Learning Concepts from Large Scale Imbalanced Data Sets Using Support Cluster Machines[*]

Jinhui Yuan, Jianmin Li and Bo Zhang
State Key Laboratory of Intelligent Technology and Systems
Department of Computer Science and Technology
Tsinghua University, Beijing, 100084, P. R. China
yuan-jh03@mails.tsinghua.edu.cn
{lijianmin, dcszb}@mail.tsinghua.edu.cn

ABSTRACT

This paper considers the problem of using Support Vector Machines (SVMs) to learn concepts from large scale imbalanced data sets. The objective of this paper is twofold. Firstly, we investigate the effects of large scale and imbalance on SVMs. We highlight the role of linear non-separability in this problem. Secondly, we develop a both practical and theoretical guaranteed meta-algorithm to handle the trouble of scale and imbalance. The approach is named Support Cluster Machines (SCMs). It incorporates the *informative* and the *representative* under-sampling mechanisms to speedup the training procedure. The SCMs differs from the previous similar ideas in two ways, (a) the theoretical foundation has been provided, and (b) the clustering is performed in the feature space rather than in the input space. The theoretical analysis not only provides justification, but also guides the technical choices of the proposed approach. Finally, experiments on both the synthetic and the TRECVID data are carried out. The results support the previous analysis and show that the SCMs are efficient and effective while dealing with large scale imbalanced data sets.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Abstracting methods,Indexing methods*; I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*

General Terms

Algorithms, Theory, Experimentation

[*]Supported by National Natural Science Foundation of China (60135010, 60321002) and Chinese National Key Foundation Research & Development Plan (2004CB318108).

Keywords

Support Vector Machines, Kernel k-means, Clustering, Concept Modelling, Large Scale, Imbalance

1. INTRODUCTION

In the context of concept modelling, this paper considers the problem of how to make full use of the large scale annotated data sets. In particular, we study the behaviors of Support Vector Machines (SVMs) on large scale imbalanced data sets, not only because its solid theoretical foundations but also for its empirical success in various applications.

1.1 Motivation

Bridging the semantic gap has been becoming the most challenging problem of Multimedia Information Retrieval (MIR). Currently, there are mainly two types of methods to bridge the gap [8]. The first one is relevance feedback which attempts to capture the user's precise needs through iterative feedback and query refinement. Another promising direction is concept modelling. As noted by Hauptmann [14], *this splits the semantic gap between low level features and user information needs into two, hopefully smaller gaps: (a) mapping the low-level features into the intermediate semantic concepts and (b) mapping these concepts into user needs.* The automated image annotation methods for CBIR and the high level feature extraction methods in CBVR are all the efforts to model the first mapping. Of these methods, supervised learning is one of the most successful ones. An early difficulty of supervised learning is the lack of annotated training data. Currently, however, it seems no longer a problem. This is due to both the techniques developed to leverage surrounding texts of web images and the large scale collaborative annotation. Actually, there is an underway effort named Large Scale Concept Ontology for Multimedia Understanding (LSCOM), which intends to annotate 1000 concepts in broadcast news video [13]. The initial fruits of this effort have been harvested in the practice of TRECVID hosted by National Institute of Standards and Technology (NIST) [1]. In TRECVID 2005, 39 concepts are annotated by multiple participants through web collaboration, and ten of them are used in the evaluation.

The available large amount of annotated data is undoubtedly beneficial to supervised learning. However, it also brings out a novel challenge, that is, how to make full use of the data while training the classifiers. On the one hand, the annotated data sets are usually in rather large scale. The development set of TRECVID 2005 includes 74523 keyframes. The data set of LSCOM with over 1000 annotated concepts might be even larger. With all the data, the training of SVMs will be rather slow. On the other hand, each concept will be the minority class under one-against-all strategy. Only a small portion of the data belong to the concept, while all the others are not (In our case, the minority class always refers to the positive class). The ratio of the positive examples and the negative ones is typically below 1 : 100 in TRECVID data. These novel challenges have spurred great interest in the communities of data mining and machine learning[2, 6, 21, 22, 29]. Our first motivation is to investigate the effects of large scale and imbalance on SVMs. This is critical for correct technical choices and development. The second objective of this paper is to provide a practical as well as theoretical guaranteed approach to addressing the problem.

1.2 Our Results

The major contribution of this paper can be summarized as follows:

1. We investigate the effects of large scale and imbalance on SVMs and highlight the role of linear nonseparability of the data sets. We find that SVMs has no difficulties with linear separable large scale imbalanced data.

2. We establish the relations between the SVMs trained on the centroids of the clusters and the SVMs obtained on the original data set. We show that the difference between their optimal solutions are bounded by the perturbation of the kernel matrix. We also prove the optimal criteria for approximating the original optimal solutions.

3. We develop a meta-algorithm named Support Cluster Machines (SCMs). A fast kernel k-means approach has been employed to partition the data in the feature space rather than in the input space.

Experiments on both the synthetic data and the TRECVID data are carried out. The results support the previous analysis and show that the SCMs are efficient and effective while dealing with large scale imbalanced data sets.

1.3 Organization

The structure of this paper is as follows. In Section 2 we give a brief review of SVMs and kernel k-means. We discuss the effects of the large scale imbalanced data on SVMs in Section 3. We develop the theoretical foundations and present the detailed SCMs approach in Section 4. In Section 5 we carry out experiments on both the synthetic and the TRECVID data sets. Finally, we conclude the paper in Section 6.

2. PRELIMINARIES

2.1 Support Vector Machines

Here, we present a sketch introduction to the soft-margin SVMs for the convenience of the deduction in Section 4. For a binary classification problem, given a training data set $\mathcal{D}$ of size n

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathcal{R}^N, y_i \in \{1, -1\}\},$$

where $\mathbf{x}_i$ indicates the training vector of the ith sample and y_i indicates its target value, and $i = 1, \ldots, n$. The classification hyperplane is defined as

$$\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b = 0,$$

where $\Phi(\cdot)$ is a mapping from $\mathcal{R}^N$ to a (usually) higher dimension Hilbert space $\mathcal{H}$, and $\langle \cdot, \cdot \rangle$ denotes the dot product in $\mathcal{H}$. Thus, the decision function $f(\mathbf{x})$ is

$$f(\mathbf{x}) = sign(\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b).$$

The SVMs aims to find the hyperplane with the maximum margin between the two classes, i.e., the optimal hyperplane. This can be obtained by solving the following quadratic optimization problem

$$\begin{aligned} \min_{\mathbf{w}, b, \xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \\ \text{subject to} \quad & y_i(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle + b) \ \geq 1 - \xi_i \\ & \xi_i \geq 0, \forall i = 1, \ldots, n. \end{aligned} \tag{1}$$

With the help of Lagrange multipliers, the dual of the above problem is

$$\begin{aligned} \min_{\alpha} \quad & G(\alpha) = \frac{1}{2}\alpha^T \mathbf{Q} \alpha - \mathbf{e}^T \alpha \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \forall i = 1, \ldots, n \\ & \alpha^T \mathbf{y} = 0, \end{aligned} \tag{2}$$

where α is a vector with components α_i that are the Lagrange multipliers, C is the upper bound, $\mathbf{e}$ is a vector of all ones, and $\mathbf{Q}$ is an $n \times n$ positive semi-definite matrix, $Q_{ij} = y_i y_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. Since the mapping $\Phi(\cdot)$ only appears in the dot product, therefore, we need not know its explicit form. Instead, we define a kernel $K(\cdot, \cdot)$ to calculate the dot product, i.e., $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. The matrix $\mathbf{K}$ with components $K(\mathbf{x}_i, \mathbf{x}_j)$ is named Gram Matrix (or kernel matrix). With kernel $K\langle \cdot, \cdot \rangle$, we can implicitly map the training data from *input space* to a *feature space* $\mathcal{H}$.

