# Semi–Supervised Clustering with Metric Learning using Relative Comparisons

Nimit Kumar, Krishna Kummamuru, Deepa Paranjpe
IBM India Research Lab,
Block-1, Indian Institute of Technology,
New Delhi-110016, INDIA.
nimitk, kkummamu, dparanjp@in.ibm.com

## Abstract

*Semi-supervised clustering algorithms partition a given data set using limited supervision from the user. In this paper, we propose a clustering algorithm that uses supervision in terms of relative comparisons, viz.,* $x$ is closer to $y$ than to $z$. *The success of a clustering algorithm also depends on the kind of dissimilarity measure. The proposed clustering algorithm learns the underlying dissimilarity measure while finding compact clusters in the given data set. Through our experimental studies on high-dimensional textual data sets, we demonstrate that the proposed algorithm achieves higher accuracy than the algorithms using pairwise constraints for supervision.*

## 1. Introduction

Semi-supervised clustering algorithms partition a given set of data points using additional supervisory information. The most popular form of supervision used in clustering algorithms is in terms of pairwise constraints viz., whether two points belong to the same cluster or not. The constraints derived from the feedback given in terms of similar data points is called *must-link* and of dissimilar data points *cannot-link* constraints. However, the points in cannot-link constraints may actually lie in wrong clusters and still satisfy the cannot-link constraints. When the pairwise feedback is generated from the labeled part of the training set, the must-link constraints would mislead the clustering algorithm if the points in the constraint belong to two different clusters of the same class.

In this paper, we consider supervision to be available in terms of relative comparisons: $x$ is close to $y$ than to $z$. We refer to the relative comparisons as *triplet constraints*. We propose a clustering algorithm that not only considers the triplet constraints but also simultaneously learns the dissimilarity measure underlying the given data set. We use SVaD measures [8] to model the dissimilarity between data points. The proposed algorithm is called as the *Semi-Supervised SVaD* (SSSVaD, pronounced as "triple SVaD") algorithm. Our experiments show that the proposed algorithm performs better than MPCK-Means algorithm [5, 3], the most recent semi-supervised clustering algorithm that uses pairwise constraints.

In many practical scenarios, large set of unlabeled data and a small set of labeled data are available and supervisory information for semi-supervised clustering is obtained from the small set of labeled data. Once we assume a set of labeled data, relative comparisons can be obtained from any three points from the set if two of them belong to a class different from the class of the third point. One may note that triplet constraints give more information on the underlying dissimilarity measure than the pairwise constraints.

The type of clusters determined by the clustering algorithm is dependent on the assumed dissimilarity measure. Quite a few efforts have been aimed toward learning distance measures for improving the performance of clustering algorithms [8, 5, 3, 9, 6, 12]. These algorithms can be categorized into unsupervised and semi-supervised algorithms.

Most of the semi-supervised clustering algorithms also learn the distance measure. The majority of these algorithms use pairwise feedback. For example, Xing et. al. [12] propose an algorithm to learn distances from the pairs of points that are similar or dissimilar to each other and use the new distance for clustering. In [5, 3], the authors assume pairwise constraints where the pairs of instances belonging to same or different clusters. Whereas, Pedrycz et. al [9], use labels on the subset of training data to learn the underlying metric.

Relative comparisons were first used in [10] to learn distance measures using SVMs. The problem of learning a distance metric from the triplet constraints has been formulated as a quadratic optimization where the triplet constraints would directly become the constraints in the optimization problem.

## 2. Notation

Let $U = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ be the set of unlabeled data. The relative comparisons is assumed to be given in terms of triplets $\tau_i = (\mathbf{x}_1^i, \mathbf{x}_2^i, \mathbf{x}_3^i); i = 1, 2, \ldots, r$ which imply that $\mathbf{x}_1^i$ is *more similar* (or *less dissimilar*) to $\mathbf{x}_2^i$ than to $\mathbf{x}_3^i$ for $i = 1, 2, \ldots, r$. Note that $\mathbf{x}_k^i$, $k = 1, 2, 3$ and $i = 1, 2, \ldots, m$ belong to $U$. Let $\lambda_i$ represent the cluster label of $\mathbf{x}_i$.

