Visual Object detection using Frequent Pattern Mining

A THESIS

submitted by

YOUSUF A

for the award of the degree

of

MASTER OF SCIENCE
(by Research)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY, MADRAS.

January 2011

THESIS CERTIFICATE

This is to certify that the thesis entitled **Visual Object detection using Frequent Pattern Mining**, submitted by **Yousuf A**, to the Indian Institute of Technology, Madras, for the award of the degree of **Master of Science (by Research)**, is a bona fide record of the research work carried out by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. B. Ravindran
Research Guide
Associate Professor
Dept. of Computer Science and Engineering
IIT-Madras, 600 036

Place: Chennai

Date:

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisor Dr. B Ravindran for his untiring and sincere guidance. Without him I wouldn't have done any part of this work. He always reassured me in times of immense pressure and taught me how to do research, present a technical document, how to motivate it. I also thank him for giving me an opportunity to work with University of Birmingham, UK. I thank British council for funding the UKIERI project which enabled me to do research with University of Birmingham. Jeremy Wyatt and his PhD student Josè of University of Birmingham, also helped in bringing out this work. We had long discussions on different aspects of the problem like, formulation and evaluation.

I would like to thank Priya for patiently reading and giving valuable comments for my initial draft of thesis. My parents through their support helped me a lot to bring about this work, especially my sister, brother-in-law, their kids and my mother. I am thankful to my GTC committee which included Dr. Shankar, Dr. Hema Murthy and Dr. Giridar for their valuable comments regarding research. I thank my thesis reviewers Dr. Dipti Deodhare and Dr. Shirish Shivade for their comments on thesis. I also thank my friends Abhishek, LN, Sreejith, Yarapavan, Padmanabhan, Abhilash, Shamsudheenn for giving me ideas and for patiently listening to me while I described and argued about the problem.

Last and never the least I would thank the faculty and staff of department of computer science, for facilitating the research.

ABSTRACT

KEYWORDS: Object detection, visual operators, Frequent pattern, FP mining on images, Incremental object detection

Visual object detection is the problem of identifying an object from a visual scene or more generally, addressing object queries like, *What are the objects present in the scene ? How many objects are there in the scene ?*, etc. Performing reliable object detection is an important task as it is one of the key requirements for realizing a fully autonomous, general purpose robotic agent. Activities like playing a game, navigating a road, looking for a book in library, etc., are examples involving object detection. Humans perform these tasks incredibly well. But this is a very hard problem for a learning agent. Part of the difficulties arise from the visual processing involved as the objects can appear in different scale, lighting conditions, rotation, occlusion etc. Vision researchers have come up with representations which are robust to these visual variations and can reliably detect objects even in the case of noise. Apart from the visual challenges, there are machine learning challenges of generalization, discrimination and incremental addition. From the training samples the agent has to learn representations which can generalize well over objects within the class and discriminate between objects of different classes. For example, the representation learned for class *cup* should generalize various cups of different size, and shape model and should distinguish it from other classes say *ball*. As the agent interacts with the environment, it may encounter new objects. It needs to learn and update its knowledge base so that it can detect the new object class. This incremental update of knowledge base is very desirable for a vision system and the re-learning involved should be minimal.

In this work we propose a system which addresses the learning challenges explained above. The key idea here is to learn those visual properties which are common among the objects of the same class so that these properties can be later used for detecting the class. Shape, color, size, texture, etc., are examples of visual properties. A combination of these properties can represent each class and the learning is done in three phases - pattern mining phase, scoring phase and re-scoring phase. In the pattern mining phase, training image of each class is processed and the various visual properties are extracted. From these visual properties, those patterns that frequently occur are learned using frequent pattern mining. Since we are considering the frequency alone, it might happen that the operators learned will include those which covers more background pixels than the foreground ones or whose coverage is superseded by other operators. The scoring phase is used to eliminate such erroneous or extra operators. To handle discrimination and incremental addition, we have the re-scoring phase, which re-scores the operator sets learned based on how popular the operator sets are among other classes. Generally those operators which are common among many classes are considered as bad choices for discriminating the classes. Re-scoring phase tries to remove such operators by switching them with other alternatives. The system is empirically evaluated with the Caltech-101 data set and the results are presented.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

FP	Frequent Patterns
fg	Foreground or object pixels
bg	Background pixels
tp	true positives
tn	true negatives
ROI	Region of interest
SIFT	Scale Invariant Feature Transform
SURF	Speeded Up Robust Features
MSER	Maximally stable extremal regions
DOG	Difference of Gaussians
kNN	k Nearest Neighbours
SVM	Support Vector Machine
POMDP	Partially Observable Markov Decision Process
HiPPO	Hierarchical POMDP
5poly	a visual operator which detects any 5 sided polygon
npoly	a visual operator which detects any n sided polygon
DT	Decision tree

NOTATION

i	Item in a transaction
I	Set of items in a transaction
D	Transaction database
C_k	Candidate set in level k
L_k	Frequent patterns in level k
v	Visual Operator
V	Visual Operator set
L	Label or object class
FP_{max}	Maximally scored set of frequent operators
N^L	Number of images in class L.
$N^L_{FP_i}$	Number of images in class L with i^{th} frequent pattern.

CHAPTER 1

INTRODUCTION

Vision is one of the most heavily used sensors by humans. We interact with our environment mainly using visual sensors. Out of many high-level tasks that we can accomplish with vision, object detection stands out. Every day we recognize a multitude of familiar and novel objects. Many of the high-level tasks we perform are seamlessly integrated with object detection and in-fact we don't realize them.

Object detection helps us to do a wide range of daily activities like moving around, interacting with people, reading, playing, etc. For better understanding, these tasks can be divided into those which involve implicit and explicit object detection.

In explicit object detection tasks, the objective is to find an object like a book from a book shelf or a commodity from a store. Whereas in an implicit task, the objective would be to perform higher level tasks like crossing a road, interacting with friends, etc.

Let's look at an example of navigating a road. Here the high level task is to reach the other end of the road without hitting obstacles like cars, other persons, etc. We start by looking towards both sides of the road one by one and if they are clear we go ahead. If you look at this task more closely, it can be seen that, we are detecting cars, buses, familiar faces, friends, traffic signals, pedestrians, etc. All this involves object detection but the main task that we are interested in is different, *crossing the road*. Figure 1.1 shows some examples.

We do object detection with little effort despite the fact that the object can appear in various conditions of illumination, rotation, size, etc. Objects can even

Figure 1.1: Various tasks involving explicit and implicit object detection

be recognized under different viewpoints and also under partial occlusion. Such robust object detection helps us to easily tackle the uncertainty of the environment.

1.1 Why is Object detection important ?

In computer vision and robotics, object detection is of great importance. Applications in computer vision include video surveillance, traffic monitoring, industrial inspection, digital libraries, gadgets like mobiles, cameras etc. In robotics, the task of navigation, object manipulation, interaction with humans and environment, all requires object detection. Object detection serves as the most important goal to be realized for a fully autonomous agent.

Object detection can pave the way for building hybrid systems which use object detection with sophisticated cameras like long-range zoom, wide panorama, night-vision, etc., which can be used for surveillance and military applications. Many of the modern day techniques of visual detection is inspired by the processing involved in brain. Thus tackling object detection will give us better insights about

how vision is processed in different regions of brain. This can help us devise various remedies for vision related ailments.

1.2 Scope of the work

The scope of this thesis is to study the problem of visual object detection system from a machine learning perspective. The reason why this is challenging and interesting is that even if we are given highly reliable visual processing routines, the problem of addressing visual queries is still hard. Research [Yarbus [1967]] has shown that humans learn the sequence of visual operations to be performed in a task dependent manner. Yarbus conducted experiments, tracking eye movements of human subjects while they view images. He observed that as different tasks are given to the viewer, the eye-movement patterns and gazing time differed considerably. Another interesting aspect is to see how we combine our top level knowledge of the object and the bottom up cues from the scene [Itti *et al.* [1998], Itti and Koch [2000]].

Although this work uses various vision routines, it won't be discussing about the computer vision perspective of object detection where the emphasis is to devise more reliable and easy to compute visual features and routines.

1.3 Contribution of the thesis

In this work we present a novel idea of using data mining for object representation and detection in images. The main motivation of applying data mining is to find useful patterns that occur in images of a class. There can be multiple such patterns appearing in the images and the hope is that learning all such patterns will help us predict the unknown image. The patterns here can refer to color, shape, texture, spatial organization, depth, etc. For example, green-5polygon on

top of red-8polygon can be learned as a pattern for strawberry class. Mapping the image classification to data mining allows us to have more control over the data representation and organization. Emphasizing this, the thesis provides the following key contributions:

- A dynamically extensible visual object detection system using frequent pattern mining.
- New object classes can be incrementally added to the system.
- Majority of the computation involved is localized and independent of other class data and hence can be processed in parallel.

1.4 Organization

Chapter 2 gives a brief overview about visual systems and the various challenges involved. The motivation for this work is discussed and examines various methods present in the literature. This is followed by a short introduction on data mining and frequent pattern mining.

Chapter 3 discusses the proposed system. FP mining on images is introduced. Various phases of the general system are explained and how each phase helps in feature selection and classification is described.

Chapter 4 shows the explained system built with basic shape, color and spatial operators. Modification for each phase of the system is explained in detail. The system is evaluated on selected labels of Caltech-101 data set and results are discussed. Also performance of the system is compared with SVMs and decision trees.

Finally Chapter 5 summarizes the work carried out and the conclusions drawn from the thesis as a whole. Future research directions are also discussed.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter we discuss about visual systems in general. The various challenges in building a visual system includes feature extraction, object representation, and inferencing. This chapter gives a detailed analysis on these challenges and discusses various methods that are currently used. A brief overview of our approach towards solving the problem is also discussed towards the end of this chapter.

2.1 Introduction to Visual System

Although object detection forms an important part of the scene information, it is not the only information that the agent might be interested in. According to the task, the agent might require to know the different objects present in the scene (not looking for a particular object), spatial organization of various objects, similarity of objects, etc. Such an entire information system can be called as a visual system.

In general, a visual system should be capable of addressing different types of visual queries. The types of queries addressed in an object detection task is of the form, *What is the object present in the scene ?* or *Is the object present of type X ?*. For the system to be more useful we might also be interested in other queries like, *How many objects are there in the scene ?*, *What is the object towards the left of object X ? Find the object(s) which are of cylindrical shape ?*, etc. Such queries depend on the task the agent is trying to solve.

In this thesis we will be concentrating on object detection query of the type *Is the object present is of type X ?*. Our method learns a symbolic representation and hence suitable modifications can be made to address other queries like *Find the objects*

with a particular property (among the symbols that we learn).Throughout this thesis, the following terms are used interchangeably: visual actions, visual operators and operators.

2.2 Challenges involved

Object detection can be considered as a pattern recognition task, where the input representation is mapped to an output label. But the image domain makes the problem much harder to solve. A visual scene contains a lot of information. From the processing point of view it is just a collection of pixels with color and location information. Group of such pixels forms objects or object parts. These objects/parts combine to form bigger objects and they together define a scene. The object itself can appear in different color, texture, size or even shape. Many of these variations can occur combined in a scene, which makes it harder to detect these properties.

It can be quickly seen that learning just the pixels are not useful for defining a scene. Different distribution of the same pixel set can completely redefine a scene and convey a different meaning. Learning all such possible pixel distribution for a particular object or scene are also not possible as such data may not be available all the time and that the data involved is enormous. Also there can be completely different images which map to the same label. See the different types of cups shown in figure 2.1.

The various challenges explained above can be broadly classified into Visual Processing challenges and Machine Learning challenges. We will go through each of these in the subsequent sections.

Figure 2.1: Different varieties of cups

2.2.1 Visual processing challenges

First and foremost, the major hindrance for a reliable object detection is the difficulty posed by the visual appearance of the object. The same object can appear different while scaled, translated and rotated. The appearance might also change with different illuminations. In addition to this, there are view point changes where the objects appear differently from different viewpoint locations. Figure 2.2 shows the various visual challenges.

Figure 2.2: Image: Original, Scaled, Rotated and Change in Illumination

2.2.2 Machine learning challenges

Apart from the visual appearance changes, there are many other interesting aspects of computer vision research. Object tracking in videos, object activity recognition,

Figure 2.3: Images from Caltech-101 showing: view, scale, noise and occlusion.

spatial blur detection, Content Based Image retrieval are a few. The main learning challenges with regard to object detection are posed by generalization, discrimination and incremental addition.

Generalization requires to have a representation that can capture different variations of an object under a single label (Remember the cups example). This is usually very difficult as it is very hard to explain what constitutes an object (Try describing a cup class).

Generalization is complemented by discrimination where the aim is to well discriminate different objects. The representation shouldn't over generalise objects so as to not differentiate between a cup and a mug.

Another important requirement of a visual learning system is that it should be able to incrementally update its knowledge base. It might not be possible to have all object classes during the training phase. The agent might encounter new object classes during its life time. Also due to the resource (memory, computation) constrained environment or real time requirements, it might not be feasible to save

all training data and re-train with the new object. Knowledge of this new object should be updated such that the re-learning involved is minimal. As we will be interested on these challenges, a simple illustration is given below.

Illustration

Let's say the agent wants to learn the class *apple* from the image sample given in 2.4.

