AN ABSTRACT OF THE THESIS OF

Pavan Kumar Vatturi for the degree of Master of Science in Computer Science presented on December 30, 2008.

Title: Rare Category Detection using Hierarchical Mean Shift

Abstract approved: _____

Weng-Keen Wong

Many applications in surveillance, monitoring, scientific discovery, and data cleaning require the identification of anomalies. Although many methods have been developed to identify statistically significant anomalies, a more difficult task is to identify anomalies that are both interesting and statistically significant. Category detection is an emerging area of machine learning that can address this issue using a "human-in-the-loop" approach. In this interactive setting, the algorithm asks the user to label a query data point under an existing category or declare the query data point to belong to a previously undiscovered category. The goal of category detection is to discover all the categories in the data in as few queries as possible. In a data set with imbalanced categories, the main challenge is in identifying the rare categories or anomalies; hence, the task is often referred to as rare category detection.

We present a new approach to rare category detection using a hierarchical mean shift procedure. In our approach, a hierarchy is created by repeatedly applying mean shift with increasing bandwidth on the entire data set. This hierarchy allows us to identify anomalies in the data set at different scales, which are then posed as queries to the user. The main advantage of this methodology over existing approaches is that it does not require any knowledge of the dataset properties such as the total number of classes or the

prior probabilities of the classes. Results on real-world data sets show that our hierarchical mean shift approach performs consistently better than previous techniques.

Rare Category Detection using Hierarchical Mean Shift

by

Pavan Kumar Vatturi

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented December 30, 2008
Commencement June 2009

Master of Science thesis of Pavan Kumar Vatturi presented on December 30, 2008.

APPROVED:

Major Professor, representing Computer Science

Director of the School of Electric Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Pavan Kumar Vatturi, Author

ACKNOWLEDGEMENTS

I would like to take this opportunity and thank my advisor, Weng-Keen Wong, for his support, understanding and guidance. His approach to research helped me in reaching my potential while maintaining my enthusiasm for the subject matter. I am grateful to have him as my advisor. I would also like to thank all my friends and colleagues who provided me with all the collaborative and personal support that I needed along the way. Finally, I thank my parents and my brother for their constant love and dedication, which has imbued everything in my life with value.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

DEDICATION

For my parents.

Chapter 1 – Introduction

Many applications in surveillance, monitoring, scientific discovery, and data cleaning require the identification of anomalies. Ideally, these anomalies correspond to data points or events of interest, such as a disease outbreak in biosurveillance or a network attack in intrusion detection. Although many methods have been developed to characterize anomalies as statistically unusual events, not all statistically significant anomalies are necessarily useful. In fact, many anomalies are simply uninteresting, corresponding to known sources of noise or known combinations of features that are irrelevant to actual events of interest.

Consider the following two examples. The first example, taken from [16], involves the task of scientific discovery from the Sloan Digital Sky Survey (SDSS) [19], which is a 5 year survey of the northern skies by ground-based telescopes. Most of the images in the SDSS capture known phenomena such as stars, comets, nebulae, etc. which have already been discovered. Anomalies in the data correspond to unusual or unknown objects which could potentially lead to new scientific discoveries. However, the majority of anomalies in the SDSS are of no interest to astronomers. These anomalies include satellite trails and diffraction spikes, which are artifacts of the telescope. Such anomalies clearly do not lead to new discoveries in astronomy. The anomalies of interest, such as unusual galaxies, are extremely rare and constitute a miniscule 0.001% of the entire data set. A similar task exists in the analysis of network log files. The IT infrastructure of a company can contain thousands of computers and devices networked together. These components are often equipped with monitoring agents that generate log files that capture characteristics of the network traffic. Statistical anomalies in these log files often correspond to uninteresting events such as those arising from events that are already known or expected, such as events arising from maintenance upgrades. A very small fraction of the log file anomalies

correspond to actual network failures and attacks. Identifying these meaningful anomalies would be beneficial for the diagnosis of faults and the prevention of malicious attacks.

Thus, a challenging new task has emerged for the field of anomaly detection – identifying anomalies that are not only statistically significant but also interesting. Since the "interestingness" of an anomaly is subjectively defined, a human-in-the-loop approach is needed. Category detection is an emerging area of machine learning that can address this issue. In category detection, the algorithm asks the user to label a query data point under an existing category or declare the query data point to belong to a previously undiscovered category. The goal of category detection is to discover all the categories in the data in as few queries to the user as possible. The main challenge in this task is discovering the rare categories, which appear as small, dense clusters or isolated outliers in the data set. This task becomes especially challenging if the data set is dominated by a handful of disproportionately large categories, which makes the rare categories become extremely difficult to discover through manual inspection. As a result, category detection is often referred to as *rare* category detection.

In this thesis we present a new approach to rare category detection using a hierarchical mean shift procedure. Mean shift is a non-parametric clustering technique widely used in the areas of image processing and computer vision. It is used in this thesis for discovering clusters and cluster modes. By repeatedly calling mean shift with different bandwidths, we can create a hierarchy of clusters at different scales and thus use this information to score each cluster by how "anomalous" it is. This score can then be used to rank the representative data points of each cluster for labeling by a user. The main advantage of the proposed method over previous related work [16, 9] is that it does not require any prior knowledge regarding the properties of the data set, such as the total number of clusters present in the data or the prior probabilities of the clusters.

Chapter 2 – Related Work

2.1 Problem Definition

Previous work has defined the problem of rare category detection in the following manner. A set of unlabeled examples $S = \{x_1, x_2, ..., x_n\}$ where $x_i \in \Re^d$ are from m distinct classes labeled $y_i = \{1, 2..., m\}$. The goal is to detect at least a single instance from each class by requesting as few labels from the user as possible.

2.2 Rare Category Detection

2.2.1 Interleave

Pelleg et al. [16] proposed the Interleave algorithm in which they assumed the data is generated by a mixture model. Interleave starts with an entirely unlabeled data set and clusters it using a standard EM algorithm for mixture models. The EM algorithm maintains, for every mixture component and data point, the "degree of ownership" that the mixture component exerts over the data point. Interleave stores, for each mixture component, a list of the points that it "most owns", sorted in increasing order by the "degree of ownernship" values. Intuitively, the algorithm queries data points that are "least owned" by the components. It cycles through all the lists in a round robin fashion and constructs an output list by picking the top data point from each list that has not already been placed in the output list. This top data point is placed in the next position of the output list. The data points in the output lists are the ones selected for user labeling. The class membership values obtained from labeling data points from the previous step are clamped to the user-supplied labels before the next EM run. These iterations go

on until data points from all the classes have been discovered. Interleave is outlined in Algorithm 1.

Algorithm 1: Interleave

```
Set Labels = {};
while not all classes have been discovered do
    (PointOwnership[m][n]) = SemiSupervisedEM (S, Labels);
    where PointOwnership[i][j] is an estimate of the degree of ownership that i^th
    component exerts over j^th point;
    for i ← 1 to m do
        Sort(PointOwnership[i], 'IncreasingOrder', Points[i]);
        where Points[i] is the list of all points ranked by the degree of ownership
        that i^th component exerts over them;
    end
    OutputList = ();
    for i ← 1 to m do
        for j ← 1 to n do
            if Points[i][j] not in OutputList & Labels then
                OutputList.PushBack(Points[i][j]);
                break;
            end
        end
    end
    Labels += SelectTopK(OutputList,numHints);
    where SelectTopK(OutputList,numHints) selects the top numHints points
    from the OutputList;
end
```

The advantage of this approach is that it is model independent. The algorithm does not make any assumption that the data is noiseless like some of the existing active learning methods and is resilient to noise. However, this algorithm needs to know, at the beginning, an initial estimate of the number of classes in the data. Although the number of classes changes as the user provides feedback to the Interleave algorithm, this initial estimate of the number of classes is often not known in advance and an incorrect value at the outset can adversely affect the algorithm. Furthermore, the Interleave algorithm converges to a local optimum due to the EM clustering algorithm, thereby producing different results for different starting conditions.

2.2.2 Nearest-Neighbor-Based Rare Category Detection (NNDM)

He et al. [9] describe a nearest-neighbor based active learning for rare category detection (NNDM). NNDM uses an unsupervised local density differential sampling strategy. It makes use of nearest neighbors to measure the local density around each data point. The algorithm starts with an entirely unlabeled data set. In each iteration, the algorithm selects for labeling the data point with the largest change in local density. The algorithm halts once a data point from each class has been selected for labeling. The main advantage of this method is that it does not assume the separability of the classes. This property is useful in case of real world data sets where the support regions of the majority and minority classes often overlap. The pseudocode for NNDM is outlined in Algorithm 2. Like Interleave, NNDM needs to know the number of classes in advance. NNDM also requires the prior probabilities of the classes. While NNDM is effective at discovering categories that overlap each other, we have noticed some undesirable behavior in which NNDM repeatedly queries data points from the same already-discovered cluster if these data points have a large local density differential value.