2.2 Kernel k-means and Graph Partitioning

Given a set of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$, the standard k-means algorithm aims to find clusters $\pi_1, \ldots, \pi_k$ that minimize the objective function

$$J(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\mathbf{x}_i - \mathbf{m}_c\|^2, \tag{3}$$

where $\{\pi_c\}_{c=1}^k$ denotes the partitioning of the data set and $\mathbf{m}_c = \frac{\sum_{\mathbf{x}_i \in \pi_c} \mathbf{x}_i}{|\pi_c|}$ is the centroid of the cluster π_c. Similar to the idea of nonlinear SVMs, the k-means can also be performed in the feature space with the help of a nonlinear mapping $\Phi(\cdot)$, which results in the so-called kernel k-means

$$J(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^{k} \sum_{\mathbf{x}_i \in \pi_c} \|\Phi(\mathbf{x}_i) - \mathbf{m}_c\|^2, \tag{4}$$

where $\mathbf{m}_c = \frac{\sum_{\mathbf{x}_i \in \pi_c} \Phi(\mathbf{x}_i)}{|\pi_c|}$. If we expand the Euclidean distance $\|\Phi(\mathbf{x}_i) - \mathbf{m}_c\|^2$ in the objective function, we can find that the image of $\mathbf{x}_i$ only appears in the form of dot product. Thus, given a kernel matrix $\mathbf{K}$ with the same meaning in SVMs, we can compute the distance between points and centroids without knowing explicit representation of $\Phi(\mathbf{x}_i)$.

Recently, an appealing alternative, i.e., the graph clustering has attracted great interest. It treats clustering as a graph partition problem. Given a graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, which consists of a set of vertices $\mathcal{V}$ and a set of edges $\mathcal{E}$ such that an edge between two vertices represents their similarity. The affinity matrix $\mathbf{A}$ is $|\mathcal{V}| \times |\mathcal{V}|$ whose entries represent the weights of the edges. Let $links(\mathcal{V}_1, \mathcal{V}_2)$ be the sum of the edge weights between the nodes in $\mathcal{V}_1$ and $\mathcal{V}_2$, that is

$$links(\mathcal{V}_1, \mathcal{V}_2) = \sum_{i \in \mathcal{V}_1, j \in \mathcal{V}_2} A_{ij}.$$

Ratio association is a type of graph partitioning objective which aims to maximize within-cluster association relative to the size of the cluster

$$RAssoc(G) = \max_{\mathcal{V}_1, \ldots, \mathcal{V}_k} \sum_{c=1}^{k} \frac{links(\mathcal{V}_c, \mathcal{V}_c)}{|\mathcal{V}_c|}. \quad (5)$$

The following theorem establishes the relation between kernel k-means and graph clustering [10]. With this result, we can develop some techniques to handle the difficulty of storing the large kernel matrix for kernel k-means.

Theorem 1. *Given a data set, we can construct a weighted graph $G = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, by treating each sample as a node and linking an edge between each other. If we define the edge weight $A_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$, that is, $\mathbf{A} = \mathbf{K}$, the minimization of (4) is equivalent to the maximization of (5).*

3. THE EFFECTS OF LARGE SCALE IMBALANCED DATA ON SVMS

3.1 The Large Scale Data

There are two obstacles yielded by large scale. The first one is the kernel evaluation, which has been intensively discussed in the previous work. The computational cost scales quadratically with the data size. Furthermore, it is impossible to store the whole kernel matrix in the memory for common computers. The decomposition algorithms (e.g., SMO) have been developed to solve the problem [20, 22]. The SMO-like algorithms actually transform the space load to the time cost, i.e., numerous iterations until convergence. To reduce or avoid the kernel reevaluations, various efficient caching techniques are also proposed [16]. Another obstacle caused by large scale is the increased classification difficulty, that is, the more probable data overlapping. We can not prove it is inevitable but it typically happens. Assume we will draw n randomly chosen numbers between 1 to 100 from a uniform distribution, our chances of drawing a number close to 100 would improve with increasing values of n, even though the expected mean of the draws is invariant [2]. The checkerboard experiment in [29] is an intuitive example. This is true especially for the real world data, either because of the weak features (we mean features that are less discriminative) or because of the noises. With the large scale data, the samples in the overlapping area might be so many that the samples violating KKT conditions become abundant. This means the SMO algorithm might need more iterations to converge.

Generally, the existing algorithmic approaches have not been able to tackle the very large data set. Whereas, the under-sampling method, e.g., active learning, is possible. With unlabelled data, active learning selects a well-chosen subset of data to label so that reduce the labor of manual annotations [24]. With large scale labelled data, active learning can also be used to reduce the scale of training data [21]. The key issue of active learning is how to choose the most "valuable" samples. The *informative* sampling is a popular criterion. That is, the samples closest to the boundary or maximally violating the KKT conditions (the misclassified samples) are preferred [24, 26]. Active learning is usually in an iterative style. It requires an initial (usually random selected) data set to obtain the estimation of the boundary. The samples selected in the following iterations depend on this initial boundary. In addition, active learning can not work like the decomposition approach which stops until all the samples satisfy the KKT conditions. This imply a potential danger, that is, if the initial data are selected improperly, the algorithm might not be able to find the suitable hyperplane. Thus, another criterion, i.e., *representative*, must be considered. Here, "representative" refers to the ability to characterize the data distribution. Nguyen *et al.* [19] show that the active learning method considering the *representative* criterion will achieve better results. Specifically for SVMs, pre-clustering is proposed to estimate the data distribution before the under-sampling [31, 3, 30]. Similar ideas of *representative* sampling appear in [5, 12].

3.2 The Imbalanced Data

The reason why general machine learning systems suffer performance loss with imbalanced data is not yet clear [23, 28], but the analysis on SVMs seems relatively straightforward. Akbani *et al.* have summarized three possible causes for SVMs [2]. They are, *(a) positive samples lie further from the ideal boundary, (b) the weakness of the soft-margin SVMs, and (c) the imbalanced support vector ratio.* Of these causes, in our opinion, what really matters is the second one. The first cause is pointed out by Wu *et al.* [29]. This situation occurs when the data are linearly separable and the imbalance is caused by the insufficient sampling of the minority class. Only in this case does the "ideal" boundary make sense. As for the third cause, Akbani *et al.* have pointed out that it plays a minor role because of the constraint $\alpha^T \mathbf{y} = 0$ on Lagrange multipliers [2].

The second cause states that the soft-margin SVMs has inherent weakness for handling imbalanced data. We find that it depends on the linear separability of the data whether the imbalance has negative effects on SVMs. For linearly separable data, the imbalance will have tiny effects on SVMs, since all the slack variables ξ of (1) tend to be zeros (, unless the C is so small that the maximization of the margin dominates the objective). In the result, there is no contradiction between the capacity of the SVMs and the empirical error. Unfortunately, linear non-separable data often occurs. The SVMs has to achieve a tradeoff between maximizing the margin and minimizing the empirical error. For imbalanced data, the majority class outnumbers the minority one in the overlapping area. To reduce the overwhelming errors of misclassifying the majority class, the optimal hyperplane will inevitably be skew to the minority. In the extreme, if C is not very large, SVMs simply learns to classify everything as negative because that makes the "margin" the largest, with zero cumulative error on the abundant negative examples. The only tradeoff is the small amount of cumulative error on the few positive examples, which does not count for much.

Several variants of SVMs have been adopted to solve the problem of imbalance. One choice is the so-called one-class SVMs, which uses only positive examples for training. Without using the information of the negative samples, it is usually difficult to achieve as good result as that of binary SVMs classifier [18]. Using different penalty constants C_+ and C_- for the positive and negative examples have been reported to be effective [27, 17]. However, Wu *et al.* point out that the effectiveness of this method is limited [29]. The explanation of Wu is based on the KKT condition $\alpha^T \mathbf{y} = 0$, which imposes an equal total influence from the positive and negative support vectors. We evaluate this method and the result shows that tuning $\frac{C_+}{C_-}$ does work (details refer to Section 5). We find this also depends on the linear separability of the data whether this method works. For linearly separable data, tuning $\frac{C_+}{C_-}$ has little effects, since the penalty constants are useless with the zero-valued slack variables. However, if the data are linearly non-separable, tuning $\frac{C_+}{C_-}$ does change the position of separating hyperplane. The method to modify the kernel matrix is also proposed to improve SVMs for imbalanced data [29]. A possible drawback of this type approach is its high computational costs.