Figure 2.4: A sample from class *apple*

A possible representation of the object is shown in table 2.1,

class	hypothesis
apple	shape = *round*

Table 2.1: Hypothesis for class apple

Discrimination - now to accommodate a class *orange*, the shape attribute is not sufficient as it is satisfied by both the classes. Now to discriminate class *apple* and *orange* we have,

class	hypothesis
apple	shape = *round* & color = *red*
orange	shape = *round* & color = *orange*

Table 2.2: Hypothesis for class apple and orange

Figure 2.5: New class *orange*

Figure 2.6: Third class *ball*

Incremental Learning - the agent now wants to learn class *ball*

From the agents hypothesis, shape = *round* & color = *red*, will satisfy classes *apple* and *ball*.

A possible representation to accommodate all three classes is shown in table 2.3,

class	hypothesis
apple	texture = 
orange	shape = *round* and color = *orange*
ball	texture = 

Table 2.3: Hypothesis for class apple, orange and ball

Here the hypothesis for class *apple* was modified to accommodate new class incrementally. Note that this assumes that texture property was also learned for

all classes.

2.3 Current approaches

Finding an interest point and a feature vector around the interest point (describing the locality) is one of the widely used methods for solving appearance differences. Here the emphasis is to learn features which are scale, translation and illumination invariant. It essentially builds the object detection on local features.

State of the art in this method include Scale Invariant feature transform (SIFT Lowe [2004]), Maximally stable extremal regions (MSER Forssen and Lowe [2007]), Speeded Up Robust Features (SURF Bay *et al.* [2008]), etc. Most popular among these is the SIFT features.

SIFT extracts a large collection of feature vectors which are invariant to image translation, scaling and rotation and partially invariant to illumination changes. Key locations are defined as maxima and minima of the result of difference of Gaussians (DOG) function, applied in scale-space to a series of smoothed and resampled images. Low contrast candidate points and edge response points along an edge are discarded. Dominant orientations are assigned to localized keypoints. The feature vector is formed by specifying the orientation relative to the keypoint, essentially making the vector rotation invariant.

Many variants to the SIFT method are also proposed like RIFT (Lazebnik *et al.* [2004]), which is rotation invariant SIFT, PCA-SIFT (Ke and Sukthankar [2004]), GLOH (Mikolajczyk and Schmid [2005]), etc.

The whole approach of building object detection on local features is challenged by researchers who have shown that extracting global features along with local features might help in better detection-accuracy and speeds (Oliva and Torralba [2006]). They argue that the ambiguity in local features can be reduced by global features of the image - *gist* of the scene.

Figure 2.7: Image from Dickinson [2009]. From the image we can easily detect two humanoids facing each other but mapping the image into local features is very difficult.

Another line of approach in object detection is to consider part-based representation for objects. Felzenszwalb *et al.* [2008] describes object detection system which uses multi scale deformable part models. In this model, each part encodes local appearance properties of an object and the deformable configuration is characterized by spring like connections between certain pair of parts. Agarwal and Awan [2004] describes object detection using a sparse part-based representation. Here a vocabulary of distinctive object parts are automatically constructed from the set of sample images of the object class and the objects are represented using this vocabulary together with the spatial relation among these parts.

Object detection using shape context is another popular approach. Mori *et al.* [2005] discuss about using similarity between shapes for object recognition. A measurement of similarity between shapes is estimated by finding an aligning transform for correspondences between points on the two shapes. Each point is associated with a descriptor - the shape context and the remaining points are described relative to the shape context. Given the point correspondences, a transformation that best aligns the two shapes is found. For recognition a nearest-neighbor classifier is used which finds the maximally similar prototype to that in the image.

Researchers have also tried developing a visual grammar for object representa-

tion. The visual grammar describes an object using variable hierarchical structures. The object will be represented in terms of parts and each part is again described by sub-parts or by properties of that part. Zhu and Mumford [2006] explain such an approach but the major difficulty is defining what constitutes the various parts.

Many hybrid systems have also been proposed. Frintrop [2006] describes a visual detection system in which the object training and detection happens in two phases. Initially the object class is trained to find out how the object "stands out" from its surroundings. This is done by learning the weights for various Gabor filters which responds to variations in intensity, color and orientation. These potential object areas are later fed to a Viola-Jones classifier which identifies the features of the object.

There has been work from the machine learning community with emphasis on learning visual operators for object detection and gaze control. Minut and Mahadevan [2001] have proposed a selective visual attention model based on reinforcement learning in which the agent has to choose the next fixation point (action) based only upon the visual information.

Sridharan *et al.* [2008] proposes a Hierarchical POMDP system called HiPPO which plans a sequence of visual operators associated with a task. It is a probabilistic planning frame work that enables the agent to plan a sequence of visual operators, which when applied on an input scene enable it to determine the answer to a visual query with high confidence. The unreliability of the visual operators are captured using a POMDP framework. Plan execution corresponds to traversing a policy tree, repeatedly choosing the action with the highest value at the current belief state, and updating the belief state after executing that action and getting a particular observation. As solving POMDP becomes exponential in the combined space of visual actions and ROIs, they propose a hierarchical POMDP framework. A lower level (LL) POMDP is used for the visual actions to be taken and a higher level (HL) POMDP describing the various ROIs to choose from. The actions in the LL POMDP depends on the agents task and the HL POMDP maintains the belief

over the entire scene. The visual operators used include various colors and shapes. The result of their experiments has shown that it is computationally more efficient to plan a set of operators for the visual query rather than doing the naive exhaustive search. In order to add new labels to HiPPO, it would require reformulating the model with new states, actions, and observations and do re-planning on the query.

2.4 Motivation

In this work we describe an object detection system which tries to extract the common visual properties among a set of images. The approach is motivated by Shimon Ullman's visual routines theory (Ullman [1984]) which states that the human vision system is composed of basic visual operators which are combined in different ways for complex object detection tasks. Visual operators can extract properties like color, shape, texture, etc.

2.5 Our approach

The main idea of our approach is to identify the common properties that exist among the set of images in a class. The properties can be color, texture, shape, spatial orientation, etc. These properties can occur in different combinations within the same class. The learning phase for an object class tries to find out the common visual properties that occur in the set of images within the class. The representation learned is later combined with other object classes for discriminating among them. The combining step is done incrementally so that the re-learning involved is minimal. Object in an unknown image can be later verified by checking for these properties.

The main difference from the existing approach is that, we use FP mining for

feature selection. The features selected for each class are later combined incrementally with other classes and used for classification. Typically incremental learning adds the data points of a class incrementally to the training set but in our case the classes themselves are added incrementally.

For learning the common properties in an object class, we use Frequent Pattern (FP) mining, which is a well-known technique in data mining. Subsequent sections will provide a brief overview on FP mining and its application on images.

2.6 Introduction on Data mining

Data mining comprises of a set of machine learning techniques which can be used to find patterns from large sets of data, which otherwise is very difficult to find manually. It has immense applications in the field of data analysis like customer purchase patterns, medical diagnosis, text analysis, etc. These patterns will contain useful information which can be used to predict the future occurrences of data.

Data mining initially found its application in market analysis which was later known as *market-basket* analysis. It was used to find interesting customer purchase patterns from purchase transactions. The various commodities in a transaction is known as items and set of items present in a transaction is termed as the itemset. The transactions are analyzed for the occurrence of patterns in the items and the co-occurrence of these patterns. This helps to better understand customer purchase patterns and is used in marketing.

Today data mining has evolved into an important machine learning technique for analysis and prediction of data.

Data mining mainly involves four classes of tasks,

- Clustering is the task of finding groups or structures in the data. Usually a measure of similarity is defined between the data and this is used for the grouping. This is completely unsupervised approach.

- Classification is the task of generalizing learned structure of data to new data. Common algorithms include decision tree learning, kNN, SVM etc.
- Regression analysis finds a function that can model the data. There will be measure of error defined on the training samples and the objective is to minimize this error.
- Pattern analysis or Rule learning is the task of learning useful patterns or sequences from the data. Finding such frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data.

We will be more interested in Pattern analysis also known as Frequent Pattern (FP) mining.

2.6.1 FP mining

Frequent patterns are patterns (or itemsets) that occur more than a minimum number of times in a data set. For example, a set of items, such as milk and bread, which appear frequently together in a transaction data set is a frequent itemset. Finding such frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data.

Terminology

Considering the market-basket analysis, a commodity in a transaction (T) is known as an item(i). The set of items present in a transaction is termed as the itemset(I). The itemset with n items is known as an n-itemset. The set of transactions forms the transaction database (D).

For example: milk, bread, jam, etc., are items, and ⟨milk, bread⟩, ⟨bread, jam⟩, ⟨bread, eggs⟩ are examples of 2-itemsets.

The number of times an itemset occurs in a transaction database is known as the count or support of that itemset. An itemset is said to be "frequent" if its

occurrence in the transaction database are more than a minimum number of times, known as the 'minimum support' or 'min-support'.

Let A be an itemset such that $A \subseteq I$ then,

$$support(A) =\mid A \cap D \mid$$

$$frequent(A) = \begin{cases} 1 & \text{if support(A)} \geq n \\ 0 & \text{if support(A)} < n \end{cases}$$

where n is the min-support.

FP mining can be defined as finding all such itemsets which have support greater than the min-support. i.e., for a given database D and value of minimum support n we have,

$$FP(D, n) = \{F \mid \forall A \subseteq D, frequent(A) \Leftrightarrow A \in F\}$$

where F denotes the frequent patterns identified from D.

Based on the completeness of patterns to be mined, we can mine the complete set of frequent itemsets, or the maximal frequent itemsets, given a minimum support threshold. In the maximal frequent itemsets mining, only itemsets of length k are considered such that there are no k+1 frequent itemsets. In complete set of frequent itemsets, itemset of length 1 to k (both inclusive) are considered.

In the next section two popular algorithms used for FP mining will be discussed.

2.6.2 Algorithm

An exhaustive approach of finding all subsets of itemsets and checking if it is frequent is exponential in time. There are many algorithms proposed for frequent pattern mining and one of the earliest and simplest of these is the Apriori Pattern Mining algorithm (Agarwal and Awan [2004].)

Apriori Algorithm

This is a basic FP mining algorithm which uses prior knowledge of frequent itemset properties. Apriori employs an iterative approach known as a level-wise search, where k-itemsets are used to explore (k + 1)-itemsets. Let all the frequent patterns in level k be noted as L_k. The finding of each L_k requires one full scan of the database. To improve the efficiency of the level-wise generation of frequent itemsets, an important observation that, *All nonempty subsets of a frequent itemset must also be frequent* is used. This implies that a k-itemset is frequent if and only if all of its (k-1) subsets are frequent. This is called the Apriori property. This property can be used to reduce the search space of finding frequent itemsets.

The procedure starts by finding the set of frequent 1-itemsets by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support. The resulting set is denoted L_1. These 1-itemsets are combined together to form 2-itemsets. These are known as the candidate 2-itemsets denoted as C_2. The database is scanned for the occurrence of these 2-itemsets to find the actual support. The support of each of 2-itemsets are recorded and those 2-itemsets are pruned which have support less than the min-support. From these frequent 2-itemsets, candidate 3-itemsets are generated and so on and so forth until no more frequent itemsets can be found. The steps are enumerated in algorithm 1. The routine *apriori_combine* combines all frequent L_{k-1} items to form C_k.

The caveats here are that the scanning of database can be huge. Also the candidate generation step might result in a large number of candidates, making it practically infeasible. There are several enhancements proposed for Apriori algorithm like hash based buckets for looking up FPs, scanning and recording support of more than one itemset etc.

Algorithm 1: Apriori FP mining

Input: Database D, min-support n
Output: set of frequent itemsets FP

```
1 L_1 = find_all_1_itemsets(D) ;
2 for k=2; L_{k-1} ≠ φ; k++ do
3     C_k = apriori_combine(L_{k-1});
4     L_k ← φ ;
5     for c ∈ C_k do
6         if support(c) ≥ n then
7             L_k ← L_k ∪ c ;
8 L ← ∪_{i=1}^{i=k} L_i
```

FP growth

Another interesting algorithm for FP mining is the Frequent pattern growth algorithm (Han *et al.* [2000]). FP growth algorithm constructs a tree representing the database and finding frequent pattern is by scanning this tree recursively. Advantage here is that the whole process employs a divide-and-conquer strategy. The main motivation of including this algorithm is that it is highly scalable for large databases and for resource constrained environments.

The FP-tree algorithm constructs all frequent 1-itemset (L_1). Each item is ordered according to its support and the items in a transaction are considered in this order. Each transaction in the database is scanned and a tree is constructed by inserting nodes for items. Initially the root of the tree is marked as null and for the first transaction all items are inserted as nodes with their count marked. Now it considers the second transaction starting from the root, traversing down and checking if the nodes and incoming items match. If there is no node for an incoming item, a new node is created and its count initialized to 1. Remember that we are considering the transactions in a sorted order of the items and all common prefixes get accumulated starting from the root and becoming more specific towards the leaves. Each time the node is scanned for a place, the corresponding count is also incremented. A item header table which points to the items occurrences in the tree

is build using a chain of node links.