2.2.3 Active Sampling for Multiple Output Identification

Fine et al. [5] abstract the rare category detection problem as an output identification task in a learning model. The learning model has an unknown target function f which maps every input in χ to one of m output values. The output identification task is to find m inputs, one for each output value. The algorithm knows that the target function f is in a given function class F. The work assumes an unknown distribution over the inputs and describes algorithms for many classes of functions. The algorithms are shown to have an expected sample bound of the order of m and $log(1/\epsilon)$ where ϵ is the lower bound on the probability of each output class. The work similarly reports sample bounds in special

Algorithm 2: Nearest-Neighbor-Based Rare Category Detection for Multiple Classes (NNDM)

Let $p_1, p_2 ... p_m$ be the priors of m classes. Let n be the number of data points.;
for $i \leftarrow 2$ **to** m **do**
    Let $K_i = np_i$;
    For each example, calculate the distance between this example and its K_i^{th} nearest neighbor;
    Set $r_i^{'}$ to be the minimum value among all the examples;
end
Let $r_1^{'} = max_{i=2}^{m} r_i^{'}$;
for $i \leftarrow 1$ **to** m **do**
    $\forall x_j \in S$, let NN(x_j,$r_i^{'}$) = $\{x | x \in S, \|x - x_j\| \leq r_i^{'}\}$, and $n_j^i = |\text{NN}(x_j, r_i^{'})|$;
end
while *not all classes have been discovered* **do**
    Let $r^{'} = \min\{r_i^{'} | 1 \leq i \leq m$, and class i has not been discovered$\}$, and s be the corresponding index, i.e. $r^{'} = r_s^{'}$;
    for $t \leftarrow 1$ **to** n **do**
        for each x_i that has been selected and labeled y_i, $\forall x \in S, s.t. \|x - x_i\| \leq r_{y_i}^{'}, s_i = -\infty$;
        for all other examples, $s_i = \max_{x_j \in NN(x_i, tr^{'})}(n_i^s - n_j^s)$;
        Query $x = argmax_{x_i \in S} s_i$;
        If x belongs to a class that has not been discovered, break;
    end
end

cases like binary outputs i.e. $m = 2$, specific distributions (the uniform distribution) and a concept class defined over the Boolean hypercube $\{0, 1\}^n$.

2.3 Semi-supervised Clustering

Semi-supervised clustering incorporates background knowledge about the domain or the data set into the clustering process. This knowledge is in the form of pairwise must-link or cannot-link constraints between data points. Bilenko et al.[1] propose approaches that use labeled data or pairwise constraints on data in a semi-supervised clustering setup. The pairwise constraints are used in constructing an objective function defined as the sum total of the squared distances between the points and their cluster centroids with an attached cost for violating any pairwise constraint. The clustering algorithm aims to minimize this objective function. The work also highlights a metric learning approach to further utilize the pairwise constraints obtained from the labeled data. The distance metric is modified in such a way that the distance between the objects of the same cluster is minimized and that of different clusters is maximized. The generally used Euclidean distance is parameterized using a symmetric positive-definite weight matrix A. Using the pairwise constraints, a weight matrix is learned for each cluster and the distance metric is warped to best suit the constraints.

Another form of semi-supervised clustering is distance metric learning which is used to learn the distance metric of the input space of the data given the label information of some of the data points. Previous work has shown that distance metric learning can improve the performance of various learning tasks like clustering, classification etc. Xing et al. [23] present an algorithm which takes examples of similar and dissimilar points in the dataset in the form of pairwise constraints and learns a distance metric which respects these relationships. The learned distance metric will assign small distances between similar

points and large distance between dissimilar points. The metric learning problem is posed as a convex optimization problem of finding a distance metric which minimizes the distance between similar points. They show that the learned metric significantly improves clustering performance. Similarly Wagstaff et al. [21] modify the k-means clustering algorithm to incorporate the pariwise constraints. The new algorithm named constrained k-means clustering ensures that none of the constraints are violated while updating the cluster assignments of the data points. An attempt to assign a point to its closest cluster C will fail if there is another similar point assigned to C' or a dissimilar point assigned to C. The next closest clusters are checked where the assignment can be made without breaking any constraints. Both of these approaches output a global distance metric which keeps all the points in the same class close together and those of different classes well separated. In the scenario where classes in the dataset exhibit multimodal behavior, these goals conflict and cannot be simultaneously satisfied. Yang et al. [25] propose a local metric learning algorithm which aims to optimize local compactness and local separability. The work learns the local distance metric in a probabilistic setup by using eigenvector analysis and bound optimization. Instead of focusing on all the pairwise constraints, the algorithm learns a metric which brings only pairs of the same mode of a class closer and separates nearby pairs from different classes.

2.4 Active Learning for Object Categorization

Active learning has been applied for category recognition previously. Kapoor et al. [12] describe a probabilistic model to categorize visual objects. They introduce an active learning paradigm into the probabilistic model to optimally select unlabeled points for interactive labeling. The strategy is to select the next point based on its level of uncertainty. A point which has the maximum posterior mean (distance from the classification boundary i.e. margin) and posterior variance is the most uncertain to categorize. The

knowledge of its label is most likely to maximize the discriminatory capability of the system and hence the best choice for interactive labeling. The work aims to learn as accurate a classifier in as few queries as possible. It is to be noted that the focus is on active learning rather than category detection.

Chapter 3 – Mean Shift

3.1 Introduction

Mean shift is a non-parametric clustering algorithm proposed by Fukunaga and Hostetler [6]. It is based on the concept of nonparametric estimation of probability density functions in which the value of a density function at a point can be estimated using the sample observations that fall within a small region around that point. The popular Parzen window technique generalized this concept for density estimation. The mean shift algorithm remained fairly obscure until researchers in vision recognized its usefulness in image segmentation [2, 3]. The subsequent sections follow the notations used in Dorin et al. [3].

3.2 Kernel Density Estimation

Let x_i, $i = 1, ..., n$ be n independent and identically distributed d-dimensional random vectors. The multivariate kernel density estimator computed at point x is given by

$$\hat{f}(x) = n^{-1} \sum_{i=1}^{n} K_H(x - x_i) \tag{3.1}$$

where

$$K_H(x - x_i) = |H|^{-1/2} K(H^{-1/2} x) \tag{3.2}$$

H is a symmetric positive definite d x d bandwidth matrix. $K(x)$ is d-variate kernel that is non-negative and integrates to one. It satisfies the conditions for asymptotic unbiasedness, consistency, and uniform consistency of the gradient of the density estimate

[6]. One of the widely used kernels are the radially symmetric kernels obtained by rotating a univariate kernel in R^d. The kernel is of the form

$$K(x) = c_{k,d}k(\|x\|^2) \tag{3.3}$$

$k(x)$ is called the profile of kernel $K(x)$ for $x \geq 0$ and $c_{k,d}$ is a normalization constant that makes $K(x)$ integrate to one [2]. In order to reduce the complexity of estimation, H is assumed to be either of the form of a diagonal matrix $H = diag[h_1^2, ..., h_d^2]$ or proportional to the identity matrix $H = h^2 I$ where $h > 0$. Assuming H to be proportional to the identity matrix, the multivariate kernel density estimator in 3.1 becomes

$$\hat{f}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{x - x_i}{h}) \tag{3.4}$$

Various multivariate kernel functions can be used for density estimation using the above expression. Two commonly used kernels are the multivariate Gaussian kernel and the Epanechnikov kernel. We will be using the Gaussian kernel function with zero mean and identity covariance matrix.

$$K_N(x) = (2\pi)^{-d/2} \exp(-\frac{1}{2}\|x\|^2) \tag{3.5}$$

whose profile is given by

$$k_N(x) = \exp(-\frac{1}{2}x) \qquad x \geq 0. \tag{3.6}$$

The density estimator expression based on the multivariate normal kernel will now become

$$\hat{f}(x) = \frac{c_{k,d}}{nh^d} \sum_{i=1}^{n} k(\|\frac{x - x_i}{h}\|^2) \tag{3.7}$$

3.3 Density Gradient Estimation

The density gradient estimator can be obtained by the gradient of the density estimator in 3.7 [3]. It is given by,

$$\begin{aligned} \bigtriangledown \hat{f}_{h,K}(x) &= \frac{2c_{k,d}}{nh^{d+2}} \sum_{i=1}^{n} (x - x_i) k^{'}(\|\frac{x - x_i}{h}\|^2) \\ &= c_{k,g} \hat{f}_{h,G}(x) \left[\frac{\sum_{i=1}^{n} x_i g(\|\frac{x - x_i}{h}\|^2)}{\sum_{i=1}^{n} g(\|\frac{x - x_i}{h}\|^2)} - x \right] \end{aligned}$$

where $g(x) = -k^{'}(x)$, is the profile of kernel $G(x)$. This makes kernel $K(x)$ the shadow of $G(x)$ [2]. $\hat{f}_{h,K}(x)$ is the density estimate with the kernel G. $c_{k,g}$ is the normalization constant. The second term is the mean shift which is the difference between the weighted mean, using the kernel G for weights, and x, the center of the kernel.

$$m(x) = \frac{\sum_{i=1}^{n} x_i g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)} - x \tag{3.8}$$

Using the normal kernel, the mean shift vector becomes

$$m(x) = \frac{\sum_{i=1}^{n} x_i \exp\left(-1/2 \left\| \frac{x - x_i}{h} \right\|^2\right)}{\sum_{i=1}^{n} \exp\left(-1/2 \left\| \frac{x - x_i}{h} \right\|^2\right)} - x \tag{3.9}$$

$m(x)$ is proportional to the normalized density gradient and always points toward the steepest ascent direction of the density function.