SVaD measure represents a general class of dissimilarity measures in which the weights associated with various dimensions vary within the feature space. In this work, we use the SVaD measure as the dissimilarity measure and learn the weights associated. Let $W = \{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$ be the set of $K$ weights corresponding to $K$ regions. And, let $g_1, g_2, \ldots$, and $g_m$ be $m$ dissimilarity measures. Then, the SVaD measure of $\mathbf{x}$ from $\mathbf{y}$ is defined as

$$d_W^C(\mathbf{x}, \mathbf{y}) \equiv \sum_{l=1}^{m} w_{jl} g_l(\mathbf{x}, \mathbf{y}), \text{ if } \mathbf{y} \in R_j.$$

It may be noted that $d_W^C$ is not symmetric. Moreover, weighted Euclidean and weighted cosine distance measures are special cases of SVaD measure. For a detailed discussion on SVaD measures, refer to [8].

## 3. Problem Definition

Given a set of unlabeled samples and triplet constraints, the objective of SSSVaD is to find a partition of the data set along with the parameters of SVaD measure that minimize the within-cluster dissimilarity (the traditional clustering measure) at the same time satisfying as many triplet constraints as possible. We translate the triplet constraints into an appropriate objective function which is minimized along with the within cluster dissimilarity measure. In the following subsection, we derive this objective function.

### 3.1. Relative Comparisons

As stated above, the objective is to minimize the number of unsatisfied triplet constraints. We approximate the number of unsatisfied triplet constraints by the sum of the extent to which the constraints are not satisfied to make the objective function differentiable. Each triplet constraint $\tau_i = (\mathbf{x}_1^i, \mathbf{x}_2^i, \mathbf{x}_3^i)$ implies that

$$d_W^C(\mathbf{x}_3^i, \mathbf{x}_1^i) > d_W^C(\mathbf{x}_2^i, \mathbf{x}_1^i). \tag{1}$$

Let,

$$e_{W,C}(\tau_i) = \begin{cases} 0 \text{ if } d_W^C(\mathbf{x}_3^i, \mathbf{x}_1^i) - d_W^C(\mathbf{x}_2^i, \mathbf{x}_1^i) > 0, \\ 1 \text{ Otherwise.} \end{cases}$$

Then, the task of the learning process includes the minimization of $E$ (which is the number of dissatisfied inequalities) defined as:

$$E(W, C) = \sum_{i=1}^{r} e_{W,C}(\tau_i).$$

Note that $E(W, C)$ represents the number of unsatisfied triplet constraints.

In this paper, we convert the constraints into an appropriate objective function and minimize it along with within cluster dissimilarity measure. Let $u_W^C(\tau^i) = h(d_W^C(\mathbf{x}_3^i, \mathbf{x}_1^i) - d_W^C(\mathbf{x}_2^i, \mathbf{x}_1^i))$, for $i = 1, \ldots, m$ where,

$$h(z) = \begin{cases} -z \text{ if } z < 0, \\ 0 \text{ otherwise.} \end{cases}$$

Note that $u_W^C(\tau^i)$ approximates $e_{W,C}(\tau_i)$. The cumulative error is then defined as in (2).

$$J_T(W, C) = \sum_{i=1}^{m} u_W^C(\tau^i). \tag{2}$$

Let $I(W, C) = \{i : u_W^C(\tau^i) > 0\}$ denote the set of inequalities not satisfying (1). Then,

$$J_T(W, C) = \sum_{i \in I(W,C)} [d_W^C(\mathbf{x}_2^i, \mathbf{x}_1^i) - d_W^C(\mathbf{x}_3^i, \mathbf{x}_1^i)].$$

From the definition of the dissimilarity measure, we get

$$J_T(W, C) = \sum_{i \in I(W,C)} \sum_{l=1}^{m} w_{j_i l}[g_l(\mathbf{x}_2^i, \mathbf{x}_1^i) - g_l(\mathbf{x}_3^i, \mathbf{x}_1^i)],$$

for $\mathbf{x}_1^i \in R_{j_i}$.