4. OVERALL APPROACH

The proposed approach is named Support Cluster Machines (SCMs). We first partition the negative samples into disjoint clusters, then train an initial SVMs model using the positive samples and the representatives of the negative clusters. With the global picture of the initial SVMs, we can approximately identify the support vectors and non-support vectors. A shrinking technique is then used to remove the samples which are most probably not support vectors. This procedure of clustering and shrinking are performed iteratively several times until some stop criteria satisfied. With such a from coarse-to-fine procedure, the *representative* and *informative* mechanisms are incorporated. There are four key issues in the meta-algorithm of SCMs: (a) How to get the partition of the training data, (b) How to get the representative for each cluster, (c) How to safely remove the non-support vector samples, (d) When to stop the iteration procedure. Though similar ideas have been proposed to speed-up SVMs in [30, 3, 31], no theoretical analysis of this idea has been provided. In the following, we present an in-depth analysis for this type of approaches and attempt to improve the algorithm under the theoretical guide.

4.1 Theoretical Analysis

Suppose $\{\pi_c\}_{c=1}^k$ is a partition of the training set that the samples within the same cluster have the same class label. If we construct a representative $\mathbf{u}_c$ for each cluster π_c, we can obtain two novel models of SVMs.

The first one is named Support Cluster Machines (SCMs). It treats each representative as a sample, thus the data size is reduced from n to k. This equals to the classification of the clusters. That is where the name SCMs come from. The new training set is

$$\mathcal{D}_\pi = \{(\mathbf{u}_c, y_c) | \mathbf{u}_c \in \mathcal{R}^N, y_c \in \{1, -1\}, c = 1, \ldots, k\},$$

in which y_c equals the labels of the samples within π_c. We define the dual problem of support cluster machines as

$$\begin{aligned} \min_{\alpha^\pi} \quad & G_\pi(\alpha_\pi) = \frac{1}{2}\alpha_\pi^T \mathbf{Q}_\pi \alpha_\pi - \mathbf{e}_\pi^T \alpha_\pi \\ \text{subjectto} \quad & 0 \le \alpha_{\pi i} \le |\pi_i| C, \forall i = 1, \ldots, k \\ & \alpha_\pi^T \mathbf{y}_\pi = 0, \end{aligned} \tag{6}$$

where α_π is a vector of size k with components $\alpha_{\pi i}$ corresponding to $\mathbf{u}_i$, $|\pi_i|C$ is the upper bound for $\alpha_{\pi i}$, $\mathbf{e}_\pi$ is a k dimension vector of all ones, and $\mathbf{Q}_\pi$ is an $k \times k$ positive semi-definite matrix, $Q_{\pi ij} = y_i y_j \langle \Phi(\mathbf{u}_i), \Phi(\mathbf{u}_j) \rangle$.

Another one is named Duplicate Support Vector Machines (DSVMs). Different from SCMs, it does not reduce the size of training set. Instead, it replace each sample $\mathbf{x}_i$ with the representative of the cluster that $\mathbf{x}_i$ belongs to. Thus, the samples within the same cluster are duplicate. That is why it is named DSVMs. The training set is

$$\tilde{\mathcal{D}} = \{(\tilde{\mathbf{x}}_i, \tilde{y}_i) | \forall \mathbf{x}_i \in \mathcal{D}, \text{if } \mathbf{x}_i \in \pi_c, \tilde{\mathbf{x}}_i = \mathbf{u}_c \text{ and } \tilde{y}_i = y_i \ \},$$

and the corresponding dual problem is defined as

$$\begin{aligned} \min_{\alpha} \quad & \tilde{G}(\alpha) = \frac{1}{2}\alpha^T \tilde{\mathbf{Q}} \alpha - \mathbf{e}^T \alpha \\ \text{subjectto} \quad & 0 \le \alpha_i \le C, \forall i = 1, \ldots, n \\ & \alpha^T \mathbf{y} = 0, \end{aligned} \tag{7}$$

where $\tilde{\mathbf{Q}}$ is is an $n \times n$ positive semi-definite matrix, $\tilde{Q}_{ij} = \tilde{y}_i \tilde{y}_j \langle \Phi(\tilde{\mathbf{x}}_i), \Phi(\tilde{\mathbf{x}}_j) \rangle$. We have the following theorem that states (6) is somehow equivalent to (7):

THEOREM 2. *With the above definitions of the SCMs and the DSVMs, if α_π^* and α^* are their optimal solutions respectively, the relation $G_\pi(\alpha_\pi^*) = \tilde{G}(\alpha^*)$ holds. Furthermore, any $\alpha_\pi \in \mathcal{R}^k$ satisfying $\{\alpha_{\pi c} = \sum_{\mathbf{x}_i \in \pi_c} \alpha_i^*, \quad \forall c = 1, \ldots, k\}$ is the optimal solution of SCMs. Inversely, any $\alpha \in \mathcal{R}^n$ satisfying $\{\sum_{\mathbf{x}_i \in \pi_c} \alpha_i = \alpha_{\pi c}^*, \forall c = 1, \ldots, k\}$ and the constraints of (7) is the optimal solution of DSVMs.*

The proof is in Appendix A. Theorem 2 shows that solving the SCMs is equivalent to solving a quadratic programming problem of the same scale as that of the SVMs in (2).

Comparing (2) and (7), we can find that only the Hessian matrix is different. Thus, to estimate the approximation from SCMs of (6) to SVMs of (2), we only need to analyze the stability of the quadratic programming model in (2) when the Hessian matrix varies from $\mathbf{Q}$ to $\tilde{\mathbf{Q}}$. Daniel has presented a study on the stability of the solution of definite quadratic programming, which requires that both $\mathbf{Q}$ and $\tilde{\mathbf{Q}}$ are positive definite [7]. However, in our situation, $\mathbf{Q}$ is usually positive definite and $\tilde{\mathbf{Q}}$ is not (because of the duplications). We develop a novel theorem for this case.

If define $\varepsilon = \|\mathbf{Q} - \tilde{\mathbf{Q}}\|$, where $\|\cdot\|$ denotes the Frobenius norm of a matrix, the value of ε measure the size of the perturbations between $\mathbf{Q}$ and $\tilde{\mathbf{Q}}$. We have the following theorem:

THEOREM 3. *If $\mathbf{Q}$ is positive definite and $\varepsilon = \|\mathbf{Q} - \tilde{\mathbf{Q}}\|$, let α^* and $\tilde{\alpha}^*$ be the optimal solutions to (2) and (7) respectively, we have*

$$\|\tilde{\alpha}^* - \alpha^*\| \le \frac{\tilde{m} C \varepsilon}{\lambda}$$

$$G(\tilde{\alpha}^*) - G(\alpha^*) \le \frac{(m^2 + \tilde{m}^2) C^2 \varepsilon}{2}$$

where λ is the minimum eigenvalue of $\mathbf{Q}$, m and $\tilde{m}$ indicate the numbers of the support vectors for (2) and (7) respectively.

The proof is in Appendix B. This theorem shows that the approximation from (2) to (7) is bounded by ε. Note that this does not mean that with minimal ε we are sure to get the best approximate solution. For example, adopting the support vectors of (1) to construct $\tilde{\mathbf{Q}}$ will yield the exact optimal solution of (2) but the corresponding ε are not necessarily minimum. However, we do not know which samples are support vectors beforehand. What we can do is to minimize the potential maximal distortion between the solutions between (2) and (7).

Now we consider the next problem, that is, given the partition $\{\pi_c\}_{c=1}^k$, what are the best representatives $\{\mathbf{u}_c\}_{c=1}^k$ for the clusters in the sense of approximating $\mathbf{Q}$? In fact, we have the following theorem:

THEOREM 4. *Given the partition $\{\pi_c\}_{c=1}^k$, the $\{\mathbf{u}_c\}_{c=1}^k$ satisfying*

$$\Phi(\mathbf{u}_c) = \frac{\sum_{\mathbf{x}_i \in \pi_c} \Phi(\mathbf{x}_i)}{|\pi_c|}, c = 1, \dots, k \qquad (8)$$

will make $\varepsilon = \|\mathbf{Q} - \tilde{\mathbf{Q}}\|$ minimum.

The proof is in Appendix C. This theorem shows that, given the partition, $\Phi(\mathbf{u}_c) = \mathbf{m}_c$ yields the best approximation between $\tilde{\mathbf{Q}}$ and $\mathbf{Q}$.