Fukunaga et al. [6] derive another mean shift formulation using a different kernel function which satisfies the conditions for asymptotic unbiasedness, consistency, and uniform consistency of the gradient estimate. The kernel function is of the form

$$K(x) = \begin{cases} c(1 - \|x\|^2) & \|x\|^2 \leq 1 \\ 0 & \|x\|^2 > 1 \end{cases} \tag{3.10}$$

where c is the normalizing constant given by

$$c = \pi^{-n/2} \left(\frac{n+2}{2}\right) \Gamma\left(\frac{n+2}{2}\right) \tag{3.11}$$

The density gradient estimator will now be of the form

$$\begin{aligned} \bigtriangledown \hat{f}_{h,K}(x) &= \frac{2c_{k,d}}{nh^{d+2}} \sum_{x_i \in S_h(x)} (x_i - x) \\ &= \left(\frac{m}{n\upsilon_h(x)}\right) \frac{n+2}{h^2} \left(\frac{1}{k} \sum_{x_i \in S_h(x)} (x_i - x)\right) \end{aligned}$$

where

$$\upsilon_h(x) \equiv \int_{S_h(x)} dy = \frac{h^n \pi^{n/2}}{\Gamma(n + 2/2)} \tag{3.12}$$

is the volume of the region

$$S_h(x) \equiv \left\{ y : \|y - x\|^2 \leq h^2 \right\} \tag{3.13}$$

and m is the number of data points falling within the region $S_h(x)$ about x. The second term in the density gradient estimator is the mean shift given by

$$m(x) \equiv \frac{1}{k} \sum_{x_i \in S_h(x)} (x_i - x) \tag{3.14}$$

The gradient estimator in 3.12 can be easily converted into a normalized gradient estimator by taking the proportionality constant$\left(\frac{m}{n\upsilon_h(x)}\right)$ to the left side and using the properties of the function ln y. The estimate of the normalized gradient is of the form

$$\bigtriangledown ln \hat{f}_{h,K}(x) \equiv \frac{n+2}{h^2} m(x) \tag{3.15}$$

The standard mean shift algorithm iteratively performs computation of the mean shift vector $m(x^k)$ and then updating the current position $x^{k+1} = x^k + m(x^k)$ until reaching the stationary point i.e. the cluster center.

3.4 Clustering Applications

Fukunaga et al. [6] describe clustering using mean shift as assigning each data point to the nearest mode along the direction of the gradient. To do this the data point is shifted by some amount proportional to the gradient at the data point. Each data point is transformed recursively according to the following formula

$$x^{i+1} = x^i + a \bigtriangledown ln\hat{f}_{h,K}(x) \tag{3.16}$$

where a is an appropriately chosen positive constant to guarantee convergence. a is selected to be equal to $\left(\frac{h^2}{n+2}\right)$ to get the next point to which the data point has to be shifted. The new shifted point is given by

$$x^{i+1} = \frac{1}{m} \sum_{x_j^i \in S_h(x^i)} x_j^i \tag{3.17}$$

Comaniciu et al. [3] state and prove a theorem which shows that if the kernel K has a convex and monotonically decreasing profile, the sequences $\{y_j\}_{j=1,2...}$ and $\{\hat{f}_{h,K}(j)\}_{j=1,2...}$ converge and $\{\hat{f}_{h,K}(j)\}_{j=1,2...}$ is monotonically increasing, where sequences $\{y_j\}_{j=1,2...}$ are successive locations of the kernel G given by

$$y_{j+1} = \frac{\sum_{i=1}^{n} x_i g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)} \qquad j = 1, 2... \tag{3.18}$$

y_{j+1} is the weighted mean at y_j computed with kernel G and y_1 is the center of the initial position of the kernel.

This theorem implies that the magnitude of the mean shift vector converges to zero and that the trajectories of such gradient methods are attracted by local maxima if they are unique stationary points. These implications indicate a practical algorithm for mode detection:

Algorithm 3: Mode detection

1 Use mean shift algorithm to find the stationary points of $\hat{f}_{h,K}$;
2 Retain the local maxima and prune the rest of the points. The local maxima are the cluster modes;

3.5 Fast Mean Shift

Mean shift is a computationally expensive method. The calculation of the mean shift vector for a given point requires a scan of the entire dataset to find out which points are covered by the kernel centered at the point. This operation is repeated M times until the mean shift vector for the point converges to zero. This procedure has to be applied for all the points in the dataset. Thus the complexity of mean shift algorithm is $O(n^2dM)$ where n is the number of points in the dataset and d is the number of dimensions. Many algorithms have been proposed to speed up the mean shift computations. For example, Georgescu et al. [7] propose an approximation technique named locality-sensitive hashing (LSH) to reduce the computational complexity of mean shift especially with high dimensional data. The work defines a data structure to improve the efficiency of the neighborhood queries. The idea is to partition the data multiple times based on different dimensions. The points which fall in the same partition as the query point are considered its neighbors. To accomplish this, the data is tessellated L times with random partitions each defined by K inequalities .Each partition maintains K pairs of random numbers (d_k, v_k) which partitions the data according to the inequality $x_{i,d_k} \leq v_k$ where $i = 1, ...n$ and x_{i,d_k} is the selected coordinate for the data point x_i. For each point x_i a

K dimensional boolean vector is generated by each partition. Each point simultaneously belongs to L partitions and their union gives the neighborhood of the point. The optimal L and K values are derived from the data by performing a numerical search procedure. The values of K and L which minimize the query time on a subset of the data points are chosen. This approach suffers from having too many tuning parameters like K and L.

Another algorithm which approximates the mean shift procedure is described by Yang et al. [24]. The work puts forward an improved fast Gauss transform (IFGT) to efficiently estimate sums of Gaussians in higher dimensions and in turn apply it to the mean shift algorithm to obtain a run time of linear complexity. The dataset is first clustered to K clusters using the simple greedy farthest-point clustering algorithm [8]. The algorithm then loops over each point and its cluster center pair, evaluating the precomputed Taylor coefficients for clusters whose distance from the point is within a threshold value. IFGT is mainly restricted by the use of only Gaussian kernels.

3.5.1 kd-trees

kd-trees [17] can also be used to speed up mean shift by approximating the calculation of mean shift vector. A kd-tree is a binary space partitioning tree that recursively splits the whole input space into partitions. Each kd-tree node represents a partition of the input space. Deng et al. [11] discuss kernel regression and use kd-trees for multi-resolution hierarchical structuring of data to summarize the dataset. This permits to query the data set with the same flexibility as a conventional linear search, but at greatly reduced computational cost. Given the dataset of input points and their labels, kernel regression aims to find the label ($y^{est}(x_q)$) for the given query point(x_q). For this it needs to compute the weighted average of all points in the dataset and the points closest to the query point

are assigned the largest weights.

$$y^{est}(x_q) = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \qquad \text{where } w_i \text{ is the weight of the ith data point.} \tag{3.19}$$

The use of kd-trees in kernel regression is based on the idea that if we have a group with k datapoints in which we know that all the weights in the group with respect to the query point x_q are close to the same value w then approximate values of $\sum w_i x_i$ and $\sum w_i$ can be used. This can be done in constant time without the need to sum the individual members of the group. A kd-tree supplemented with extra cache information is used as an implementation of this grouping idea. The nodes in the kd-tree act as hyper-rectangles enclosing all the points of the node and are treated as groups. The parts of the state space with a highly varying weight function provide limited opportunity for accurate approximation. As a result, the leaves or lower-level nodes (i.e. the small-sized partitions) are selected as groups. At the parts of the state space with little variance in the weight function, we have more opportunities for accurate approximation and the higher-level internal nodes are treated as groups. To compute $\sum w_i x_i$ and $\sum w_i$ a top-down search of kd-tree is performed where at each node a decision is made either to treat all the points in the node as a group (cutoff) or recursively continue the search of the children (recurse). Considering x_q and the hyper-rectangle of the node, the minimum and the maximum distance of x_q from the hyper-rectangle i.e. D_{min} and D_{max} can be easily computed in turn, providing the maximum and minimum possible weights w_{min} and w_{max} of any data points owned by the node. If the values of w_{min} and w_{max} are close enough then the cutoff option is taken. The algorithm makes a cutoff if

$$\frac{(w_{max} - w_{min})N_B}{\text{weight so far in search}} < \tau \tag{3.20}$$

where τ is a system constant. This cutoff is similar to the simple cutoff rule $w_{max} - w_{min} < \epsilon_{max}$ and also makes sure that a larger total error will not occur if the node contains many points than if the node contains only a few points.

Similar to kernel regression, finding the mean shift vector requires the calculation of the weighted average of points within the bandwidth distance of the query point. Using the above kd-tree methodology, the points within the bandwidth distance can be found quickly using the minimum and maximum distances of query point from the hyper rectangles. Their weights can be approximated based on cutoff value. Thus the computation of the mean shift can be sped up.

3.5.2 Dual Trees

Wang et al. [22] introduce another fast way of computing mean shift using dual kd-trees. The core idea is to compare groups of query-reference point pairs rather than the single kd-tree strategy of comparing a single query point against a group of reference points. Here, the reference points are simply the data points from the original data set while the query points are locations that the mean shift operations are applied to; the query points need not be actual data points. At each iteration of mean shift, the query points are moved to according to the mean shift vector.

To accomplish the idea of dual trees, two kd-trees are built using query points and reference points separately. These trees are traversed simultaneously and each reference tree node's weight contribution to a query tree node is recursively updated by comparing these two nodes and their children. Once both trees have been traversed, the dual-tree algorithm produces a memory-efficient cache of the mean shift values of all the query points. Thus, for every iteration of mean shift, the query tree has to be rebuilt since the query points have been changed while the reference tree remains fixed. When more

mean shift iterations are needed to find the stationary modes of the data, the rebuilding of the query tree takes up the majority of the run time of the algorithm. As a result, this method is ideally suited for datasets which need a small number of mean shift iterations. Image data sets where each datum represents the normalized CIE LUV color space are good examples that require very few mean shift iterations.

3.6 Bandwidth Selection

The bandwidth parameter h plays an important role in the mean shift algorithm. An optimal bandwidth value is necessary for the correct estimation of the density function. Smaller h values produce a spiky estimate of the density function resulting in splitting of larger clusters into smaller ones. Larger h values lead to over-smoothed estimates and results in the grouping of several clusters together into one cluster. In fact, the bandwidth choice is more critical than that of the kernel function K. Figure 3.1 - 3.4 show the effect of using different bandwidth values with mean shift on a 3 class synthetic dataset. The mean shift for each point (denoted by a + sign) is displayed by the lines and the cluster modes are denoted by triangular shapes.

Different criteria have been used to evaluate the optimality of the bandwidth parameter [10, 20]. All the criteria provide a measure of the distance between the true density f and the estimated density $\hat{f}_h$. The value of h which minimizes the criteria is selected. The criteria commonly used are Integrated Squared Error (ISE), Mean Integrated Squared Error (MISE) and Asymptotic Mean Integrated Squared Error (AMISE) [10].