FP construction is done by considering the items with lowest support first and upwards. Each item is taken and there conditional FP tree is constructed. This can be done by enumerating all paths which leads to the current item using the link pointers. Once we have the conditional tree, the FPs are generated recursively by concatenating the FP tree prefix and the item. A detailed algorithm of FP growth can be found in the original publication Han *et al.* [2000].

2.7 Summary

In this chapter we discussed about visual systems. Visual systems are generic systems which can answer a variety of visual queries about a scene or object. The various challenges we saw in building such a system are the visual challenges which included the difference in scale, orientation, view point variations, etc. Apart from the visual changes are the machine learning challenges which ensure that the representation learned can be generalized and discriminated over a range of examples and different classes. Another important requirement of a vision system is incremental learning. The agent should be able to update its knowledge base such that its existing knowledge about the objects are preserved and there is minimal processing involved in this process of re-learning.

In this thesis we will be concentrating on the machine learning challenges. We also discussed about data mining concepts especially frequent pattern mining. FP mining involves finding frequent patterns in data. The idea is that this patterns can be later used for detection in unknown samples. FP mining algorithms, Apriori pattern mining and FP tree mining were also discussed.

CHAPTER 3

OBJECT DETECTION SYSTEM

One of the major contributions of this work is the application of frequent pattern mining for object detection. So far there has been little work done on pattern mining in images for object detection. In this chapter we will discuss on extending FP mining to images. A detailed explanation on the proposed object detection system is provided. The chapter is concluded with some initial approaches that we tried and which helped in evolving the current FP mining method. For ease of understanding, most of the examples use the features: shape and spatial-relation. But the methodology is generic and the system can be extended to use different pattern mining algorithms or image features as applicable.

3.1 FP mining on Images

As previously seen, raw image data is of very high dimension and low dimension feature vectors need to be extracted for any useful processing. Most of the data mining algorithms (including those we have seen) do not work on real features. Thus the feature vectors need to be "itemized" so that the data mining algorithms can work.

Analogous to the FP mining terminology (see section 2.6.1), we define a visual operator (o) as an item. That is to say, any feature vector that we use is to be mapped as an item. A set of visual operators(O) forms a visual itemset. The visual operators identified for an image forms a visual transaction and the set of visual transactions identified for all the training samples in a class forms the visual database of that particular class.

For example: square, circle, square-inside-circle, etc., are examples of visual items. ⟨square, circle⟩, ⟨circle, rectangle⟩ are examples of 2-itemsets.

Similarly, the number of times a visual itemset occurs in a transaction database is known as the count or support of that itemset. A frequent visual itemset is an itemset with support greater than or equal to the minimum support. The FP mining process is defined as the process which finds visual patterns in an image database.

3.2 Proposed System

In this work we explore the theme of using of FP mining for selection of image features, how to rank these features and finally use them for object detection. We assume a supervised setup where a set of images with class labels are provided for training. We also assume that the training samples have the pixels marked as foreground and background.

Our system can be logically partitioned into feature selection and classification. Feature selection includes 3 phases. The FP mining phase, which finds useful patterns from the images, Scoring phase to rank the operator sets depending on its coverage and finally a re-scoring phase which re-scores the operators based on its discriminative qualities among the given set of classes. Classification phase does the actual classification task using the operators selected from the feature selection phase. Subsequent sections describe each phase in detail.

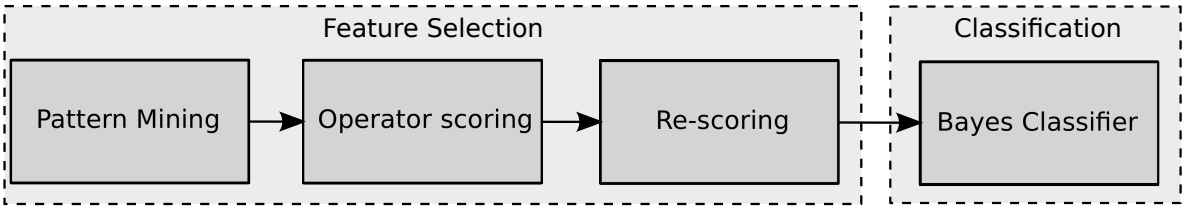

Figure 3.1: Different phases of the system

Figure 3.2: Input image with fg pixels highlighted

3.2.1 FP mining phase

FP mining phase includes the feature extraction or visual database creation and pattern mining the data. Feature extraction is the process of mapping images into a set of features or visual properties (o). The input of this phase is the set of images and output is the frequent patterns extracted for each class. Even though the images are grouped with class label, this phase does not use the label information to correlate with other classes. Since there is no dependency with other classes, this processing can be done in parallel for all the classes.

The feature extraction is usually done by a set of routines which can extract specific properties known as visual operators. The operators can be simple property operators which extract features like color, shape, texture, etc., or relational operators like spatial relations, ternary relation operators, etc.

A property operator is said to have *fired* (or relevant) on an image i.e., the operator is included as an item in the visual transaction of an image, if it selects at least some minimum number of pixels from the foreground. This condition is to make sure that we include only foreground selection operators and discard background operators. All available property operators are applied on an image and those which fire are found.

The relationship among the instances identified by the property operators are determined by applying the relational operators on them. Thus the property operators together with the relational operators form the visual transaction for

Figure 3.3: Relevant 6polygon operator and a discarded triangle bg operator

that image. This procedure is repeated for all available images in the class and the identified visual transactions constitute the visual database of the class.

Figure 3.4 shows an example of visual transactions extracted for an image of stop_sign class from the Caltech-101 dataset.

Figure 3.4: Property operators, Relational operators and the visual transaction for stop sign image

Algorithm

The algorithm starts by constructing the visual database for each class as explained above. Apriori frequent pattern mining (refer section 2.6.2) is used on these transactions to find the frequently occurring visual itemsets. Candidate generation is done on the apriori property which states that, for an itemset to be a frequent itemset candidate, all its subsets should also be frequent. For bigger itemsets where

candidate generation is expensive, FP tree algorithm can be used for mining.

Similarly the frequent patterns are found for each class independently and these FPs forms the input to the next phase, viz. Scoring phase. Apart from fixing the values of parameters like minimum-support, thresholds for visual operators, etc., no user intervention is required in this phase. Algorithm 2 outlines the steps required for a single class.

Routine *apply_operator* applies the property operator on to the image and returns a score proportional to the foreground pixels it selected. *find_relational_opers* is used to find the relational operators among the identified property operators. The pseudo-code of these routines is not shown as they are very specific to the operators that we use.

Algorithm 2: FP mining

```
Input: label L, Training Data T
Output: set of frequent operators, FP
1 for each image i and foreground info. f ∈ T do
2     O[i] ← NIL ;
3     for each property operator p ∈ P do
4         fgpixels ← apply_operator(p, i, f);
5         if fgpixels > select_threshold then
6             O[i] ← O[i] ∪ p
7     O[i] ← O[i] ∪ find_relational_opers(O[i], R);
8 FP ← Apriori_FPMine(O, minsupport);
```

Regarding the mode of mining, we consider only maximal FPs. Also FP mining can give multiple frequent visual patterns which show the different object configurations. Choosing the min-support value here can be tricky as it controls how much generalization we want with the training samples. Too high a value can cause very generalized representation to be learned which might miss some specifics. Too low a value can cause many different object configurations including some noisy configurations to be learned. More about this value is discussed in the experiments section of chapter 4.

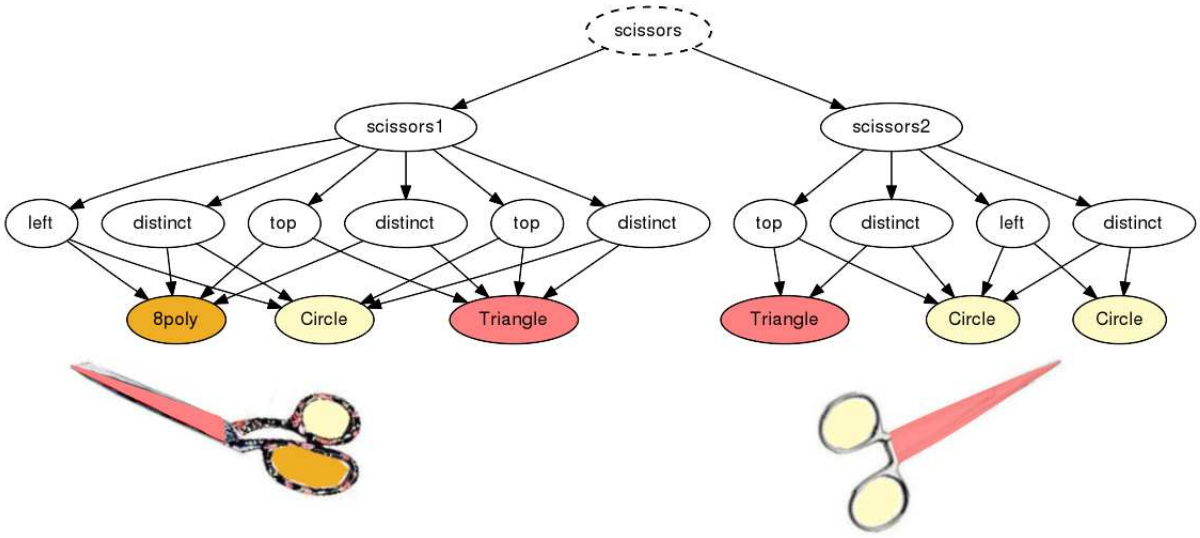

Figure 3.5: Multiple configurations detected for *scissor class*

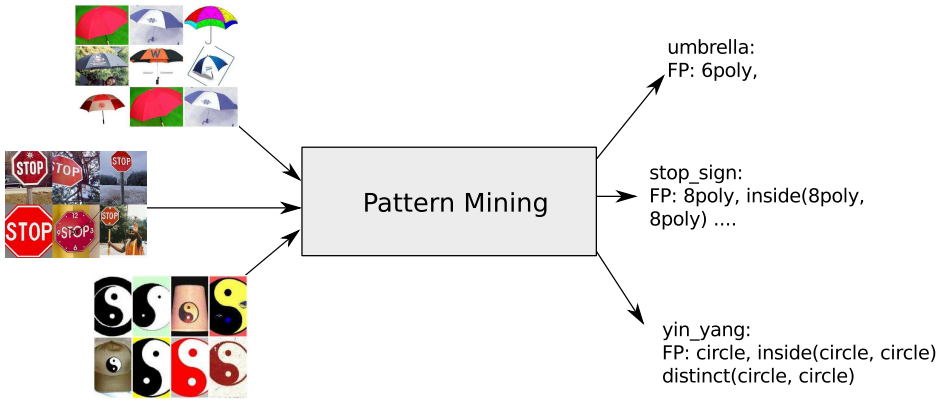

Figure 3.6: Learning features using Pattern mining

The set of FPs thus forms a AND-OR graph where each configuration is OR-ed together with its parts AND-ed to its parent. Figure 3.5 shows a sample AND-OR graph.

Thus to summarize this phase, the input is the sample images grouped in classes. The feature extraction is done, creating a feature database of that class. Frequent pattern mining is done on this database and the output is the frequent patterns thus identified. Figure 3.6 summarizes the processing involved in this phase.

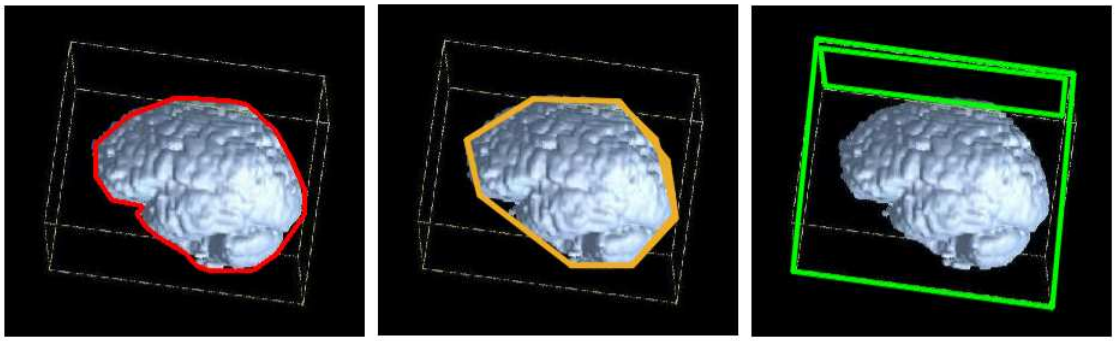

Figure 3.7: Brain image showing 8poly and square operator

3.2.2 Scoring of frequent operators

Now we have represented an image class with a set of frequent visual patterns. While adding an operator in the frequent pattern mining phase, we considered only the coverage of the operator. The FP mining procedure ensures that the patterns obtained would be present at least in *minimum-support* images of the class. But this does not say how accurate the operator sets are. An operator may select more background pixels than foreground pixels and this operator might be selected as a frequent pattern because the object is usually shown in a similar background (like a soccer ball usually shown in a play ground).

Figure 3.7 shows brain image with the left most one showing the foreground pixels and others showing operators 8polygon and square respectively. Here you can see that even though both operators are relevant (as they select fg pixels) the square operator selects more bg pixels than the 8polygon operator.

Another scenario is that the coverage of an operator might be superseded by some other operator. Here both operators might be frequent but we want to rank the higher operator more than the lower operator in terms of its coverage.