### 3.2. Overall Objective Function

Let $J_U(W, C)$ represents the within-cluster dissimilarity measure i.e., the sum of the dissimilarity measures of all the data points from their closest centroids. Then,

$$J_U(W, C) = \sum_{i=1}^{n} \sum_{l=1}^{m} w_{\lambda_i l} g_l(\mathbf{x}_i, \mathbf{c}_{\lambda_i}).$$

where $\lambda_i$ is the cluster label of $\mathbf{x}_i$ determined by

$$\arg\min_j d_W^C(\mathbf{x}_i, \mathbf{c}_j). \tag{6}$$

The overall objective function that captures both the within-cluster dissimilarity and triplet constraints is given by

$$J(W, C) = J_U(W, C) + \gamma_T J_T(W, C), \tag{7}$$

where $\gamma_T$ reflects the relative importance of $J_T(W, C)$. One way to avoid trivial solutions to the above optimization problem is to impose the normalization conditions on $w_{jl}$. The normalization conditions are given by

$$\sum_{l=1}^{m} w_{jl} = 1 \text{ for all } j. \tag{8}$$

Moreover, regularization term is needed to be added to the objective function to avoid weights drifting toward unit vectors. We consider entropy measure for regularization in this paper. Thus, the overall objective function would become:

$$J(W, C) = J_U(W, C) + \gamma_T J_T(W, C)$$

$$+\delta \sum_{j=1}^{K} \sum_{l=1}^{m} w_{jl} log(w_{jl}) + \sum_{j=1}^{K} \alpha_j (\sum_{l=1}^{m} w_{jl} - 1), \tag{10}$$

where $\alpha_j$ are Lagrangian multipliers and $\delta$ represent relative importance given to the regularization term. Ideally, $\delta$ should be specific to each cluster and needs to be estimated from the data as in [7]. We have experimented with this strategy and found that it does not help in improving the performance. Hence, we made it the same for all the clusters.

## 4. The Learning Algorithm

As in the SVaD learning algorithm [8], the overall objective function given in (10) can be optimized using an algorithm similar to $K$-Means Algorithm (KMA) except that in each iteration $w_{jl}$ also need to be updated. The proposed algorithm (SSSVaD) starts with unit weights and random cluster assignments, and updates $C$, $W$ and $\Lambda$ in each iteration, where $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$. It can be shown that SSSVaD is an instance of alternating optimization (AO) algorithms [4]. AO algorithms partition the search dimensions into various sets of dimensions and in each iteration, they optimize the objective function with respect to each set of dimensions keeping the other sets of dimensions constant.

We expand the expression for $J(W, C)$ in (10) to make the following observations easily understandable.

$$J(W, C) = \sum_{i=1}^{n} \sum_{l=1}^{m} w_{\lambda_i l} g_l(\mathbf{x}_i, \mathbf{c}_{\lambda_i})$$

$$+ \sum_{i \in I(W,C)} \sum_{l=1}^{m} w_{j_i l} [g_l(\mathbf{x}_2^i, \mathbf{x}_1^i) - g_l(\mathbf{x}_3^i, \mathbf{x}_1^i)]$$

$$+\sum_{j=1}^{K} \delta_j \sum_{l=1}^{m} w_{jl} log(w_{jl}) + \sum_{j=1}^{K} \alpha_j (\sum_{l=1}^{m} w_{jl} - 1). \tag{13}$$

It may be noted that the update of $\Lambda$ is straight forward. Given $C$ and $W$, the objective function is optimized by assigning the points to the cluster represented by the nearest centroid, viz., as given in (6). Here, the nearest centroid is computed using dissimilarity with the given $C$ and $W$. We explain ways to update $C$, and $W$ in the sequel.