The bandwidth values obtained by any of the above criteria can be treated as optimal for density estimation. However, all these criteria depend upon on the unknown density f and hence cannot be used in practice. Several bandwidth estimation techniques have be proposed to deal with this problem. One of the most commonly used heuristics is the "Rule of Thumb".

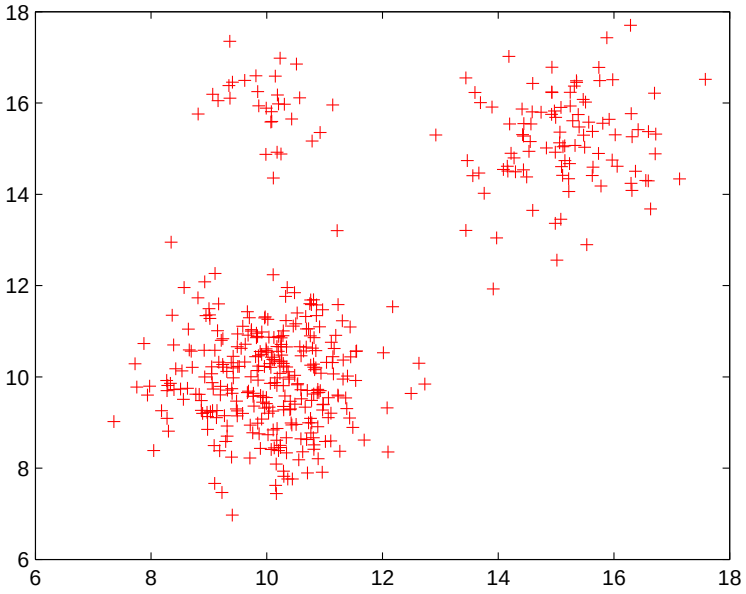

Figure 3.1: 3 class dataset

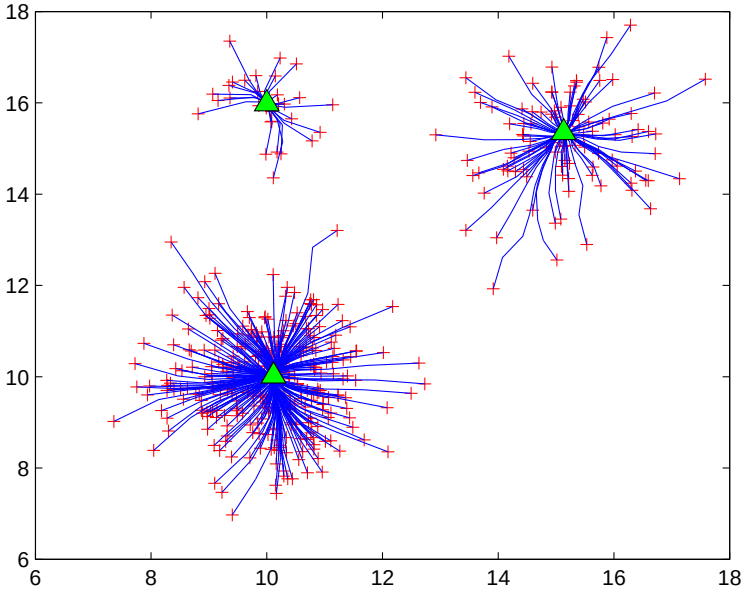

Figure 3.2: Mean shift using an ideal bandwidth

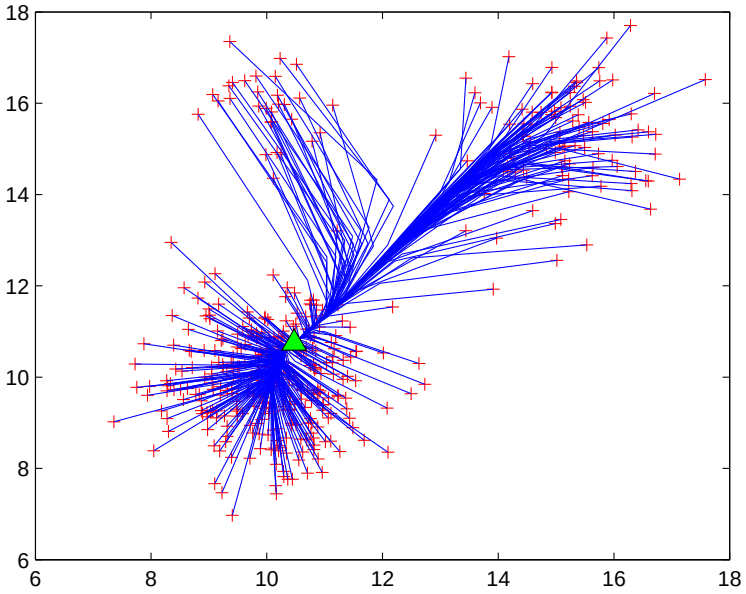

Figure 3.3: Mean shift using too large a bandwidth

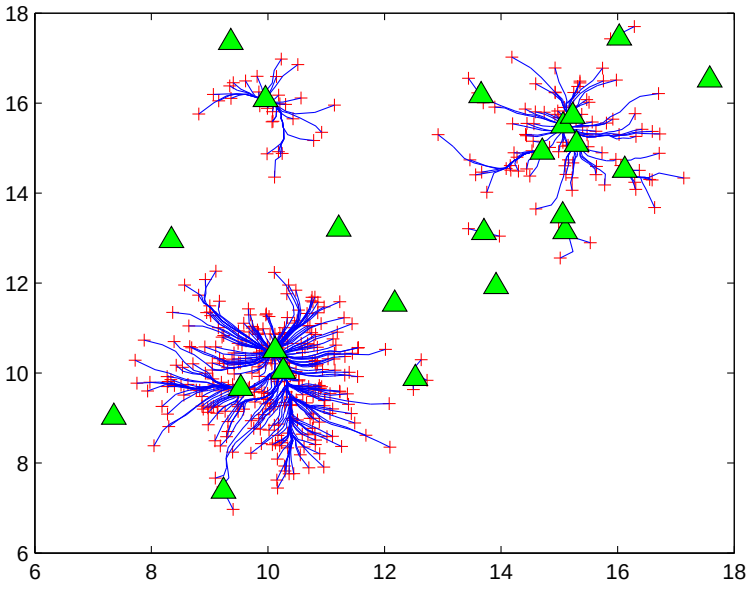

Figure 3.4: Mean shift using too small a bandwidth

3.6.1 Rule of Thumb

The Rule of Thumb (ROT) [18] methodology is based on AMISE. It replaces the unknown density f with a reference distribution function which is rescaled to have a variance equal to the sample variance. If the reference distribution function is a normal function and K is a Gaussian kernel, then the Rule of Thumb yields

$$\hat{h}_{rot} = 1.06\hat{\sigma}n^{1/5} \tag{3.21}$$

where $\hat{\sigma}^2$ is the sample variance.

Other bandwidth selection techniques include Maximal Smoothing Principle, Likelihood Cross Validation, Least Squares Cross Validation and Plug-in methods [10].

Chapter 4 – Methodology

4.1 Mean Shift - Interleave

The Interleave algorithm has a number of drawbacks. First, it requires the user to specify K, which is the number of components in the mixture model. Secondly, one typically needs to impose a parametric form on the mixture components in order to perform an EM-based clustering routine. For instance, a Gaussian mixture model assumes that the clusters take on an elliptical shape. There is, of course, no apriori reason to believe that the clusters in an arbitrary data set would take the form of a Gaussian.

To overcome these two problems, we initially approached the rare category detection problem by proposing a nonparametric version of Interleave called MS-Interleave (for Mean Shift Interleave). In this nonparametric version, we use mean shift instead of EM as the underlying clustering algorithm. Pseudocode for MS-Interleave is presented in Algorithm 4. In MS-Interleave, mean shift [3, 6] is first used to cluster the data set. The bandwidth for mean shift is chosen by applying the "Rule of Thumb" for bandwidth estimation. Mean shift outputs the cluster modes, the list of points belonging to each cluster and the total distance each point moved during the mean shift procedure (which we will call the total mean shift distance). Note that we are applying a blurring mean shift procedure [2], in which the reference and query points are the same initially and the converged query point locations serve as the reference data points in the next iteration. As in the original Interleave algorithm, each cluster maintains a list containing the points owned by that cluster. Each list is sorted in increasing order of the degree of ownership that each cluster expresses over the data points in the list. The degree of ownership is measured in terms of the total mean shift distance moved by each data point to the cluster

mode; the further the total mean shift distance, the less the degree of ownership by the cluster mode. The algorithm cycles through these sorted lists in a round-robin fashion and constructs a query list by picking the top data point from the current list. This top data point is placed in the next position of the query list. This process continues until the query list contains all the points of the data set.

Algorithm 4: Mean Shift - Interleave approach to rare category detection

```
Let S = {x_1, x_2, ...., x_n} be the data set where x_i ∈ ℜ^d and h be the bandwidth
parameter;
h = GetBandwidth(S,' RuleofThumb');
(ClusterCenters[m],MeanShiftDist[n],PointsList[m]) = MeanShift(S, h);
where m is the number of clusters;
for i ← 1 to m do
    Sort(MeanShiftDist, 'DecreasingOrder', PointsList[i]);
    where 'Sort' sorts the PointsList[i] based on their mean shift distances to
    the i^th cluster center;
end
OutputList = ();
for j ← 1 to n do
    for i ← 1 to m do
        if If all points in PointsList[i] are added to OutputList then
            continue;
        end
        OutputList.PushBack(PointsList[i][j]);
    end
    if If all n points are added to OutputList then
        break;
    end
end
while not all classes have been discovered do
    Pick the top point in OutputList and select it for labeling;
end
```

The performance of MS-Interleave, and the quality of mean shift in general, is highly dependent on the bandwidth parameter. A well-chosen bandwidth results in a very good density estimate of the underlying data distribution. There are a variety of techniques for data driven bandwidth selection [10]. Some of these techniques select a single scalar band-

width parameter like Uniform Likelihood Cross Validation (ULCV) and Least Squares Cross Validation (LSCV). Other techniques such as Rule of Thumb, Maximal Smoothing Principle and plug-in bandwidth estimation produce a diagonal bandwidth matrix. Many of these bandwidth estimates are expensive to compute. We selected the Rule of Thumb technique as it can be computed quickly and the results using bandwidth estimation from other techniques did not show much improvement in the performance of MS-Interleave. Nevertheless, MS-Interleave does not perform well because it is difficult to efficiently determine the right bandwidth for the density estimation. Typically, the bandwidth selection techniques produce a large number of modes found by the mean shift algorithm, which in turn results in poorly selected data points for queries to the user.