For pruning such inefficient operators or ranking the extra operators from the set of identified frequent operators, we attach a score with each of the property operators. The score is defined as the accuracy measure, i.e. the ratio of sum of

true positives (tp) and true negatives (tn) detected by an operator to the total pixels of the image. The true positives for an operator are those pixels selected by the operator which are true fg pixels (as per the object foreground information) and the true negatives are true bg pixels.

$$\text{score} = \frac{\text{tp} + \text{tn}}{\text{total pixels}}$$

Figure 3.8 shows example for both cases mentioned above and how the scoring helps. Here operator o_1 has a relatively lower score than o_2 or o_3 as it selects more bg pixels. Also operators o_2 and o_3 overlaps in their coverage but o_2 has a better score.

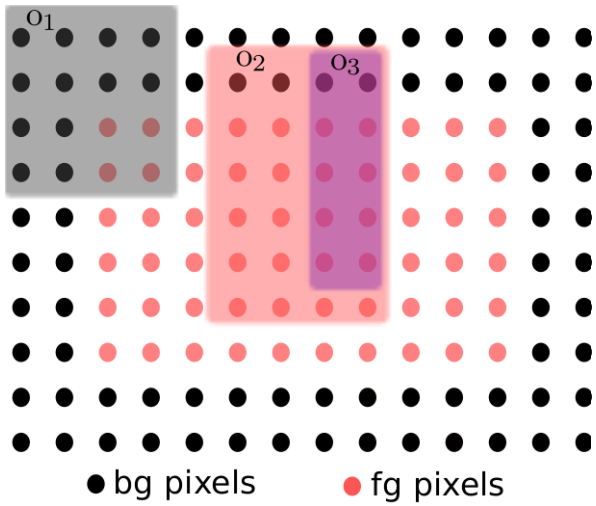

Figure 3.8: Operators showing the scoring scenarios

It must also be noted that neither the scoring phase can be combined with the frequent pattern mining phase nor they can be swapped (fp mining accurate operators). FP mining among the accurate operators might fail to capture operators which are frequent but not very accurate (due to noise or operator errors). Whereas the approach of selecting accurate among the frequent operators will capture all operators and then rank them with their accuracy. Figure 3.9 shows a simple illustration. Here the score of the pentagons and hexagons might be less than the outer circle and mining-scored-operators will fail to capture the pentagons

and hexagons. Detecting them as frequent operators is very crucial as in the next section we will see that these operators give a better discrimination.

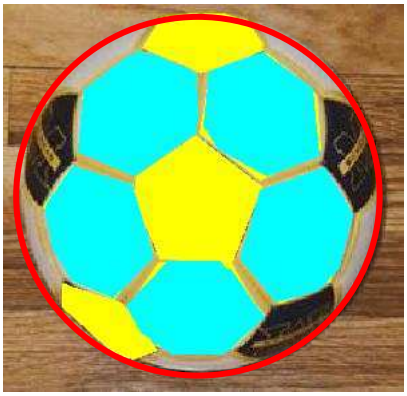

Figure 3.9: Soccer ball with detected 5polygon and 6polygons

Algorithm

From the identified set of frequent patterns (FP), we have to find those subsets which give the maximum score (FP_{max}) for the class. This is done by constructing a lattice of frequent operator sets by considering inclusion on the FPs (see figures 3.10, 3.11).

Each node in this lattice is frequent because they themselves are subsets of the frequent itemsets and is associated with a score. The score is computed as the average accuracy of the operator set computed across all the images of that class. The search for the maximum-score operator set starts from the first level (single operators). In each level the operator set which has the maximum score (maximal operator set) is found. This maximal operator set is compared with the maximal operator set of second level and so on. The search continues till no improvement in the maximum score is seen i.e., till the score of i^{th}level is less than or equal to $i\text{-}1^{th}$level. Once the maximal operator set is found out, it is included to the FP_{max} set and the maximal operator set along with all of its subsets and supersets are not further considered in the search process. As the maximal operator set had the maximum score within the level and among the different levels, there is no need to

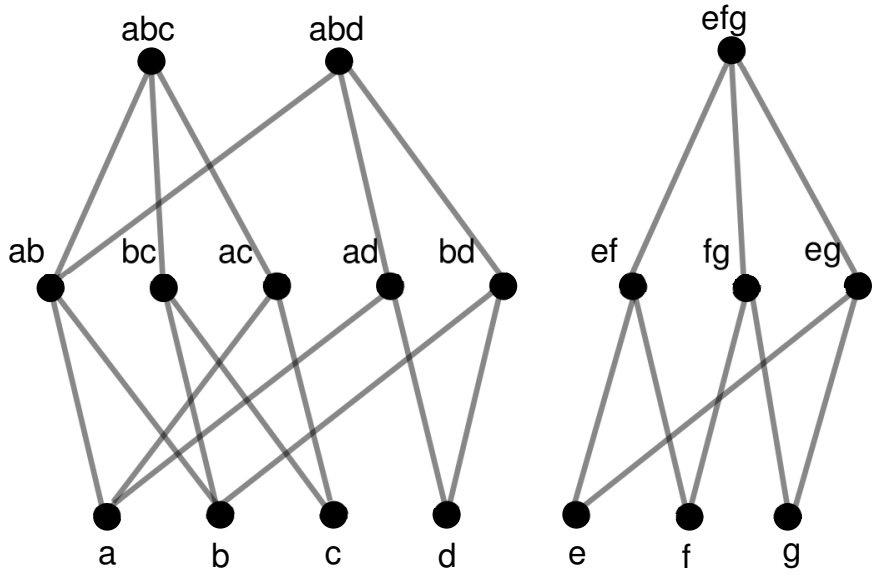

Figure 3.10: Lattice constructed for 3 operator sets *abc*, *abd* and *efg*

consider its subsets or supersets during further search. The maximal operator set covers for its supersets and its subsets and hence they are removed from further analysis. The above steps are repeated until all the nodes of lattice are removed.

Algorithm 3 enumerates the steps involved in scoring the frequent patterns. Routine *find_max* finds the maximum scored operator set among all the operator sets in the specified level. The processing involved in this phase is also independent with other classes and all the processing can be done in parallel with other classes.

Algorithm 3: Scoring the FPs

Input: set of frequent operators, FP
Output: set of operators, FP_{max}

```
1  construct the lattice for FP ;
2  while there are still nodes in the lattice do
3      max ← find_max(O_1, T) ;
4      for level ← 2 to max do
5          max_curr ← find_max(O_level, T) ;
6          if max_curr.score ≤ max.score then
7              break ;
8          max ← max_curr ;
9      FP_max ← FP_max ∪ max.op ;
10     Remove max, its subsets and supersets from further consideration;
```

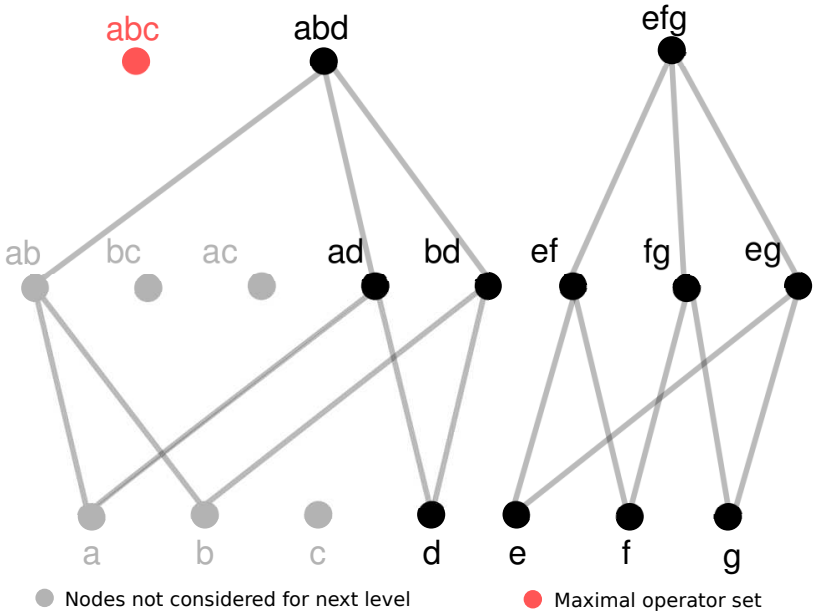

Figure 3.11: Lattice considered for second iteration of algorithm, maximum operator set in first iteration is *abc*

3.2.3 Re-scoring of operators

To summarize the processing so far, we have frequent visual patterns among the images of a class and we have ranked them according to their accuracy. We still have processed object classes independent of each other and have not considered the discrimination and incremental requirement of the system.

While adding new classes to the system, it might happen that the operator set with the maximum score identified for one class would be popular for other classes too. For example, a circle operator will have a good score for detecting a pizza class and a soccer ball class.

In such cases the "popular operators" must be replaced with other alternatives (if present). This has to be done in an online fashion as all the object classes may not be present during training. Thus in the pizza-soccer ball example, the system should identify a slightly less scored frequent operator set which detects the various hexagons and pentagons in the soccer ball. This rescoring of the operator sets should be done as and when new labels are added to the system.

Figure 3.12: Circle as the highest score operator for both pizza and socccer_ball class

Algorithm

From the previous scoring phase, we have identified the frequently occurring maximum scored operator sets. From the frequency of occurrence of an operator set in a class, we can calculate the probability of the operator set given the label i.e $P(\text{FP}_i|\text{L})$ where L is a label and FP_i a frequent operator set. This probability can be written as,

$$P(\text{FP}_i|\text{L}) = \frac{N_{\text{FP}_i}}{\Sigma_i N_{\text{FP}_i}}$$

where N_{FP_i} is the number of images FP_i has fired for the class.

For testing the presence of a label in an image, we require the probability of a label given that we observed a particular operator set. Using Bayes rule, we can find the probability of label given the observation of a frequent pattern FP_i as,

$$P(L|\text{FP}_i) = \frac{P(\text{FP}_i|\text{L}) * P(\text{L})}{P(\text{FP}_i)}$$

P(L) is the probability of the label which is assumed constant, given by N^L/T where N^L is the number of images of the class L, and T is the total number of images across all classes. We assume equal number of training images for all classes i.e, $N^{L_i} = N^{L_j}$. With the above assumption $P(\text{FP}_i|\text{L})$ can be rewritten as $N^L_{\text{FP}_i}/N^L$. The probability of observing a frequent pattern $P(\text{FP}_i)$ is N_{FP_i}/T i.e., the number of images on which FP_i fired regardless of the label, divided by the total number of

Figure 3.13: Soccer ball with alternate operator set and pizza with circle operator

images. Substituting all of these in the above equation we have,

$$
\begin{aligned}
P(\mathrm{L}|\mathrm{FP}_i) &= \frac{P(\mathrm{FP}_i|\mathrm{L}) * P(\mathrm{L})}{P(\mathrm{FP}_i)} \\
&= \frac{N^L_{\mathrm{FP}_i}}{N^L} \times \frac{N^L}{T} \times \frac{T}{N_{\mathrm{FP}_i}} \\
&= \frac{N^L_{\mathrm{FP}_i}}{\not{N}^L} \times \frac{\not{N}^L}{\not{T}} \times \frac{\not{T}}{N_{\mathrm{FP}_i}} \\
\therefore P(\mathrm{L}|\mathrm{FP}_i) &= \frac{N^L_{\mathrm{FP}_i}}{N_{\mathrm{FP}_i}}
\end{aligned}
$$

The above result shows that for testing a label, the operator sets should be ordered according to the ascending order of $\frac{N^L_{\mathrm{FP}_i}}{N_{\mathrm{FP}_i}}$. Thus for an operator set to get a better score in this phase, either the frequency of observing the operator set FP_i for the particular label is high or that the probability of the operator set FP_i firing for other classes is less.

Algorithm 4 enumerates the steps.

Algorithm 4: Re-scoring the FPs

Input: $FP_{\max}$for all classes

Output:

1 **for** *each class*, $c \in C$ **do**

2 **for** *each* $FP \in c.FP_{max}$ **do**

3 $freq$[FP] $\leftarrow$ $freq$[FP] + FP.freq

4 **for** *each class*, $c \in C$ **do**

5 $sort(c.\text{FP}_{max})$ by $\frac{c.\text{FP}_{max}.freq}{freq[\text{FP}_{max}]}$

3.2.4 Classification task

As already seen, for testing the presence of a label in an image, we need to check the presence of operator set in the descending order of rescore value, $\frac{N^L_{FP_i}}{N_{FP_i}}$.

A re-score value of 1 ($N^L_{FP_i} = N_{FP_i}$) indicates that the operator set fires only for the label under consideration. Firing of such an operator set on the test image gives a full confidence of that label. But if the value is less than 1, it signifies the cross-firing of the operator set on other labels ($N_{FP_i} < N^L_{FP_i}$). Firing of such an operator indicates that the label can be either of those labels on which FP_i fires.

We call these labels as confused labels for operator set FP_i denoted as $C(FP_i)$. Let's assume that for a particular label, all the operator sets that we have are confused with more than 1 label. So with the above method, there will be no single operator set which can give us full confidence for the label. But we can use the partial information of multiple operator sets and improve the detection of the labels. The justification here is that if multiple operator sets for the query label fire on the test image, it gives more evidence on the presence of that label.