Here we come to the last question, i.e., what partition $\{\pi_c\}_{c=1}^k$ will make $\varepsilon = \|\mathbf{Q} - \tilde{\mathbf{Q}}\|$ minimum. To make the problem more clearly, we expand ε^2 as

$$\|\mathbf{Q}-\tilde{\mathbf{Q}}\|^2 = \sum_{h=1}^{k}\sum_{l=1}^{k}\sum_{\mathbf{x}_i\in\pi_h}\sum_{\mathbf{x}_j\in\pi_l}\left(\langle\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_j)\rangle - \langle\mathbf{m}_h,\mathbf{m}_l\rangle\right)^2. \qquad (9)$$

There are approximately $k^n/k!$ types of such partitions of the data set. An exhaustive search for the best partition is impossible. Recalling that (9) is similar to (4), we have the following theorem which states their relaxed equivalence.

THEOREM 5. *The relaxed optimal solution of minimizing (9) and the relaxed optimal solution of minimizing (4) are equivalent.*

The proof can be found in Appendix D. Minimizing ε amounts to find a low-rank matrix approximating $\mathbf{Q}$. Ding *et al.* have pointed out the relaxed equivalence between kernel PCA and kernel k-means in [11]. Note that minimizing (9) is different from kernel PCA in that it is with an additional block-wise constant constraint. That is, the value of $\tilde{Q}_{ij}$ must be invariant with respect to the cluster π_h containing $\tilde{\mathbf{x}}_i$ and the cluster π_l containing $\tilde{\mathbf{x}}_j$. With Theorem 5 we know that kernel k-means is a suitable method to obtain the partition of data.

According to the above results, the SCMs essentially finds an approximate solution to the original SVMs by smoothing the kernel matrix $\mathbf{K}$ (or Hessian matrix $\mathbf{Q}$). Fig.1 illustrates the procedure of smoothing the kernel matrix via clustering. Hence, by solving a smaller quadratic programming problem, the position of separating hyperplane can be roughly determined.

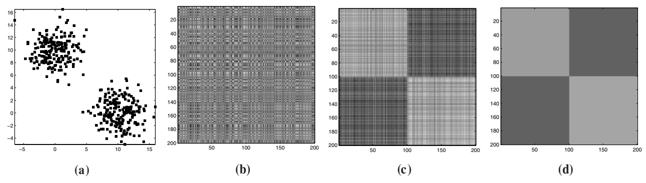

Figure 1: (a) 2D data distribution, (b) the visualization of the kernel matrix Q, (c) the kernel matrix Q by re-ordering the entries so that the samples belonging to the same cluster come together, (d) the approximate kernel matrix $\tilde{\mathbf{Q}}$ obtained by replacing each sample with the corresponding centroid.

4.2 Kernel-based Graph Clustering

In the previous work, k-means [30], BIRCH [31] and PDDP [3] have been used to obtain the partition of the data. None of them performs clustering in the feature space, though the SVMs works in the feature space. This is somewhat unnatural. Firstly, recalling that the kernel $K(\cdot,\cdot)$ usually implies an implicitly nonlinear mapping from the input space to the feature space, the optimal partition of input space is not necessarily the optimal one of feature space. Take k-means as an example, due to the fact that the squared Euclidean distance is used as the distortion measure, the clusters must be separated by piece-wise hyperplanes (i.e., voronoi diagram). However, these separating hyperplanes are no longer hyperplanes in the feature space with nonlinear mapping $\Phi(\cdot)$. Secondly, the k-means approach can not capture the complex structure of data. As shown in Fig.2, the negative class is in a ring-shape in the input space. If the k-means is used, the centroids of positive and negative class might overlap. Whereas in the feature space, the kernel k-means might get separable centroids.

Several factors limit the application of kernel k-means to large scale data. Firstly, it is almost impossible to store the whole kernel matrix $\mathbf{K}$ in the memory, e.g., for $n = 100\ 000$, we still need 20 gigabytes memory taking the symmetry into account. Secondly, the kernel k-means relies heavily on an effective initialization to achieve good results, and we do not have such a sound method yet. Finally, the computational cost of the kernel k-means might exceeds that of SVMs, and therefore, we lose the benefits of under-sampling. Dhillon *et al.* recently propose a multilevel kernel k-means method [9], which seems to cater to our requirements. The approach is based on the equivalence between graph clustering and kernel k-means. It incorporates the coarsening and initial partitioning phases to obtain a good initial clustering. Most importantly, the approach is extremely efficient. It can handle a graph with 28,294 nodes and 1,007,284 edges in several seconds. Therefore, here we adopt this approach. The detailed description can be found in [9]. In the following, we focus on how to address the difficulty of storing large scale kernel matrix.

Theorem 1 states that kernel k-means is equivalent to a type of graph clustering. Kernel k-means focuses on grouping data so that their average distance from the centroid is minimum ,while graph clustering aims to minimizing the average pair-wise distance among the data. Central grouping and pair-wise grouping are two different views of the same approach. From the perspective of pair-wise grouping, we can expect that two samples with large distance will not belong to the same cluster in the optimal solution. Thus,

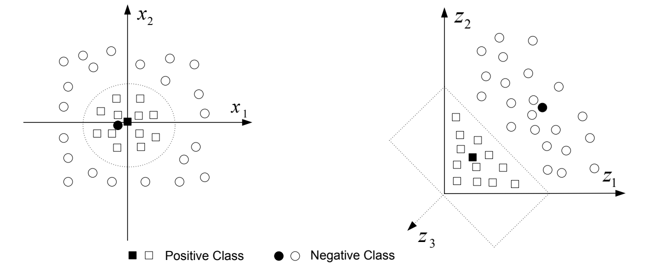

Figure 2: The left and right figures show the data distribution of input space and feature space respectively. The two classes are indicated by squares and circles. Each class is grouped into one cluster, and the solid mark indicates the centroid of the class.

we add the constraint that two samples with distance large enough are not linked by an edge, that is, transforming the dense graph to a sparse graph. This procedure is the common practice in spectral clustering or manifold embedding. Usually, two methods have been widely used for this purpose, i.e., k-nearest neighbor and ϵ-ball. Here, we adopt the ϵ-ball approach. Concretely, the edges with weight $A_{ij} < \epsilon$ is removed from the original graph, in which the parameter ϵ is pre-determined. By transforming a dense graph into a sparse graph, we only need store the sparse affinity matrix instead of the original kernel matrix. Nevertheless, we have to point out that the time complexity of constructing sparse graph is $O(n^2)$ for data set with n examples, which is the efficiency bottleneck of the current implementation. With the sparse graph, each iteration of the multilevel kernel k-means costs $O(ln^-)$ time, where ln^- is the number of nonzero entries in the kernel matrix.

4.3 Support Cluster Machines

According to Theorem 4, choosing the centroid of each cluster as representative will yield the best approximation. However, the explicit form of $\Phi(\cdot)$ is unknown. We don't know the exact pre-images of $\{\mathbf{m}_c\}_{c=1}^k$, what we can get are the dot products between the centroids by

$$\langle \mathbf{m}_h, \mathbf{m}_l \rangle = \frac{1}{|\pi_h||\pi_l|} \sum_{\mathbf{x}_i \in \pi_h} \sum_{\mathbf{x}_j \in \pi_l} \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle,$$

which requires $O(n^2)$ costs. Then the pre-computed kernel SVMs can be used. The pre-computed kernel SVMs takes the kernel matrix $\mathbf{K}_\pi$ as input, and save the indices of support vectors in the model [15]. To classify the incoming sample $\mathbf{x}$, we have to calculate the dot product between $\mathbf{x}$ and all the samples in the *support clusters*, e.g., π_c (If $\mathbf{m}_c$ is a support vector, we define the cluster π_c as *support cluster*.)

$$\langle \mathbf{x}, \mathbf{m}_c \rangle = \frac{1}{|\pi_c|} \sum_{\mathbf{x}_i \in \pi_c} \langle \mathbf{x}, \mathbf{x}_i \rangle.$$

