupervised GM results in a clustering show in Figure 1b. Clearly there is confusion between clusters 2 and 3 and the user expectations are not met. On the other hand, if we use TCS-GM with clusters 4, 5, 6 as supervision, we learn a transformation that successfully captures user expectations i.e. the transformation simply collapses all points to the X-axis. When we apply this learnt transformation to the raw data and cluster using GM, we can accurately recover the ground-truth (Fig 1c)

5.2 PERFORMANCE ON REAL-WORLD DATA

Due to lack of space, we report the results only using the NMI metric (the supplementary material contains detailed results using all metrics). Table 3 reports the results of the all supervised and unsupervised models. We defer the results of the vMF-based methods to Section 5.2.1. The results show that our proposed TCS-GM and TCS-GM-L2 achieve the best performance in 19 out of the 23 datasets. We now discuss the results from each domain.

In the time-series domain, *Aussign* and *Libras* are sign-language datasets where each instance represents hand-movement over time, *DSPA* is a human activity recognition dataset where each instance is a human performing one of many predefined actions such as walking, running, etc, *Char* is a handwriting recognition dataset where the instances are (x, y) co-ordinates of pen-tip velocity over time. For all these datasets, the instances are represented as a sequence of vectors (sensor measurements) over time and the goal is to cluster the instances by the associated classes i.e. handsign (*Aussign*, *Libras*) or activity (*DSPA*) or character (*Char*). To represent each instance as a fixed dimesional vector, we used fast fourier transform to extract the first several high-frequency components of each feature and concatenated them. In this domain TCS-GM-L2 performs the best by showing a 14.2% average improvement over unsupervised GM, followed by TCS-GM with a 9.6% average improvement.

CNAE, *K9*, *TDT4*, *TDT5* are popular text datasets for classification and clustering (Banerjee et al., 2006). We used the standard bag-of-words with 'ltc' term-weighting [5]. On all datasets, only TCS-GM and CSP show any improvement at all. The rest are mostly negatively impacted by supervision. The improvement of TCS-GM is higher than that of CSP.

In the handwritten characters domain (*Penbased, Letter, USPS, Binalpha, Optrec* and *MNIST*) each instance is an image-representation of a single character and the task is to cluster the instances characterwise. For feature representation, on the latter four datasets, we used histogram of oriented gradients (HOG) (Srikantan et al., 1996) representation with a patchsize of 2 across 9 different orientations. *Penbased* and *Letter* datasets have predefined set of features such as mean pixel value, edgewise mean, etc. Performance wise, TCS-GM and TCS-GM-L2 achieve the best performance, TCS-GM seems to be more suited to HOG-based features whereas TCS-GM-L2 works better with pixel-based statistical features.

In the face clustering tasks (*AT & T, Umist, Faces96* and *LFW*) each instance is an image of a single person's face and the task is to cluster the instances by faces. We represented each instance using HOG features extracted from the image in 9 orientations and varying patchsizes. Both TCS-GM-L2 and LMNN show competitive performance in this task. However, we believe that all datasets except LFW are highly contrived - the images on these other datasets were taken in ideal lighting and posing conditions. The

[5] http://nlp.stanford.edu/IR-book/html/htmledition/document-and-query-weighting-schemes-1.html

Table 3: The NMI of the various supervised and unsupervised methods on unnormalized data. The percentage improvement over the unsupervised GM baseline is given in paranthesis. The results of the significance tests between the best method against the other methods on each dataset is denoted by a † for significance at 1% level. NS denotes the method could not be scaled to the dataset.