4.2 Hierarchical Mean Shift for Rare Category Detection

In order to avoid the problem of selecting a single optimal bandwidth, we turn to a rare category detection method based on hierarchical mean shift. Hierarchical mean shift is closely related to scale-space theory [13], which is a multi-scale data representation framework designed to model how the human visual system sees details in an image as the image is progressively smoothed. Leung et al. [13] develop a clustering approach based on scale space theory in which a repeated blurring process creates a hierarchical clustering.

Using the notation in [13], consider a two-dimensional data set given by a continuous mapping $p(x) : R^2 \rightarrow R$. In scale space theory, $p(x)$ is embedded into a continuous family $P(x, \sigma)$ of gradually smoother versions of the data. The scale $\sigma = 0$ corresponds to the original data set. $P(x, \sigma)$ is the convolution of $p(x)$ with the Gaussian kernel $g(x, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\|x\|^2/2\sigma^2}$ given by

$$P(x,\sigma) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2\pi\sigma^2}e^{-\frac{\|x-x_i\|^2}{2\sigma^2}} \tag{4.1}$$

The family $P(x,\sigma)$ can be described as the Parzen estimation with the Gaussian window function. At each σ, $P(x,\sigma)$ is a smooth distribution function where the clusters and their centers can be determined by finding the gradient $\bigtriangledown_x P(x,\sigma)$. This is the same methodology explained in the section 3.3 where σ is denoted as the bandwidth parameter (h).

Each $P(x,\sigma)$ represents a clustering where each data point is deterministically assigned to a cluster. As the scale σ or the bandwidth parameter h is changed, we get a series of clusters arranged in a hierarchical format. The output in our case is what Leung et al. call a nested hierarchical clustering, in which a dendrogram-like hierarchy illustrates a sequence of clusters with each cluster being a partition of the data set. In this clustering, each small cluster either remains unchanged or is nested inside a larger cluster at each level of the dendrogram. Once the assignment is made, each data point is not allowed to change the cluster membership. DeMenthon et al. [4] use this technique for spatio-temporal segmentation of video and call it hierarchical mean shift.

In this section we describe a hierarchical clustering approach for detecting rare categories. Mean shift is used as the clustering algorithm for nested hierarchical clustering of the data. The procedure consists of 3 phases

- Data standardization
- Building the dendrogram
- Calculating the validity criterion for all clusters and querying data points.

4.2.1 Data Standardization

The data sets are first standardized before being used to prevent one dimension dominating the distance calculations used in the analysis. Sphering is perfomed on the datasets to achieve the standardization using the following transformation [14].

$$Z_i = \Lambda^{-1/2} Q^T (x_i - \bar{x}) \qquad i = 1, ..., n \tag{4.2}$$

In the equation above, $\bar{x}$ is the sample mean, the columns of Q are the eigen vectors of the sample covariance matrix, Λ is a diagonal matrix of corresponding eigenvalues and x_i is the i^{th} data point in the dataset. Note that the diagonal elements in the sample covariance matrix with the value 0 are replaced with a small value $1e-5$ before the eigenvectors and values are computed. This is done to prevent imaginary values in the sphered data set.

4.2.2 Building the Dendrogram

In the first phase, a dendrogram is built from the data set by repeatedly blurring the data set using mean shift. Initially, each data point is treated as a separate cluster and this clustering forms the lowest level in the dendrogram. We compute the initial bandwidth value for mean shift by computing the minimum non-zero distance between any two points in the data set. The bandwidth matrix is computed by multiplying this minimum distance with the identity matrix. For convenience, we will refer to the bandwidth as a scalar value. Then, mean shift with this bandwidth matrix is applied to the individual data points at the first level to produce clusters. The modes of these clusters now become the data points for mean shift at the next level of the dendrogram. By summarizing the data points in terms of their modes, we are in effect blurring the data. This blurring effect continues

at each level, with the bandwidth increasing by multiplying the current bandwidth scalar value by a constant factor k, where $k = 1.1$ is used in all of our experiments. The entire process terminates when there is only one cluster left. The single kd-tree approach was used in the computation of the mean shift vectors for the dataset as opposed to dual kd-trees which required rebuilding the query tree for each mean shift iteration (refer to Section 3.5). Figure 4.1 shows a dendrogram for a synthetic dataset with 30 examples from 8 classes.

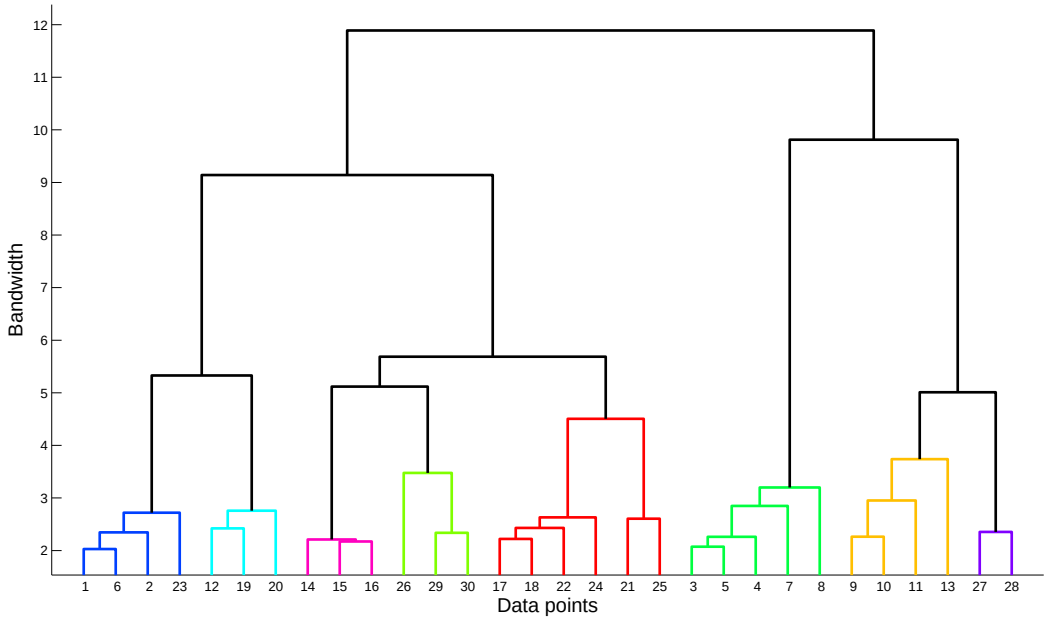

Figure 4.1: Dendrogram for a synthetic dataset with 8 classes

The algorithm to build the dendrogram is described in Algorithm 5.

4.2.3 Cluster Validity Criterion

One of the advantages of hierarchical mean shift is its ability to preserve "the structure and integrity of the outliers in the clustering process" [13]. Leung et al. define different cluster validity metrics such as lifetime, compactness, isolation and outlierness to measure

Algorithm 5: Hierarchical Mean Shift - Building the dendrogram

```
1 Let S = {x_1, x_2, ...., x_n} be the data set where x_i ∈ ℜ^d and h be the bandwidth
  parameter;
2 Set h = minimum non-zero distance between any two points in the data set;
3 Set Dendrogram[0].ClusterCenters = S;
4 ht = 0;
5 repeat
6 |   ht = ht + 1; h = h * 1.1;
7 |   (ClusterCenters,MeanShiftDist,PointsList) =
  |   MeanShift(Dendrogram[ht].ClusterCenters,h);
8 |   Save (ClusterCenters, MeanShiftDist, PointsList) in Dendogram[ht++]
  |   while keeping track of cluster membership of all points at each level of
  |   dendrogram;
9 until size(Dendrogram[ht].ClusterCenters) == 1 ;
```

the "goodness" of a cluster in the dendrogram. This section describes two types of criteria, each of which can be used with hierarchical mean shift resulting in two different approaches to rare category detection. We refer to these two approaches as Hierarchical mean shift - Outlierness (HMS-Out) and Hierarchical mean shift - CompactIsolation (HMS-CI).

4.2.3.1 Outlierness

The outlierness criterion is based on the concept of the lifetime of a cluster. The lifetime of a cluster is defined as the range of logarithmic scales over which the cluster survives. Similarly, we can define the lifetime in the hierarchical mean shift scenario as the range of logarithmic bandwidths over which the cluster survives i.e. the logarithmic difference between the point when the cluster is formed and the point when the cluster is absorbed into or merged with other clusters. The outlierness metric helps in identifying outliers in the data. The cluster which has a long lifetime and fewer points will have high outlierness value. Intuitively, rare categories are more likely to have high outlierness value as they tend to have fewer points and longer lifetime.

$$outlierness_i = \frac{lifetime\ of\ C_i}{number\ of\ data\ points\ in\ C_i} \tag{4.3}$$

The dendrogram built in the first phase is traversed and the list of unique clusters is formed. The same cluster can exist in more than one level of the dendrogram and all the different occurrences are treated as one unique cluster. The outlierness value for each cluster is calculated and the list is sorted in the decreasing order of their outlierness value. To query a cluster, we ask the user to label its representative data point, which is the data point closest to the mode of the cluster.

