# Table of Contents

Workshop Co-Chairs, Steering and Program Committees .................................................. 5

Foreword ............................................................................................................................. 6

A multiversion model for multimedia data warehouse  
Anne-Muriel Arigon, Anne Tchounikine, and Maryvonne Miquel .................... 7

Collaborative Multi-strategy Classification of Complex Data: Application to per-pixel Analysis of Images  
Pierre Gançarski and Cedric Wemmert ................................................................. 15

A Framework to Support Multiple Query Optimization for Complex Mining Tasks  
Ruoming Jin, Kaushik Sinha and Gagan Agrawal ................................................ 23

A generalized metric distance between hierarchically partitioned images  
Maude Manouvrier and Marta Rukoz ................................................................. 33

Effective Image and Video Mining: An Overview of Model-Based Approaches  
Rokia Missaoui and Roman M. Palenichka ............................................................ 43

Multiple Sensor Integration For Indoor Surveillance  
Valery A. Petrushin, Gang Wei, Rayid Ghani and Anatole V. Gershman .............. 53

Classify By Representative Or Associations (CBROA): A Hybrid Approach for Image Classification  
Vincent. S. Tseng, Chon-Jei Lee and Jia-Hwuei Su .............................................. 61

OCRS: An Interactive Object-based Image Clustering and Retrieval System  
Chengcui Zhang and Xin Chen ................................................................. 71

A Framework for a Video Analysis Tool for Suspicious Event Detection  
Gal Lavee, Latifur Khan and Bhavani Thuraisingham ........................................... 79

Message Correlation in Automated Communication Surveillance through Singular Value Decomposition  
Ryan Layfield, Latifur Khan and Bhavani Thuraisingham ..................................... 85

A Similarity Measure for Motion Stream Segmentation and Recognition  
Chuanjun Lee and B. Prabhakaran ................................................................. 89
Analyzing user’s behavior on a video database
Sylvain Mongy, Fatma Bouali and Chabane Djeraba.................................95

Seeing and Reading Red: Hue and Color-word Correlation in Images and Attendant Text on the WWW
Shawn Newsam.................................................................101

Learning a Distance Metric to preserve the Semantic Content in Mining Graphical Data
Aparna S. Varde, Elke A. Rundensteiner, Carolina Ruiz, Mohammed Maniruzzaman, Richard D and Sisson Jr..........................................................107
**Workshop Co-Chairs, Steering and Program Committees**

**WORKSHOP CHAIRS**
Fatma Bouali - University of Lille2, France  
Latifur Khan - University of Texas at Dallas  
Florent Masseglia - INRIA Sophia Antipolis, France

**WORKSHOP STEERING COMMITTEE**
Chabane Djeraba - University of Sciences and Technologies of Lille, France  
Valery A. Petrushin - Accenture Technology Labs, USA  
Simeon J. Simoff - University of Technology Sydney, Australia

**WORKSHOP PROGRAM COMMITTEE**
Ramazan S. Aygun - University of Alabama, Huntsville  
John Risch Battelle - Pacific Northwest National Lab, USA  
Omar Boussaid - University of Lyon 2, France  
K. Slecuk Candan - Arizona State University, USA  
Claude Chirsment - University of Toulouse, France  
Chitra Dorai - IBM Thomas J. Watson Research Center, USA  
Alex Duffy - University of Strathclyde, UK  
Pierre Gançarski - University of Strasbourg, France  
William Grosky - University of Michigan, USA  
Howard J. Hamilton - University of Regina, Canada  
Wynne Hsu - National University of Singapore, Singapore  
Oktay Ibrahimov - Institute of Cybernetics, Azerbaijan  
Manfred Jeusfeld - Tilburg University, Netherlands  
Joemon M Jose - University of Glasgow, UK  
Odej Kao - Technical University of Clausthal, Germany  
Paul Kennedy - University of Technology-Sydney, Australia  
Brian Lovell - University of Queensland, Australia  
Mark Maybury - MITRE Corporation  
Dennis McLeod - University of Southern California, USA  
Dunja Mladenic - J. Stefan Institute, Slovenia  
Milind Naphade - IBM T.J. Watson Research Center, USA  
Mario Nascimento - University of Alberta, Canada  
Monique Noirhomme-Fraiture - Institut d'Informatique, FUNDP, Belgium  
Andreas Nürnberg - University of Magdeburg, Germany  
Vincent Oria - New Jersey Institute of Technology, USA  
Tom Osborn - The NTF Group, Australia  
Balakrishnan Prabhakaran - University of Texas at Dallas  
Christian Preda - University of Lille 2, France  
Simone Santini - University of California, San Diego, USA  
John R. Smith - IBM T. J. Watson Research Center, USA  
Yuqing Song - The University of Michigan at Dearborn  
Sundar Venkataraman - (Rockwell Scientific Corporation)  
Zhaohui Tang - Microsoft, USA  
Brigitte Trousse - INRIA Sophia Antipolis, France  
Duminda Wijesekera - George Mason University, USA  
Wensheng Zhou - Hughes Research Lab, USA  
Djamel Zighed - University of Lyon 2, France
Foreword

Data mining efforts are getting beyond the area of databases. They have been focusing on data collected in fields like art, design, hypermedia and digital media production, case-based reasoning and computational modeling of creativity, including evolutionary computation, and medical multimedia data. These exotic fields use variety of data sources and structures, interrelated by the nature of the phenomenon that these structures describe. As a result there is an increasing interest in new techniques and tools that can detect and discover patterns that can lead to a new knowledge in the problem domain where the data has been collected. There is also an increasing interest in the analysis of multimedia data generated by different distributed applications, like collaborative virtual environments, virtual communities, and multi-agent systems. These heterogeneous multimedia data records require sophisticated preprocessing, synchronization and other transformation procedures before even getting to the analysis stage.

On the other hand, researchers in multimedia information systems, in the search for techniques for improving the indexing and retrieval of multimedia information are looking into new methods for discovering indexing information. Variety of techniques from machine learning, statistics, databases, knowledge acquisition, data visualization, image analysis, high performance computing, and knowledge-based systems, have been used mainly as a research handcraft activity. The development of multimedia databases and their query interfaces recall again the idea of incorporating multimedia data mining methods for dynamic indexing. The emerging international standard for multimedia content description (MPEG-7) promises to foster the collaboration in the field giving a uniform data representation.

The 6th Workshop on Multimedia Data mining continues the series of successful workshops that have been held in conjunction with KDD Conferences in Boston (2000), San Francisco (2001), Edmonton (2002), Washington (2003) and Seattle (2004). These workshops brought together numerous experts in spatial data analysis, digital media, multimedia information retrieval, state-of-art data mining and knowledge discovery in multimedia database systems, and data analysis in collaborative virtual environments. For more information about the workshops see the reports in SIGKDD Exploration (2(2), 3(2) and 4(2)).

The topic of the 6th Workshop is Multimedia Mining. It means that the Workshop focuses on issues related to mining information from multi-modality, multisource, multi-format data in an integrated way. The selected papers represent a wide range of approaches, techniques, and tool that cover different aspects of image, audio and video indexing and annotation, and their integration into intelligent decision making systems. Furthermore, in order to encourage submissions of greenhouse work, which present early stages of cutting-edge research and development, this year a "short papers" session is added to the program of MDM 2005. This session will give to research teams the opportunity to present their research problems. In addition to the program of accepted papers, the workshop will enjoy an invited speaker: Professor Christos Faloutsos, Dept of Computer Science, Carnegie Mellon University, USA.

As a part of the SIGKDD conference series the Workshop follows a rigid peer-review and paper selection process. Each paper has been reviewed by at least two reviewers. We would like to thank all reviewers and all members of Steering and Program Committees who helped to organize this year workshop. We also would like to thank all those who submitted their papers to the Workshop. This year we received submissions from five countries: Azerbaijan, Canada, France, Taiwan and the United States of America. This year, the proceedings of the MDM workshop are also published in the ACM digital library (ACM DL).

Fatma Bouali, Latifur Khan and Florent Masseglia
July 2005
A multiversion model for multimedia data warehouse

Anne-Muriel Arigon
Laboratoire de Biométrie et Biologie Evolutive
UMR CNRS 5558
UCBL, 43 bd. du 11 novembre 1918
69622 Villeurbanne Cedex, France
arigon@biomserv.univ-lyon1.fr

Anne Tchounikine
Laboratoire d'InfoRmatique en Images et Systèmes d'information
UMR CNRS 5205
INSA, 7 avenue Capelle
69621 Villeurbanne Cedex, France
Anne.Tchounikine@insa-lyon.fr

Maryvonne Miquel
Laboratoire d'InfoRmatique en Images et Systèmes d'information
UMR CNRS 5205
INSA, 7 avenue Capelle
69621 Villeurbanne Cedex, France
Maryvonne.Miquel@insa-lyon.fr

ABSTRACT

The traditional multidimensional models have a static structure where members of dimensions are computed in a unique way. However, multimedia data is often characterized by descriptors that can be obtained by various computation modes. We define these computation modes as “functional versions” of the descriptors. We propose a Functional Multiversion Multidimensional Model by integrating the concept of "version of dimension". This concept defines dimensions with members computed according to various functional versions. This new approach integrates a choice of computation modes of these members into the model, in order to allow the user to choose the best representation of data. We implement a multimedia data warehouse in the medical field by integrating the multimedia data of a therapeutic study into a multidimensional model. We formally define a conceptual model and we present a prototype for this study.

Keywords
Data warehouse, OLAP, multimedia, functional version, descriptor

1. INTRODUCTION

Modern areas produce increasingly voluminous amounts of electronic complex data. As an example, in the medical area, conventional administrative data, therapeutic and diagnostic data now are completed with complex multimedia data such as X-ray pictures, echography, electrocardiogram, etc... captured by electronic medical devices. This statement of fact, like in business or retail areas, raised the idea to extract and gather these data lying in heterogeneous and distributed system (HIS, PACS...) in order to discover useful information. Indeed, medical research requires large sets of data coming from various sources collected for the purpose of analysis and extraction of information. These databases are often organized around a pathology or class of pathologies, and are used to validate research hypothesis and to build clinical knowledge. Before being introduced into the database, data must be validated by experts in order to guarantee the quality and the coherence of the unit. In that case, the constitution of an expertized database is an expensive work in time and human investment. It is only natural that clinical researchers are interested in data warehousing and in On Line Analysis Processing (OLAP) technologies in order to constitute and manage high quality data. With these technologies, they also have the capacity to navigate into large data sets according to their needs (epidemiological studies, population follow-up, and evaluation of indicators...). In a field such as cardiology, data, which are used for clinical studies, are not only alphanumeric, but can also be composed of images or signals. The environment of analysis must include processing methods for these types of data in order to compute or extract the knowledge that is embedded into raw data. OLAP commercial environment such as it is currently designed in traditional applications can only implement very simple operations on data and their use for multimedia data imposes an adaptation.

Another important problem which is barely addressed is due to the semantic of medical data, and to the way physicians deal with it. In cardiology area, for instance, the main goal of many researches is to find indicators and descriptors to understand and characterize heart pathologies. This goal induces the scientists to develop efficient algorithms (based on signal or image processing, pattern recognition, statistical methodologies...) to transform the initial data (e.g. an electrocardiogram ECG, representing the twelve leads of a standard electrocardiogram) or a set of data (current ECG, past ECG, biological patients' data,...) into relevant information (data descriptors, risk factors, diagnosis class...) and to validate them on a large scale database. Beyond the difficulties encountered in extracting and modeling medical data, this example emphasizes the difficulty in manipulating, interpreting the data and the need in promoting raw data into useful information. These processes are often very hard to elaborate, and one should be able to share this knowledge, and then to allow the user to choose how to calculate his indicators. Thus, these processes should be fully, tightly integrated in the warehouse and be part of it.

In this article we propose a model for multimedia data warehouse. This model integrates different functional versions for dimension members and allows navigation and comparison between these different modes of calculus. Section 2 presents definitions and requirements for a multimedia data warehouse and related works. In section 3 we describe our case study in the cardiology field. Section 4 details our multiversion model. The prototype and data
warehouse used for the case study is presented in section 5. Section 6 concludes.

2. MULTIMEDIA DATA WAREHOUSES

2.1 Data warehouse: definition and architecture

A Data Warehouse is a "subject-oriented, integrated, non-volatile and time-variant collection of data in support of management's decisions" [10]. A data warehouse is a single site repository of information collected from multiple sources. Information in the data warehouse is organized around major subjects and is modeled so as to allow pre-computation and fast access to summarized data. "OLAP" as defined in warehouse [5,19] refers to analysis techniques used to explore the data.

Data warehouse has become a leading topic in the commercial world as well as in the research community. Until now, data warehouse technology has been mainly used in business world, in retail or finance areas for examples. The leading motivation is to take benefits from the enormous amount of data that relies in operational databases. According to Kimball [12], the data-modeling paradigm for a data warehouse must comply with requirements that are profoundly different from the data models in OLTP environments. The data model of the data warehouse must be easy for the end-user to understand and write queries, and must maximize the efficiency of queries. Data warehouse models are called multidimensional models or hypercubes and have been formalized by several authors [1,4,7,14,18]. They are designed to represent measurable facts or indicators and the various dimensions that characterize the facts. As an example, in a retail area, typical facts are price and amount of a purchase, dimensions being product, location, time and customer. A dimension can be organized in hierarchy, for example the location dimension can be aggregated in city, state, country. The "star schema" models the data as a simple cube, in which the hierarchical relationship in a dimension is not explicit but is rather encapsulated in attributes. The "snowflake schema" normalizes dimension tables, and makes it possible to explicitly represent the hierarchies by separately identifying dimension in its various granularities. At last, when multiple fact tables are required, the "galaxy schema", or "fact constellation" model allows the design of collection of stars.