### 4.1. Update of Centroids

The update of centroid depends on the nature of $g_l$. Let $\mathbf{x}, \mathbf{y} \in \Re^m$ and $\mathbf{x} = (x_1, \ldots, x_m)$ and $\mathbf{y} = (y_1, \ldots, y_m)$. For Euclidean SVaD, in which $g_l(\mathbf{x}, \mathbf{y}) = (x_l - y_l)^2$ the centroid update equation is same as that of KMA, i.e.,

$$c_{jl} = \frac{1}{|R_j|} \sum_{\mathbf{x}_i \in R_j} x_{il}$$

In other words, $c_{jl}$ given above optimizes $J(W, C)$ when $W$ and $\Lambda$ are kept constant. Similarly, in case of cosine SVaD, in which $g_l(\mathbf{x}, \mathbf{y}) = (1/m - x_l \cdot y_l)$,

$$c_{jl} = \frac{1}{|R_j|} \sum_{\mathbf{x}_i \in R_j} x'_{il}$$

where, $x'_{il} = w_{jl} x_{il}$. It is not necessary that the update equation for the centroids is in closed form. In fact, existence of such a close form could potentially decide the use of a particular $g_l$. In cases where a particular $g_l$ is more appropriate for a given data set and a closed form update equation does not exists, one can use gradient descent approach to find the optimizing centroids in each iteration.

### 4.2. Update of Weights

Let,

$$V_{jl}^U = \sum_{\mathbf{x}_i : \lambda_i = j} g_l(\mathbf{x}_i, \mathbf{c}_j) \tag{16}$$

and

$$V_{jl}^T = \sum_{\substack{i \in I(W,C) \\ \mathbf{x}_1^i \in R_j}} \left( g_l(\mathbf{x}_2^i, \mathbf{x}_1^i) - g_l(\mathbf{x}_3^i, \mathbf{x}_1^i) \right). \tag{17}$$

Then, differentiating $J(W, C)$ as in (13) with respect to $w_{jl}$ and equating it to zero, we get, after some algebra,

$$w_{jl} = \exp\left( \frac{-(V_{jl}^U + \gamma_T V_{jl}^T + \alpha_j)}{\delta} - 1 \right). \tag{18}$$

Applying the normalizing condition, solving for $\alpha_j$ and substituting the value of $\alpha_j$ in the above equation, we obtain

$$w_{jl} = \frac{\exp\left(-(V_{jl}^U + \gamma_T V_{jl}^T)/\delta\right)}{\sum_{p=1}^{m} \exp\left(-(V_{jp}^U + \gamma_T V_{jp}^T)/\delta\right)}. \tag{19}$$

```
Input:
1: Set of unlabeled points, $U = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$
2: Number of Clusters, $K$
3: Type of distance measure $g_l$
4: Relative qualitative feedback, $\tau_i$,
   $i = 1, \dots, m$
5: Parameters:  $\gamma_T, \eta < 1, \rho(0) < 1$
Output:
  Partitioning of the data set $U$ in terms
  of $\Lambda$, and the parameters $C$ and $W$ that
  minimizes $J(W, C)$
SSSVaD Algorithm:
1: Set t=0
2: Randomly assign the data points to K
   clusters
3: Compute the centroid of each cluster $\mathbf{c}_j$
4: Determine the set of unsatisfied in-
   equalities $I(W, C)$
5: Compute the intermediate weights $\mathbf{w}'_j$ for
   $j = 1, \dots, K$ using (19)
6: Update the weights as $\mathbf{w}_j(t+1) = (1 -$
   $\rho(t))\mathbf{w}_j(t) + \rho(t)\mathbf{w}'_j$ for $j = 1, \dots, K$
7: $\rho(t+1) = \eta\rho(t), t = t + 1$
8: Reassign the data points to the cluster
   with the nearest centroid
9: Go to STEP 3 until the termination cri-
   teria is satisfied
```

**Figure 1. SSSVaD Algorithm**

It may be noted that, in case of Euclidean SVaD measure, $V_{jl}^U$ is equal to the variance of $l$th attribute of data points in the $j$th cluster. According to (19), the weight of $l$th dimension in $j$th cluster, $w_{jl}$ is inversely proportional to the dimension's variance in the cluster. Moreover, $V_{jl}^T$ is equal to the cumulative error along $l$th dimension among all the unsatisfied triplet constraints whose $\mathbf{x}_1$ is in the $j$th cluster. It is quite natural to decrease the weight of the dimension whose contribution to the error is high. This is reflected in (19).