Domain	Dataset	Supervised Learning Method						Unsupervised
		TCS-GM	TCS-GM-L2	LMNN	PCC	BP	CSP	GM
Time series	Aussign	0.89† (8.9%)	0.93† (13.2%)	**0.94** (**14.7**%)	0.78† (-5.1%)	0.84† (2.6%)	0.71† (-13.5%)	0.82†
	Char	0.75† (9.5%)	**0.75** (**9.8**%)	0.67† (-2.3%)	0.69† (0.4%)	0.74† (7.3%)	0.59† (-14.5%)	0.69†
	DSPA	0.69† (1.0%)	**0.73** (**8.3**%)	0.60† (-11.9%)	0.67† (-1.5%)	0.64† (-5.3%)	0.71† (4.6%)	0.68†
	Libras	0.61† (18.7%)	**0.64** (**25.4**%)	0.50† (-2.7%)	0.60† (17.0%)	0.59† (15.6%)	0.45† (-11.7%)	0.51†
Avg Improvement		9.5%	**14.2**%	-0.6%	2.7%	5.0%	-8.04%	
Text	CNAE	**0.61** (**25.4**%)	0.24† (-50.7%)	0.31† (-35.3%)	0.43† (-11.1%)	0.48† (-0.8%)	0.59† (21.7%)	0.48†
	K9	**0.58** (**3.6**%)	0.41† (-27.8%)	0.38† (-33.2%)	0.56† (0.2%)	0.51† (-9.1%)	0.58 (3.4%)	0.56†
	TDT4	**0.91** (**0.8**%)	0.69† (-23.3%)	0.90 (0.2%)	0.89† (-0.8%)	0.89† (-0.4%)	0.87† (-2.7%)	0.90†
	TDT5	**0.70** (**0.6**%)	0.69 (0.4%)	0.68† (-2.5%)	0.69† (0.0%)	0.69 (0.3%)	NS	0.69
Avg Improvement		**7.6**%	-25.3%	-17.7%	-2.9%	-2.5%	-	
Handwritten	Penbased	0.56† (6.5%)	**0.60** (**15.7**%)	0.57† (9.4%)	0.40† (-23.2%)	0.49† (-6.7%)	NS	0.52†
Characters	Letter	0.48† (36.2%)	**0.50** (**43.6**%)	0.43† (23.6%)	0.27† (-24.5%)	0.37† (6.3%)	NS	0.35†
	USPS	**0.84** (**3.7**%)	0.46† (-44.2%)	0.79† (-3.7%)	0.81† (-1.0%)	0.79† (-3.6%)	NS	0.81†
	Binalpha	**0.79** (**10.4**%)	0.70† (-2.2%)	0.70† (-2.2%)	0.72† (0.8%)	0.68† (-5.4%)	0.67† (-7.2%)	0.72†
	Optrec	**0.94** (**2.6**%)	0.73† (-20.3%)	0.91† (-0.8%)	0.86† (-5.3%)	0.91† (0.2%)	NS	0.91†
	MNIST	**0.84** (**1.4**%)	0.70† (-15.5%)	0.72† (-13.6%)	0.55† (-33.4%)	0.74† (-10.7%)	NS	0.83†
Avg Improvement		**10.1**%	-3.8%	2.1%	-14.4%	-3.3%	-	
Faces	AT & T	0.84† (1.1%)	**0.88** (**5.4**%)	0.84† (1.0%)	0.82† (-1.4%)	0.85† (2.2%)	0.78† (-6.1%)	0.83†
	Umist	0.59† (6.1%)	0.74† (33.4%)	**0.79** (**43.0**%)	0.57† (2.7%)	0.56† (1.6%)	0.48† (-13.2%)	0.55†
	Faces96	0.92† (4.1%)	0.93† (4.9%)	**0.94** (**6.0**%)	0.89† (0.1%)	0.89† (0.9%)	0.81† (-8.4%)	0.89†
	LFW	0.39† (36.1%)	**0.41** (**45.6**%)	0.33† (16.1%)	0.31† (9.5%)	0.30† (4.2%)	NS	0.28†
Avg Improvement		11.9%	**22.3**%	16.5%	2.7%	2.2%	-	
Speech	Isolet	**0.83** (**2.2**%)	0.80† (-2.5%)	0.81† (-0.5%)	0.75† (-8.5%)	0.83† (1.6%)	NS	0.82†
	WSJ	**0.81** (**46.5**%)	0.81† (46.1%)	0.81† (45.9%)	0.52† (-6.1%)	0.71† (27.4%)	NS	0.56†
Avg Improvement		**24.3**%	21.8%	22.7%	-7.3%	14.5%	-	
Other datasets	Image	**0.84** (**7.2**%)	0.40† (-48.8%)	0.49† (-36.5%)	0.60† (-22.9%)	0.70† (-10.4%)	0.75† (-4.5%)	0.78†
	Vowel	**0.41** (**63.6**%)	0.39† (55.7%)	0.35† (36.8%)	0.14† (-46.6%)	0.23† (-7.1%)	0.22† (-13.2%)	0.25†
	Leaves	0.82† (5.9%)	0.82† (5.7%)	**0.85** (**9.5**%)	0.77† (-1.4%)	0.81† (4.8%)	0.73† (-6.4%)	0.78†
Avg Improvement		**25.6**%	4.2%	3.2%	-23.7%	-4.2%	-4.2%	

LFW dataset, on the other hand, represents a more *realistic* distribution of images as found on the web. On this dataset, TCS-GM-L2 works best with a 45.6% improvement.