2.2 Data Warehouses for complex multimedia data

Multidimensional models usually consider facts as the dynamic part of data warehouses, and dimensions as static entities [15]. Thus members of dimensions are computed once, and in one single manner. However, this can restrict the analysis of data, particularly in the case of multimedia data. Indeed, multimedia data are bulky data, with various formats (text, graph, video, sound...). These data are generally described by content-based or description-based descriptors. Description-based descriptors are extrinsic information (e.g. key words, date of acquisition, author, topic...). Content-based descriptors represent the content of data (e.g. color, texture, form,...) and are generally automatically extracted from the data [8,22]. Another type of content-based descriptors is data-specific, i.e. descriptors computed by specific processes applied on the multimedia data, such as the various measurements calculated on an ECG signal. In multimedia data warehouses, descriptors are usually used to become some of the dimensions of the multidimensional model. Warehousing multimedia data require specific extraction processes, adapted tools for visualization and the definition of specific multimedia aggregation functions.

Candidate descriptors for multimedia data are numerous and the same descriptor can be extracted in various ways. These computation modes can be seen as functional versions of a descriptor, and make possible the characterization of the data by various ways. Thus dimensions should integrate multiple methods...
to represent the data according to the various functional versions of each descriptor. Therefore, we argue that the integration of functional versions of dimension into the multidimensional model, corresponding to the data descriptors computed by different functions, can improve the characterization and the analysis of the data. Indeed, it will enable the user to choose the computation modes for each descriptor in order to define the best interpretation and representation of the multimedia data. He will also be able to compare the different results obtained using different versions of computation modes for the descriptors. The aim of our study is to define a multidimensional conceptual model capable of managing multimedia data characterized by descriptors obtained by various computation modes. This model is illustrated by a case study on a medical multimedia data warehouse.

2.3 Related Work

Various works about the design of multimedia data cubes were undertaken to improve multidimensional analysis of large multimedia databases [20]. The MultiMediaMiner project [21] uses a multimedia data cube in order to store the multidimensional data and to incorporate them with different granularities. In the medical domain, the problems of exploration and analysis of huge multimedia data are omnipresent. For example, [23] is a study undertaken in the domain of breast cancer detection, and proposes a data warehouse integrating numerical mammography. Another study [17] deals with the problem of storage and restitution of medical image data from a warehouse by comparing the data warehouse with a pyramid of means of storage. All these studies use multimedia data modeling based on data descriptors computed at loading time. Then, the design of the multimedia data cube follows the design of traditional data cubes paradigm, i.e. dimension members are computed during the ETL process and remain static. Dynamic aspects in multidimensional models can be found in studies that deal with time-related evolutions. Two types of approaches are proposed in order to take evolutions of the analysis structures into account: the first one consists in updating models [2,9,10] that map data into the most recent version of the structure; the second is based on tracking history models [3,6,15,16] that keep trace of evolutions of the system. This last type of model is particularly interesting because it makes it possible to analyze data in their various versions and their evolutions. However these models focus on temporal evolutions and mapping processes applied on facts. They do not offer navigation into functional version of dimensions which could enable comparison between extraction and/or interpretation processes.

3. A CASE STUDY IN CARDIOLOGY

In the case of disease studies, researchers try to extract information that best reflect the derangement from a chosen population. It is difficult to establish before the study the set of descriptors that characterizes the multimedia data associated to a patient as well as the methodological approach to process these indicators. However, the design of the methods for information extraction is often a part of the research task. When several methods compete, the aim is to evaluate them. If the purpose is to study the characteristics and the evolution of descriptors taken of a specific population, the descriptor calculation is at least as significant as the selection of the raw data. In all these cases, one notes that an environment of analysis of medical multimedia data must jointly allow the selection of the data to be treated and the methods to be applied on the data for their processes. In order to organize a large set of multimedia data, to allow different ways to calculate their descriptors and to help to constitute study population, we propose a model for a multimedia data warehouse.

We worked with an INSERM team (ERM 107) specialized in the methodology of cardiological information. In the EMIAT project (European Myocardial Infarct Amiodarone Trial), the aim of the ERM107 team is to extract knowledge about descriptors that can be used to evaluate a cardiac pathology evolution. The EMIAT study was designed to compare the drug Amiodarone to placebo. It included a total of 1486 survivors of acute myocardial infarction. As a result this study provides a significant amount of data to be exploited and analyzed. Among this data, there is multimedia data: the Electrocardiogram signals (ECGs) (figure 3) of the various patients who participated in the study.

Figure 3. ECG signal and some descriptors

Several descriptors or indicators can be computed from an ECG to characterize the cardiac health state of a patient. The QT interval measure (time after which the ventricles are repolarized) is the most studied. The noise level is an often used criterion for the selection of a study population because it gives a quality indication. Other data are associated with these ECGs, such as the patient’s pathology. Thus, two types of descriptors characterize ECGs: description-based descriptors (principal pathology, age, gender of the patient, ECG acquisition slot, technology with which the ECG is obtained) and content-based descriptors (the QT duration and the noise level of the signal). The facts are ECGs characterized by descriptors (corresponding to the dimensions) organized in complex hierarchies. As an example, the ECG acquisition slots can be classified in hours and then in periods (night, waking, day). Some sets of hours (ex., 6 am) can belong to several periods (waking and night). The dimension Time corresponding to this descriptor is then organized in non-strict hierarchy. A simple multidimensional model of the data warehouse shows the fact table (ECGs Signal) and the dimensions we use (Time, Duration of QT, gender,…) (Figure 4).

Figure 4. Schema of the EMIAT data warehouse

This data warehouse will be used in order to analyze the ECGs of a population with selected age, pathology… Furthermore, some descriptors can be computed by various computation modes. As an example, the duration of the QT can be obtained thanks to
several algorithms. Thus the user should be able to choose the relevant computation mode (or the functional version) of this descriptor.

On the one hand, our model will incorporate the versions of dimension that we define as dimensions with members that are calculated according to various functional versions of descriptors, and on the other hand, we will integrate explicit hierarchies and complex hierarchies.

### 4. THE MODEL

#### 4.1 General Principle of our Approach

Our approach is based on a fact table that gathers the set of measures which represent the data to analyze (references on multimedia data) and dimensions that represent the descriptors of this multimedia data. To take the problem of functional multiversion into account, we redefine the multidimensional structure by adding the concept of functional version. Thus, we introduce the concepts of version of dimension, multiversion dimension, functional multiversion fact table and function of version of dimension. A multiversion dimension is composed of several versions of a dimension, each one being a dimension for a given version with its own schema and its own instance. The functional multiversion fact table gathers all data by combining the various versions of dimension of a multiversion dimension with the others. Finally, the functions of versions of dimension correspond to the computation modes that make it possible to obtain its members. We define the schemas of various dimensions by describing the hierarchical levels and links that bind them. We also describe the instances of these dimensions by describing the set of members belonging to a hierarchical level and their parent-child relationships. Thus, our approach makes it possible to have explicit dimensions since the schemas of dimension are defined explicitly and our model also supports complex hierarchies (multiple, non-onto, non-strict and non-covering hierarchies) since the instances of dimensions are built from the members and the hierarchical links.

### 4.2 Concept Definitions

**Definition 1 (Schema of version of dimension).** A schema of version of dimension is a schema of dimension for a given version. A version is a computation mode used to obtain the members of a dimension. The schema of version of dimension VDid is defined by the tuple \( <\text{VDid}, \mathcal{N}, [<\text{TId}>] > \) where:

- \( \text{VDid} \) is the identifier of the version of dimension
- \( \mathcal{N} = \{n_j, j=1,...,k\} \) is the set of levels of the schema \( S_{TD} \). A level in \( S_{TD} \) represents a set of values with the same granularity associated with the same version of dimension. A level \( n_j \) is defined by the tuple \( <\text{levelId}, \text{levelName}_j, \mathcal{A}, \text{description}> \) where:
  - \( \text{levelId} \) is the identifier for the level of version of dimension
  - \( \text{levelName}_j \) is the name for the level of version of dimension
  - \( \mathcal{A} \) is an optional property representing descriptive attributes of this level
  - \( \text{description} \) is an optional property representing textual information on the level \( n_j \)

- \( \mathcal{N} \times \mathcal{N} \) is a partial order on the set \( \mathcal{N} \) which defines the hierarchical links between the levels of schema \( S_{TD} \) when the number of levels is more than 1. The partial order \( \leq_{\text{TId}} \) is defined such as: \( \forall (n_1,n_2) \in \mathcal{N} \times \mathcal{N} \), if \( n_1 \leq_{\text{TId}} n_2 \) then \( n_1 \) has a granularity finer than \( n_2 \).

Thus, a schema of version of dimension can be seen as a directed graph, where nodes are elements of the set \( \mathcal{N} \) and arcs are relations according \( \leq_{\text{TId}} \). This graph must be acyclic in order to enable aggregations to the least fine hierarchical levels. One defines a level ALL as the root of the hierarchy, i.e. the highest level of granularity.

**Example 1.** Suppose we want to analyze the influence of age on ECG. Age is a dimension of the data warehouse but its members can be ordered in various ways, the ages can be classified by intervals of age, i.e. five years interval, ten years, fifty years.

Let \( S_{\text{agePerInterval}} \) be a schema of the version of dimension "agePerInterval" of which \( VDid = 1 \). The schema of this version of dimension is defined by:

\[
S_{\text{agePerInterval}} = \{ [n_1, n_2, n_3, n_4, n_5, n_6, n_7], \mathcal{A}, \mathcal{A} \}
\]

with

\[
\begin{align*}
(n_1, IntervalOf5) & = <1, 5> \quad \text{IntervalOf5} \\
(n_2, IntervalOf10) & = <2, 10> \quad \text{IntervalOf10} \\
(n_3, IntervalOf50) & = <3, 50> \quad \text{IntervalOf50}
\end{align*}
\]

and the following order: \( n_1 < n_2 < n_3 \). The schema of the version of dimension \( S_{\text{agePerInterval}} \) can be represented by the following graph (Fig. 2).

![Figure 2. Schema of the version of dimension S_agePerInterval](image)

In a 2nd way, one can gather these ages in age classes (young child, child, teenager, young adult, adult, senior) then in categories (minor, major) and one can define the schema of this version of dimension \( S_{\text{agePerClass}} \).

Now let us regard the duration of the QT of these electrocardiograms as another dimension of the data warehouse. It can be computed by several algorithms, i.e. algo1 and algo2. The schema of dimension characterizing the duration of the QT has as hierarchies, the values of the duration of the QT for the finest level, gathered into interval of 100ms, then in interval of 400ms. So one can define the schemas of these two versions of dimension \( S_{\text{Qalgo1}} \) and \( S_{\text{Qalgo2}} \):

**Definition 2 (Version of dimension).** A version of dimension is a dimension for a given version. The version of dimension \( VD \) of schema \( S_{TD} = < VDid, \mathcal{N}, [<\text{TId}>] > \) is defined by the tuple \( <\text{VDid}, \text{VName}, \mathcal{M}, [\text{VDescription}> \) where:

- \( \text{VDid} \) is the unique identifier for the version of dimension
- \( \text{VName} \) is the name for the version of dimension
- \( \mathcal{M} = \{m_i, j=1,...,l\} \) is the set of members of this version of dimension. A member of version of dimension is a member computed by the computation mode corresponding to the version of dimension. It belongs to one of the levels of the schema \( S_{TD} \). Thus, one gathers in a level the members with the same granularity. A member \( m_i \) is represented by a tuple \( <\text{id}_i, \text{val}_i, \text{levelId}> \) where:
  - \( \text{id}_i \) is a unique identifier for this member of version of dimension
  - \( \text{val}_i \) is a set of values for this member of version of dimension
  - \( \text{levelId} \) is a set of levels for this member of version of dimension
In the same way, one can also define the versions of dimension "agePerClass", "QTalgo1" and "QTalgo2" whose the schemas are $S_{agePerClass}$, $S_{QTalgo1}$ and $S_{QTalgo2}$ as well as the sets $LM_{agePerClass}$, $LM_{QTalgo1}$ and $LM_{QTalgo2}$.

Definition 3 (Multiversion dimension). A multiversion dimension $MVD$ is a dimension that contains 1 to $n$ versions of dimension. It is defined by the tuple $< MVDId, MVDname, VD, \{ MVDdescription \} >$ where:

- $MVDId$ is the unique identifier for the multiversion dimension
- $MVDname$ is the name for the multiversion dimension
- $VD = \{ VD_i, i=1,...,n \}$ is the set of versions of dimension associated with this multiversion dimension
- $MVDdescription$ is an optional property containing textual information on the multiversion dimension.

One notes $LM_{MVD}$ the set of leaf members of the versions of dimension contained in the multiversion dimension $MVD$. This set is defined by:

$$LM_{MVD} = \bigcup_{i=1}^{n} LM_{VD_i}$$

with $n$ the number of versions of dimension contained in the multiversion dimension $MVD$.

Example 3. The versions of dimension "agePerInterval" and "agePerClass" defined previously belong to the multiversion dimension "Age" with the identifier 1. This multiversion dimension is defined by:

$$\text{Age} = \langle 1, \"Age\", \{ \"agePerInterval\", \"agePerClass\" \} \rangle$$

In the rest of the paper, we will use the names of the members of versions of dimension to identify them. One defines the set $LM_{Age}$ by:

$$LM_{Age} = \{ 0-5, 6-10, 11-15, 16-20, \text{young child}, \text{child}, \text{teenager}, \text{young adult}, \text{adult}, \text{senior} \}$$

The versions of dimension "QTalgo1" and "QTalgo2" belong to multiversion dimension "DurationQT" with the identifier 2. This multiversion dimension is defined by: $\text{DurationQT} = \langle 2, QT, \{ QTalgo1, QTalgo2 \} \rangle$

In the same way, one defines the set $LM_{DurationQT}$.