### 4.3. SSSVaD Learning Algorithm

We summarize the algorithm in Figure 1. The SSSVaD algorithm takes $U$, $K$, type of $g_l$, $\tau_i$, $\gamma_T$, $\eta$ and $\rho(0)$ as input. The algorithm starts with random cluster assignments to the data points. And, in each iteration, $C$, $W$ and $\Lambda$ are updated. The only difference of the proposed algorithm with that of AO algorithms is in the way the weights are updated. In AO algorithms, the weights are updated such that they optimize the objective function as in (19). We have experimented with this way of updating the weights and observed that the algorithm either converges to a bad solution or oscillates between two solutions. Hence, we update the weights as explained below. Once the optimizing weight vector $\mathbf{w}'$ for each cluster is computed using (19), the current weight vector is updated such that it is only slightly moved toward $\mathbf{w}'$ instead of making it equal to $\mathbf{w}'$. The amount of movement is controlled by the learning rate $\rho(0)$ and momentum, $\eta$. After a termination condition is reached, typically completion of a fixed number of iterations, the algorithm outputs $\Lambda$, $C$ and $W$.

## 5. Comparison with MPCK-Means Algorithm

In this section, we compare the effect of pairwise constraints and triplet constraints as supervision in clustering. We do this by comparing SSSVaD and MPCK-Means [5], the most recent semi-supervised algorithm using pairwise constraints. In our experiments, we also use MPCK-Means with a random cluster initialization - this version of MPCK-Means is referred to as rMPCK-Means.

Even though it is easier to generate pairwise constraints from a small set of labeled samples, the number of pairwise constraints that can be obtained is quite smaller than the number of triplet constraints. In other words, given a set of labeled samples, triplet constraints capture more information than pairwise constraints.

MPCK-Means uses must-link constraints which represent points belonging to the same cluster. However, if the constraints are generated from the class labels, these constraints could be misleading especially when a particular class has more than one cluster in it. Similarly, cannot-link constraints are not sufficient conditions, because two data points can lie in incorrect clusters and still satisfy the constraints.

It may be noted that computing the weight updates (17) needs only $\tau_{il}, l = 1, \dots, m$ from each triplet $\tau_i$ where $\tau_{il} = g_l(\mathbf{x}_2^i, \mathbf{x}_1^i) - g_l(\mathbf{x}_3^i, \mathbf{x}_1^i)$. Therefore, $\tau_{jl}$ need not be computed in every iteration. Whereas, in MPCK-Means, the weight updates involve more computation. Thus, the weight updates in SSSVaD are faster than that in MPCK-Means.

In SSSVaD, the process of cluster assignments is the same as in K-Means algorithm. However, in case of MPCK-Means, the cluster labels explicitly figure in the objective function involving pairwise constraints. Hence, an iterative algorithm needs to be used to find an optimal cluster assignments for each data point. This makes the MPCK-Means slower.

## 6. Experiments & Comparisons

We performed our experiments on the 20NewsGroup Dataset [1]. Similar to [11], we considered random samples of two subsets of the data set - *Binary*, *Multi10*. *Binary* has 250 documents each from *talk.politics.mideast* and *talk.politics.misc*. *Multi10* has 50 documents each from *alt.atheism*, *comp.sys.mac.hardware*, *misc.forsale*, *rec.autos*, *rec.sport.hockey*, *sci.crypt*, *sci.electronics*, *sci.med*, *sci.space*, and *talk.politics.gun*. The documents are represented using a set of vocabulary terms. Vocabulary sets are generated by stopword removal, stemming and thresholding on the frequency of occurrence of terms. The size of the vocabulary used to represent the documents in Binary data set is about 4000 and Multi10 about 2800. We use normalized term frequency vectors to represent the documents.

We generate the pairwise and triplet constraints using labels of the small percentage of training samples. The triplet constraints are generated by finding ($x_1$,$x_2$,$x_3$) from the labeled subset such that the class labels of $x_1$ and $x_2$ are the same and different from that of $x_3$. The must-link constraint pairs ($x_1$,$x_2$) are generated such that the class labels of $x_1$ and $x_2$ are the same. Similarly, the cannot link constraints ($x_1$,$x_2$) are generated such that the class labels of $x_1$ and $x_2$ are different.