4.2.3.2 Compactness-Isolation

Compactness and isolation are another set of criteria that can measure the quality of a cluster. Intuitively, a cluster is well-defined if the distance between the data points inside the cluster is small (ie. it is compact) and those outside is large (ie. it is isolated). Given p_i is the cluster center of C_i and h is the scalar bandwidth parameter, the *isolation* and *compactness* of C_i can be calculated as

$$isolation = \frac{\sum_{x \in C_i} e^{-\|x-p_i\|^2/2h^2}}{\sum_x e^{-\|x-p_i\|^2/2h^2}} \tag{4.4}$$

$$compactness = \frac{\sum_{x \in C_i} e^{-\|x-p_i\|^2/2h^2}}{\sum_{x \in C_i} \sum_{j \neq i} e^{-\|x-p_j\|^2/2h^2}} \tag{4.5}$$

The *isolation* and *compactness* metric are close to one for a good cluster. These two criteria can be combined into one single $Compactness - Isolation$ criterion (CI) for the cluster which is simply the sum of *isolation* and *compactness* values of the cluster. This criterion is calculated for each cluster in the dendrogram and stored in a list sorted in

decreasing order. The lowest level of the dendrogram will provide the worst case run time complexity of calculating this criterion value for all clusters at that level which is $O(n^2)$.

4.2.4 Querying Data Points

The list created by using any of the above two criteria is traversed in the third phase. Clusters with higher validity criterion are selected first and the points which have the least mean shift distance from the mode of the clusters are picked for labeling. If the point has already been selected for labeling then the cluster is skipped. The entire algorithm is described in Algorithm 6.

Algorithm 6: Hierarchical Mean Shift for rare category detection

```
1  Build dendrogram using hierarchichal mean shift. Let ht be its height.
2  for i ← ht to 0 do
3  |   For each cluster C at height i calculate its validity criteria value if it is not
   |   done previously and add it to the list;
4  end
5  Sort list in decreasing order of the validity criteria values of the clusters;
6  while not all classes have been discovered do
7  |   Let C be the next cluster to select in list;
8  |   Let p be the data point which has least mean shift distance from the cluster
   |   center of C_i;
9  |   If p has not been selected for labeling then do so. Else continue.
10 end
```

4.2.4.1 Tiebreaker

The sorted list may contain cluster entries with the same criterion values. This can happen for clusters at the lower levels of the hierarchy as low bandwidth values lessen the effect of neighboring points for a given cluster resulting in high compactness and isolation values. In such cases a tiebreaker condition can be applied. For each cluster with the same criterion values, the distance between the cluster mode and each data point already

labeled by the expert is calculated. Two different tiebreakers can be considered. The point with the highest average of the distances can be picked first for labeling. This is highest average distance tiebreaker (HAD). The point with the highest minimum distance can be picked first for labeling. This is called highest minimum distance tiebreaker (HMD). The time complexity for running the tiebreaker is bounded by $O(ab)$ where a is the number of clusters with the same criterion value and b is the number of points already labeled by the expert. Both a and b are generally far less than n, the size of the dataset.

Chapter 5 – Results

5.1 Experimental Setup

5.1.1 Data

The data consists of various real data sets taken from the UCI data repository [15]. Table 5.1 has a summary of their properties. All the datasets except for Abalone and Yeast are sub-sampled. This sub-sampling is done to create imbalanced data sets that suit the rare category detection problem where some classes dominate the data and the remaining classes have only a few records. Shuttle is randomly sub-sampled from the original dataset to produce a smaller data set with 4000 examples. The original Optical Digits, Optical Letters, Image Segmentation (Statlog) datasets contain almost the same number of examples for each class. The class distributions for the Optical Digits and Image Segmentation datasets have been changed into a geometric series with the largest class owning half of the data and each subsequent class being half as small with the smallest class containing 8 examples; this is the same sub-sampling technique used by Pelleg and Moore [16]. The Optical Letters data set has been sampled in such a way that the two largest classes own half the data and the subsequent pairs of classes being half as small with the smaller class containing 8 examples. The data sets are first standardized before running the experiments.

5.1.2 Experiments

All the algorithms are evaluated based on the number of hints requested from the expert before being presented with at least one example from all the classes in the data set. The

Name	Dims	Records	Classes	Smallest class	Largest class
Abalone	7	4177	20	0.34%	16%
Shuttle	8	4000	7	0.02%	64.2%
Optical Digits	64	1040	10	0.77%	50%
Optical Letters	16	2128	26	0.37%	24%
Image Segmentation (Statlog)	19	512	7	1.5%	50%
Yeast	8	1484	10	0.33%	31.68%

Table 5.1: Properties of the data sets.

assumption with this performance metric is that a single example would help the expert to generalize about the class to which the example belongs. This evaluation technique is the standard metric used in rare category detection [16, 9]. Intuitively, the metric measures the effort the expert expends in order to discover all the classes in the data set.

We compare the results from the methods described in the previous chapters. The performance of 'Hierarchical mean shift - Outlierness' (HMS-Out), 'Hierarchical mean shift - CompactnessIsolation' (HMS-CI), 'Mean shift - Interleave' (MS-Interleave) on the 6 real world data sets is compared with existing rare category detection techniques NNDM [9] and Interleave [16]. The effect of the two tiebreakers 'Highest Average Distance' (HAD) and 'Highest Minimum Distance' (HMD) on the Hierarchical mean shift methods is also shown in the results. For Interleave we ran the experiment 10 times and reported the best result. The performance of Interleave depends on the initial random point to cluster assignments at the beginning of EM and Interleave can converge to a local optimum which is why we ran the experiment 10 times.

DATASET	HMS-CI	HMS-Out	NNDM	Interleave	MS-Interleave
Abalone	1195	603	**124**	193	1306
Shuttle	44	36	162	**35**	136
Optical Digits	**100**	160	576	117	174
Optical Letters	**133**	161	420	489	721
Image Segmentation (Statlog)	**18**	34	228	54	28
Yeast	**73**	103	88	111	83

Table 5.2: Number of hints needed to identify all classes

DATASET	HMS-CI +HAD	HMS-CI +HMD	HMS-Out +HAD	HMS-Out +HMD
Abalone	**93**	387	385	1066
Shuttle	32	39	28	**27**
Optical Digits	**100**	100	118	119
Optical Letters	**133**	133	182	423
Image Segmentation (Statlog)	20	**17**	124	65
Yeast	91	**73**	77	114

Table 5.3: Number of hints needed to identify all classes for the HMS methods using Highest Average Distance (HAD) and Highest Minimum Distance (HMD) tiebreakers