Definition 4 (Functional multiversion fact table). A functional multiversion fact table provides the measures according to various versions of dimension. Let $\{ \mu_i, i=1,...,m \}$ be the set of measurements, a functional multiversion fact table $ft$ is defined by a function such as:

$$f: MVD_1 \times MVD_2 \times \ldots \times MVD_n \rightarrow \text{dom}(\mu_1) \times \ldots \times \text{dom}(\mu_n)$$

where $n$ is the number of multiversion dimensions of the data warehouse, $mi \in LM_{MVD}$ with $i=1,...,n$ and $\text{dom}(\mu_i)$ is the range for the measure $\mu_i$. This function associates the set of the values $V_i$ of measures $\mu_i$ with a set of leaf member of the versions of dimension of each multiversion dimension.

Definition 5 (Function of version of dimension). The functions of version of dimension are the computation modes that make it possible to obtain the members of a version of dimension $VD$ from the data of the production database. A function of version of dimension $f_{VD}$ is defined by the tuple $< functionId_{VD}, VDid, functionName_{VD}, functionDefinition_{VD} >$ where:

- $functionId_{VD}$ is the identifier for the function of version of dimension $VD$
- $VDid$ is the identifier for the version of dimension $VD$ whose members are computed by using this function of version of dimension
- $functionName_{VD}$ is the name for the function of version of dimension
5. PROTOTYPE

The model has been implemented in three tier architecture:

- a functional multiversion multimedia data warehouse in which the multiversion dimensions and the functional multiversion fact table are stored,
- an OLAP cube built from the functional multiversion multimedia data warehouse, using aggregations, which enables requests against the functional versions of the dimensions,
- a tool for data navigation and visualization.

We use Microsoft SQL Server and Analysis Services to implement the prototype. The aggregations functions are implemented in Visual Basic; an interface using Proclarity 4.0 is built for navigation. The data is loaded from the production database to the functional multiversion multimedia data warehouse using the functions of versions of dimension.
The user can visualize in a frame the hierarchies and the members of the versions of dimensions and build his request choosing the appropriate data aggregation operator (“ECG-count”, “ECG-list” or “medium-ECG”) (figure 5, part C). The result is a multidimensional table (figure 5, part B). It is also possible to visualize and to choose several versions of dimension of a multiversion dimension in order to make comparisons (figure 5, part A). The selected multimedia data can be visualized in a frame (figure 5, part D). Moreover, metadata can be visualized in order to have a global view of all versions of dimension and to navigate the data cube more easily (e.g. schemas, instances of each version of dimension).

6. DISCUSSION AND CONCLUSION

We present a model that takes functional versions into account. The model helps to manage multimedia data and allows the user to choose different views that represent various functional versions of the descriptors. We define the concept of version and multiversion in dimensions in order to compare results obtained by various versions. This model is particularly well suited to multimedia data because they require various computation modes.

We also use specific aggregation functions for multimedia data that are integrated into the data warehouse. This model is used to develop an OLAP application for the navigation into a hypercube integrating signal data. We propose a tool to explore this complex data which improves navigation in the multidimensional data cube. Thus, we enable the visualization of data according to several methods of analysis and we provide the possibility to visualize the representation of this multimedia data.

However, the data storage of our model could be improved. It is possible to have some redundancy in the schemas of versions of dimension, the functional multiversion fact table storage is not optimised, and the treatment of non-strict and multiple hierarchies imply duplications. Our model could be extended by associating the notion of functional version with the facts, in the same way that our model associates functional versions with dimensions. This could be possible by adding a version of fact dimension so as to enable the user to choose the version of fact.

7. REFERENCES


Collaborative Multi-strategy Classification: Application to per-pixel Analysis of Images

Pierre Gançarski and Cédric Wemmert
LSIIT - Data Mining Research Group
University Louis Pasteur
Strasbourg, France
{gancarski, wemmert}@lsiit.u-strasbg.fr

ABSTRACT
This paper presents a new process of collaborative multi-step multi-strategy classification of complex data. Our goal is to be able to handle in the same system several instances of classifiers in order to make them collaborate. In this paper, we highlight how the classifiers collaborate. We present the implementation of our method dedicated to remote sensing images. Finally, we validate it with a pixel based classification application.

Keywords
Complex data. Collaborative clustering. Classification combining. Per-pixel image analysis.

1. INTRODUCTION
For many years data mining methods are being used on more and more complex data: intervals, distributions, histogram, fuzzy data, temporal data, images, etc.

Many discussions are currently going on to define and formalize what should be considered as complex data. Nevertheless, complex data are commonly seen as sets of strongly heterogeneous data, often unstructured which may arise from different theoretical approaches (observations, knowledge a priori, trainings, etc.).

Unfortunately, in general, in objects described by a large set of features, many features are correlated, some are noisy or irrelevant. For example, in per-pixel clustering of remote sensing images, the abundance of noisy, correlated and irrelevant bands disturb the classical clustering procedures. In fact, traditional methods are not as effective as they seem. Consequently, the "traditional" process of knowledge discovery from such data becomes more and more complex. In particular, new models of classifiers (supervised or not) which are able to handle complex data of different types (numerical, symbolic or structured) are necessary.

A relatively recent approach can be used. It is based on the idea that the information offered by different classifiers about objects are complementary [17]. And thus the combination of different classification methods may increase their efficiency and accuracy. A single classification is produced from results of methods having different points of view: all individual classifier opinions are used to derive a consensual decision.

There are many different ways to combine multiple classifiers depending on the representational methodology. In [1, 17], the authors divide them in:

- **multi-expert approaches** like boosting [22, 6, 20, 3], bagging [5, 21] or stacked generalization [27, 24, 23]: the different classification methods work in parallel and give all their classifications; then the final classification is computed by a separate combiner;

- **multi-step approaches** like cascading [2, 10]: the methods work in a serial way; each method is trained or consulted only for the patterns rejected by the previous classifier(s).

In both cases, the approach is described as:

- **multi-strategy** if different types of algorithms are used at the same time[4];

- **multi-representationnal** if the algorithms use different data or different points of view on the data [9].

In our work, we focus on unsupervised classification. The goal of clustering is to identify subsets called clusters (or classes) from the data, where a cluster usually corresponds to objects that are more similar to each other than they are to objects from other clusters. Clustering is carried out in an unsupervised way by trying to find these subsets without having a priori knowledge on the clusters: it operates only starting from the intrinsic properties of the objects. A complete panorama of the unsupervised classification existing methods is given in [16, 11]. In this case, we believe that the combination of classifiers should be able to propose a classification method that decreases the importance of the initial choices. And secondly, it should also solve some of the limitations of the methods by using the complementarity of the different classification methods used. For example, some classifiers only propose a partitioning of the data space, whereas others give a hierarchy of classes or concepts as a result. So it could be interesting to automatically produce a hierarchy of classes with the partitioning methods according to the results presented by the hierarchical methods.
Many techniques for combining supervised classifiers exist. Unfortunately, it is hard to apply the same schemes to the unsupervised case. First, these techniques are often monostatirical. Secondly, the fusion of decisions is harder because there is no direct correspondence between the clusters found by the different classifiers. And finally, few of these methods are able to use different representations of data.

Nevertheless, we believe that these traditional approaches for combining classifiers can be used and improved if the methods collaborate during the entire classification process. Each method can use a different strategy and/or representation of data.

Thus, we propose a new method including the two significant aspects:

- the collaborative multi-strategy aspect with a classification method of complex data based on an automatic and mutual refinement of several classification results;

- the multi-step aspect with a method which enables us to represent complex data (numeric or symbolic data, ...) and to use the preceding classes to form objects of higher level.

In this paper, we highlight the collaborative multi-strategy aspect. Then, we present the implementation of our method dedicated to remote sensing images. We validate it with a pixel based classification application.

2. COLLABORATIVE MULTI-STRATEGY CLASSIFICATION

To combine \( P \) classifications \( \{R^p\} \) using a voting method for example, it is necessary to define a correspondence function, associating to each class \( C^i_k \) of a classification \( R^k \) one class of the classification \( R^j \), for each couple \( (R^k, R^j) \).

To carry out this operation in an optimal way, this function should be bijective, as it is in supervised approaches. In the case of unsupervised classification, the results may not have the same number of classes and we do not have any information about the correspondence between the different classes of the different results. We propose to carry out a pretreatment based on a collaborative process which consists in an automatic and mutual refinement of the classification results. These refinements are done to make the results of the various classifications converge, that is they should have almost the same number of classes, and all these classes should be statistically similar. In many cases, it is then possible to define a bijective correspondence function and to apply a unifying technique, such as our new voting method [25].

The entire classification process is presented in figure 1. It is decomposed in three main phases:

1. First a phase of initial classifications is performed: classifications are computed by each method with its parameters.

2. An iterative phase of convergence of the results which corresponds to alternations between two steps, as long as the convergence and the quality of the results improve:
   - 2.1 A step of evaluation of the similarity between the results and mapping of the classes;

2.2 A step of refinement of the results.

3. A phase of combination of the refined results.

The mechanism we propose for refining the results is based on the concept of distributed local resolution of conflicts by the iteration of four phases:

2.2.1 Detection of the conflicts by evaluating the dis-similarities between couples of results

2.2.2 Choice of the conflicts to solve;

2.2.3 Local resolution of these conflicts (concerning the two results implied in the conflict);

2.2.4 Management of these local modifications in the global result (if they are relevant).

2.1 Convergence of the results

Evaluation and mapping. A class \( C^j_k \) of the result of a classification \( R^j \) is the corresponding class of the class \( C^i_k \) of the result \( R^i \) if it is most similar to \( C^i_k \). We define the similarity without using a concept of distance between objects by:

\[
\omega_{k}^{i,j} = \rho_k^{i,j} \alpha_{k,m}^{i,j} \text{ where } \alpha_{k,m}^{i,j} = \max \{ \alpha_{k,m}^{i,j} \}_{1 \leq i \leq n} \]

It is evaluated by observing:

- the relationship between the size of their intersection and the size of the class itself:

\[
\alpha_{k,m}^{i,j} = \frac{|C^i_k \cap C^j_m|}{|C^j_m|}
\]

- and by taking into account the distribution of the data in the other classes

\[
\rho_k^{i,j} = \sum_{l=1}^{n_j} \alpha_{k,l}^{i,j}
\]

Refinement. During a phase of refinement of the results, several local resolutions are performed in parallel. The detection of the conflicts consists, for all the classes of all the results, in seeking all the couples \( \{C^i_k\} R^j \) for two classifications \( R^i \) and \( R^j \), such as \( C^i_k \neq C^j_k \), which is its corresponding class. A conflict importance coefficient is calculated according to the interclass similarity between the two classes. Then a conflict is selected according to the conflict importance coefficient and its resolution is started. This conflict and all those concerning the two associated methods are removed from the list of conflicts. This process is reiterated until the list of conflicts is empty.

The resolution of a conflict consists in applying an operator to \( R^j \) and an operator to \( R^j \). These operators are chosen according to the classes \( C^i_k \) and \( C^j_m \) involved in the conflict:

- merging of classes: the classes to merge are chosen according to the representative classes of the treated class. The representative classes of class \( C^i_k \) from result \( R^i \) compared to result \( R^j \) are the set of classes from \( R^j \) which have more than \( p_{cr} \% \) of their objects included in \( C^i_k \) (\( p_{cr} \) is given by the user).
Figure 1: Collaborative multi-strategy classification process

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Data set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Step 2.1</td>
<td>Results evaluation</td>
</tr>
<tr>
<td></td>
<td>Evaluation coefficients</td>
</tr>
<tr>
<td>Step 2.2</td>
<td>Results refining</td>
</tr>
<tr>
<td></td>
<td>Refined result 1</td>
</tr>
<tr>
<td></td>
<td>Refined result m</td>
</tr>
<tr>
<td>Step 3</td>
<td>Results unification</td>
</tr>
<tr>
<td></td>
<td>Unified result</td>
</tr>
</tbody>
</table>

- splitting a class into subclasses: all the objects of one class are classified in a certain number of subclasses,
- reclassification of a group of objects: one class is removed and its objects are reclassified in all the other existing classes.

But, the simultaneous application of operators on $\mathcal{R}^i$ and $\mathcal{R}^j$ is not always relevant. Indeed, it does not always increase the similarity of the results implied in the conflict treated (Red Queen effect: "success on one side is felt by the other side as failure to which must be responded in order to maintain one’s chances of survival"[19]), and the iteration of conflict resolutions may lead to a trivial solution where all the methods are in agreement: a result with only one class including all the objects to classify, or a result having one class for each object.

So we defined the local concordance and quality rate which estimates the similarity and the quality for a couple of results by

$$\gamma^{i,j} = \frac{1}{2} \sum_{k=1}^{n_i} p \omega^{i,j}_k + p \delta^{i}_k + \sum_{k=1}^{n_j} p \omega^{i,j}_k + p q \delta^{j}_k$$

where $p + q = 1$ and $\delta^{i}_k$ is the class quality criterion chosen by the user ($0 < \delta^{i}_k \leq 1$). For example, with methods which include a distance measure, the user can select intra-class inertia. Without distance measure, it can use class predictivity (Cobweb), class variance (EM algorithm), ...