In the speech domain, we tested on two datasets *Isolet, WSJ*. In *Isolet* the task is to cluster the instances by the content of the speech (more specifically the letter uttered) and in *WSJ* the task is to cluster by the speaker. The instances are represented using well known audio features such as MFCC's. On both datasets TCS-GM achieves the best performance. The results on *WSJ* particularly highlight the importance of having supervision with a 46% improvement over unsupervised GM.

Image, Vowel and *Leaves* are other popularly used classification datasets. The instances in *Image* and *Leaves* are pictures of outdoor scenes and leaves, whereas in *Vowel* the instances are various utterances of different vowels. On two out of the three datasets, TCS-GM achieves the best results.

To further validate our results, we conducted two-sided significance tests using paired t-test between the best-performing method against the rest of the methods on each dataset. The NMI on the 50 different restarts are considered as observed samples. The null hypothesis is that there is no significance difference between the methods. The results of the testing (Table 3) show that almost all the results are significant.

Table 4: The NMI of TCS-VM with other supervised and unsupervised methods on normalized data. The results of the significance tests between the best method against the other methods on each dataset is denoted by a † for significance at 1% level.

Domain	Dataset	Supervised Learning Method							Unsupervised	
		TCS-VM	TCS-GM	TCS-GM-L2	LMNN	PCC	BP	CSP	VM	GM
Text	CNAE	**0.905**	0.678†	0.678†	0.692†	0.557†	0.462 †	0.588†	0.857†	0.669†
	K9	**0.638**	0.621†	0.443†	0.485†	0.617†	0.584†	0.589†	0.615†	0.616†
	TDT4	**0.933**	0.916†	0.755†	0.929†	0.914†	0.911†	0.870†	0.930†	0.915†
	TDT5	**0.781**	0.750†	0.746†	0.707†	0.750†	0.756†	-	0.766 †	0.755 †

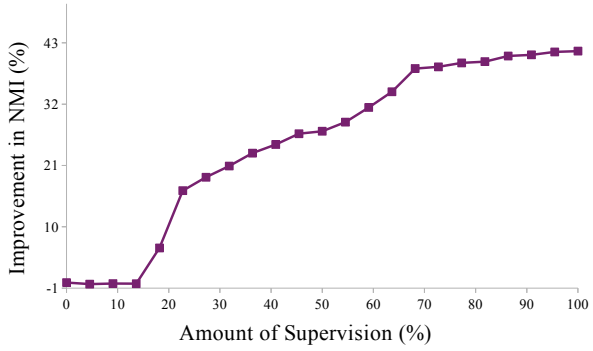


Figure 2: Effect of amount of supervision on the quality of clusters. We plot the improvement in NMI achieved by TSC-GM over baseline GM as we increase the number of training clusters.

5.2.1 PERFORMANCE OF vMF-BASED METHODS

Due to lack of space we present the results of vMF-based models only in the text domain. Normalization to a unit sphere has often shown to work very well for text data, especially vMF-based models have been particularly effective in generating good clusters (Banerjee et al., 2006). We compare the following vMF-based methods on text data,

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 (section 3.2) and $\|A - I\|^2$ regularization, followed by VM on the transformed unlabeled data.

For data representation, we used the bag-of-words with 'ltc' term-weighting and svd-based dimension reduction to 30 dimensions, followed by a normalization to unit-sphere. The results are shown in Table 4. For an informative comparison we also included the results of the other methods - TCS-GM, TCS-GM-L2, LMNN, PCC, BP, CSP and GM. On all text datasets the proposed TCS-VM model performs best. In the complete set of results for all domains for normalized data (presented in the supplementary material), our proposed TCS models achieve the best performance in 20 out of the 23 datasets.

5.3 EFFECT OF AMOUNT OF SUPERVISION

We analyze how the quality of clusters generated depend on the amount of supervision provided. We used a subset of the WSJ dataset with 25 training and 25 testing clusters for this task.

Figure 2 plots the improvement in NMI achieved by TCS-GM over baseline GM as we increase the number of training clusters. Initially there is no improvement in performance, but as training clusters increase, there is a gradual improvement in performance, until it reaches a saturation level. This shows that (a) there is some minimum amount of supervision needed to see any improvement (b) providing more supervision does not increase performance indefinitely but saturates at certain level. This kind of curve is typical of most machine learning algorithms.