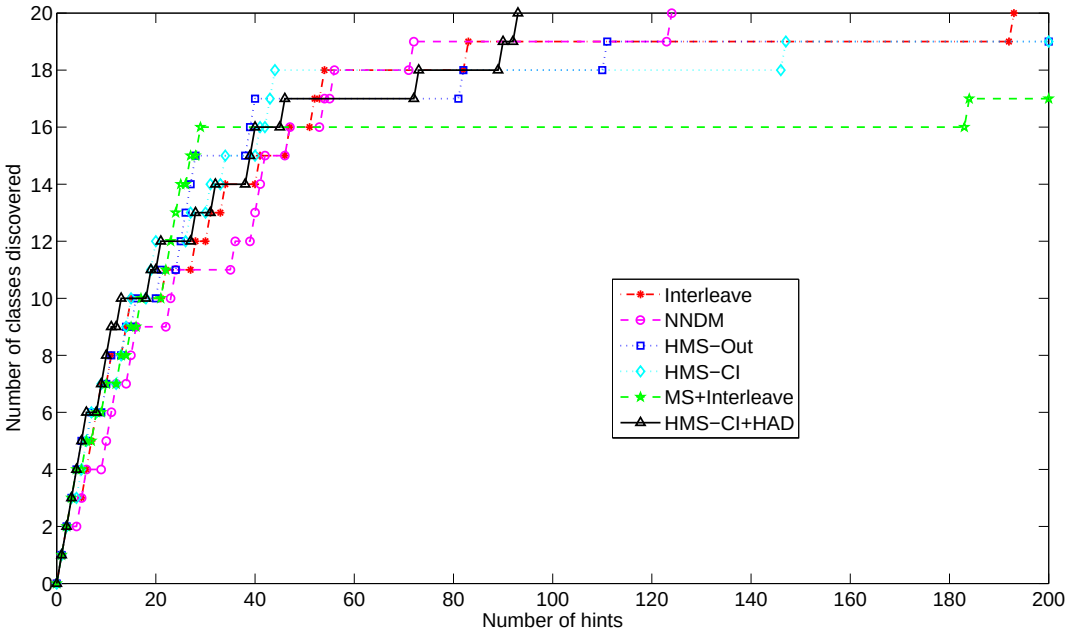

Figure 5.1: Learning curves for Abalone

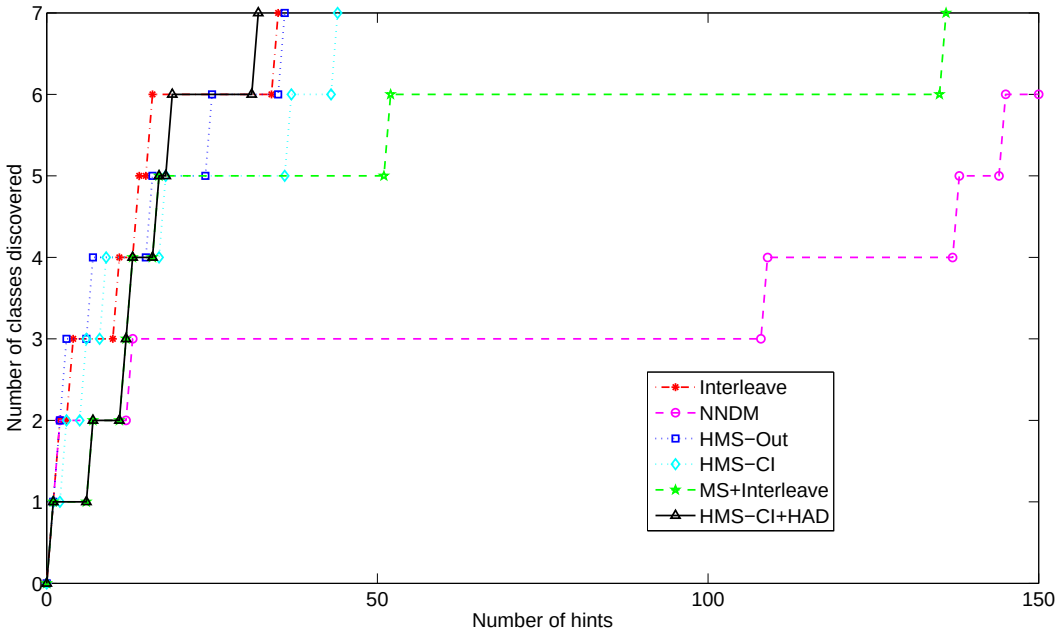

Figure 5.2: Learning curves for Shuttle

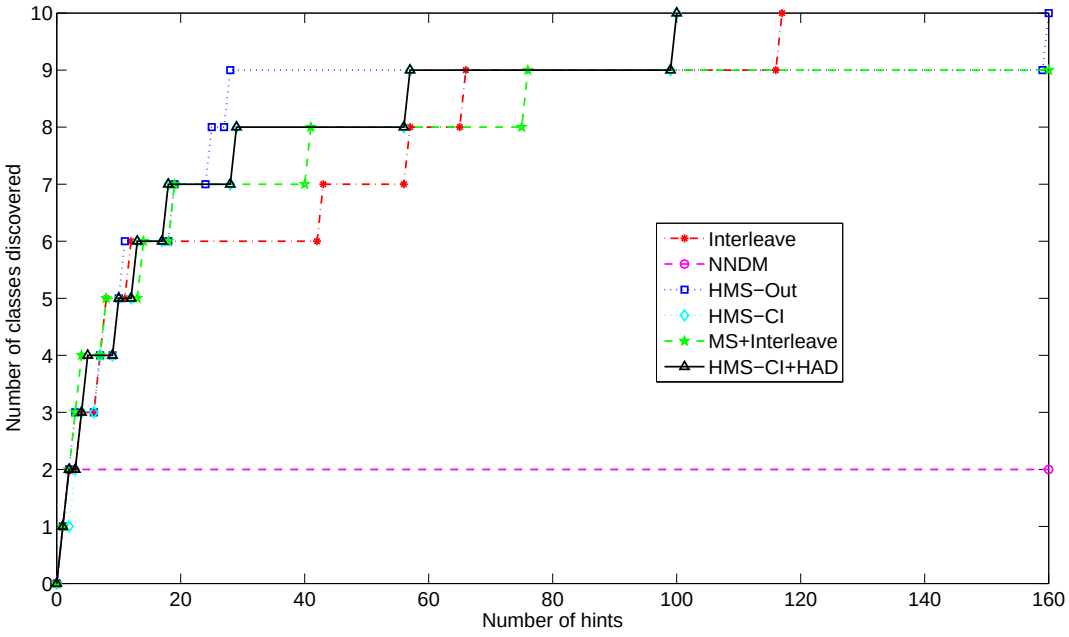

Figure 5.3: Learning curves for Optical Digits

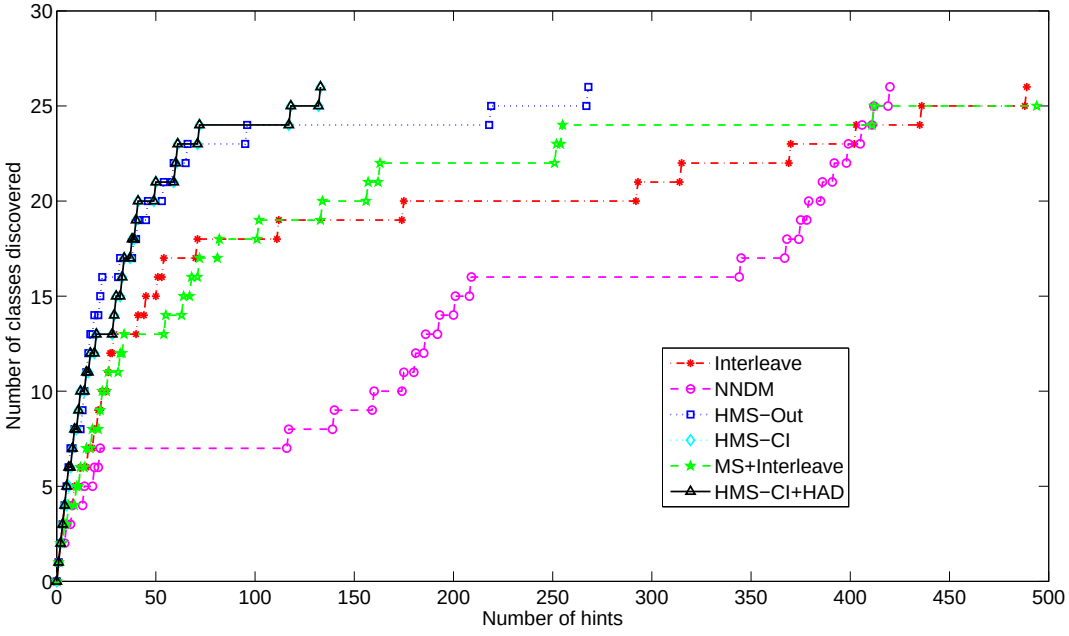

Figure 5.4: Learning curves for Optical Letters

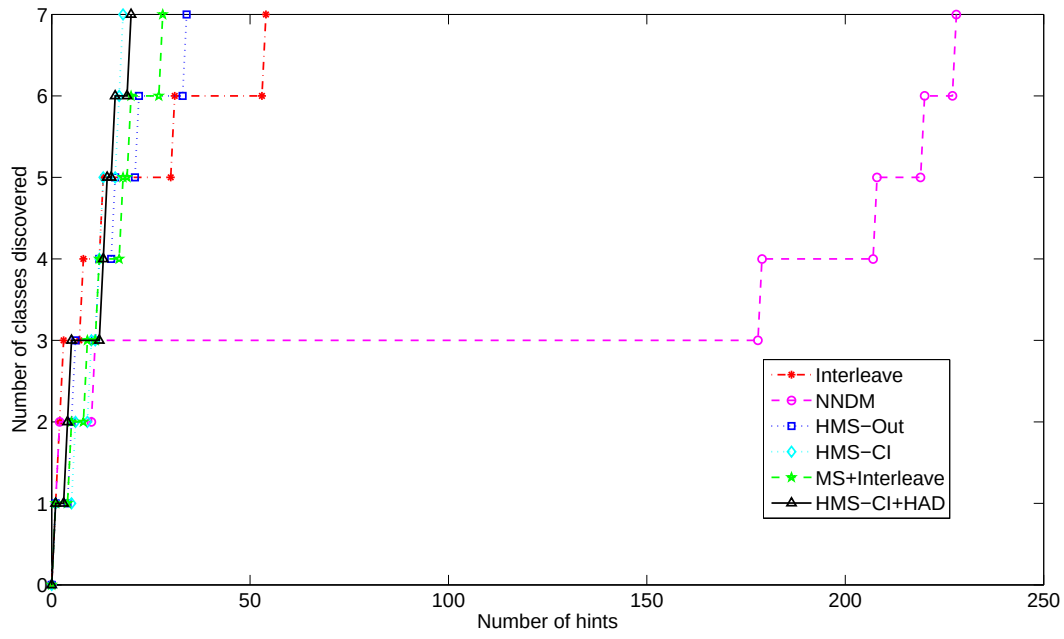

Figure 5.5: Learning curves for Image Segmentation (Statlog)

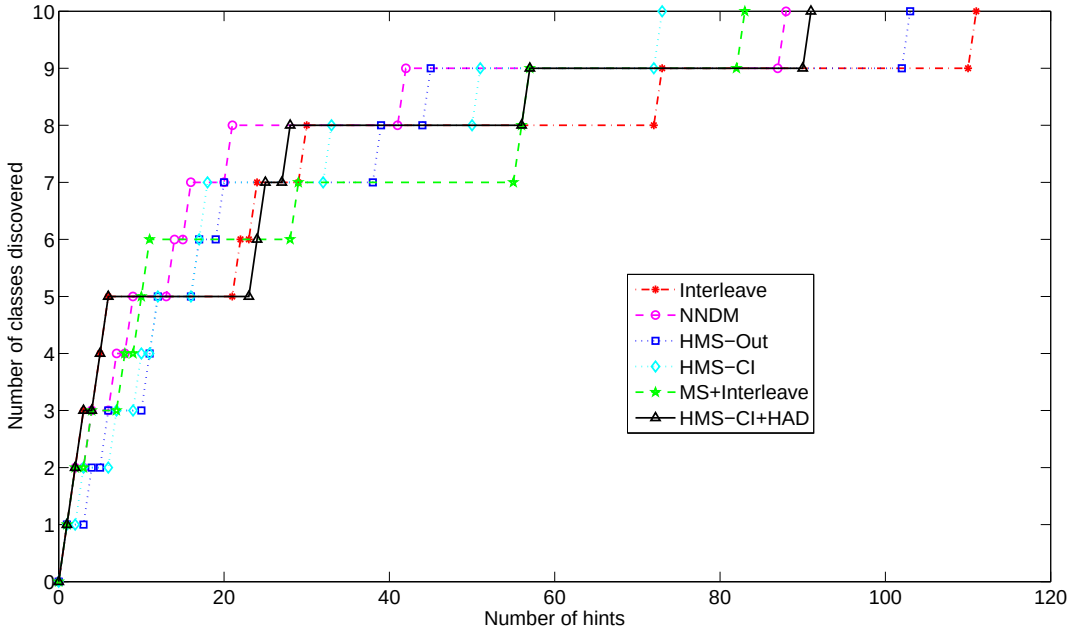

Figure 5.6: Learning curves for Yeast

Chapter 6 – Discussion

Table 5.3 shows that the hierarchical mean shift (HMS) approaches perform well. In the case of the Optical Letters and Image Segmentation data sets, both HMS-CI and HMS-Out perform better than the existing methods NNDM and Interleave. For the Yeast and Optical Digits datasets, it can be seen that HMS-CI produces better results than NNDM and Interleave.