At the end of each conflict resolution, after the application of the operators, the couple of results (the two new results, the two old results, or one new result with one old result) which maximizes this rate is kept.

After the resolution of all the conflicts, a global application of the modifications proposed by the refinement step is decided according to the improvement of the global agreement coefficient:

$$\Gamma = \frac{1}{m} \sum_{i=1}^{m} \Gamma^{i}$$

where

$$\Gamma^{i} = \frac{1}{m - 1} \sum_{j=1, j \neq i}^{m} \gamma^{i,j}$$

is the global concordance and quality rate of the result $\mathcal{R}^i$ with all the other results.

Then a new iteration of the phase of convergence is started if the global agreement coefficient has increased and an intermediate unified result is calculated by combining all the results.

### 2.2 Combination of the results

All the results tend to have the same number of classes which are increasingly similar. There are two cases:

- it is possible to define a bijective correspondence function: it is then possible to apply a unifying technique
as a voting algorithm (bagging or boosting) [22, 6, 20, 3, 5, 21] or a reclassification algorithm (stacked generalization [27, 24, 23, 7]).

• it is not possible: we have defined a new voting method to perform this combination of the results [25].

Relevant classes and nonconsensual objects. It is possible to define two new concepts which are relevant classes and nonconsensual objects.

The relevant classes correspond to the groups of objects of a same class, which were classified in an identical way in a majority of the results used. Moreover, we can quantify this relevance by using the percentage of classifications that are in agreement.

These classes are interesting to highlight, because they are, in the majority of the cases, the relevant classes for the user.

Reciprocally, a nonconsensual object is an object that has not been classified identically in a majority of results, i.e. that does not belong to any of the relevant classes. These objects often correspond to the edges of the classes in the data space (for example in remote sensing image classification they may correspond to mixed pixels).

3. RESULTS

3.1 Automatic medical image segmentation

Our system was used to segment a medical image. The data was extracted from a sequence of images of slices of a human brain, obtained by magnetic resonance. Five attributes were assigned to each pixel of the image:

• its own value (grayscale between 0 and 255);
• the value of each of its 4-neighbors.

Two segmentations, $S_1$ et $S_2$, were achieved: the first ($S_1$) with three instances of the Kmeans algorithm, with 6, 10 and 14 initial nodes; the second one ($S_2$) was obtained by five instances of the Kmeans algorithm, with 4, 7, 10, 13 et 16 initial nodes. In both cases, the initial nodes were randomly chosen among the data.

The different coefficients found by the system are presented below:

<table>
<thead>
<tr>
<th>Segmentation $S_1$</th>
<th>$n^i$ initial</th>
<th>$\Gamma^i$ initial</th>
<th>$n^f$ final</th>
<th>$\Gamma^f$ final</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1^1$</td>
<td>6</td>
<td>0.53</td>
<td>5</td>
<td>0.76</td>
</tr>
<tr>
<td>$M_2^1$</td>
<td>10</td>
<td>0.48</td>
<td>5</td>
<td>0.80</td>
</tr>
<tr>
<td>$M_3^1$</td>
<td>14</td>
<td>0.46</td>
<td>5</td>
<td>0.79</td>
</tr>
</tbody>
</table>

$\Gamma = 0.78$

$\Gamma = 0.79$

Figure 2: Results of the two segmentations

As shown in the two results presented in Fig. 2 (final results of the two hybrid classifications), the two segmentations $S_1$ et $S_2$ are visually very similar. Moreover they both have 5 classes that are very similar (as shown by the confusion matrix between the two results, presented in table 1) and no result differs too much compared to the others ($0.74 \leq \Gamma^i \leq 0.82$). We can suppose that these 5 classes are relevant, because they were found by two different learnings, initialized differently. This proves that the importance of the initial choices is decreased because of the collaboration between different classification methods.

Table 1: Confusion matrix between the two results of $S_1$ et $S_2$

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_0$</td>
<td>14283</td>
<td></td>
</tr>
<tr>
<td>$C_1$</td>
<td>5227</td>
<td></td>
</tr>
<tr>
<td>$C_2$</td>
<td>937</td>
<td>8881</td>
</tr>
<tr>
<td>$C_3$</td>
<td>197</td>
<td>5452</td>
</tr>
<tr>
<td>$C_4$</td>
<td>741</td>
<td>3075</td>
</tr>
</tbody>
</table>

As shown in the two results presented in Fig. 2 (final results of the two hybrid classifications), the two segmentations $S_1$ et $S_2$ are visually very similar. Moreover they both have 5 classes that are very similar (as shown by the confusion matrix between the two results, presented in table 1) and no result differs too much compared to the others ($0.74 \leq \Gamma^i \leq 0.82$). We can suppose that these 5 classes are relevant, because they were found by two different learnings, initialized differently. This proves that the importance of the initial choices is decreased because of the collaboration between different classification methods.

Table 1: Confusion matrix between the two results of $S_1$ et $S_2$

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_0$</td>
<td>14283</td>
<td></td>
</tr>
<tr>
<td>$C_1$</td>
<td>5227</td>
<td></td>
</tr>
<tr>
<td>$C_2$</td>
<td>937</td>
<td>8881</td>
</tr>
<tr>
<td>$C_3$</td>
<td>197</td>
<td>5452</td>
</tr>
<tr>
<td>$C_4$</td>
<td>741</td>
<td>3075</td>
</tr>
</tbody>
</table>

As shown in the two results presented in Fig. 2 (final results of the two hybrid classifications), the two segmentations $S_1$ et $S_2$ are visually very similar. Moreover they both have 5 classes that are very similar (as shown by the confusion matrix between the two results, presented in table 1) and no result differs too much compared to the others ($0.74 \leq \Gamma^i \leq 0.82$). We can suppose that these 5 classes are relevant, because they were found by two different learnings, initialized differently. This proves that the importance of the initial choices is decreased because of the collaboration between different classification methods.

Table 1: Confusion matrix between the two results of $S_1$ et $S_2$

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_0$</td>
<td>14283</td>
<td></td>
</tr>
<tr>
<td>$C_1$</td>
<td>5227</td>
<td></td>
</tr>
<tr>
<td>$C_2$</td>
<td>937</td>
<td>8881</td>
</tr>
<tr>
<td>$C_3$</td>
<td>197</td>
<td>5452</td>
</tr>
<tr>
<td>$C_4$</td>
<td>741</td>
<td>3075</td>
</tr>
</tbody>
</table>

As shown in the two results presented in Fig. 2 (final results of the two hybrid classifications), the two segmentations $S_1$ et $S_2$ are visually very similar. Moreover they both have 5 classes that are very similar (as shown by the confusion matrix between the two results, presented in table 1) and no result differs too much compared to the others ($0.74 \leq \Gamma^i \leq 0.82$). We can suppose that these 5 classes are relevant, because they were found by two different learnings, initialized differently. This proves that the importance of the initial choices is decreased because of the collaboration between different classification methods.

Table 1: Confusion matrix between the two results of $S_1$ et $S_2$

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_0$</td>
<td>14283</td>
<td></td>
</tr>
<tr>
<td>$C_1$</td>
<td>5227</td>
<td></td>
</tr>
<tr>
<td>$C_2$</td>
<td>937</td>
<td>8881</td>
</tr>
<tr>
<td>$C_3$</td>
<td>197</td>
<td>5452</td>
</tr>
<tr>
<td>$C_4$</td>
<td>741</td>
<td>3075</td>
</tr>
</tbody>
</table>

We can notice that eventhough the two segmentations did not evolve in the same way, they both found very similar results. Fig. 3 shows the evolution graphs of the percentage of pixels classified identically by all the method occurrences at each step of each hybrid learning. The segmentation $S_2$ (with 5 agents) started with a less relevant result than $S_1$ (27% of pixels classified identically against 57%) but it converged faster (12 refinement steps against 14).
3.2 Per-pixel classification of remote sensing images

In order to make our method usable for remote sensing images analysis, we implemented the MuStIC system developed in the GeoDM project\textsuperscript{2}. It was initially intended for the geographers and ecological experts to classify pixels from remote sensing images. It integrates some unsupervised classification tools for images using K-means \cite{14}, Cobweb \cite{8} or S.O.M algorithms \cite{18}, and the Samarah module which implements the collaborative method using all of the algorithms below.

We have applied this system to remote sensing data from the city of Strasbourg (France): Spot 4 data with three channels (XS1, XS2, XS3) at standard resolution (200×250 pixels - 20 meters/pixel) (Fig. 4).

We carried out two series of tests:

1. because the inflexion of the curve of empirical average, we have configured our system (10 experimentations) to find about 6 classes.

2. according to the geographers from the LI\textsuperscript{3}, we have configured (15 experimentations) our system to find about 10 classes. (for example, 3 Kmeans with 9, 10 and 11 seeds or 2 Kmeans and a Cobweb)

First we present a test with 6 classes expected. The unsupervised classification methods used are:

- $\mathcal{M}_1$: K-means with 4 initial random nodes;
- $\mathcal{M}_2$: K-means with 8 initial random nodes;
- $\mathcal{M}_3$: SOM with a 4×4 map;
- $\mathcal{M}_4$: Cobweb with an acuity of 7.5.

We have obtained the results\footnote{In all the results, the colors have been aleatorily chosen by the authors} below:

- $\mathcal{R}_1$: 4 classes;
- $\mathcal{R}_2$: 8 classes;
- $\mathcal{R}_3$: 14 classes;
- $\mathcal{R}_4$: 27 classes.

These results were then refined according to our refining algorithm presented in [26]. We then obtained these results (Fig. 5):

- $\mathcal{R}_1$: 5 classes;
- $\mathcal{R}_2$: 6 classes;
- $\mathcal{R}_3$: 5 classes;
- $\mathcal{R}_4$: 4 classes.

We applied to these results our multi-view voting algorithm and obtained the unifying result presented on figure 6(a). This result is composed of 5 different classes.

Finally we present on figure 6(b) the voting result for all the objects:

- in white: all the methods agreed on the classification;
- in gray: one method disagreed with the other ones;
- in black: the non consensual objects (two results or more classified these objects differently).

Secondly, we present a test with 10 classes expected. The unsupervised classification methods used are:

- $\mathcal{M}_1$: K-means with 8 initial random nodes;
- $\mathcal{M}_2$: K-means with 10 initial random nodes;
- $\mathcal{M}_3$: K-means with 12 initial random nodes.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.png}
\caption{Evolution graph of the pixels classified identically by all the method occurrences during each segmentation}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure4.png}
\caption{Spot Image of Strasbourg}
\end{figure}
is really a hard problem. Today, no real evaluation process has been proposed [12, 13]. There exists many different statistical measures that can be used to have an idea of the quality of our results. The most frequent cluster validation index proposed in the literature is the inter-classes inertia (and cluster compactness [15] and the Xie-Beni index [28]). In order to evaluate the quality of our results, we first use Kmeans to evaluate the intra-classes inertia according to the number of classes. We calculated an empirical average for inertia for each number of classes (see Fig. 8): we carried out the algorithm 200 times for each number of classes with a random initialization of the centers.

In this test, all the computations needed about 20 steps to propose their final number of classes (see Fig. 9). Compared to all the executions, the inertia of our result is better than the empirical average for a same number of classes (see Fig. 10).

4. CONCLUSIONS

We presented a new process of collaborative multi-strategy classification of complex data which enables us to carry out an unsupervised multi-strategy classification on complex
Table 2: Relation between number of classes found and inertia

<table>
<thead>
<tr>
<th>#classes</th>
<th>Inertia</th>
<th>Empirical average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First test (10 classes expected)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1465, 1377, 1408, 1384, 1373, 1382, 1375, 1401</td>
<td>1502</td>
</tr>
<tr>
<td>11</td>
<td>1292, 1293, 1244, 1290, 1270, 1275</td>
<td>1373</td>
</tr>
<tr>
<td>12</td>
<td>1156</td>
<td>1282</td>
</tr>
<tr>
<td></td>
<td>Second test (6 classes expected)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>2302, 2305, 2338, 2139, 2401, 2327, 2331, 2304, 2329</td>
<td>2401</td>
</tr>
<tr>
<td>7</td>
<td>2075</td>
<td>2080</td>
</tr>
</tbody>
</table>

This enables them to converge towards a single result (without necessarily reaching it), and to obtain very similar classes. Doing this, it is possible to put in correspondence the classes found by the various methods and finally to apply an unification algorithm like a voting method for example. This way, we can give to the user a single result representing all the results found by the various methods of unsupervised classification.

Within the framework of the research on this approach, we were brought to study the theoretical bases of the integration of classification methods and the unification of classification results.

On one hand, we proposed the concepts allowing the combination of unsupervised classification methods extending the results on the combination of supervised methods.

And on the other hand, we presented a new theoretical approach to distributed multi-strategy classification, not based on the fusion of methods but on the collaboration, inspired by the multi-agent paradigm.

This approach is not specialized in a particular domain and allows to integrate directly any unsupervised classification method, without modification. The definition of this collaboration gave place to a theoretical study and the definition of an objective method of conflict resolution, a new concept that we introduced to represent the disagreements between the methods used.

To quantify these disagreements, we introduced a new criterion, called similitude, which can be used as well with numerical data as with symbolic attributes, since it is based on the recovery of classes and not on a distance. This means our approach can be applied to symbolic data (obviously, only if the integrated classification methods used in this case are able to treat such data).

With this criterion, we proposed a new definition of the concept of relevant classes based on the similitude, we gave a method to determine these classes within the framework of unsupervised hybrid classification.