We need another $O(nm)$ costs to predict all the training samples if there are m samples in support clusters. This is unacceptable for large scale data. To reduce the kernel reevaluation, we adopt the same method as [3], i.e., selecting a pseudo-center for each cluster as the representative

$$\mathbf{u}_c = \arg\min_{\mathbf{x}_i \in \pi_c} \|\Phi(\mathbf{x}_i) - \frac{1}{|\pi_c|} \sum_{\mathbf{x}_j \in \pi_c} \Phi(\mathbf{x}_j)\|^2,$$

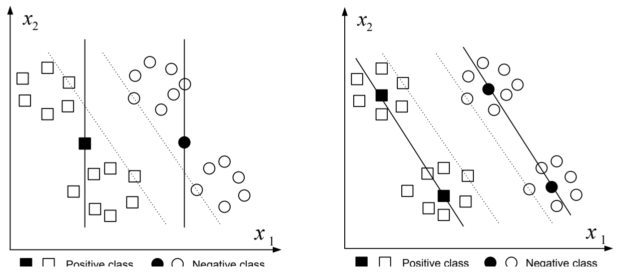

Figure 3: (a) Each class is grouped into one cluster, (b) each class is grouped into two clusters. The solid mark represents the centroid of the corresponding class. The solid lines indicate the support hyperplanes yielded by SCMs and the dot lines indicate the true support hyperplanes.

which can be directly obtained by

$$\mathbf{u}_c = \arg\max_{\mathbf{x}_i \in \pi_c} \sum_{\mathbf{x}_j \in \pi_c} \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle. \tag{10}$$

Thus, the kernel evaluation within training procedure requires $O(\sum_{c=1}^k |\pi_c|^2 + k^2)$ time, which be further reduced by probabilistic speedup proposed by Smola [25]. The kernel evaluation of predicting the training samples is reduced from $O(nm)$ to $O(ns)$, where s indicates the number of support clusters.

4.4 Shrinking Techniques

With the initial SCMs, we can remove the samples that are not likely support vectors. However, there is no theoretical guarantee for the security of the shrinking. In Fig. 3, we give a simple example to show that the shrinking might not be safe. In the example, if the samples outside the margin between support hyperplanes are to be removed, the case (a) will remove the true support vectors while the case (b) will not. The example shows that the security depends on whether the hyperplane of SCMs is parallel to the true separating hyperplane. However, we do not know the direction of true separating hyperplane before the classification. Therefore, what we can do is to adopt sufficient initial cluster numbers so that the solution of SCMs can approximate the original optimal solution enough. Specifically for large scale imbalanced data, the samples satisfying the following condition will be removed from the training set:

$$|\langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b| > \gamma, \tag{11}$$

where γ is a predefined parameter.

4.5 The Algorithm

Yu [31] and Boley [3] have adopted different stop criteria. In Yu *et al.*'s approach, the algorithm stops when each cluster has only one sample. Whereas, Boley *et al.* limit the maximum iterations by a fixed parameter. Here, we propose two novel criteria especially suitable for imbalanced data. The first one is to stop whenever the ratio of positive and negative samples is relatively imbalanced. Another choice is the Neyman-Pearson criterion, that is, minimizing the total error rate subject to a constraint that the miss rate of positive class is less than some threshold. Thus, once the miss rate of positive class exceeds some threshold, we stop the algorithm.

The overall approach is illustrated in Algorithm 1. With large scale balanced data, we carry out the data clustering for both classes separately. Whereas with imbalanced data, the clustering and shrinking will only be conducted on the majority class. The computation complexity is dominated by kernel evaluation. Therefore, it will not exceed $O((n^-)^2 + (n^+)^2)$, where n^- and n^+ indicate the number of negative and positive examples respectively.

Algorithm 1: Support Cluster Machines

```
    Input  : Training data set $\mathcal{D} = \mathcal{D}^+ \cup \mathcal{D}^-$
    Output: Decision function $f$
1   repeat
2       $\{\pi_c^+, \mathbf{m}_c^+\}_{c=1}^{k^+}$ =KernelKMeans($\mathcal{D}^+$)
3       $\{\pi_c^-, \mathbf{m}_c^-\}_{c=1}^{k^-}$ =KernelKMeans($\mathcal{D}^-$)
4       $\mathcal{D}_\pi = \{\mathbf{m}_c^+\}_{c=1}^{k^+} \cup \{\mathbf{m}_c^+\}_{c=1}^{k^-}$
5       $f_\pi$ =SVMTrain($\mathcal{D}_\pi$)
6       $f(\mathcal{D})$ =SVMPredict($f_\pi, \mathcal{D}$)
7       $\mathcal{D} = \mathcal{D}^+ \cup \mathcal{D}^-$ =Shrinking($f(\mathcal{D})$);
8   until stop criterion is true
```

5. EXPERIMENTS

The experiments on both the synthetic and the TRECVID data are carried out. The experiments on synthetic data are used to analyze the effects of large scale and imbalance on SVMs and the experiments on TRECVID data serve to evaluate the effectiveness and efficiency of SCMs. The multi-level kernel graph partitioning code *graclus* [9] is adopted for data clustering and the well-known *LibSVM* software [15] is used in our experiments. All our experiments are done in a Pentium 4 3.00GHz machine with 1G memory.

5.1 Synthetic Data Set

We generate two-dimensional data for the convenience of observation. Let x is a random variable uniformly distributed in $[0, \pi]$. The data are generated by

$$\mathcal{D}^+ = \{(x, y) | y = \sin(x) - \alpha + 0.7 \times [rand(0,1) - 1], x \in [0, \pi]\}$$
$$\mathcal{D}^- = \{(x, y) | y = -\sin(x) + 1 + 0.7 \times rand(0,1), x \in [0, \pi]\},$$

where $rand(0, 1)$ generates the random numbers uniformly distributed between 0 and 1, and α is a parameter controlling the overlapping ratio of the two classes. Fig. 4 and Fig. 5 show some examples of the synthetic data. We use the linear kernel function in all the experiments on synthetic data.

5.1.1 *The Effects of Scale*

We generate two types of balanced data, i.e., $n^+ = n^-$, but one ($\mathcal{D}_1 = \mathcal{D}(\alpha = 1.5)$) is linearly separable and the other ($\mathcal{D}_2 = \mathcal{D}(\alpha = 0.6)$) is not, as shown in Fig.4. We observe the difference of the behaviors of time costs for $\mathcal{D}_1$ and $\mathcal{D}_2$ when the scale increases. With the same parameter settings, the time costs of optimizing the objective for $\mathcal{D}_1$ and $\mathcal{D}_2$ are shown in Table 1, from which we can get two conclusions, (a) time costs increase with the scale, and (b) in the same scale, the linearly non-separable data will cost more time to converge.

Table 1: The effects of scale and overlapping on the time costs of training SVMs (in seconds).

$n^+ + n^-$	200	2000	4000	8000	20000	40000	80000
$time(\mathcal{D}_1)$	0.01	0.03	0.04	0.07	0.23	0.63	1.32
$time(\mathcal{D}_2)$	0.02	0.70	3.24	14.01	58.51	201.07	840.60

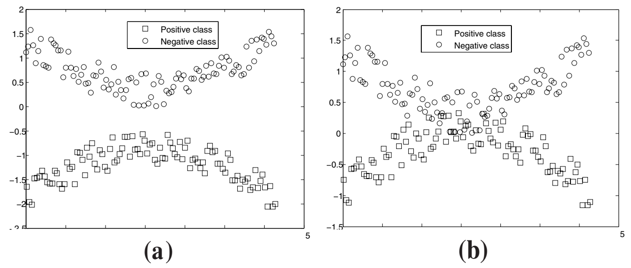

Figure 4: (a) example of non-overlapped balanced data sets, (b) example of overlapped balanced data sets.

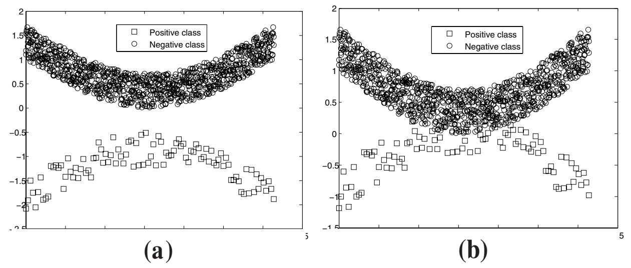

Figure 5: (a) example of non-overlapped imbalanced data sets, (b) example of overlapped imbalanced data sets.