To make use of this information, we keep a list of confused labels. We start off

by adding all labels to this list. As each operator set fires, the new set of confused labels will be those which are common in the current set of confused labels and those for which the operator set fires ($C(FP_i)$). The subsequent operators for the queried label is re-scored only on these confused set of labels. So the next operator set for the label would be the one which tries to discriminate only among the confused labels as opposed to all available labels. This procedure is continued until the label is finalized or until the operator sets are exhausted.

Even with this procedure, if we don't have enough discriminating operator sets, we could still end up with a set of confused labels. Note that the rescore value of such confused labels is an approximation of $P(L|O)$ and it would be equal when the operator set O fires only for those confused operator sets.

Algorithm 5 enumerates the various steps.

3.3 Initial approaches

This section talks about some of the initial ideas that we tried before the current FP mining approach. Although they were failed attempts it nevertheless contributed to the current thoughts and ideas.

3.3.1 Heuristic Operator Search

We started off with the idea of abstracting the image class with visual operators and to find out the combination of these operators which will help in classification.

We assume that we have a set of labeled images with background and foreground information for training. We define a set of visual operators (O) which included shape and color features.

The initial method we tried was to find out the set of operators present in an

Algorithm 5: Algorithm used for testing object presence

```
   Input: FPs for query class,Q and Image M
   Output: Class
 1 Test(FP, M)
 2 begin
 3     for each f←1 to size(FP) do
 4         FP ← FPs(f)
 5         if FP fires on image then
 6             if Pr(Q | FP) == 1 then
 7                 return Q ;
 8             else
                   // let C' be the set of confused labels
                      associated with FP
 9                 FPs' = { FPs(f+1) , ... FPs(n)) } ;
10                 for FP' ∈ FPs' do
                       // rescore based on the universe of confused
                          labels
11                     score(FP', Q) = N^Q_FP' / Σ_{c∈C'}(N^c_FP)
12                 sort(FP', score);
13                 return Test(FP, M);
14     // no operator set fires
15     return NIL;
16 end
```

image. Each visual operator when applied to an image or more generally to a ROI, select certain pixels. The score of the operator is calculated as proportional to the true foreground pixels it selected on the image. The fg, bg information of the image is obtained from the training data.

The operator search procedure tries each operator one by one on the image and finds the score of the operator. The best operator and the best score is found out for the single operator case. Then the procedure is repeated for 2 operator case, finding the best 2 operator representation of the image and so on. The procedure is repeated until the score is less or is the same between two consecutive levels. This procedure is repeated for each image in the training set and the operator sets with score greater than a certain threshold are selected.

The number of times the selected operator sets fired on the images are found and it gives the probability of finding the image given the operator set, i.e., $P(O|L)$. For the classification, the operator sets are applied on the images in the order of $P(O|L)$.

Some of the caveats are, each level requires $\binom{n}{m}$ operator checks where n is the total number of operators and m is the level thus making the search exponential. The information on best operator from each level cannot be used in the next level. It need not be the case that the best operator in n^{th}level will always have the best operators from $n-1^{th}$level. So a greedy algorithm cannot find an optimal solution in our case.

Figure 3.14 shows an example. Here in level1 - o_1 is the best operator and for level2 - o_2, o_3 is the best operator set.

We later tried a simple heuristic, for computing the operators for n^{th} level, (n-2) operators from $(n-2)^{th}$ level are kept fixed and remaining 2 operators are tried for all $\binom{n}{2}$ combinations. Empirically the results for the exhaustive search and the heuristic search were similar.

The FP method that we proposed is a modification of the heuristic search method. To handle the exponential computation complexity, in the FP method, the search procedure was split in to 2 phases. The FP mining phase finds those operators which are frequent. Here we exploit the apriori property and incrementally builds the operator set from each level. The scoring phase handles the actual scoring of the operator by evaluating only those operators which are selected from the pattern mining phase.

3.3.2 POMDP approach

Extending the idea of HiPPO we decided to model the object representation as a hierarchical POMDP. Basic features such as color, shape, etc., forms the first level,

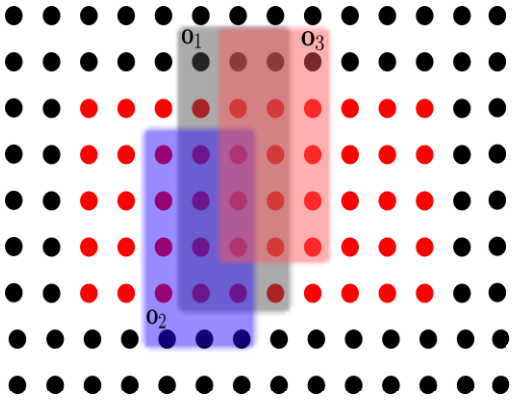

Figure 3.14: Counter example showing search procedure cannot be greedy

combination of these features leads to the second level like spatial relations and so on. Modeling the features as probabilistic helps to capture the uncertainty of the operators.[1]

The whole scene is described as a hierarchical tree. The lowest level POMDP takes care of the primitive visual operators. The policy of this POMDP will be the object detection sequence. This can be modeled either as one single large POMDP whose policy does the classification. Another way to model the same is to have different POMDP for each object. The initial belief of which POMDP to be chosen should be either decided by higher POMDPs or from initial probabilities set by the query. We also found similar work from the Machine learning community. Pineau and Thrun [2001] talks about planning under uncertainty and hierarchical POMDPs. Simon *et al.* [2002] proposes a method for hierarchical object classification. Dietterich [1998] explains the MAX-Q framework for hierarchical reinforcement learning which can be used for policy learning.

Figure 3.15 shows a sample hierarchical decomposition of a scene.

One of the major problems we encountered was the large state space this model has. We want to represent the fact that objects can be decomposed in different parts. Each part could be represented by the set of basic features defined above.

[1]This part of the work was done with the help of Jeremy Wyatt and his PhD student José from University of Birmingham, UK.

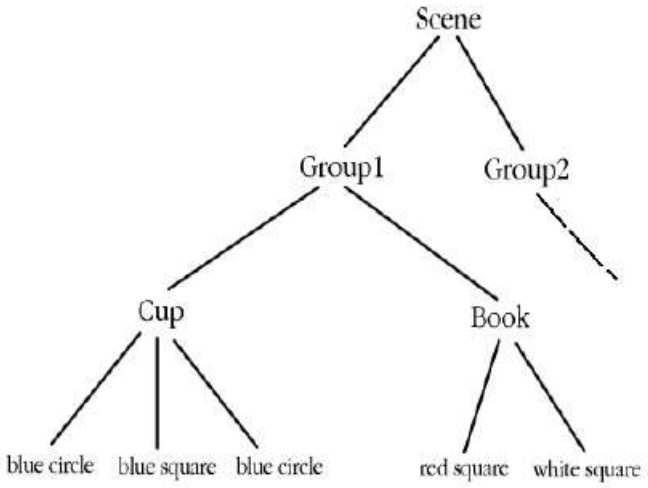

Figure 3.15: Hierarchical decomposition of a scene

It is also important to define how different parts are related in order to compose an object. Furthermore, we faced the problem of how to represent the relations among different objects in the scene. Therefore, the state space would be defined over all possible trees of scenes, which is too big and possibly infinite too. Also as new operators are added the lower POMDP also becomes intractable.

3.3.3 Learning visual operators

Learning the visual operators themselves was also an important area that caught our attention. The system learns the set of visual operators suitable for the image class from more primitive visual operations like, segmentation, edge detection, contour detection, etc. The operators need to be modeled as probabilistic to capture the uncertainty. The sequence of actions can be learned using planning.

Again the obstacles here are the large number of parameters involved. To tackle the visual changes like illumination, translation, etc. the plan needs to be executed on various parameters like edge-thresholds, scale values, etc.

3.4 Summary

In this chapter we presented a way of extending FP mining on images for object detection. The various phases of the object detection were discussed. The feature extraction or the pattern mining phase is used to map the image into feature vectors. From this feature vectors, the feature selection is done by applying fp mining algorithms on this visual transaction. To assess the accuracy of the frequent operators identified, a score is attached to the frequent operator set, which is the average score of the frequent operator set across all the training images of that class. Thus the various FP sets are ranked according to their score. Finally to ensure the discriminative power of the identified operator set, a simple statistical inferencing is used. Among the set of the FPs those which are popular with other classes are swapped with lower scoring but more descriptive alternatives. This is done by considering the firing of the FP across all the classes. Those operators which have a high occurrence count among other classes are considered bad as they are too popular. This ordering is done among the present set of classes. As more classes are added, the operator sets are dynamically re-ordered so that each class is always tested with the most discriminative operator set the class has.

The various machine learning challenges we discussed in the previous chapter is answered in different phases. The pattern mining phase controls the generalization of the representation learned. The re-scoring phase takes care of the discriminative quality of the operator sets. Each class is processed in an incremental manner. The operators re-ordering cost is very less as we require to keep track of only a global count of the operator sets and the re-computation of the probabilities.

CHAPTER 4

DETECTION USING BASIC OPERATORS

This chapter discusses about the experimental validation of our approach. Brief description on the operators we have chosen, various parameters, test bed, results and implementation details are provided. A discussion on extending the experiments with real valued features is given. The chapter concludes with comparison of our approach with SVMs and Decision tree and experiments using SIFT operators.

4.1 Visual operators

For experimental validation of our approach we selected some basic operators. As property operators we have shape operators like square, triangle, circle, etc., and color operators red, green, and blue. Relational operators include binary spatial operators like left, right, top, bottom, inside, distinct, etc.

The shape operators includes circle and a polygon detector. Circle detection is done using Hough transform for circles (Ballard [1981]) The polygon operator *npoly* is a generic n-sided polygon detector (regular or irregular). Actual detectors we used where 5poly, 6poly, etc., which were instantiations of npoly operator with corresponding values for n. When applied to an image, shape operator returns all the instances of the respective shape.

For the color operators, the image colorspace is converted from RGB to HSV and specific range of values for these channels are used to detect red, green and blue colors. For detection, color operators are used in conjunction with the shape

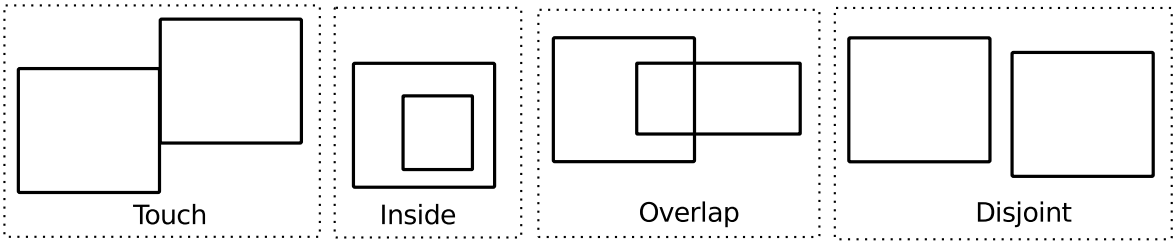

Figure 4.1: Rotation invariant spatial operators

operator, i.e., the color operator is applied to only those regions where shape operator has fired.

Relational operators are binary operators which can detect spatial relations among two shapes. Relational operators included rotation dependent operators like left, right, top, bottom and rotation invariant operators (spri) like inside, distinct, touch and overlap. Figure 4.1 shows the various rotation-invariant spatial operators used.

For selecting the set of shape operators we did some testing on the data set. Shape operators for detecting polygons with 3 to 15 sides were used and usage frequency of various operators were analyzed (training over all labels). It was found that the usage dropped sharply after 9 sided polygon operator.

The main motivation of selecting these operators is ease of understanding. The representation learned would be symbolic in terms of these operators and can be easily verified. Also the shape operators are scale and rotation invariant. Relational operators had both rotation specific and rotation invariant operators See table 4.1 for the complete set of operators that we used.

4.2 Dataset

The image data set we used was Caltech 101-object categories (Fei-Fei *et al.* [2004]). The database consisted of 101 classes with images in each class ranging from 30-400. Each class had a minimum of at least 45-50 images. The images are mainly of size

Class	Operator	Description
Shape	triangleop, squareop, circleop, 5polyop, 6polyop,7polyop 8polyop, 9polyop	identifies shapes
Color	redop greenop blueop	identifies color
Spatial	left,right,top,bottom inside,touch,overlap,distinct	defines relation

Table 4.1: Visual operators

300x500 pixels and contained various views, scales, orientation of the object. Most of the images had only single instance of the object and hence single ROI. Many objects are embedded in real world scenes and are complex (like faces, animals, etc.)

As there are very few geometrical figures in the real world images, shape operators fails to capture all the features. There were classes (like *cougar*) where none of the shape operators worked. So out of the 101 object labels, we empirically selected a set of 18 labels in which the operators performed well. Figure 4.2 shows the various labels that we selected. The system was trained on 30 images from each class and testing was done on 15 images different from the training set.

4.3 Various Phases

The following sections explains the various modifications and implementation considerations that we did for each phase explained in the previous chapter.