We compare the performance of SSSVaD algorithm against MPCK-Means, rMPCK-Means, SVaD and K-Means Algorithm (KMA). MPCK-Means augments the pairwise constraints using transitive closure and use the augmented set of constraints in initializing the centroids and updating the weights. However, in this paper, we uniformaly initialize SSSVaD, rMPCK-Means and SVaD with the clusters obtained using KMA. The results reported here are obtained by averaging the accuracies over 10 random initializations of KMA. For MPCK-Means, we use the publicly available code at [2].

Throughout the experiments, we use weighted Euclidean distance measure, viz., $g_l(\mathbf{x}, \mathbf{y}) = (x_l - y_l)^2$. Since we use the normalized frequency vector, the Euclidean measure is similar to the cosine measure.

## 6.1. Results

We compare the performance of the algorithms varying the number of possible partitions (clusters) and the amount of supervision (percentage of labeled samples). Figures 2 and 3 compare the performance of the five algorithms on Binary, Multi5 and Multi10 data sets respectively. In most cases, we observe a noticeable improvement in the clustering accuracies using SSSVaD algorithm over other algorithms.

**Effect of number of clusters:** Part (a) of the Figures 2 and 3 show the performance of different algorithms with increasing number of clusters. The optimal number of clusters for a given problem depends on the data distribution. It can be seen that for *Binary1* (Figure 2), which has two highly overlapping classes, the accuracy attains a maximum and then falls with increasing number of clusters. However, for *Multi10* data set, the performance improves as the number of clusters increases. *Multi10* is a typical example of the problems that are difficult to cluster because of the large number of classes. MPCK-Means performs best only when the number of classes is identical to the number of clusters and its accuracy falls as the number of clusters increases. This is expected, as the pairwise constraints are often misleading and confusing when the number of clusters is greater than the number of classes.

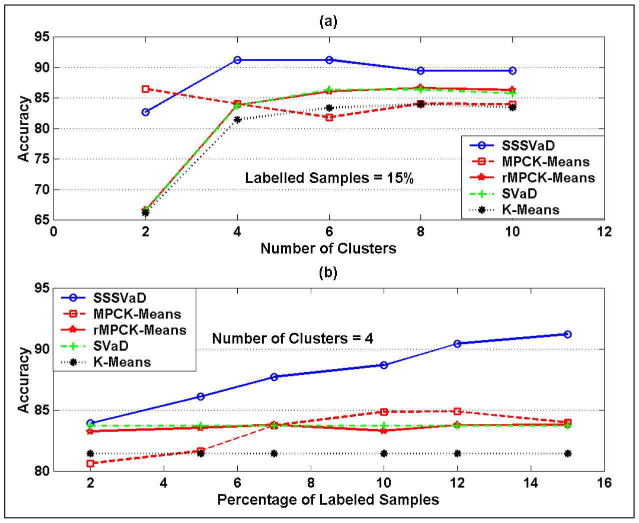

**Figure 2. Performance of SSSVaD, MPCK-Means, rMPCK-Means, SVaD and K-Means on *Binary* data set with varying (a) number of clusters and (b) percentage of labeled samples.**

**Effect of the amount of supervision:** Part (b) of the Figures 2 and 3 show the performance of the different algorithms as the size of the labeled data increases. Note that the accuracies of KMA and SVaD do not change with the size of the labeled samples. It can be observed that with increasing supervision, the performance of SSSVaD improves significantly as compared to that using MPCK-Means in the case of *Binary* data set. In case of Multi10, MPCK-Means performs slightly better than SSSVaD at 15% of labeled samples.

**MPCK-Means and rMPCK-Means:** Note that rMPCK-Means, which uses KMA for initialization, could not show an improvement in accuracies with increasing number of labeled samples. Thus, the real success of the MPCK-Means algorithm seems to be through the deterministic cluster initialization which considers the augmented closure of all the constraints.