Finally, we proposed a theoretical solution to the problem of automatic unification of unsupervised classification by extension of the traditional voting methods.

Further research focuses first on the possibility of the collaboration of various classification methods, where each one uses a different model from the data. In the case of remote sensing image for example, we could use various data sources on the same zone (radar, radiometry, photo...).
Secondly, we are studying the possibility of using this collaborative approach in a multi-step process. We have applied a collaborative multi-step multi-strategy classification method on the problem of thematical urban zone extraction from remote sensing images. The first results are encouraging: we highlighted an improvement of the results at each step of the process.

Lastly, we are interested by the integration of domain knowledge (ontology, training, ...) in our system to improve each step of the collaborative mining.

5. REFERENCES


A Framework to Support Multiple Query Optimization for Complex Mining Tasks

Ruoming Jin  
Department of Computer Science and Engineering  
Ohio State University, Columbus OH 43210  
jnr@cse.ohio-state.edu

Kaushik Sinha  
Department of Computer Science and Engineering  
Ohio State University, Columbus OH 43210  
sinhak@cse.ohio-state.edu

Gagan Agrawal  
Department of Computer Science and Engineering  
Ohio State University, Columbus OH 43210  
agrawal@cse.ohio-state.edu

ABSTRACT

With an increasing use of data mining tools and techniques, we envision that a Knowledge Discovery and Data Mining System (KDDMS) will have to support and optimize for the following scenarios: 1) Sequence of Queries: A user may analyze one or more datasets by issuing a sequence of related complex mining queries, and 2) Multiple Simultaneous Queries: Several users may be analyzing a set of datasets concurrently, and may issue related complex queries.

This paper presents a systematic mechanism to optimize for the above cases, targeting the class of mining queries involving frequent pattern mining on one or multiple datasets. We present a system architecture and propose new algorithms for this purpose. We show the design of a knowledgeable cache which can store the past query results from queries on multiple datasets. We present algorithms which enable the use of the results stored in such a cache to further optimize multiple queries.

We have implemented and evaluated our system with both real and synthetic datasets. Our experimental results show that our techniques can achieve a speedup of up to a factor of 9, compared with the systems which do not support caching or optimize for multiple queries.

1. INTRODUCTION

The iterative and exploratory nature of knowledge discovery or data mining has increasingly become a bottleneck in the KDD process, especially in view of the need for interactive response to the users. One of the relatively recent developments has been the focus on constraint mining. Constraint mining involves queries with constraint conditions, which help reduce the execution time of mining algorithms, and also help guide the KDD process. Many researchers have focused on designing languages or models to express these complex mining queries and developing new algorithms to evaluate each single complex query efficiently [19, 16].

Despite such developments, evaluation of mining queries is still a relatively slow process, and users often cannot perform data mining in an interactive manner. As the amount of data available for analysis in both scientific and commercial domains is increasing dramatically, efficiency in the data mining process is likely to become the crucial issue. With an increasing use of data mining tools and techniques, we envision that a Knowledge Discovery and Data Mining System (KDDMS) will have to support and optimize for the following scenarios:

- **Sequence of Queries**: A user may analyze one or more datasets by issuing a sequence of related complex mining queries. This may be due to the iterative and exploratory nature of the process, where the mining parameters and constraints are modified till desired insights are gained from the dataset(s).
- **Multiple Simultaneous Queries**: Several users may be analyzing a set of datasets concurrently, and may issue related complex queries.

The need for supporting and optimizing such scenarios has been well recognized in database and OLAP systems. Views have been used to optimize a sequence of database operations [12], and similarly, techniques such as reducing common subexpressions [25, 24] have been used. However, because the nature of the mining operations is very different from that of database and OLAP operations, these techniques cannot apply to a KDDMS system.

Some efforts have been made towards addressing these issues for mining environments. Nag et al. have studied how a knowledgeable cache can be used to help perform interactive discovery of association rules [18]. They maintain a cache to record (in)frequent itemsets with their support levels, and then modify the frequent itemset mining algorithm to utilize the itemsets in the cache. The focus of their research is on frequent itemset mining without complex mining conditions. Ng et al. have studied constraint association rule mining [19]. In their method, multiple queries can be merged as a single query for evaluation. Hipp and Gunther have argued that execution of data mining queries with constraints can be very expensive [13]. Therefore, they have proposed to use precomputation of frequent itemsets of certain support levels to answer constraint itemset mining queries.

The above efforts have two important limitations. First, sequence of queries and multiple simultaneous queries have not been studied together. Second, the techniques involving the use of knowledgeable cache have been restricted to deal with simple data mining queries.

In this paper, we focus on the problem of efficiently evaluating an important class of complex mining queries in a query intensive environment, where one needs to optimize multiple simultaneous queries, as well as a sequence of related queries. The class of complex mining queries we target are the ones involving frequent pattern mining on one or multiple datasets. Particularly, we show how multiple simultaneous queries can be optimized, and how the results from past mining queries can be utilized to evaluate the current
ones. Due to the complexity and characteristics of such queries, simultaneous optimization of multiple queries and caching of their query results is challenging, and quite different from the existing work in this area.

Overall, this paper makes the following contributions:

1. We present a novel system architecture to deal with a query intensive environment that needs to support and optimize both multiple simultaneous queries and a sequence of queries.
2. We propose new algorithms to perform multiple-query optimization for frequent pattern mining on multiple datasets.
3. We show the design of a knowledgeable cache which can store the past query results from queries on multiple datasets. We present algorithms which enable the use of the results stored in such a cache to further optimize multiple queries.
4. We have implemented and evaluated our system with both real and synthetic datasets. Our experimental results show that our techniques can achieve a speedup of up to a factor of 9, compared with the systems which do not support caching or optimize for multiple queries.

The rest of the paper is organized as follows. In Section 2, we will briefly introduce our target class of queries, which involve frequent pattern mining on multiple datasets. We describe three different representations of these queries, which are using SQL, an Algebra, and a new format, which we call the M-table. In Section 3, we present our framework to deal with both multiple simultaneous queries and a sequence of queries. In Section 4, we discuss the important properties of the M-table, which form the basis for our multiple query optimizations and caching of query results. In Section 5, we present our optimization algorithms. In Section 6, we discuss the major implementation issues for our system, and present our experimental results. We compare our work with related research efforts in Section 7, and conclude in Section 8.

2. FREQUENT PATTERN MINING ON MULTIPLE DATASETS: SQL EXTENSIONS, ALGEBRA, AND M-TABLE

Frequent pattern mining focuses on discovering frequently appearing sub-structures in datasets. The structures explored include itemsets, sequences, sub-trees, sub-graphs, and other topological structures [29, 3, 33, 4]. Frequent pattern mining has emerged as a very useful class of techniques for analyzing datasets from a variety of domains, including retail transactions, DNA sequences, chemical compounds, XML documents, among others.

An important class of frequent pattern mining tasks involve discovering interesting patterns from multiple distinct datasets. For example, a manager of a nation-wide store will like to know what itemsets are frequent in stores in New York and New Jersey, but very infrequent in stores in California. Similarly, biology researchers are interested in sequences that are frequent in human gene but infrequent in chicken gene, and/or, the sequences are frequent in both the species.

In this section, we briefly describe the major issues in expressing as well as evaluating frequent pattern mining tasks on multiple datasets. Also, in order to simplify our discussion, we will focus on frequent itemset mining tasks.

2.1 SQL and M-Table for Mining Multiple Datasets

Assume we have four transaction datasets $A, B, C, D$, and Item is the set of all the items appearing in the four datasets. To extract the interesting patterns from these datasets, a Frequency table $F$ is defined with a schema $\text{Frequency}(I, A, B, C, D)$. The column with attribute $FI$ stores all possible itemsets, i.e., the power-set of

<table>
<thead>
<tr>
<th>$I$</th>
<th>$A$</th>
<th>$B$</th>
<th>$C$</th>
<th>$D$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.05</td>
<td>0.2</td>
</tr>
<tr>
<td>2</td>
<td>0.3</td>
<td>0.8</td>
<td>0.4</td>
<td>0.1</td>
</tr>
<tr>
<td>3</td>
<td>0.5</td>
<td>0.01</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>1,2</td>
<td>0.08</td>
<td>0</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>1,2</td>
<td>0.1</td>
<td>0.1</td>
<td>0</td>
<td>0.1</td>
</tr>
</tbody>
</table>

| Table 1: $F$ Table for Datasets $A, B, C$ and $D$ |

Item. The columns with attribute $F.A, F.B, F.C, F.D$ store the frequency of the itemsets in the four datasets $A, B, C, D$, respectively. Table 1 contains a portion of this $F$ table.

\[
\text{SELECT } F.I, F.A, F.B, F.C, F.D \\
\text{FROM } \text{Frequency}(I, A, B, C, D) \quad F \\
\text{WHERE } (F.A \geq 0.1 \text{ AND } F.B \geq 0.1) \\
\text{OR } (F.C \geq 0.1 \text{ AND } F.D \geq 0.1 \text{ AND } (F.A \geq 0.2 \text{ OR } F.B \geq 0.2))
\]

(a) SQL query for query $Q_1$

\[
\begin{align*}
&F_1 \cup (SF(A, 0.1) \cap SF(B, 0.1)) \Rightarrow F_1 \\
&F_2 \cup (SF(A, 0.2) \cap SF(C, 0.1) \cap SF(D, 0.1)) \Rightarrow F_2 \\
&F_3 \cup (SF(B, 0.2) \cap SF(C, 0.1) \cap SF(D, 0.1)) \Rightarrow F_3
\end{align*}
\]

(c) Necessary Information for the query $Q_1$

Figure 1: Query $Q_1$

Note that the $F$ table only serves as a virtual table or a logical view. A frequent pattern mining task on multiple datasets is expressed as a SQL query to partially materialize this table. The query $Q_1$ in Figure 1(a) is an example. Here, we want to find the itemsets that are either frequent with support level 0.1 in both $A$ and $B$, or frequent (with support level 0.2) in both $C$ and $D$, and also frequent in either $A$ or $B$ (with support level 0.2).

Consider any query whose constraint condition (the WHERE clause) does not contain any negative condition, i.e., a condition which states that support in a certain dataset is below a specified threshold. Clearly, the constraint condition of such a query can be expressed in a tabular format, where 1) each row of the table represents a dataset, 2) each column corresponds to a conjunctive-clause (involving only the AND operation) in the disjunctive normal form (DNF) of the constraint condition, and 3) a cell at i-th row and j-th column will have the value $\alpha$ if the j-th conjunctive-clause requires that the support in the i-th dataset is at least $\alpha$. The table thus computed is referred to as an $M$-table. For example, Figure 1(b) illustrates the $M$-table for the query $Q_1$.

An $M$-table provides a systematic way to describe the information required to answer a query involving multiple datasets. It turns out that $M$-table can be used to 1) generate efficient query plans for a given query, 2) detect common computations across multiple queries, and 3) summarize the results obtained from multiple queries in a cache. Thus, our presentation in the rest of this paper will be based on $M$-table representation of the queries.

In the rest of this section, we focus on two issues. In the next subsection, we show how we can formally express the conditions captured by an $M$-table in terms of a basic mining operator, which
is the frequent itemset operator on a single dataset. Finally, in Subsection 2.3, we show that an M-table can also be used to represent a class of queries involving negative conditions.

2.2 Algebra for Evaluating Frequent Mining Tasks

In this subsection, we introduce an algebra to express the information required to answer a mining query over multiple datasets. This algebra contains only one mining operator \( SF \) and two operations, intersection (\( \cap \)) and union (\( \cup \)). Formally, they are as follows:

The frequent itemset mining operator \( SF(A_j, \alpha) \) returns a two-column table, where the first column contains itemsets in \( A_j \) which have the support level \( \alpha \), and the second column contains their corresponding frequency in the dataset \( A_j \).

Intersection \( (F_1 \cap F_2) \): Let \( F_1 \) and \( F_2 \) be two tables whose first column contains a collection of itemsets, and other columns contain the corresponding frequency (possibly empty) in different datasets. The intersection operation \( (F_1 \cap F_2) \) returns a table whose first column contains the itemsets appearing in the first columns of both \( F_1 \) and \( F_2 \), and other columns contain frequency information for these itemsets in the datasets appearing in \( F_1 \) and \( F_2 \).

Union \( (F_1 \cup F_2) \): The union operation \( (F_1 \cup F_2) \) returns a table whose first column contains the itemsets appearing in the first columns of either \( F_1 \) or \( F_2 \), and other columns contain the frequency of these itemsets in the datasets appearing in \( F_1 \) or \( F_2 \).

Given an \( M \)-Table, the information required for answering the corresponding query can be described as follows. Each nonempty cell, \( M_{i,j} \), maps to a \( SF \) operator, \( SF(A_i, M_{i,j}) \). The \( SF \) operators in the same column are connected by the intersection (\( \cap \)) operation, and the expressions corresponding to each column are connected by the union (\( \cup \)) operation. The resulting expression is referred to as the necessary information for the query. For example, the necessary information of \( Q_1 \) is shown in Figure 1(c).

2.3 Admissible Conditions and M-Table

In this subsection, we consider a broader class of queries, which could involve negative predicates as well. We establish that under certain restrictions, they can be represented through an \( M \)-table as well.

For a given query, we transform the constraints into the disjunctive normal form (DNF),

\[
C \equiv C_1 \lor C_2 \lor \cdots \lor C_k
\]

where, \( C_i \) is a conjunctive-clause, i.e., it involves AND operation on one or more predicates. A query is considered admissible if each conjunctive-clause in the DNF format contains at least one positive predicate, i.e., \( F.A_1 \geq \alpha \). For example, a query involving the following condition is not admissible:

\[
F.A_1 < 0.1 \text{ OR } (F.A_2 \geq 0.2 \text{ AND } F.A_3 < 0.05)
\]

This is because the first conjunctive-clause, \( F.A_1 < 0.1 \), contains only a negative predicate.