4.3.1 Visual Pattern Mining

The pattern mining phase has feature extraction and mining stages. Feature extraction (visual database creation) was done by applying each shape operator to

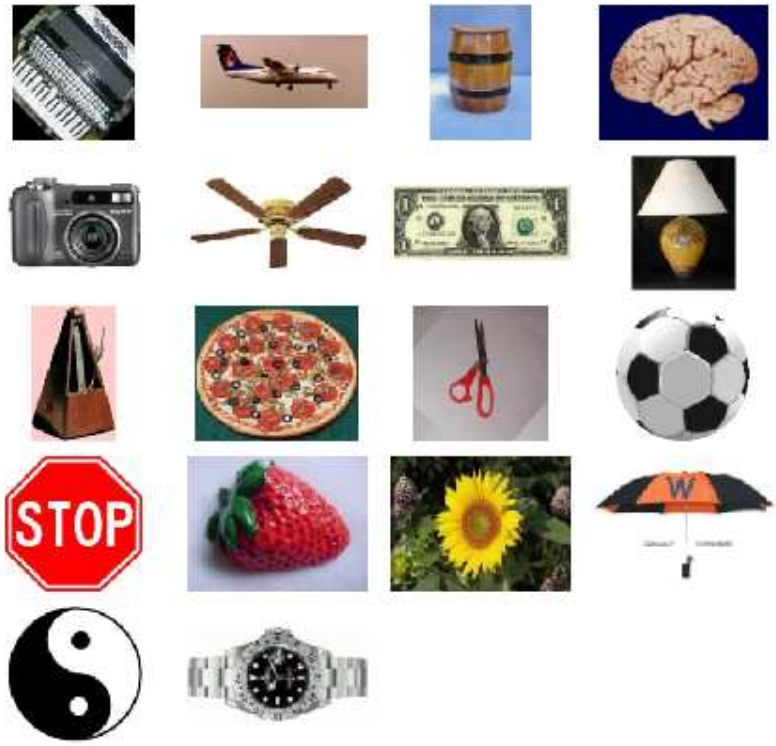

Figure 4.2: Selected 18 labels from Caltech-101 dataset

the training image and then color operators on these detected shapes. Once we have all the shapes and colors identified, the spatial operators are applied by considering the shapes two at a time. The rotation specific and rotation invariant relations are identified on these shapes and the whole features are saved as the visual transaction for that image. Similarly the procedure was repeated for all training images in the class and across all classes.

Apriori mining algorithm was applied on this database and the frequent patterns were recorded. We used 20% as min-support value. Thus any feature which occurs in more than 20% of the training images is considered to be frequent. The value was empirically found out by inspecting the frequent patterns each value gave. It was seen that a value from 20%-40% did not change the accuracy much. Although the min-support value selection can be done in a systematic way using cross validation, we are not aware of a method which can select this non-empirically. Also it was seen that for some classes a specific value is better than a

global minimum support value. For example, in the case of *soccer_ball* class a value of 20% was too small as it constructed many FPs using pentagons and hexagons with slight variations in orientation.

Following list shows frequent patterns detected for certain classes.

- *pizza*
  1. 8poly
  2. circle, left(circle, circle), distinct(circle,circle), top(circle, circle), bottom(circle,circle)
- *stop_sign*
  1. 8poly-red,inside(8poly-red, 8poly-red),left(8poly-red, 8poly-red)
  2. 8poly-red,circle-red
- *sunflower*
  1. 8poly, 7poly, inside(8poly,7poly)
  2. 7poly, square, inside(7poly, square)
  3. 8poly, square, inside(8poly, square)
  4. 6poly, square, inside(6poly, square)
- *watch*
  1. circle, 6poly, inside(circle, 6poly)
  2. circle,8poly,inside(circle, 8poly)

From the FPs it can be seen that: In the *pizza* class many instances of circle operator were detected For *stop_sign* class operator 8poly-red occurred frequently but for other classes no (available) colors were frequent. Also in the case of classes *watch*, and *sunflower* the FPs included polygons which occurred one inside the other.

4.3.2 Scoring phase

Each operator were scored according to the scoring method prescribed in section 3.2.2. The scoring lattice was constructed with the frequent operators. The score of each node was taken as the average score across all the training images of each class.

4.3.3 Re-scoring

Maximal FPs found in the previous stage is further scored for discrimination in this phase. For some labels, the topmost accurate operator set contained only a single operator. We observed that those operators were re-scored to lower values as they are too popular. For example, circle operator was popular among classes soccer_ball, pizza and watch. In all the these cases operator sets were re-ranked for more discriminative operators.

4.3.4 Classification

Classification was done by matching the operators against the unknown image in the re-scored order of each FP set. The testing algorithm used was the same as mentioned in section 3.2.4.

4.4 Results

Table 4.2 shows the selected labels and percentage of detection accuracy, top-2, top-3 and rejection accuracy for the system build with only the shape operators and no spatial relation operators. The detection accuracy, or recall, for a label is the percentage of true positives(TP) detected by the classifier among all the positive image. Note that the TP used here is the true positive images detected. This is different from *tp* used in scoring phase which is true positive pixels. The system was trained on all the 18 labels. The results were generated by doing a 10 fold cross validation with 27 training images and 18 test images from each class. The match for a particular label is given as explained in the Classification section. In top-2 and top-3 matches, the object is considered to be detected if the queried label occurs at least in the top-2 or top-3 of P(L|FP) ordering of the fired operator set.

The low scores towards the end of the table shows those labels where the

operator sets learned are similar or same. For example, in the case of labels umbrella, accordion, etc., the same 5poly operator were learned. We observed that in those cases there were no alternate operators to apply in the re-scoring phase.

The rejection accuracy, or specificity, is the percentage of true negatives (TN) detected by the classifier among all the negative samples. The system was trained on all the labels, and the testing was done on 306 images per class, i.e., 18 images from each class except the true class. Note that this measure cannot be directly correlated with the TP%.

Similarly in table 4.3 the results are given for the system built with shape and spatial operators. Comparing with table 4.2 we can observe that, the labels have become more selective and in case of dollar_bill, pizza, etc., the TN rate improves. For some labels like soccer_ball, scissors, etc., the TP rate decreases because the labels now impose the spatial constraints also for a successful match.

Class	**TP%**	**Top-2 TP%**	**Top-3 TP%**	**TN%**
soccer_ball	72.2	73.3	73.3	79.9
stop_sign	52.8	62.8	66.1	98.5
airplanes	49.4	65.6	73.3	75.7
dollar_bill	36.1	76.7	78.9	97.5
barrel	35.6	44.4	44.4	76.5
sunflower	27.8	54.4	62.8	95.3
accordion	25.6	66.1	68.3	89.5
scissors	25.6	68.9	68.9	91.8
brain	24.4	45.6	60.6	69.8
camera	23.9	28.9	28.9	89.2
pizza	21.7	74.4	81.7	82.5
metronome	20.0	32.2	53.9	90.4
watch	16.1	46.7	51.7	87.7
strawberry	13.3	25.6	32.8	74.5
yin_yang	9.4	76.7	77.2	100.0
lamp	3.3	80.6	93.3	100.0
ceiling_fan	2.2	49.4	64.4	100.0
umbrella	1.7	15.6	29.4	100.0

Table 4.2: Selected labels from 101-dataset and their detection/rejection accuracy with shape operators

Class	TP%	Top-2 TP%	Top-3 TP%	TN%
soccer_ball	65.6	66.1	66.1	98.0
stop_sign	43.9	44.4	44.4	100.0
airplanes	42.2	56.7	56.7	97.1
dollar_bill	46.7	47.8	47.8	99.3
barrel	30.6	43.9	47.2	74.5
sunflower	40.0	57.8	57.8	87.6
accordion	48.9	48.9	48.9	82.7
scissors	21.1	56.1	56.1	99.7
brain	38.9	54.4	54.4	90.8
camera	21.1	21.1	21.1	79.1
pizza	25.6	33.3	35.0	94.4
metronome	26.7	36.7	49.4	90.2
watch	39.4	47.2	47.2	71.2
strawberry	17.8	38.3	45.0	52.6
yin_yang	20.6	41.7	41.7	92.8
lamp	2.8	85.0	87.8	100.0
ceiling_fan	28.9	47.2	65.0	87.3
umbrella	11.7	42.8	61.1	100.0

Table 4.3: Selected labels from 101-dataset and their detection/rejection accuracy with shape and spatial operators

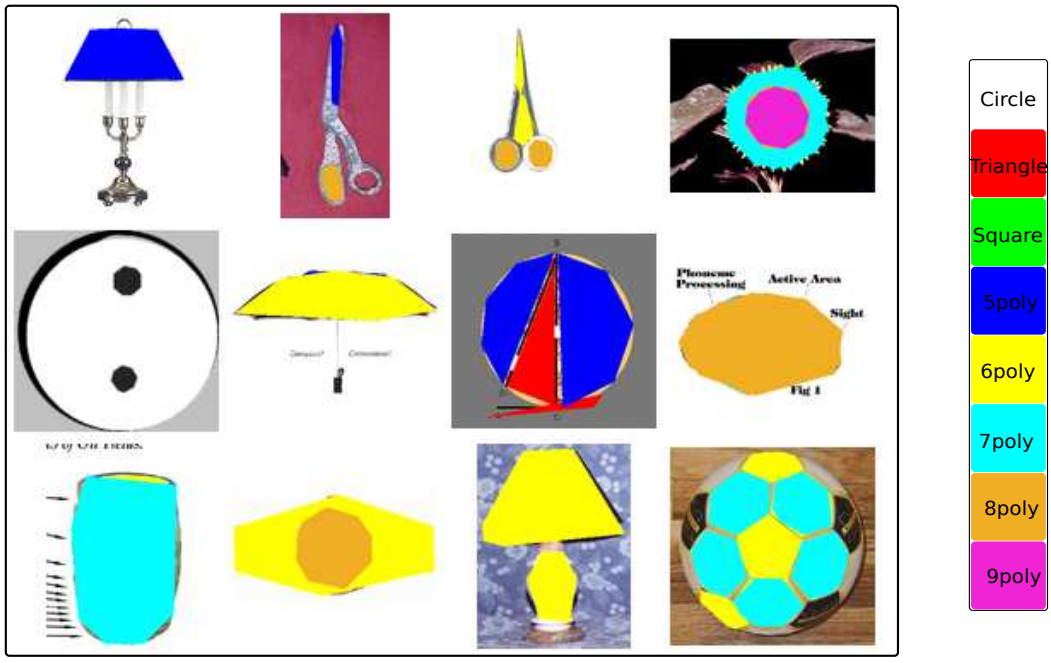

Figure 4.3: Various object selection masks detected

SVM	shape	ri	spri	shape-fp	ri-fp	spri-fp
Linear	33.3	50.7	49.6	13.7	24.4	28.5
Polynomial	14.1	8.9	7.0	5.6	5.6	5.6
RBF	28.5	32.2	32.2	5.9	5.6	5.6

Table 4.4: Percentage of accuracy for SVM on full vs. fp-features on different kernels

4.5 Comparison Experiments

The results discussed here are not state-of-the art in object detection. This is mainly accounted to the set of operators that we chose. Within this limitation we were keen to know how well the method performed and we conducted a set of comparative experiments with Decision trees and SVMs.

4.5.1 Feature performance

To see how well the FP features performed, we compared performance of SVMs on full data and SVM on FP features using different kernels. In full data case, all features from the visual database are used for classification, whereas in the latter, frequent patterns from the scoring phase are used. The problem was setup as a classification task of 18 classes. Results were gathered for Linear, Polynomial and RBF kernels.Among them Linear kernel gave the best performance.

Table 4.4 shows the performance of SVM on full data and FP data using various kernels. Each column denotes various feature sets. Shape only features are denoted as *shape*, shapes and rotation invariant features as *ri* and finally, shapes, rotation invariant and rotation dependent relational features as *spri*. A minimum support value of 20% was used for frequent pattern extraction.

The performance of SVM using Frequent patterns as features is observed to be lower than SVM running on full feature set. This is because FPs are not optimized to run with SVM. Another important aspect to consider here is the computation

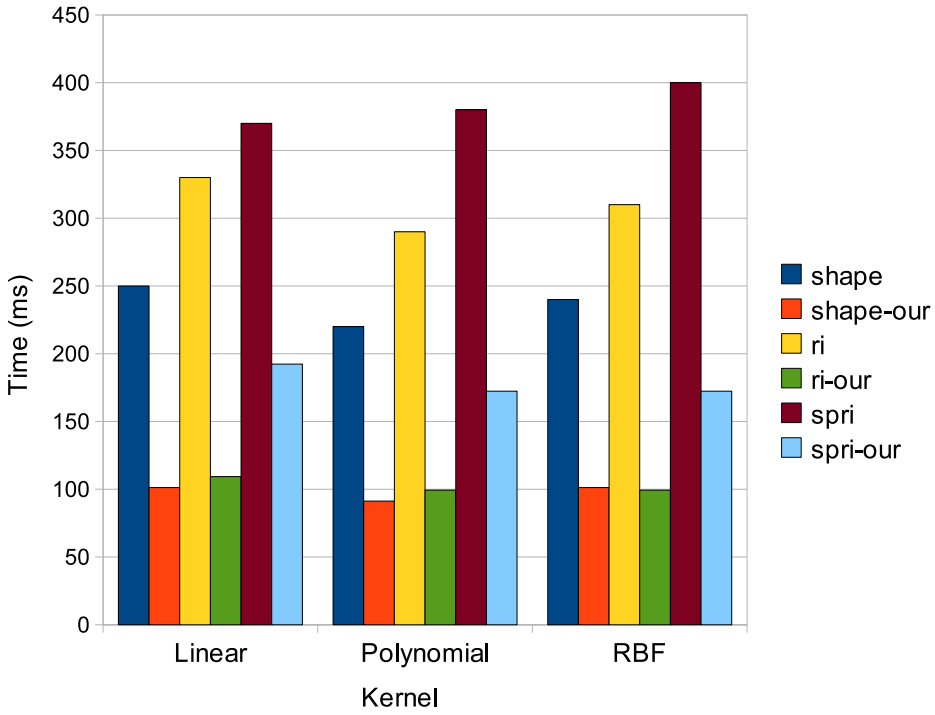

Figure 4.4: Computation time for SVM and our method on various kernels

time required to build the model. Building the model from the whole training data will take more time whereas from the FP operator set is less. In the FP operator case we have to consider time taken for initial FP mining and scoring phase from which the FP operator set are obtained.