In general it can be said that HMS-CI outperforms HMS-Out with the exception of Shuttle and Abalone. In case of the Shuttle dataset, both the HMS approaches perform only slightly worse than best performing method for this dataset. For Abalone, the HMS methods need more hints. From Figure 5.1 we see that the HMS methods find 18 of the total 20 classes more quickly than NNDM and Interleave, but fail to find the last 2 classes. Analyzing the Abalone dataset reveals that these classes are small compact clusters that are very close to more than one large cluster. As a result, in the dendrogram, their lifetime is low as they are merged into one of the larger clusters early on in the dendrogram building phase, resulting in low outlierness value. Hence HMS-Out fails to find these clusters quickly. Similarly in case of HMS-CI, the points from nearby larger clusters contribute substantially to the denominators of the compactness and isolation criteria of the small cluster, resulting in lower value for compactness-isolation.

The value of h also plays a crucial role in the calculations of compactness and isolation values. The smaller the value of h, the less the effect of neighboring points on the compactness and isolation values and vice versa. Thus, at higher levels of the dendrogram where h is large, the small compact clusters will have their criteria values affected widely by the

nearby larger clusters resulting in smaller compactness-isolation. To identify these small clusters, the lower level of the dendrogram where h values are small should be searched. However, at the lower levels of the dendrogram the data set is divided into many smaller sized clusters with higher compactness-isolation due to smaller h values, resulting in many clusters that are tied in terms of the compactness-isolation criterion values.

In the case of clusters having the same criterion value in the sorted list, a tiebreaker makes sure that we intelligently order the clusters with the same criterion values in the query list. Our HAD tiebreaker condition places the cluster mode with the highest average distance from the queried points in the next position of the query list. Another tiebreaker condition is to select the cluster mode with the highest minimum distance from the queried points. This means that cluster modes near already queried points are left to be queried later thus decreasing the possibility of picking hints from the classes that have already been discovered. In this way, tiebreakers use the labels that have been provided by the user to improve performance. The results show that the HAD tiebreaker helps in improving the performance of HMS-CI for Abalone. On the other hand, the HMD tiebreaker does not provide the same improvement. This is in accordance with the previous explanation given for Abalone that the hard to find small clusters are closer to many large clusters. The mode of the nearby large cluster is queried first and so the HMD tiebreaker would not select the mode of the nearby small cluster as it takes only the minimum distance into consideration. The highest minimum distance for the small cluster will be less as the queried mode of the larger cluster is close. The HAD tiebreaker avoids this pitfall as it takes all the queried points into consideration leading to better results.

The MS-Interleave generally performs poorly. This is due to its dependence on the value of the bandwidth parameter. It can be expected to perform well when the mean shift

algorithm is able to identify nearly all the clusters in the original dataset using the bandwidth obtained from one of the bandwidth selection techniques. For real-world datasets, experiments show that the bandwidth output by the selection techniques is generally too small. A small bandwidth results in mean shift identifying more clusters than the original number of clusters present in the dataset, as a large cluster would be divided into a group of separate small clusters by mean shift. For all the modes of these small clusters, the points with the highest mean shift distance will be points that originally belong to the same large cluster resulting in poor performance. The bandwidths returned for Abalone, Shuttle, Optical Digits and Optical Letters are small values thus requiring MS-Interleave to ask more hints to identify all the classes. For Yeast and Image Segmentation, mean shift is able to identify all the original clusters in the datasets approximately using the bandwidth returned by the selection technique. For these datasets, it can be seen that the results are close to the best performance.

6.1 Conclusion

We have proposed a rare category detection method using hierarchical mean shift. A dendrogram is built by successively running mean shift first on the dataset and then on the cluster centers using a series of bandwidth values. The scale space theory acts as theoretical basis for the hierarchical mean shift approach. Different criteria are used to identify examples from rare categories from the dendrogram. Tiebreaker conditions have also been defined to intuitively query the cluster modes in cases of criterion value ties between clusters. Mean shift clustering or mode seeking is central to the methodology. kd-trees or dual trees (for image datasets) can be used to speed up mean shift calculations thus helping in scaling up the method. The main advantage of this methodology is that it does not require any information regarding the datasets like the number of classes or the prior probabilities of the classes. The dependence of mean shift on bandwidth values has

been eliminated with the hierarchical mean shift approach. No assumptions about the datasets are made. The methodology has been extensively tested with 6 real datasets and the performance analyzed. The results show that an expert is asked only a few hints before presented with examples from rare categories. The results also show that the approach performs consistently better than the existing methodologies Interleave and NNDM.

6.2 Future Work

Currently, the labels provided by the expert during the querying of the data points are only used by the algorithm to resolve ties between clusters having the same criterion value. In cases when there are no ties the labels are totally ignored. It would be desirable to use these labels in a more effective manner in such a way that the next cluster that is selected for labeling is picked by taking into account both its criterion value and the labels already provided by the expert.

All the experiments have been conducted on image data sets or continuous data with less than 10 dimensions. It would be interesting to see the performance of the algorithm on sequences or other structured data. More research needs to be done on how to build a hierarchy for sequential data and ways to better find the anomalous sequences.

Non-parametric techniques are known not to scale well with the dimension of the space. This is due to the empty space phenomenon by which most of the mass in a high-dimensional space is concentrated in small regions of the space. Hence for high dimensional data the mean shift clustering algorithm should be used carefully. Most of the datasets used in the experiments have dimensions less than 20. It would be interesting to see how the rare category detection technique using hierarchical mean shift performs on data sets with dimensions higher than 100.

Bibliography

[1] Sugato Basu and Raymond J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. pages 81–88. ACM Press, 2004.

[2] Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 17(8):790–799, 1995.

[3] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, 2002.

[4] Daniel Dementhon. Spatio-temporal segmentation of video by hierarchical mean shift analysis. In *SMVP 2002 (Statistical Methods in Video Processing Workshop)*, 2002.

[5] Shai Fine and Yishay Mansour. Active sampling for multiple output identification. *Mach. Learn.*, 69(2-3):213–228, 2007.

[6] K. Fukunaga and L. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *Information Theory, IEEE Transactions on*, 21(1):32–40, 1975.

[7] Bogdan Georgescu, Ilan Shimshoni, and Peter Meer. Mean shift based clustering in high dimensions: A texture classification example. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 456, Washington, DC, USA, 2003. IEEE Computer Society.

[8] Teofilo F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[9] Jingrui He and Jaime Carbonell. Nearest-neighbor-based active learning for rare category detection. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 633–640. MIT Press, Cambridge, MA, 2008.

[10] M. Chris Jones, James S. Marron, and Simon J. Sheather. A brief survey of bandwidth selection for density estimation. *Journal of American Statistical Association*, 91(433):401–407, March 1996.

[11] Andrew Moore Kan Deng. Multiresolution instance-based learning. In *Proceedings of the Twelfth International Joint Conference on Artificial Intellingence*, pages 1233–1239, San Francisco, 1995. Morgan Kaufmann.

[12] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with gaussian processes for object categorization. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[13] Yee Leung, Jiang-She Zhang, and Zong-Ben Xu. Clustering by scale-space filtering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1396–1410, 2000.

[14] Wendy L. Martinez. *Exploratory Data Analysis with MATLAB (Computer Science and Data Analysis)*. Chapman & Hall/CRC, November 2004.

[15] C.L. Blake D.J. Newman and C.J. Merz. UCI repository of machine learning databases, 1998.

[16] Dan Pelleg and Andrew Moore. Active learning for anomaly and rare-category detection. In *Advances in Neural Information Processing Systems 18*, December 2004.

[17] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry - An Introduction*. Springer, 1985.

[18] B. W. Silverman. *Density estimation: for statistics and data analysis*. London, 1986.

[19] Alexander S. Szalay. The sloan digital sky survey. *Comput. Sci. Eng.*, 1(2):54–62, 1999.

[20] Berwin A. Turlach. Bandwidth selection in kernel density estimation: A review. In *CORE and Institut de Statistique*, pages 23–493, 1993.

[21] Kiri Wagsta, Claire Cardie, and Stefan Schroedl. Constrained k-means clustering with background knowledge. In *Proc. 18th International Conf. on Machine Learning*, pages 577–584. Morgan Kaufmann, 2001.

[22] Ping Wang, Dongryeol Lee, Alexander Gray, and James Rehg. Fast mean shift with accurate and stable convergence. In *In Proceedings of AISTATS 2007*, 2007.

[23] Eric P. Xing, Andrew Y. Ng, Michael I. Jordan, and Stuart Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems 15*, pages 505–512. MIT Press, 2003.

[24] Changjiang Yang, Ramani Duraiswami, Nail A. Gumerov, and Larry Davis. Improved fast gauss transform and efficient kernel density estimation. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 464, Washington, DC, USA, 2003. IEEE Computer Society.

[25] Liu Yang and Rong Jin. An efficient algorithm for local distance metric learning. In *in Proceedings of AAAI*, 2006.