Through a set of transformations described in a related publication [14], we are able to achieve the following:

**Lemma 1.** The query constraints of an admissible query can be expressed as an \( M \)-table.

Note that in such cases, the necessary information corresponding to the \( M \)-table may represent a superset of the results of the query. In such cases, a selection operation can be used to obtain the results of the query. As an example, Figure 2(a) shows the query \( Q_2 \). The \( M \)-table representing its query constraints is illustrated in Figure 2(b), and the corresponding necessary information is in Figure 2(c). We can see that the first conjunctive-clause \( F.A \geq 0.1 \text{ AND } F.B \geq 0.1 \text{ AND } F.D < 0.05 \) has been mapped to the first two columns of the \( M \)-table. Particularly, in the necessary information format (\( P_2 \)), we use \( SF(A, 0.1) \cap SF(B, 0.1) \) to intersect with \( SF(D, 0.05) \). Note that such repetition of \( SF(A, 0.1) \cap SF(B, 0.01) \) is for minimizing the necessary information.

3. SYSTEM ARCHITECTURE AND OPTIMIZATION OVERVIEW

Let us envision a KDDMS system in which there are multiple datasets and multiple users. If different users issue queries each of which involves multiple datasets, it is quite likely that the queries could have a significant overlap.

For example, consider the following two queries, \( Q_1 \) and \( Q_2 \), which are issued simultaneously.

\[
Q_1: \text{SELECT } F.I, F.A, F.B, F.X \text{ FROM Frequency } (I, A, B, X) \text{ F WHERE } F.A \geq 0.2 \text{ AND } F.B \geq 0.1 \text{ AND } F.X < 0.1
\]

\[
Q_2: \text{SELECT } F.I, F.A, F.B, F.Y, F.Z \text{ FROM Frequency } (I, A, B, Y, Z) \text{ F WHERE } (F.A \geq 0.1 \text{ AND } F.B \geq 0.1 \text{ AND } F.Y \geq 0.1 \text{ AND } F.Z < 0.01) \text{ OR } (F.Z > 0.2 \text{ AND } F.Y < 0.01)
\]

These two queries overlaps on the datasets \( A \) and \( B \). The question for us is, “How can we exploit the overlap in the two queries to generate query plans that are more efficient than the independently generated query plans for each query?”

Furthermore, we consider the following possibility. As we had described earlier, it is very likely that a single user issues a sequence of related queries. For example, the system might have evaluated the query \( Q \) (described in the previous section), before it receives the queries \( Q_1 \) and \( Q_2 \). In such a case, we have the following two additional questions: “How can we effectively store the results from the recent queries in a cache?”, and, “How can we efficiently utilize such cached results to speedup computation of new queries?”

Before discussing how we address these issues, we describe our system architecture. This architecture is shown in Figure 3. Our system primarily contains four components, a Query queue, a Query plan optimizer, a Query evaluation engine, and a Cache. The queries
However, our method for detecting and optimizing the common proach. Our method is based on ferent queries. This is similar to what is done for database queries. 1. Simultaneous optimization of multiple queries provides a more detailed account. The query plan is mapped to an global query plan, which can be generated for the large -tables. We show how we can use -tables to detect the overlapping computations. Further, different -tables can be obtained by the answer set for the query corresponding to -table, which is based on the cells of -tables. These relationships provide a simple mechanism to define common computations among different queries.

- Definition 1. If -table is contained in -table for each corresponding pair of cells, -table, 1 ≤ i ≤ n, either both the cells are empty, or both the cells are non-empty and -table ≥ -table.

- Definition 2. If -table and -table are multi-column -tables, -table is contained in -table if each column in -table is contained by some column in -table.

This lemma helps us detect the common computations among queries. Next, we study a more generalized containment relationship among -tables, which is based on the cells of -tables. The motivation for this is as follows. In many cases, the results of a query cannot be completely answered by one or more of the past queries, but part of its result can be derived from them. This containment helps answer these questions. To facilitate our discussion, we first define the following inequalities for empty cells. Let e be the empty cell and let r be a positive (non-zero) threshold. Then, our discussion assumes the following inequalities, e ≥ e, r ≥ e, 0 ≥ e, and e ≥ 0.

For the following definition, we again assume that we have two -tables, -table and -table, with the same number of rows (n), and the same row in the two tables corresponds to the same dataset.
DEFINITION 3. Consider a cell \( c \), which is at the row \( i \) in the column \( C_1 \) of the \( M \)-table \( M_1 \). This cell is contained in \( M_2 \) if there exists a column \( C_2 \) in \( M_2 \), denoted as \( C_2[j] \), such that 1) \( C_1[i] \geq C_2[j] \) for all \( j \) and 2) \( C_1[i] \geq C_2[j] \) for all \( i \), where \( j \) is at least \( 1 \) and \( j \) is at most \( n \).

We denote such containment as \( c \subseteq M_2 \). Intuitively, \( c \) is contained in \( M_2 \) if we can use the corresponding cell in the column \( C_2 \) to color the cell \( c \). The reason we require \( C_1[i] \geq C_2[j] \) for each pair of corresponding cells in the two columns, is that we need information in \( C_2 \) to be a superset of the information required for the cell \( c \).

As an example, in Table 5, the cell at the row three in the single-column \( M \)-table corresponding to \( O_4 \), denoted as \( O_4[9] \), is contained in the \( M \)-table for \( O_2 \). Formally, we say, \( O_4[9] \subseteq O_2 \).

Based upon the above definition, we have the following definition to relate one \( M \)-table to a set of \( M \)-tables.

DEFINITION 4. An \( M \)-table, \( M' \), is cell-contained in the group of \( M \)-tables, \( M_1, \ldots, M_k \), if each non-empty cell in \( M' \) is contained by at least one \( M \)-table in the set \( M_1, \ldots, M_k \).

Formally, we denote this as

\[
M' \subseteq c \{ M_1, \ldots, M_k \}
\]

As an example, in Table 2, we have \( M_5 \subseteq c \{ M_3, M_4 \} \).

Given this definition, we have the following lemma to detect if the necessary information of a query can be derived from a group of other queries.

LEMMA 3. Let \( Q' \) be a query with an \( M \)-table, \( M' \), and let \( Q_1, \ldots, Q_k \) be a group of queries with the corresponding \( M \)-tables \( M_1, \ldots, M_k \), respectively. If \( M' \) is cell-contained in \( M_1, \ldots, M_k \), then the necessary information of \( Q' \) can be derived from the necessary information of \( Q_1, \ldots, Q_k \).

Our discussion in this subsection has so far assumed that the \( M \)-tables have the same number of rows, and the same row in each table corresponds to the same dataset. However, this is not a serious limitation. If two \( M \)-tables do not satisfy this condition, we can align them to meet this condition. Briefly, this alignment procedure is as follows. First, we take a union of the two sets of datasets. Then, we extend the two \( M \)-tables to have the same number of rows, corresponding to the union of the set of datasets. This will involve adding rows where each cell will be empty. Finally, we shuffle the rows in the two \( M \)-tables to let each row represent the same dataset.

### 4.2 The Merge Operation for \( M \)-Tables

We now define the merge operation for the \( M \)-Tables. This operation helps in replacing multiple queries by a single large query, and also helps maintain a high-level summary of the contents of the cache. Again, our definition assumes that the \( M \)-tables being merged have been aligned, i.e., they have the same number of rows and the same row in each table corresponds to the same dataset.

DEFINITION 5. The merge operation, denoted as \( \oplus \), on two \( M \)-tables, \( M_1 \) and \( M_2 \), results in a table with the same rows, and a set of columns that is the union of the set of columns in \( M_1 \) and \( M_2 \).

As an example, Table 3 shows the merged table, \( M_1 \oplus M_2 \), where, \( M_1 \) and \( M_2 \) are \( M \)-tables for the queries \( Q_1 \) and \( Q_2 \), respectively.

Clearly, the original tables are contained in the merged table, that is

\[
M_1, M_2 \subseteq M_1 \oplus M_2
\]

The implication of the above observation is as follows. For two \( M \)-tables \( M_1 \) and \( M_2 \), corresponding to the queries, \( Q_1 \) and \( Q_2 \), respectively, the answering set of both \( Q_1 \) and \( Q_2 \) can be derived from the result of the merged \( M \)-table, \( M_1 \oplus M_2 \). This fact will be used to process multiple queries, as well as to update the knowl-edgeable cache with different mining operators.

### 5. MULTIPLE QUERY OPTIMIZATION APPROACH

In this section, we present our optimization algorithms which are based on \( M \)-tables. Specifically, in Subsection 5.1, we first review how the query plan for a single query is generated from an \( M \)-table. In Subsection 5.2, we study how each mining operator can be mapped to the \( M \)-table and how the redundant mining operators can be detected. In Subsection 5.3, we discuss how local plans from several queries can be optimized together. In Subsection 5.4, we introduce another approach for optimizing multiple queries, which involves merging multiple queries into one query, and then generating a global query plan. Subsection 5.5 focuses on how \( M \)-table can be used to summarize and update the cache, and how the cache can help us reduce the evaluation costs.

#### 5.1 Single Query Plan Generation

We begin with introducing a new mining operator \( CF \). We introduce this operator because using only the \( SF \) operator to evaluate queries can be very expensive.

**Frequent itemset mining operator with constraints \( CF(A_j, \alpha, X) \)** finds the itemsets that are frequent in the dataset \( A_j \) with support \( \alpha \) and also appears in the set \( X \). \( X \) is a set of itemsets that satisfies the down-closure property, i.e., if an itemset is frequent, then all its subsets are also frequent. This operator also reports the frequency of these itemsets in \( A_j \). Formally, \( CF(A_j, \alpha, X) \) computes the following view of the \( F \) table:

\[
X \cap SF(A_j, \alpha)
\]

Note that we can also define and use other mining operators to speedup the evaluation process [14]. For simplicity, we will only use \( CF \) and \( SF \) in this paper. Our overall approach can be easily extended to include other mining operators as well.

Now, we focus on query plan generation using the \( M \)-table. One of the important features of \( M \)-table is it can capture the evaluation process for a query by a simple coloring scheme. This coloring scheme is as follows. Initially, all the cells are black. Each invocation of a mining operator (like \( SF \) and \( CF \)) can color a number of non-empty cells red. This implies that the information corresponding to these cells has been computed. The query evaluation process is complete when all non-empty cells are colored red.

As a running example, consider applying \( SF(A, 0.05) \), \( SF(C, 0.1) \), \( CF(B, 0.1, SF(A, 0.1)) \), and \( CF(D, 0.1, SF(C, 0.1)) \) consecutively on an initially black-colored table \( M \) of the query \( Q \). Table 4 shows the resulting colored table (unshaded for black-colored, shaded for red-colored).
and shaded for red-colored). In the following, we look at how the $SF$ and $CF$ operators color the table.

Frequent mining operator $SF(A_i, \alpha)$: An invocation of the frequent mining operator on the dataset $A_i$, with support $\alpha$, will turn each non-empty cell at row $i$ who is greater than or equal to $\alpha$ red. In our example, the first operator, $SF(A, 0.05)$, will turn the cells $M_{1,1}$, $M_{1,2}$, and $M_{1,4}$ red, and the second operator, $SF(C, 0.1)$, will turn the cells $M_{3,3}$, $M_{3,4}$, and $M_{5,5}$ red.

Frequent mining operator with constraint $CF(A, \alpha, X)$: The coloring impacted by this operator is dependent on the current coloring of the table $M$. Let $X$ be the set of frequent itemsets defined by all the red cells, and let $S$ be the set of columns where these red cells appear. Then, by applying this operator on dataset $A_i$ with support $\alpha$, all cells on row $i$ whose column is in the set $S$, and whose value is less than or equal to $\alpha$, will turn red.

In our running example, the third operator $CF(B, 0.1, SF^2(A, 0.1))$ picks the red cells $M_{1,1}$ and $M_{1,2}$ by the parameter

$$X = SF^2(A, 0.1)$$

The set $S$ includes the first two columns. Therefore, this operator turns the cells $M_{2,1}$ and $M_{2,2}$ red. Similarly, the fourth operator turns the cells $M_{4,3}$, $M_{4,4}$, and $M_{4,5}$ red.

By the above formulation, the query evaluation problem has been converted into the problem of coloring the table $M$. Different operators can be used, and in different order, to color the entire table red. Generating optimal query plan is $NP$-hard, and a number of heuristic algorithms have been developed to find efficient query plans [14]. Here, we will only discuss one of the algorithms, the Algorithm-CF, which uses $SF$ and $CF$ operators to optimize the query evaluation. Algorithm-CF splits the evaluation into two phases. In the first phase, we use the $SF(A_i, \alpha)$ operators so that each column has at least one red cell. In the second phase, we use the $CF(A_j, \alpha, X)$ operators to compute all other non-empty cells in the table.

The sketch of Algorithm-CF is listed in Figure 4. It involves minimizing costs for each of the two phases. Since precise cost functions for each operator are not available, a simple heuristic based on the support level is used to estimate the cost. In general, for a single dataset, higher support level for the $SF$ operator implies lower computation.