5.1.2 *The Effects of Imbalance*

We generate two types of imbalanced data, i.e., $n^+ \ll n^-$, but one ($\mathcal{D}_1 = \mathcal{D}(\alpha = 1.5)$) is linearly separable and the other ($\mathcal{D}_2 = \mathcal{D}(\alpha = 0.6)$) is not, as shown in Fig.5. We observe the difference of the effects of imbalance for linearly separable data $\mathcal{D}_1$ and linearly non-separable $\mathcal{D}_2$. For the space limitation, we will not describe the detailed results here but only present the major conclusions. For linearly separable data, SVMs can find the non-skew hyperplane if C is not too small. In this situation, tuning $\frac{C^+}{C^-}$ is meaningless. For linearly non-separable data, the boundary will be skew to positive class if $C^+ = C^-$. In this case, increasing $\frac{C^+}{C^-}$ dose "push" the skewed separating hyperplane to the negative class. For both $\mathcal{D}_1$ and $\mathcal{D}_2$, if the C is too small, underfitting occurs, that is, the SVMs simply classify all the samples into negative class.

5.2 TRECVID Data Set

5.2.1 *Experimental Setup*

In this section, we evaluate the proposed approach on the high level feature extraction task of TRECVID [1]. Four concepts, including "car","maps","sports" and "waterscape", are chosen to model from the data sets. The development data of TRECVID 2005 are employed and divided into training set and validation set in equal size. The detailed statis-

Table 2: **The details of the training set and validation set of TRECVID 2005.**

Concept	$\|\mathcal{D}_{train}\|$		$\|\mathcal{D}_{val}\|$	
	Positive	Negative	Positive	Negative
Car	1097	28881	1097	28881
Maps	296	30462	296	30463
Sports	790	29541	791	29542
Waterscape	293	30153	293	30154

tics of the data is summarized in Table 2. In our experiments, the global 64-dimension color autocorrelogram feature is used to represent the visual content of each image. Conforming to the convention of TRECVID, average precision (AP) is chosen as the evaluation criterion. Totally five algorithms have been implemented for comparison:

Whole	All the negative examples are used
Random	Random sampling of the negative examples
Active	Active sampling of the negative examples
SCMs_I	SCMs with k-means in the input space
SCMs	SCMs with kernel k-means

In the **Active** method, we firstly randomly select a subset of negative examples. With this initial set, we train an SVMs model and use this model to classify the whole training data set. Then the maximally misclassified negative examples are added to the training set. This procedure iterates until the ratio between the negative and the positive examples exceeding five. Since both the **Random** and **Active** methods depend on the initial random chosen data set, we repeat each of them for ten times and calculate their average performances for comparison. Both **SCMs_I** and **SCMs** methods adopt the Gaussian kernel during the SVMs classification. The only difference is that **SCMs_I** performs data clustering with k-means in the input space while **SCMs** with k-means in the feature space.

5.2.2 Parameter Settings

Currently, the experiments focus on the comparative performance between the different approaches based on the the same parameter settings. Therefore, some of the parameters are heuristically determined and might not be optimal. The current implementation of SCMs involves the following parameter settings: (a) Gaussian kernel is adopted and the parameters are selected via cross-validation, furthermore, the kernel function of kernel k-means clustering is adopted the same as that of SVMs, (b) the threshold for transforming dense graphs to sparse ones is experimentally determined as $\epsilon = 0.6$, (c) the parameter of shrinking technique is experimentally chosen as $\gamma = 1.3$, (d) for SCMs, the data are imbalanced for each concept, we only carry out data clustering for negative classes, therefore, k^+ always equals $|\mathcal{D}^+|$ and k^- is always chosen as $\frac{|\mathcal{D}^-|}{10}$, (e) we stop the iteration of SCMs when the number of the negative examples are not more than the five times of that of the positive examples.

5.2.3 Experiment Results

The average performance and time costs of the various approaches are in Table 3 and Table 4 respectively. We can see that both the **Random** and **Active** methods use fewer time than the others, but their performances are not as good as the others. Furthermore, the **SCMs** achieves

Table 3: **The average performance of the approaches on the chosen concepts, measured by average precision.**

Concept	**Whole**	**Random**	**Active**	**SCMs_I**	**SCMs**
Car	**0.196**	0.127	0.150	0.161	0.192
Maps	**0.363**	0.274	0.311	0.305	0.353
Sports	0.281	0.216	0.253	0.260	**0.283**
Waterscape	**0.269**	0.143	0.232	0.241	0.261

Table 4: **The average time costs of the approaches on the chosen concepts (in seconds).**

Concept	**Whole**	**Random**	**Active**	**SCMs_I**	**SCMs**
Car	4000.2	431.0	1324.6	1832.0	2103.4
Maps	402.6	35.2	164.8	234.3	308.5
Sports	1384.5	125.4	523.8	732.5	812.7
Waterscape	932.4	80.1	400.3	504.0	621.3

the comparable performance with that of **Whole** while uses fewer time costs. Note that **SCMs_I** also achieves satisfying results. This might be due to the Gaussian kernels, in which $e^{-\|x-y\|^2}$ is monotonic with $\|x-y\|^2$. Therefore, the order of the pair-wise distances is the same for both the input space and feature space, which perhaps leads to similar clustering results.

6. CONCLUSIONS

In this paper, we have investigated the effects of scale and imbalance on SVMs. We highlight the role of data overlapping in this problem and find that SVMs has no difficulties with linear separable large scale imbalanced data. We propose a meta-algorithm named Support Cluster Machines (SCMs) for effectively learning from large scale and imbalanced data sets. Different from the previous work, we develop the theoretical justifications for the idea and choose the technical component guided by the theoretical results. Finally, experiments on both the synthetic and the TRECVID data are carried out. The results support the previous analysis and show that the SCMs are efficient and effective while dealing with large scale imbalanced data sets.

However, as a pilot study, there is still some room for improvement. Firstly, we have not incorporated the caching techniques to avoid the kernel reevaluations. Therefore, we have to recalculate the dot product on line whenever it is required. Secondly, the parameters within the algorithms are currently selected heuristically, which depend on the tradeoff of efficiency and accuracy.

7. ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their insightful suggestions. We also thank Dr. Chih-Jen Lin for the code of *libSVM*, Brian J. Kulis for the code of *graclus* and National Institute of Standards and Technology for providing the TRECVID data sets. Finally, special thanks go to Dr. Ya-xiang Yuan for his helpful discussions on optimization theory.

8. REFERENCES

[1] TREC Video Retrieval. National Institute of Standards and Technology, http://www-nlpir.nist.gov/projects/trecvid/.

[2] R. Akbani, S. Kwek, and N. Japkowicz. Applying Support Vector Machines to Imbalanced Datasets. In *Proceedings of ECML'04*, pages 39–50, 2004.

[3] D. Boley and D. Cao. Training Support Vector Machine using Adaptive Clustering. In *Proceeding of 2004 SIAM International Conference on Data Mining*, April 2004.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[5] K. Brinker. Incorporating Diversity in Active Learning with Support Vector Machines. In *Proceedings of ICML'03*, pages 59–66, 2003.

[6] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: Special Issue on Learning from Imbalanced Data Sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, 2004.

[7] J. W. Daniel. Stability of the Solution of Definite Quadratic Programs. *Mathematical Programming*, 5(1):41–53, December 1973.

[8] R. Datta, J. Li, and J. Z. Wang. Content-based Image Retrieval: Approaches and Trends of the New Age. In *Proceedings of ACM SIGMM workshop on MIR'05*, pages 253–262, 2005.

[9] I. Dhillon, Y. Guan, and B. Kulis. A Fast Kernel-based Multilevel Algorithm for Graph Clustering. In *Proceeding of ACM SIGKDD'05*, pages 629–634, 2005.

[10] I. S. Dhillon, Y. Guan, and B. Kulis. A Unified View of Graph Partitioning and Weighted Kernel k-means. Technical Report TR-04-25, The University of Texas at Austin, Department of Computer Sciences, June 2004.

[11] C. Ding and X. He. K-means clustering via principal component analysis. In *Proceedings of ICML'04*, pages 29–36, 2004.