Figure 4.4 shows the time required for both our method and SVM on different kernels.

4.5.2 Model performance

To compare the performance of our model, we compared accuracy with SVM. The problem was formulated as a verification task similar to our original experiment. The positive samples are taken from the true class. All classes except the true class forms the negative samples. Since the number of negative samples is more than the positive ones, a weight of 80% was added for true class and 20% for the other

Description	Accuracy %	Time
SVM on data	49.2	3.7
SVM on FP	33.1	2.1
Our method	31.9	1.4

Table 4.5: Accuracy and time comparison for SVM and our method

class. A minimum support value of 20% was used for frequent pattern extraction.

Table 4.5 shows the accuracy and time for SVM on full data, SVM on fp data and our method for the verification task. From the table it can be seen that the performance of SVM on FP versus our method is comparable and the time taken is less in our case. The incremental addition of classes is a harder problem to solve. For example, Note that for the SVM all the classes are available at the beginning. Whereas in our approach the classes are added incrementally. Hence our approach is solving a harder problem, which would require us to train the SVM from scratch for each additional class.

4.5.3 Effect of min-support on accuracy

The selection of min-support value for the FP mining process is a crucial parameter. To show the accuracy variation with minsupport value, we compared accuracy of our method and SVMs on various value of min-support.

Figure 4.5 shows the performance graph. The performance drops in both cases where the value is too low or too high. When the value is too low, the FP sets found will be large and thus might include some noise data points also. Too high a value will fail to recognize the actual patterns that exist in the data set. For our experiments we found that a value between 20%-30% worked well.

The computation time for classification is compared with decision trees. As each class is added, the tree needs to re-build as it is not incremental.

Figure 4.6 compares the computation time to build the decision tree and our

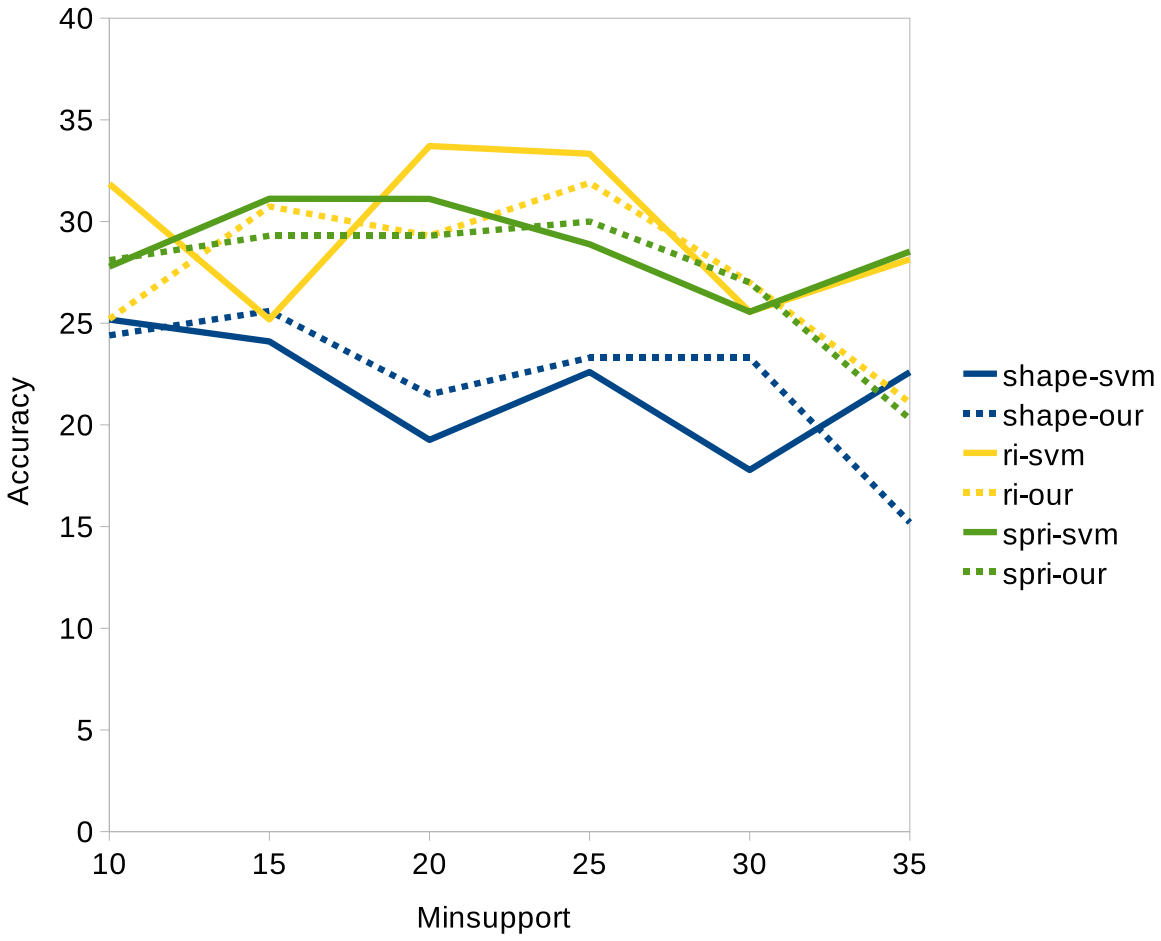

Figure 4.5: Accuracy vs. Minsupport

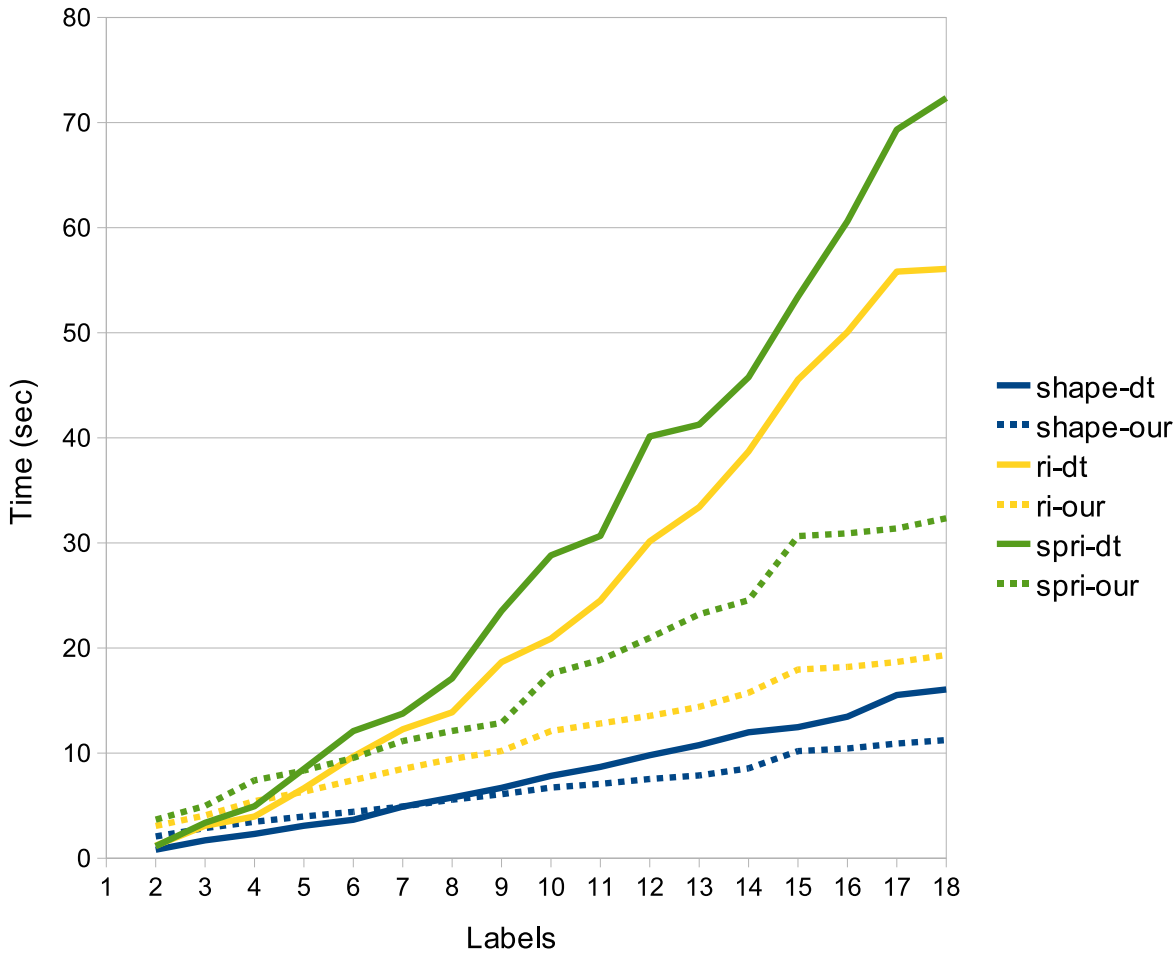

Figure 4.6: Computation time for decision tree as each class is added

method as each new class is added. From the graph it can be seen that our method have a significant advantage on time as the classes are added incrementally.

We wanted to study the impact of the incremental nature of the problem on the feature selection process. To this effect we used decision trees as a feature selection mechanism for SVM classifier. A full decision tree was built on all the image features and feature selection is done by level wise inclusion of features.

Figure 4.7 shows the decision tree. The X-axis shows the feature sets selected from each level of the decision tree and the Y-axis shows the corresponding accuracy of the SVM classifier.

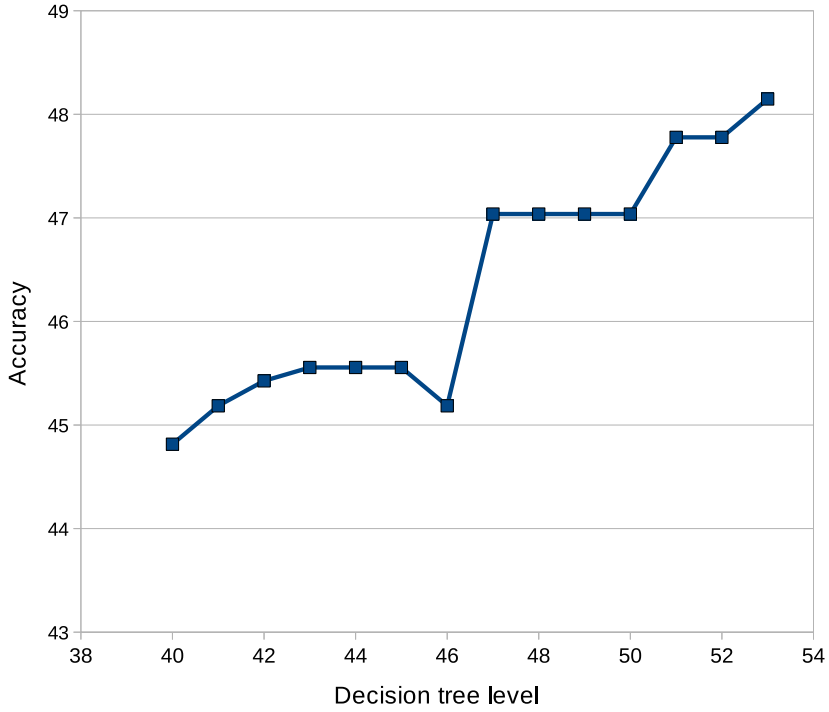

Figure 4.7: Accuracy of SVM classifier using Decision Tree for feature selection

As was expected these features resulted in better performance since they operated with the entire data. But surprisingly our FP mining method chose a lot fewer features (59 out of 864 or 6.8%) compared to the DT method (126 out of 864 or 14.6%).

4.6 Experiments with SIFT

For more robust object detection, more complicated feature extractor operators are required. To see how the FP mining approach would work with feature extraction operators, we did the detection experiments with SIFT operators. Here as the operators are more robust, we conducted detection experiments where the aim was to actually find the class of object in the image.

For an image usually thousands of SIFT feature vectors would be generated.

SIFT features works particularly well for translated, rotated and scaled features of the same image. SIFT features were designed for high recall accuracy of the transformed images even in presence of noise.

We included SIFT features to see who the classifier performs. Apart from the different phases we discussed above, we had a feature extraction phase were SIFT features from the training set are extracted. The feature vectors from the image are initially clustered to find the representatives and these representatives are used to generate the visual transactions. For clustering we tried k-means and density based clustering DB-scan.