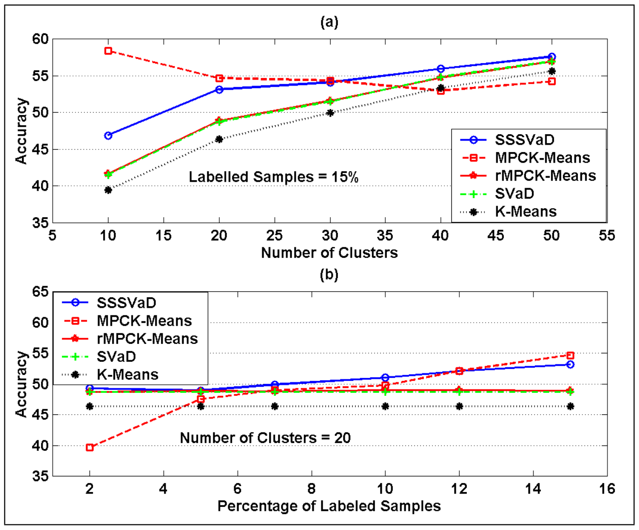

**Figure 3. Performance of SSSVaD, MPCK-Means, rMPCK-Means, SVaD and K-Means on *Multi10* data set with varying (a) number of clusters and (b) percentage of labeled samples.**

## 7. Summary and Conclusions

In this work, an attempt has been made to use limited supervision for obtaining improved clustering performance. The SSSVaD algorithm uses a generalized dissimilarity measure (SVaD) and supervision in the form of relative comparisons. The use of SVaD helped to identify compact partitions in the data set by learning the inherent metric. Also, the joint objective function for SSSVaD helps in obtaining a closed form solution for the parameter updates. This makes the implementation of the algorithm much more simpler. The overall objective function was compactly designed so that these triplet constraints could be used effectively. The proposed algorithm was compared with a similar semi-supervised algorithm, MPCK-Means, that uses pairwise constraints. The efficiency of relative comparisons over pairwise constraints was established through exhaustive experimentation. The results demonstrate both preciseness and robustness of the proposed SSSVaD algorithm.

We have used KMA to initialize the cluster centers in all the algorithms. It is shown in pairwise constraint case that initialization using the augmented constraints would improve the performance. We would like to explore the ways to initialize the centroids using triplet information. We would also like to investigate the performance of SSSVaD on non-textual datasets.

## References

[1] *http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/news20.tar.gz*.
[2] *http://www.cs.utexas.edu/users/ml/risc/code/*.
[3] S. Basu, M. Bilenko, and R. Mooney. A probabilistic framework for semi–supervised clustering. In *Proceedings of SIGKDD*, pages 59–68, 2004.
[4] J. C. Bezdek and R. J. Hathaway. Some notes on alternating optimization. In *Proceedings of the 2002 AFSS International Conference on Fuzzy Systems. Calcutta*, pages 288–300. Springer-Verlag, 2002.
[5] M. Bilenko, S. Basu, and R. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of ICML*, pages 81–88, 2004.
[6] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. *Technical Report TR2003-1892, Cornell University*, 2003.
[7] H. Frigui and O. Nasraoui. Simultaneous categorization of text documents and identification of cluster-dependent keywords. In *Proceedings of FUZZIEEE*, pages 158–163, Honolulu, Hawaii, 2001.
[8] K. Kummamuru, R. Krishnapuram, and R. Agrawal. Learning spatially variant dissimilarity (svad) measures. In *Proceedings of SIGKDD*, pages 611–616, 2004.
[9] W. Pedrycz and J. Waletzky. Fuzzy clustering with partial supervision. *IEEE Transaction on SMC - Part B: Cybernetics*, 27(5), Oct 1997.
[10] M. Schultz and T. Joachims. Learning a distance metric with relative comparisons. *Advances in Neural Information Processing Systems*, 16, 2003.
[11] N. Slonim and N. Tishby. Document clustering using word clusters via the information bottleneck method. In *Proceedings of SIGIR*, page 208215, 2000.
[12] E. Xing, A. Ng, M. Jordon, and S. Russell. Distance metric learning, with application to clustering with side-information. *Advances in Neural Information Processing Systems*, 16:505–512, 2003.