6 CONCLUSION

In this paper we proposed a novel framework that can exploit supervision to generate *better* clustering of unlabeled data. By learning a common underlying transformation function, our framework is able to successfully generalize the observed clusters to discover new clusters that match user expectations. We explored two instantiations of our framework with Gaussian and von Mises-Fisher mixture models. For both the models we developed fast optimization alorithms to estimate the model parameters. Our extensive testing on 23 datasets provide strong empirical support in favour of our proposed framework. In future, we would like to adapt our framework to spectral clustering based methods.

7 Acknowledgements

We thank the anonymous reviewers for their helpful suggestions and feedback. This work is supported in part by the National Science Foundation (NSF) under grant IIS 1216282.

References

David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth*

annual ACM-SIAM symposium on Discrete algorithms, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Arindam Banerjee, Inderjit S Dhillon, Joydeep Ghosh, and Suvrit Sra. Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, 6(2): 1345, 2006.

S. Basu, A. Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *ICML*, pages 19–26, 2002.

S. Basu, M. Bilenko, and R.J. Mooney. A probabilistic framework for semi-supervised clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 59–68. ACM, 2004.

S. Basu, M. Bilenko, A. Banerjee, and R.J. Mooney. Probabilistic semi-supervised clustering with constraints. *Semi-supervised learning*, pages 71–98, 2006.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

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.

C.M. Bishop. *Pattern recognition and machine learning*. 2006.

John Blitzer, Kilian Q Weinberger, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *NIPS*, pages 1473–1480, 2005.

Stephen Poythress Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Jason V Davis, Brian Kulis, Prateek Jain, Suvrit Sra, and Inderjit S Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th international conference on Machine learning*, pages 209–216. ACM, 2007.

Chris Ding and Xiaofeng He. K-means clustering via principal component analysis. In *Proceedings of the twenty-first international conference on Machine learning*, page 29. ACM, 2004.

Petros Drineas, Alan Frieze, Ravi Kannan, Santosh Vempala, and V Vinay. Clustering large graphs via the singular value decomposition. *Machine learning*, 56(1-3):9–33, 2004.

T. Finley and T. Joachims. Supervised clustering with support vector machines. In *Proceedings of the 22nd international conference on Machine learning*, pages 217–224. ACM, 2005.

Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

Jacob Goldberger, Sam Roweis, Geoff Hinton, and Ruslan Salakhutdinov. Neighbourhood components analysis. 2004.

Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, and R Tibshirani. *The elements of statistical learning*, volume 2. Springer, 2009.

Armand Joulin, Francis Bach, and Jean Ponce. Discriminative clustering for image co-segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1943–1950. IEEE, 2010.

Michel Journée, Francis Bach, P-A Absil, and Rodolphe Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010.

Ravi Kannan, Santosh Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM (JACM)*, 51(3): 497–515, 2004.

Andreas Krause, Pietro Perona, and Ryan G Gomes. Discriminative clustering by regularized information maximization. In *NIPS*, pages 775–783, 2010.

Z. Lu and T. Leen. Semi-supervised learning with penalized probabilistic clustering. *NIPS*, 17:849–856, 2005.

Zhengdong Lu and Miguel A Carreira-Perpinán. Constrained spectral clustering through affinity propagation. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

Shiqian Ma, Donald Goldfarb, and Lifeng Chen. Fixed point and bregman iterative methods for matrix rank minimization. *Mathematical Programming*, 128(1-2):321–353, 2011.

Boaz Nadler and Meirav Galun. Fundamental limitations of spectral clustering. *NIPS*, 19:1017, 2007.

Syama S Rangapuram and Matthias Hein. Constrained 1-spectral clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 1143–1151, 2012.

Noam Shental, Tomer Hertz, Daphna Weinshall, and Misha Pavel. Adjustment learning and relevant component analysis. In *Computer VisionECCV 2002*, pages 776–790. Springer, 2006.

Geetha Srikantan, Stephen W Lam, and Sargur N Srihari. Gradient-based contour encoding for character recognition. *Pattern Recognition*, 29(7):1147–1160, 1996.

Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schrödl. Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pages 577–584, 2001.

Xiang Wang and Ian Davidson. Flexible constrained spectral clustering. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 563–572. ACM, 2010.

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.

Eric P Xing, Andrew Y Ng, Michael I Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. *NIPS*, 15:505–512, 2002.

Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *NIPS*, pages 1537–1544, 2004.

Hongyuan Zha, Xiaofeng He, Chris Ding, Ming Gu, and Horst D Simon. Spectral relaxation for k-means clustering. In *NIPS*, volume 1, pages 1057–1064, 2001.