<table>
<thead>
<tr>
<th>$F_1$</th>
<th>$F_2$</th>
<th>$F_3$</th>
<th>$F_4$</th>
<th>$F_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.1</td>
<td>0.1</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.1</td>
<td></td>
<td>0.05</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td>0.05</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 4: Colored M Table for the query $Q$

<table>
<thead>
<tr>
<th>$O_1$</th>
<th>$O_2$</th>
<th>$O_3$</th>
<th>$O_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.1</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>C</td>
<td>0.1</td>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>0.1</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: M-Tables of different mining operators

5.2 Mapping Mining Operators to $M$-Tables

Each mining operator in a query plan can be uniquely mapped to an $M$-table. This mapping plays an important role in multiple query optimization and cache management. This is because common computations among the mining operators can be easily captured using $M$-table, and similarly, the result of each mining operator can be uniformly expressed using $M$-tables.

We had earlier described how the two operators, $SF$ and $CF$, contribute to the coloring of the table, and help generate query plans. Since part of our goal is to use an $M$-table to capture the cache, we define rules to map each different mining operator in a query plan to a unique $M$-table.

Frequent mining operator $SF(A_j, \alpha)$: An invocation of this operator on dataset $A_j$ and support $\alpha$ will generate a single column $M$-table whose row $j$ is $\alpha$, and other rows are empty.

Frequent mining operator with constraint $CF(A_j, \alpha, X)$: Recall that the $CF$ mining operator is used to color a set of columns, denoted as $S$, who have at least one cell to be colored red, and the cell at the row $j$ for each column in $S$ is black. Then, the $M$-table generated by the $CF$ operator is composed of these columns in the set $S$, with the following exception. The cells which are still black after the $CF$ mining operator will become empty in this new $M$-table.

Consider the following incomplete query plan for the query $Q$.

$$O_1 : SF(A, 0.1);$$
$$O_2 : SF(C, 0.1);$$
$$O_3 : CF(B, 0.05, (SF(A, 0.1) ∪ SF(C, 0.1))^2);$$
$$O_4 : CF(D, 0.05, ((SF(A, 0.1) ∩ SF(B, 0.01)) ∪ SF(C, 0.1))^2);$$

Table 5 shows the corresponding $M$-tables for the mining operators in the above query plan.

The significance of associating an $M$-table with each mining operator is that the common computation among mining operators can be treated the same way as the query results. In particular, Lemmas 2 3 can be modified to apply to mining operators, instead of mining queries. In next subsection, we will use such methods to reduce the redundant computations among different query plans.

5.3 Optimizing Local Plans

To optimize multiple simultaneous queries, this approach generates local query plans for each query, and then tries to remove the common computations among the query plans. The common computations are categorized into two groups. In the first group, a mining operator in a query plan can be derived from another mining operator in one of the other query plans. In the second group,
a mining operator in a query plan can be derived from a group of mining operators which are in other query plans, or are in the same query plan but scheduled before this operator. As discussed in Subsection 4.1, we can detect these common computations by the containment relationship defined on the $M$-tables.

The difficulty of this approach is that different query evaluation order will result in different ways to remove the common computations. For example, assume one query plan has the mining operator,

$$CF(X, 0.1, (SF(A, 0.1) \cup SF(B, 0.2))^I)$$

and another query plan includes

$$CF(A, 0.1, SF(X, 0.1)^I), CF(B, 0.2, SF(X, 0.1)^I)$$

Since the two sets of mining operators are equivalent, depending on which query plan is evaluated first, we have different ways to eliminate the common computations. Note that in order to simplify the above problem, we are not considering combining local plans together into a global plan. This will be the topic of the next subsection.

**Input:** local query plans $Q_1, \cdots, Q_n$

$S = \{Q_1, \cdots, Q_n\}$

**While** ($S \neq \emptyset$) **Do**

**Foreach** $Q_i \in S$

**Eliminate Containment:**

If any mining operator in $Q_i$ is contained in $S - \{Q_i\}$

**Eliminate Cell-Containment:**

If any mining operator in $Q_i$ is cell-contained in mining operators in $S - \{Q_i\}$ or in $Q_i$ but scheduled before this operator

Find the savings from the above eliminations;

Let $Q_j$ in $S$ have the maximal savings;

Eliminate the contained mining operators from $Q_j$;

Scheduled $Q_j$ after $S - \{Q_j\}$;

$S = S - \{Q_j\}$.

**Figure 5: Greedy Algorithm to Remove Containment in Multiple Query Plans**

To find the evaluation order for $n$ queries to achieve the maximal savings from removing the common computations, a simple enumeration method will have the time complexity $O(n!)$. If $n$ is large, this method is very expensive. Therefore, we propose a greedy algorithm, which is sketched in Figure 5. This greedy algorithm utilizes the following property. If a query plan, $Q$, is scheduled after a set of query plans, $S$, then the contained mining operators in $Q$ do not depend on how the contained mining operators are removed within the set $S$. This is based on the transitive property of the containment relationships. To utilize this property, our algorithm finds the query plan which has the maximal savings when it is scheduled as the last one. Such a plan is then scheduled last, and then the order of the remaining operations is determined. Note that since the exact savings cannot typically be determined, we use simple heuristics, such as the number of mining operators, as the cost function.

Consider applying the greedy algorithm on the query plans of query $Q_1$ and $Q_2$, which are as follows:

$Q_1 : SF(A, 0.2); CF(B, 0.1, SF(A, 0.2)^I); CF(X, 0.1, (SF(A, 0.2) \cap SF(B, 0.1))^I)$

$Q_2 : SF(A, 0.1), SF(Z, 0.2); CF(B, 0.1, SF(A, 0.1)^I); CF(Y, 0.01, ((SF(A, 0.1) \cap SF(B, 0.1)) \cup SF(Z, 0.2))^I); CF(Z, 0.01, ((SF(A, 0.1) \cap SF(B, 0.1)) \cap SF(C, 0.1))^I)$

The algorithm will schedule the query $Q_2$ before $Q_1$, and the first two mining operators in the query plan of $Q_1$ will be eliminated.

<table>
<thead>
<tr>
<th>A</th>
<th>0.2</th>
<th>0.2</th>
<th>0.1</th>
<th>0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.1</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>X</td>
<td></td>
<td></td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td></td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Z</td>
<td>0.01</td>
<td>0.2</td>
<td>0.2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$Q_1$</th>
<th>$Q_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$SF(A, 0.2)$</td>
<td>$SF(Z, 0.2)$</td>
</tr>
<tr>
<td>$CF(B, 0.1, SF(A, 0.2)^I)$</td>
<td>$CF(Y, 0.01, ((SF(A, 0.1) \cap SF(B, 0.1)) \cup SF(Z, 0.2))^I)$</td>
</tr>
<tr>
<td>$CF(X, 0.1, (SF(A, 0.2) \cap SF(B, 0.1))^I)$</td>
<td>$CF(Z, 0.01, ((SF(A, 0.1) \cap SF(B, 0.1)) \cap SF(C, 0.1))^I)$</td>
</tr>
</tbody>
</table>

**5.4 Global Query Plans**

A drawback of the above approach is that it is very sensitive to the local plans, and often cannot find efficient query plans. For example, consider the new query $Q'_2$ which is created by replacing the sub-condition in the query $Q_2$, $FB \geq 0.1$ by $FB \geq 0.15$. The query plan for $Q'_2$ is as follows.

$$SF(B, 0.15), SF(Z, 0.2); CF(A, 0.1, SF(B, 0.15)^I); CF(Y, 0.01, ((SF(A, 0.1) \cap SF(B, 0.15)) \cup SF(Z, 0.2))^I); CF(Z, 0.01, ((SF(A, 0.1) \cap SF(B, 0.15)) \cap SF(C, 0.1))^I);$$

If we are evaluating queries $Q_1$ and $Q'_2$ together, the above approach cannot find any common computations between the two query plans, and the mining operators will be invoked 8 times.

However, the $M$-table format of queries enables us to perform more aggressive optimizations. This new approach does not depend on the local query plans. Instead, this approach combines the local $M$-tables from different queries into a single large $M$-table by the merge operation (§). Then, it generates a global query plan based on this merged $M$-table. Consider the merged $M$-tables for query $Q_1$ and $Q'_2$ in Table 6.

We can have the following global query plan which needs only 6 mining operators.

$$SF(A, 0.1), SF(Z, 0.2); CF(A, 0.1, SF(B, 0.1)^I); CF(X, 0.1, ((SF(A, 0.2) \cap SF(B, 0.1))^I)); CF(Y, 0.01, ((SF(A, 0.1) \cap SF(B, 0.15)) \cup SF(Z, 0.2))^I); CF(Z, 0.01, ((SF(A, 0.1) \cap SF(B, 0.15)) \cap SF(C, 0.1))^I);$$

Compared with the first approach, this global query plan replaces the four mining operators $SF(B, 0.15), SF(A, 0.2), CF(A, 0.1, SF(B, 0.15)^I), SF(B, 0.1, SF(A, 0.2)^I)$ by two mining operators, $SF(A, 0.1), CF(A, 0.1, SF(B, 0.1)^I)$. This is likely to be more efficient.

**5.5 Knowledgeable Cache Management and Utilization**

We now discuss how the $M$-table can be used for summarizing our cache. Assume in our system, there are a total of $p$ distinct datasets. Then, our cache can use an $M$-table with $p$ rows, where each row corresponds to a dataset, to represent the past evaluation results that are stored in the cache. The set of columns of the $M$-table is dynamically changed after each invocation of a mining operator.

This update procedure is quite simple. Earlier, we had described how each mining operator in a query plan is mapped to an $M$-table. After invocation of a mining operator, besides inserting the mining results in the cache, the $M$-table for the mining operator will be merged with the $M$-table that summarized the cache earlier.

Consider the query plan for the query $Q$ described earlier, and assume the cache is empty initially. Then, the $M$-table of the cache after the evaluation of this query plan is shown in Table 7.

The high-level knowledge of our cache can be used to answer which part of a new query can be answered directly from the cache. Further, to help with the query plan generation, this information is represented by pre-coloring the $M$-table for the new queries. This
is done by using the generalized containment relationship of M-tables based on cells. For each non-empty cell in the M-table for queries, we search the M-table of the cache to see if a column contains it. If such a column exists, the cell will be turned red. As an example, assume we have a cache with an M-table shown in Table 7. The pre-coloring of the merged M-table for queries Q1 and Q2 is shown as Table 8.

After such pre-coloring, less cells need to be colored, and more efficient query plans can be generated. For the first approach to optimize multiple queries (Subsection 5.3), different local query plans are generated from the pre-colored M-tables, and then the common computations among them are removed. For the second approach (Subsection 5.4), a global query plan is generated from the pre-colored merged M-tables. For queries Q1 and Q2, both approaches will generate the following query plan:

\[ SF(Z, 0.2); \]
\[ CF(X, 0.1, (SF(A, 0.2) \cap SF(B, 0.1))^4); \]
\[ CF(Y, 0.01, ((SF(A, 0.1) \cap SF(B, 0.15)) \cup SF(Z, 0.2))^4); \]
\[ CF(Z, 0.01, ((SF(A, 0.1) \cap SF(B, 0.15) \cap SF(C, 0.1))^4); \]

### Table 7: M-Table for the Cache

<table>
<thead>
<tr>
<th>A</th>
<th>0.1</th>
<th>0.05</th>
<th>0.1</th>
<th>0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.05</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>C</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>0.05</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 8: Pre-Colored M-Table for Query Q1 and Q2

<table>
<thead>
<tr>
<th>A</th>
<th>0.2</th>
<th>0.2</th>
<th>0.1</th>
<th>0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.1</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>X</td>
<td></td>
<td>0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td>0.1</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>Z</td>
<td></td>
<td>0.01</td>
<td>0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>

### 5.6 Test Queries

We use a collection of query templates involving a different number of datasets, ranging from one to four. Each template involves several different thresholds. For convenience, the thresholds are classified into two groups. A threshold is positive if it is in a positive predicate, and negative if it is in a negative predicate. Table 9 illustrates several templates used in our experiment, where we use \( \alpha \) and \( \beta \) to represent the positive and negative thresholds, respectively. To generate a query from these query templates, we assign values to each threshold in the query template. For IPUMS datasets, a positive threshold ranges from 50% to 90%, and a negative threshold from 0% to 50%. For Quest datasets, a positive threshold ranges from 0% to 95%, and a negative threshold from 0.05% to 0.2%.

### Table 9: Pre-Colored M-Table for Query Q1 and Q2

<table>
<thead>
<tr>
<th>A</th>
<th>0.2</th>
<th>0.2</th>
<th>0.1</th>
<th>0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>0.1</td>
<td>0.1</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>X</td>
<td></td>
<td>0.1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td></td>
<td>0.1</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>Z</td>
<td></td>
<td>0.01</td>
<td>0.2</td>
<td>0.2</td>
</tr>
</tbody>
</table>

### 6.2 Datasets

Our experiments were conducted on two groups of datasets, each of them comprising four distinct datasets:

**IPUMS:** The first group of datasets is derived from the IPUMS 1990-5% census micro-data, which provides information about individuals and households [1]. The four datasets each comprises 50,000 records, corresponding to New York, New Jersey, California, and Washington states, respectively. Every record in the datasets has 57 attributes.

**IBM’s Quest:** The second group of datasets represents the market basket scenario, and is derived from IBM Quest’s synthetic datasets [2]. The first two datasets, dataset-1 and dataset-2, are generated from the T20.J8.N2000 dataset by some perturbation. Here, the number of items per transaction is 20, the average size of large items is 8, and the number of distinct items is 2000. For perturbation, we randomly change a group of items to other items with some probability. The other two datasets, dataset-3 and dataset-4, are similarly generated from the T20.J10.N2000 dataset. Each of four datasets contains 1,000,000 transactions.