[12] K.-S. Goh, E. Y. Chang, and W.-C. Lai. Multimodal Concept-dependent Active Learning for Image Retrieval. In *Proceedings of ACM MM'04*, pages 564–571, 2004.

[13] A. G. Hauptmann. Towards a Large Scale Concept Ontology for Broadcast Video. In *Proceedings of CIVR'04*, pages 674–675, 2004.

[14] A. G. Hauptmann. Lessons for the Future from a Decade of Informedia Video Analysis Research. In *Proceedings of CIVR'05*, pages 1–10, 2005.

[15] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A Practical Guide to Support Vector Classification. 2005. available at http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

[16] T. Joachims. Making Large-scale Support Vector Machine Learning Practical. *Advances in kernel methods: support vector learning*, pages 169–184, 1999.

[17] Y. Lin, Y. Lee, and G. Wahba. Support Vector Machines for Classification in Nonstandard Situations. *Machine Learning*, 46(1-3):191–202, 2002.

[18] L. M. Manevitz and M. Yousef. One-class SVMs for Document Classification. *Journal of Machine Learning Research*, 2:139–154, 2002.

[19] H. T. Nguyen and A. Smeulders. Active Learning Using Pre-clustering. In *Proceedings of ICML'04*, pages 79–86, 2004.

[20] E. Osuna, R. Freund, and F. Girosi. An Improved Training Algorithm for Support Vector Machines. In *IEEE Workshop on Neural Networks and Signal Processing*, September 1997.

[21] D. Pavlov, J. Mao, and B. Dom. Scaling-Up Support Vector Machines Using Boosting Algorithm. In *Proceeding of ICPR'00*, volume 2, pages 2219–2222, 2000.

[22] J. C. Platt. Fast Training of Support Vector Machines using Sequential Minimal Optimization. *Advances in kernel methods: support vector learning*, pages 185–208, 1999.

[23] R. C. Prati, G. E. A. P. A. Batista, and M. C. Monard. Class Imbalances *versus* Class Overlapping: an Analysis of a Learning System Behavior. In *Proceedings of the MICAI 2004*, pages 312–321, 2004.

[24] G. Schohn and D. Cohn. Less is More: Active Learning with Support Vector Machines. In *Proceddings of ICML'00*, pages 839–846, 2000.

[25] A. J. Smola and B. Schökopf. Sparse Greedy Matrix Approximation for Machine Learning. In *Proceedings of ICML'00*, pages 911–918, 2000.

[26] S. Tong and E. Chang. Support Vector Machine Active Learning for Image Retrieval. In *Proceedings of ACM MM'01*, pages 107–118, 2001.

[27] K. Veropoulos, N. Cristianini, and C. Campbell. Controlling the Sensitivity of Support Vector Machines. In *Proceedings of IJCAI'99*, 1999.

[28] G. M. Weiss and F. J. Provost. Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction. *Journal of Artificial Intelligence Research (JAIR)*, 19:315–354, 2003.

[29] G. Wu and E. Y. Chang. KBA: Kernel Boundary Alignment Considering Imbalanced Data Distribution. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):786–795, 2005.

[30] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang. Representative Sampling for Text Classification Using Support Vector Machines. In *Proceedings of ECIR'03*, pages 393–407, 2003.

[31] H. Yu, J. Yang, J. Han, and X. Li. Making SVMs Scalable to Large Data Sets using Hierarchical Cluster Indexing. *Data Min. Knowl. Discov.*, 11(3):295–321, 2005.

APPENDIX

A. PROOF OF THEOREM 2

Firstly, we define $\hat{\alpha}_\pi$ which satisfies $\hat{\alpha}_{\pi c} = \sum_{\mathbf{x}_i \in \pi_c} \alpha_i^*, \ \forall c = 1, \ldots, k$. It is easy to verify that $\hat{\alpha}_\pi$ is a feasible solution of SCMs. Secondly, we define $\check{\alpha}$ satisfying $\check{\alpha}_i = \frac{\alpha^*_{\pi c}}{|\pi_c|}$ if $\mathbf{x}_i \in \pi_c, i = 1, \ldots, n$. It is easy to verify that $\check{\alpha}$ is a feasible solution of DSVMs. According to the relation of $\mathcal{D}_\pi$ and $\tilde{\mathcal{D}}$, we can obtain the following equation

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i^* y_i \langle \Phi(\tilde{\mathbf{x}}_i), \Phi(\tilde{\mathbf{x}}_j)\rangle \alpha_j^* y_j - \sum_{i=1}^{n}\alpha_i^* =$$

$$\frac{1}{2}\sum_{h=1}^{k}\sum_{l=1}^{k}\hat{\alpha}_{\pi h} y_h \langle \Phi(\mathbf{u}_h), \Phi(\mathbf{u}_l)\rangle \hat{\alpha}_{\pi l} y_l - \sum_{h=1}^{k}\hat{\alpha}_{\pi h},$$

which means $\tilde{G}(\alpha^*) = G_\pi(\hat{\alpha}_\pi)$. Similarly, we can get $\tilde{G}(\check{\alpha}) = G_\pi(\alpha_\pi^*)$. Using the fact that α_π^* and α^* are the optimal solutions to SCMs and DSVMs respectively, we have $G_\pi(\alpha_\pi^*) \leq G_\pi(\hat{\alpha}_\pi)$ and $\tilde{G}(\alpha^*) \leq \tilde{G}(\check{\alpha})$. Thus, the equation $G_\pi(\alpha_\pi^*) = \tilde{G}(\alpha^*)$ holds. For any $\alpha_\pi \in \mathcal{R}^k$ satisfying $\{\alpha_{\pi c} = \sum_{\mathbf{x}_i \in \pi_c} \alpha_i^*, \forall c = 1, \dots, k\}$, we know it is a feasible solution to SCMs and $G_\pi(\alpha_\pi) = \tilde{G}(\alpha^*) = G_\pi(\alpha_\pi^*)$ holds, which means α_π is the optimal solution of SCMs. Similarly, for any $\alpha \in \mathcal{R}^n$ satisfying $\{\sum_{\mathbf{x}_i \in \pi_c} \alpha_i = \alpha_{\pi c}^*, \forall c = 1, \dots, k\}$ and the constraints of (7), we have $\tilde{G}(\alpha) = G_\pi(\alpha_\pi^*) = \tilde{G}(\alpha^*)$, which means α is the optimal solution of DSVMs.

B. PROOF OF THEOREM 3

Note that the feasible regions of (2) and (7) are the same. By the fact that α^* and $\tilde{\alpha}^*$ are optimal solutions to (2) and (7) respectively, we know that

$$(\tilde{\alpha}^* - \alpha^*)^T \nabla G(\alpha^*) \geq 0 \tag{12}$$

$$(\alpha^* - \tilde{\alpha}^*)^T \nabla \tilde{G}(\tilde{\alpha}^*) \geq 0 \tag{13}$$

hold, where the gradients

$$\nabla G(\alpha) = \mathbf{Q}\alpha - \mathbf{e} \ \text{ and } \ \nabla \tilde{G}(\alpha) = \tilde{\mathbf{Q}}\alpha - \mathbf{e}. \tag{14}$$

Adding (12) and (13) and then a little arrangement yields

$$(\tilde{\alpha}^* - \alpha^*)^T [\nabla \tilde{G}(\tilde{\alpha}^*) - \nabla \tilde{G}(\alpha^*)] \leq (\tilde{\alpha}^* - \alpha^*)^T [\nabla G(\alpha^*) - \nabla \tilde{G}(\alpha^*)].$$

Substituting (14) in the above inequality, we get

$$(\tilde{\alpha}^* - \alpha^*)^T \tilde{\mathbf{Q}} (\tilde{\alpha}^* - \alpha^*) \leq (\tilde{\alpha}^* - \alpha^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \alpha^*. \tag{15}$$

Adding $(\tilde{\alpha}^* - \alpha^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}})(\tilde{\alpha}^* - \alpha^*)$ to the both sides of (15), we have

$$(\tilde{\alpha}^* - \alpha^*)^T \mathbf{Q} (\tilde{\alpha}^* - \alpha^*) \leq (\tilde{\alpha}^* - \alpha^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \tilde{\alpha}^*. \tag{16}$$