Once we have the visual transactions, the processing is similar to the basic operator case where we do a FP mining and scoring. We used a KNN lookup for object classification.

We experimented by extracting SIFT feature of 5, 10, 15 and 20 images and gathered all the SIFT features. But the results were poor. We found that SIFT features were not able to generalize well over a set of images. We tried matching of the clustered features with unknown image of the same class and unknown image of a different class. But for both cases the number of matches it found was not substantially different.

Figures 4.8, 4.9, 4.10 shows some experiments we did with accordion and airplanes class.

For these images, SIFT features from all the training samples were used for matching. Matching was done by measuring the ratio of distances from first neighbour to second neighbour as explained in Lowe [2004] The lines joins the various keypoint matches in the 2 images.

SIFT features when used on the same image but rotated versions gave high number of matches, see figure 4.8. SIFT matches for accordion vs. airplanes is shown in figure 4.9 and accordion vs. unknown-accordion in figure 4.10. The number of matches in both these cases is not substantially different as compared

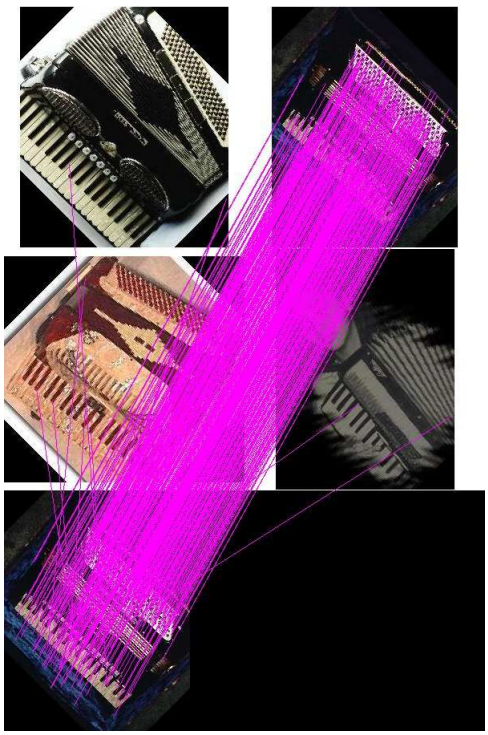

Figure 4.8: Matching accordion with known accordion image, 563 matches were found

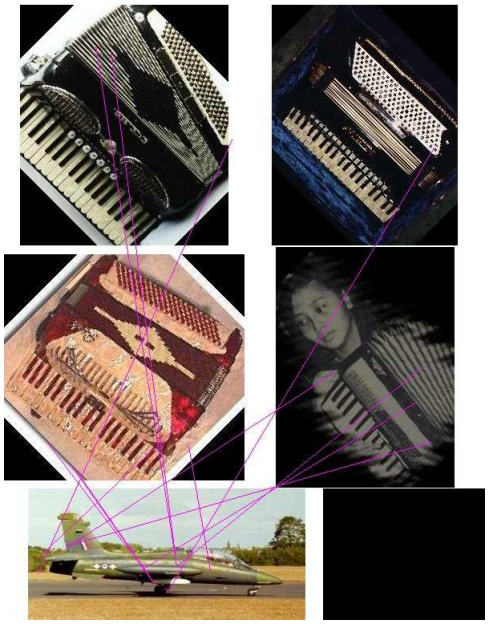

Figure 4.9: Matching accordion with airplane image, 14 matches were found

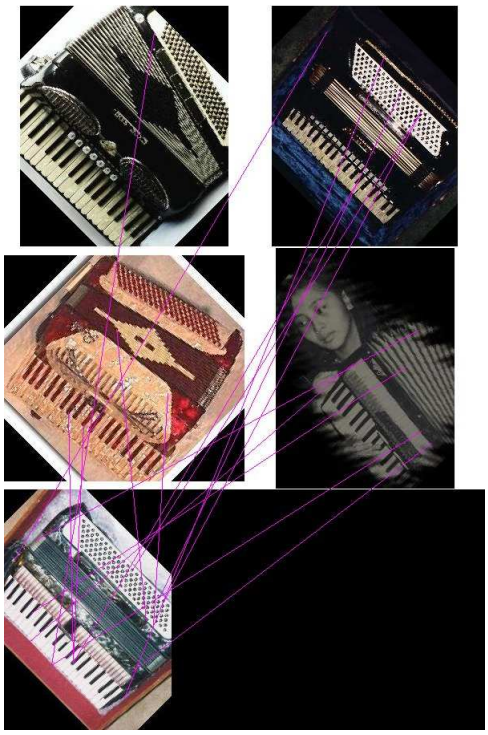

Figure 4.10: Matching accordion with unseen accordion image, 16 matches were found

to accordion vs. known-accordion image. To find features which generalize over a set of images is an incredibly hard task and SIFT features were not originally designed for this purpose.

4.7 Summary

In this section we validated our proposed method with a basic set of operators. The performance of our method in terms of detection accuracy is competitive with other methods but much more efficient in terms of execution time. With a better set of operators we can get better results with our approach.

We also tried using SIFT features in conjunction with our approach, but without much success. Even though SIFT features are good image features with good recall and tolerant to scale, view variations, they are not particularly suitable for

generalizing over a set of images. To find features which generalize over a set of images is an incredibly hard task and SIFT features were not originally designed for this purpose.

CHAPTER 5

CONCLUSION

In this thesis we discussed the problem of object detection and verification using data mining. We proposed a novel system of feature selection using frequent pattern mining and using the learned frequent patterns for detecting the object presence. The following discussion gives a quick recap of the whole process.

5.1 Summary

The main idea of our approach is to identify the common visual properties that exist among the set of images in a class. The whole detection process was divided into feature selection and classification task.

The feature selection phase is used to extract useful features from the images that can be used for object detection. This phase was divided into Frequent pattern mining phase, Operator scoring phase and Re-scoring phase. In the FP mining phase, visual operators are applied to the image and the features are extracted. Visual operators can be color, shape, texture, etc. This constitutes the visual database of the class and FP mining is applied to this feature database to find frequently occurring patterns.

The frequent patterns obtained from the FP mining phase might contain operators that select more background pixels than foreground pixels. Also, it is possible that the coverage of an operator is superseded by some other operator. Operator scoring phase was used to remove such extra operators. Each operator is attached with a score and all the highly scoring operator sets are found using a scoring lattice.

In the scoring phase we used frequent patterns independently from each class. It might happen that same operator sets were selected for discriminating different classes. To use the inter class information we had the re-scoring phase. Re-scoring phase was used to find more discriminative operator sets for each class from the learned operator sets. This was done by re-ordering the operator sets based on the popularity of the operators among the classes. Popular operators, i.e., those operator sets which appear in more than one class are switched with more discriminative ones.

Once we have the re-scored operators, these can be used for the classification task. We use a Bayes classifier for object classification. The presence of the object class is detected by applying the operators in the re-scored order.

The proposed system was tested using a basic set of operators on the Caltech-101 dataset. The operator sets we used included basic shapes, colors and spatial relations. With a better set of operators we can get better results with our approach.

We also conducted comparison experiments with SVMs and decision trees. We compared the performance of SVM as a classifier with full feature set, features from our method and features from Decision tree. We also compared our method and SVM in the object verification task. The performance of our method in terms of detection accuracy is competitive with other methods but much more efficient in terms of execution time. With a better set of operators we can get better results with our approach. Also SVMs and decision trees require all the classes at once where as our method processes the classes one at a time.

We also tried using SIFT features in conjunction with our approach, but without much success. Even though SIFT features are good image features with good recall and tolerant to scale, view variations, they are not particularly suitable for generalizing over a set of images. To find features which generalize over a set of images is an incredibly hard task and SIFT features were not originally designed for this purpose.

Mapping the image classification to data mining allows us to have more control over the data representation and organization. Emphasizing this, the thesis provides the following key contributions:

- A dynamically extensible visual object detection system using frequent pattern mining.
- New object classes can be incrementally added to the system, making it suitable for robotic agents.
- Majority of the computation involved is localized and independent of other class data and hence can be processed in parallel.

We conclude that FP mining on images have several advantages over the traditional classification approaches. FP mining learns representation of objects that can be further used to build systems which can address wide variety of visual queries.

5.2 Issues and Future Scope of Work

The model proposed is simple and works well for image classification problem, but the visual operators should be carefully chosen. The selection of operators is important because many of the tasks like texture detection require very specific operators. The classification performance would be very poor if the operators selected are not discriminative enough. Parameters like minimum-support count, clustering parameters, etc., also require careful attention.

More powerful visual feature detectors like MSER (Forssen and Lowe [2007]) can be used to improve the accuracy of detection. It would be interesting to model the system to take advantage of multiple feature sets. A cost based system can be designed where the detection can start with cheap operators and as the initial "guessing" is done, we could apply more complex operators to finalize the detection.

The sequence of operator sets used to detect the class can itself be learned using sequential decision learners like U-Trees (Mccallum [1996]). The reward function used here can be modeled such that it considers the computational cost and gain of information for each operation.

Another possible modification would be to address other visual queries like, *How many objects are there in the scene ?*, *What is the red-object near to the table ?* etc. Here the operators should be chosen cleverly to abstract the various properties queried like numerosity, similarity, etc.

Publications

1. **Yousuf, A.**, and **Ravindran, B.** (2010), Visual Object Detection using Frequent Pattern Mining. In Proceedings of the Twenty Third FLorida AI Research Society Conference (FLAIRS 2010), 98-103. AAAI Press.

REFERENCES

Agarwal, S. and **A. Awan** (2004). Learning to detect objects in images via a sparse, part-based representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, **26**(11), 1475–1490.

Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, **13**(2), 111–122.

Bay, H., **A. Ess**, **T. Tuytelaars**, and **L. V. Gool** (2008). Surf: Speeded up robust features. *Computer Vision and Image Understanding (CVIU)*, **110**, 346–359.

Dickinson, S. (2009). The evolution of object categorization and the challenge of image abstraction. *Cambridge University Press*, 1–37.

Dietterich, T. G. (1998). The MAXQ method for hierarchical reinforcement learning. *Proc. 15th International Conf. on Machine Learning*, 118–126.

Fei-Fei, L., **R. Fergus**, and **P. Perona** (2004). One-shot learning of object categories. *IEEE Trans. Pattern Recognition and Machine Intelligence*, **28**, 594–611.

Felzenszwalb, P. F., **D. A. McAllester**, and **D. Ramanan**, A discriminatively trained, multiscale, deformable part model. *In CVPR*. 2008.

Forssen, P.-E. and **D. Lowe** (2007). Shape descriptors for maximally stable extremal regions. *IEEE 11th International Conference on Computer Vision*, **CFP07198-CDR**, 1–8.

Frintrop, S., *VOCUS: A Visual Attention System for Object Detection and Goal-Directed Search*, volume 3899 of *Lecture Notes in Computer Science*. Springer, 2006.

Han, J., **J. Pei**, and **Y. Yin** (2000). Mining frequent patterns without candidate generation. *SIGMOD Rec.*, **29**, 1–12.

Itti, L. and **C. Koch** (2000). A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision Research*, **40**, 1489–1506.

Itti, L., **C. Koch**, and **E. Niebur** (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20**(11), 1254–1259.

Ke, Y. and **R. Sukthankar** (2004). Pca-sift: A more distinctive representation for local image descriptors. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, **2**, 506–513. ISSN 1063-6919.

Lazebnik, S., **C. Schmid**, and **J. Ponce** (2004). Semi-local affine parts for object recognition. *British Machine Vision Conference*, 959–968.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**, 91–110.

Mccallum, A. K. (1996). *Reinforcement learning with selective perception and hidden state,*. Ph.D. thesis, The University of Rochester. Supervisor-Ballard, Dana.

Mikolajczyk, K. and **C. Schmid** (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, **27**(10), 1615–1630.

Minut, S. and **S. Mahadevan** (2001). A reinforcement learning model of selective visual attention. *Proceedings of the Fifth International Conference on Autonomous Agents*, 457–464.

Mori, G., **S. Belongie**, and **J. Malik** (2005). Efficient shape matching using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**(11), 1832–1837.

Oliva, A. and **A. Torralba** (2006). Building the gist of a scene: the role of global image features in recognition. *Progress in brain research*, **155**, 23–36.

Pineau, J. and **S. Thrun** (2001). Hierarchical pomdp decomposition for a conversational robot.

Simon, S., **F. Schwenker**, **H. Kestler**, **G. Kraetzschmar**, and **G. Palm** (2002). Hierarchical object classification for autonomous mobile robots. *Artificial Neural Networks, ICANN 2002*, **2415**, 135–135.

Sridharan, M., **J. Wyatt**, and **R. Dearden** (2008). Hippo: Hierarchial pomdps for planning information processing and sensing actions on a robot. *International Conference on Automated Planning and Scheduling*, 346–354.

Ullman, S. (1984). Visual routines. *Cognition*, **18**, 97–156.

Yarbus, A. L. (1967). Eye movements and vision. *Vision Science: Photons to Phenomenology, Plenum Press*, **chapter VII**.

Zhu, S. C. and **D. Mumford** (2006). A stochastic grammar of images. *Found. Trends. Comput. Graph. Vis.*, **2**(4), 259–362.