If $\lambda > 0$ is the smallest eigenvalue of $\mathbf{Q}$, we have

$$\lambda \|\tilde{\alpha}^* - \alpha^*\|^2 \leq (\tilde{\alpha}^* - \alpha^*)^T \mathbf{Q} (\tilde{\alpha}^* - \alpha^*)$$

$$(\tilde{\alpha}^* - \alpha^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \tilde{\alpha}^* \leq \|\tilde{\alpha}^* - \alpha^*\| \|\mathbf{Q} - \tilde{\mathbf{Q}}\| \|\tilde{\alpha}^*\|$$

and $\|\tilde{\alpha}^*\| \leq \tilde{m}C$. Using (16) we get $\|\tilde{\alpha}^* - \alpha^*\| \leq \frac{\tilde{m}C\varepsilon}{\lambda}$. Now we turn to prove the second result. α^* is the optimal solution of (2), therefore, $0 \leq G(\tilde{\alpha}^*) - G(\alpha^*)$ is obvious. Meanwhile, we have

$$\begin{aligned}
G(\tilde{\alpha}^*) - G(\alpha^*) &= \frac{1}{2}(\tilde{\alpha}^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \tilde{\alpha}^* + \tilde{G}(\tilde{\alpha}^*) - G(\alpha^*) \\
&\leq \frac{1}{2}(\tilde{\alpha}^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \tilde{\alpha}^* + \tilde{G}(\alpha^*) - G(\alpha^*) \\
&= \frac{1}{2}(\tilde{\alpha}^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \tilde{\alpha}^* - \frac{1}{2}(\alpha^*)^T (\mathbf{Q} - \tilde{\mathbf{Q}}) \alpha^* \\
&\leq \frac{1}{2}\|\mathbf{Q} - \tilde{\mathbf{Q}}\| \|\tilde{\alpha}^*\|^2 + \frac{1}{2}\|\mathbf{Q} - \tilde{\mathbf{Q}}\| \|\alpha^*\|^2 \\
&\leq \frac{(m^2 + \tilde{m}^2) C^2 \varepsilon}{2}
\end{aligned}$$

C. PROOF OF THEOREM 4

Expanding ε to be the explicit function of $\{\Phi(\mathbf{u}_c)\}_{c=1}^k$, we get $\varepsilon^2 = \|\mathbf{YKY} - \tilde{\mathbf{Y}}\tilde{\mathbf{K}}\tilde{\mathbf{Y}}\|^2$, in which $\mathbf{Y}$ and $\tilde{\mathbf{Y}}$ denote diagonal matrices whose diagonal elements are $y_1, \dots, y_n$ and $\tilde{y}_1, \dots, \tilde{y}_n$ respectively. Using the fact that $\mathbf{Y}$ equals to $\tilde{\mathbf{Y}}$, we have $\varepsilon^2 = \|\mathbf{Y}(\mathbf{K} - \tilde{\mathbf{K}})\mathbf{Y}\|^2$. Since $\mathbf{Y}$ only change the signs of the elements of $\mathbf{K} - \tilde{\mathbf{K}}$ by $\mathbf{Y}(\mathbf{K} - \tilde{\mathbf{K}})\mathbf{Y}$, we have $\varepsilon^2 = \|\mathbf{K} - \tilde{\mathbf{K}}\|^2 = \sum_{h=1}^k \sum_{l=1}^k \sum_{\mathbf{x}_i \in \pi_h} \sum_{\mathbf{x}_j \in \pi_l} (\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle - \langle \Phi(\mathbf{u}_h), \Phi(\mathbf{u}_l) \rangle)^2$. It is a biquadratic function of $\{\Phi(\mathbf{u}_c)\}_{c=1}^k$. Therefore, this is an unconstrained convex optimization problem [4]. The necessary and sufficient condition for $\{\mathbf{u}_c^*\}_{c=1}^k$ to be optimal is $\nabla \varepsilon^2(\{\Phi(\mathbf{u}_c^*)\}_{c=1}^k) = 0$. We can verify that $\Phi(\mathbf{u}_c) = \frac{\sum_{\mathbf{x}_i \in \pi_c} \Phi(\mathbf{x}_i)}{|\pi_c|}, c = 1, \dots, k$ satisfies the condition that the gradient is zero.

D. PROOF OF THEOREM 5

We define a $n \times k$ matrix $\mathbf{Z}$ as $Z_{ic} = \begin{cases} \frac{1}{\sqrt{|\pi_c|}} & \text{if } \ \mathbf{x}_i \in \pi_c \\ 0 & \text{otherwise} \end{cases}$. We can see that $\mathbf{Z}$ captures the disjoint cluster memberships. There is only one non-zero entry in each row of $\mathbf{Z}$ and $\mathbf{Z}^T\mathbf{Z} = \mathbf{I}_k$ holds ($\mathbf{I}_k$ indicates the identity matrix). Suppose $\mathbf{\Phi}$ is the matrix of the images of the samples in feature space, i.e., $\mathbf{\Phi} = [\Phi(\mathbf{x}_1), \dots, \Phi(\mathbf{x}_n)]$. We can verify that the matrix $\mathbf{\Phi}\mathbf{Z}\mathbf{Z}^T$ consists of the mean vectors of the clusters containing the corresponding sample. Thus, the ε^2 can be written as

$$\|\mathbf{Q} - \tilde{\mathbf{Q}}\|^2 = \|\mathbf{\Phi}^T\mathbf{\Phi} - (\mathbf{\Phi}\mathbf{Z}\mathbf{Z}^T)^T \mathbf{\Phi}\mathbf{Z}\mathbf{Z}^T\|^2.$$

Using the fact that $trace(\mathbf{A}^T\mathbf{A}) = \|\mathbf{A}\|_F^2$, $trace(\mathbf{A} + \mathbf{B}) = trace(\mathbf{A}) + trace(\mathbf{B})$ and $trace(\mathbf{AB}) = trace(\mathbf{BA})$, we have

$$\varepsilon^2 = trace((\mathbf{\Phi}^T\mathbf{\Phi})^T \mathbf{\Phi}^T\mathbf{\Phi} - (\mathbf{Z}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{Z})(\mathbf{Z}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{Z})).$$

Since $trace((\mathbf{\Phi}^T\mathbf{\Phi})^T \mathbf{\Phi}^T\mathbf{\Phi})$ is constant, minimizing ε is equivalent to maximizing

$$J_1 = trace((\mathbf{Z}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{Z})(\mathbf{Z}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{Z})). \tag{17}$$

With similar procedure, we can see that minimizing $J(\{\pi_c\}_{c=1}^k)$ amounts to maximizing

$$J_2 = trace(\mathbf{Z}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{Z}). \tag{18}$$

Matrix $\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}$ is a symmetric matrix. Let $\zeta_1 \geq \dots, \geq \zeta_n \geq 0$ denote its eigenvalues and $(\mathbf{v}_1, \dots, \mathbf{v}_n)$ be the corresponding eigenvectors. Matrix $\mathbf{H} = \mathbf{Z}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{Z}$ is also a symmetric matrix. Let $\eta_1 \geq \dots, \geq \eta_k \geq 0$ denote its eigenvalues. According to Poincaré Separation Theorem, we know the relations $\zeta_i \geq \eta_i, i = 1, \dots, k$ hold. Therefore, we have $J_2 = \sum_{i=1}^k \eta_i \leq \sum_{i=1}^k \zeta_i$. Similarly, we have $J_1 = \sum_{i=1}^k \eta_i^2 \leq \sum_{i=1}^k \zeta_i^2$. In both cases, the equations hold when $\mathbf{Z} = (\mathbf{v}_1, \dots, \mathbf{v}_k)\mathbf{R}$, where $\mathbf{R}$ is an arbitrary $k \times k$ orthonormal matrix. Actually, the solution to maximizing J_2 is just the well-known theorem of Ky Fan (the Theorem 3.2. of [11]). Note that the optimal $\mathbf{Z}$ might no longer conforms to the definition of $Z_{ic} = \begin{cases} \frac{1}{\sqrt{|\pi_c|}} & \text{if } \ \mathbf{x}_i \in \pi_c \\ 0 & \text{otherwise} \end{cases}$, but it is still a orthonormal matrix. That is why it is called a relaxed optimal solution.