### 6.3 Test Queries

We use a collection of query templates involving a different number of datasets, ranging from one to four. Each template involves several different thresholds. For convenience, the thresholds are classified into two groups. A threshold is positive if it is in a positive predicate, and negative if it is in a negative predicate. Table 9 illustrates several templates used in our experiment, where we use \( \alpha \) and \( \beta \) to represent the positive and negative thresholds, respectively. To generate a query from these query templates, we assign values to each threshold in the query template. For IPUMS datasets, a positive threshold ranges from 50% to 90%, and a negative threshold from 0% to 50%. For Quest datasets, a positive threshold ranges from 0% to 95%, and a negative threshold from 0.05% to 0.2%.

### 6.4 Experimental Settings

In our experiments, we evaluate three methods to deal with multiple mining queries. The first is the naive method, which generates efficient query plan for each single query, without considering their common computations. The second method is as described in Subsection 5.3. It tries to remove the common computations among the local query plans and greedily selects an evaluation order. The third method is as described in Subsection 5.4. It merges the local queries into one single query by using the M table format, and then generates an efficient global query plan. For our discussion, we denote them as SQ (single query plan), LQ (local query plan), and GQ (global query plan), respectively. In each of these methods, we use the Algorithm-CF to generate our query plans.

We also consider the following experimental settings to study the impact of pre-computation and caching:

**Setting-I:** No pre-computation and caching.

**Setting-II:** Use pre-computation only.

**Setting-III:** Use Caching only.
removal. This observation also validates the effectiveness of our methods to optimize multiple queries.

From the experimental results, we can see that pre-computation and caching also help reduce the evaluation costs. In our experiments, Setting-IV which combines pre-computation and caching is always the best. Setting-III (purely caching) is also quite effective, and delivers a speedup quite close to Setting-IV. Compared with Setting-I (no caching and pre-computation), Setting-II (Pre-computation) achieves an average speedup of 1.2, 1.4, 1.3, and 1.2, in Tables 10, 11, 12, and 13, respectively. The gains from the Setting-III amount to a factor of 1.8, 2.2, 3.8, and 5.0, respectively. Finally, the Setting-IV achieves the highest gains, with an average speedup of 1.9, 2.6, 4.0, and 5.9, respectively.

In the Setting-IV, caching and pre-computation maximize the gains for the both local and global query plans. Specifically, compared with SQ in the Setting-I, the average speedups of LQ in the Setting-IV are 2.6, 3.5, 5.1, and 8.3, in the Tables 10, 11, 12, and 13, respectively. GQ obtains an average speedup of 4.0, 4.5, 8.8, and 9.2, respectively.

An interesting property of caching is if there is no cache replacement, as is the case in our system, it reduces the average query evaluation time as more queries are being evaluated. Table 14 shows this caching effect. Here, global query plans are used. Each row of the table corresponds to a different batch size, ranging from 1 to 6. The columns in the table correspond to the number of queries being evaluated. We issued four sets of queries, with a total of 24, 48, 96, and 144 queries, respectively, to the query queue in our system. We can see that as more queries are processed by the system which is using the cache, the average of the batch processing time is reduced. Specifically, the average evaluation time has reduced from 9.2 seconds per query when there are 24 queries, to only 5.0 seconds per query when there are a total of 144 queries.

### 7. RELATED WORK

This section compares our work with related research efforts. A number of constraint frequent itemset mining algorithms have been developed, with the goal of using additional conditions and pruning the search space [8, 16, 19, 21, 27]. More recently, Yan et al. have studied the use of connectivity constraints to mine frequent graphs [31]. However, these algorithms cannot efficiently answer our target class of queries, since the conditions in our queries correspond to a set of (in)frequent patterns. Moreover, they have not considered the multiple query optimization problem.

Raedt and his colleagues have studied the generalized inductive query evaluation problem [15, 17, 23]. Although their queries target multiple datasets, they focus on the algorithmic aspects to apply version space tree and answer the queries with the generalized monotone and anti-monotone predicates. In comparison, we are in-
tered in answering queries involving frequency predicates more efficiently.

Our research is also different from the work on Query flocks [28]. While they target complex query conditions, they allow only a single predicate involving frequency, and on a single dataset. The work on multi-relational data mining [7, 10, 22, 32] has focused on designing efficient algorithms to mine a single dataset materialized as a multi-relation in a database system.

A number of researchers have developed techniques for mining the difference or contrast sets between the datasets [6, 9, 30]. Their goal is to develop efficient algorithms for finding such a difference, and they have primarily focused on analyzing two datasets at a time. In comparison, we have provided a general framework for allowing the users to compare and analyze the patterns in multiple datasets.

As discussed in Section 1, some efforts have been made toward addressing the issues arising from sequence of queries and multiple simultaneous queries in mining environments. Nag et al. [12] have studied how a knowledgeable cache can be used to help perform interactive discovery of association rules [18]. Hipp and Gunter have proposed to use pre-computation of frequent itemsets of certain support levels to answer constraint itemset mining queries [13]. Goethals and Bussche have developed methods to support an interactive data mining session [11]. Compared with our work, these efforts have not addressed both of the issues, sequence of queries and multiple simultaneous queries, together, and the knowledgeable cache is restricted to simple data mining queries.

Multiple query optimization has been widely studied in database systems [25, 20, 24, 26]. Here, the focus has been on finding efficient query plans by dealing with the trade-offs between materialization and re-computation of common subexpressions. Zhao et al. [34] have studied simultaneous optimization of a restricted kind of queries, called multi-dimensional queries. The main differences between their study and our approach is that we assume that common computations will always be materialized, and we have developed an efficient way to detect and utilize such common computations.

Andrade et al. have studied how to simultaneously optimize a group of related scientific data processing queries [5]. However, their methods are mainly based on the spatial properties of the queries and cannot be applied to the mining tasks we have focused on.

8. CONCLUSIONS

The work presented in this paper is driven by the need to efficiently process a large number of data mining queries, which are being issued by a number of users. To speedup the evaluation of queries in such a scenario, we need to not only evaluate each single query efficiently, but also need to optimize multiple queries simultaneously. Furthermore, we need to be able to utilize mining results from past queries in a systematic fashion.

In this paper, we have presented a novel system architecture to deal with such a query intensive environment. We have proposed new algorithms to perform multiple-query optimization for frequent pattern mining queries which involve multiple datasets. We also designed a knowledgeable cache which can store the past query results from queries, and enable the use of these results to further optimize multiple queries. Finally, we have implemented and evaluated our system with both real and synthetic datasets. Our experimental results have demonstrated a speedup of up to a factor of 9.

9. REFERENCES

[16] Lukas V. S. Lakshmanan, Raymond Ng, Jiawei Han, and Alex Pang. Optimization of constrained frequent set queries with 2-attribute constraints. In Proceedings of the 1999 ACM SIGMOD international conference on Management of data, pages 157–168, 1999.
[21] Jian Pei, Jiawei Han, and Luk V. S. Lakshmanan. Mining frequent item sets with convertible constraints. In Proceedings of the 17th International Conference on Data Engineering, pages 433–442, 2001.

32
ABSTRACT
This article presents a generalized metric distance, called \( \Delta \)-distance, between images represented by a tree structure resulting from a recursive image partition. This distance is used to perform content-based image retrieval queries in databases. \( \Delta \)-distance allows to retrieve images globally similar to a query image. This distance takes into account the location of the image visual features. It can be performed using a multi-level filtering algorithm. Moreover, \( \Delta \)-distance allows region-based queries. In this case, the resulting images contain quadrants similar to the quadrants selected by the user in the query image or contain quadrants similar to the entire query image. Because it is a generalized distance function, some particular cases of the \( \Delta \)-distance appear in existing content-based image retrieval systems.

Categories and Subject Descriptors
H.2.8 [Information Systems]: Database Applications—Image databases; I.4.10 [Image Processing and Computer Vision]: Image Representation—Hierarchical; I.5.3 [Pattern Recognition]: Clustering—Similarity measures; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms
Measurement, Standardization

Keywords
Image database, quad/quin/nona-tree, recursive image decomposition, content-based image retrieval, multi-level filtering, image region similarity

1. INTRODUCTION
Content-based image retrieval (CBIR) has become a very well studied research area because of the increased number of available image databases. Readers are referred to surveys [21] and [25]. In a CBIR system, the user chooses a query image in the database and the system returns a list of images similar to the query image. The similarity between images is measured using a distance function [19]. The smaller the distance between images is, the more similar the images are.

To improve image retrieval accuracy or to enlarge the query expression, several approaches [1, 2, 5, 7, 9, 10, 11, 12, 13, 15, 20] use a tree to model a recursive image partition and compute a distance between images based on the comparison of the image tree representation. All the distances used in these approaches are analogous and can be generalized into a single generic definition. However, to the best of our knowledge, no general definition of such a distance exists.

The first contribution of this article is to formally define the generalization of the distances used in the aforementioned approaches. We called this generalized distance: \( \Delta \)-distance. By analyzing \( \Delta \)-distance, we point up that several distances (global or local) and several distance computation processes (with or without filtering) could be derived from the same definition depending on the parameters used. The last contribution of this article is a short survey showing the distance-based relations between the aforementioned approaches.

The article is organized in the following way. Section 2 deals with the recursive image decomposition for content-based image retrieval and presents the notation used in this article. Section 3 presents the contributions of this article, the \( \Delta \)-distance and the \( \Delta \)-based image retrieval processes. Finally, Section 4 concludes this article and offers directions for future work.

2. BACKGROUND AND RELATED WORK
Several content-based image retrieval approaches [1, 2, 5, 7, 9, 10, 11, 12, 13, 15, 20] deal with the same image search process. Firstly, the images from the database are decomposed into several fixed-size quadrants. Secondly, each image is represented by a tree storing the feature vectors of all its
quadrants. Finally, image similarity is computed using a distance between image trees.

This section explains in details the two first steps of the aforementioned image retrieval process – the image decomposition (see Section 2.1) and the tree representation (see Section 2.2) – and presents the notation conventions (see Section 2.3). It provides the necessary background to understand how a generalized Δ-distance, presented in Section 3, could be defined.

2.1 Recursive image partition

Figure 1: An image recursively decomposed into four or nine quadrants.

An image could be recursively decomposed into fixed-size quadrants. Using a quadtree decomposition, an image is recursively decomposed into four quadrants. Figure 1.(a) shows an example. The root of the quadtree (stored at level 0 in this article) represents the entire image. At level 1, the root node has four child nodes representing the first four quadrants of the image. If the decomposition is stopped after two iterations, level 2 contains 16 nodes, representing the four sub-quadrants of the first level quadrants. This kind of decomposition is used in [1, 2, 5, 7, 9, 10, 12, 11, 15].

Using a quin-tree, an image is recursively decomposed into five quadrants, the first fourth ones following the NW, NE, SW and SE directions and the fifth one representing the center of the image. This kind of decomposition is used in [18]. Using a nona-tree decomposition, an image is recursively decomposed into nine quadrants. As previously, the root of the tree represents the entire image. But at level 1, the root node has nine child nodes representing the first nine quadrants of the image. Figure 1.(b) shows the nona-tree decomposition of the Lena image. If the decomposition is stopped after two iterations, level 2 contains 81 nodes, each node of the first level having 9 child nodes. In [20, 27], a nona-tree is used for content-based image retrieval.

Several types of decomposition can be mixed. For example, level 1 of the hierarchical structure proposed in [13, 14] contains nine nodes, as in a nona-tree. The second level, however, corresponds to a second level of a quadtree containing only 16 nodes.

2.2 Tree structures for CBIR

Content-based image retrieval systems allow to retrieve images from a database according to their visual features. The features, including color [24], texture [19] and shape [4], are stored as numerical vectors, called feature vectors [9]. When tree structures are used in CBIR system, each tree node stores the feature vector of the corresponding image quadrant. For example in [12], the quadtree nodes store the color histograms of the corresponding quadrant. Figure 2.(b) gives an example of such a quadtree for the Lena image. In [11], each quadtree root stores the average color vectors of each corresponding image, in a three-dimensional color space \((R_{avg}, G_{avg}, B_{avg})\), \(R\) for red, \(G\) for green and \(B\) for blue. The other nodes store a color histogram of the corresponding image quadrant. Figure 2.(c) gives an example of such a quadtree storing an average color vector in all its nodes. In [9], the tree nodes store color vectors having 9 dimensions \((E_u, E_v, E_w, \sigma_u, \sigma_v, \sigma_w, s_u, s_v, s_w)\), based on color moments where \(E\) represents the average color, \(\sigma\) represents the variance and \(s\) represents the skewness of each dimension in the color space \((R, G, B)\). Any kind of feature vector could be stored into the tree nodes, like a shape feature vector [10], a combination of color and texture captured via histograms [15], or a dominant color [5]. We call such a structure a Multi-Level Feature Vector. This structure has several purposes in content-based image retrieval.

Figure 2: The Lena image: (a) decomposed into a quadtree and represented by (b) a multi-level color histogram and (c) a multi-level average color feature vector.
Firstly, a multi-level structure can be used as a multi-level filtering structure as it is suitable to a coarse-to-fine representation [9]. In this case, each image is first compared with the query image according to its global feature vectors (stored in the root node of the tree). If they are similar enough (with regard to a similarity metric and a given threshold), the first sub-quadrants of the images are compared, and so on. For more details about multi-level filtering using a partition-based tree, readers are referred to [9, 12]. Section 3.2 shows how the generalized distance, proposed in this article, is used for multi-level filtering.

Secondly, a multi-level feature vector allows different kinds of query image called, in [9], global search, sub-image search or pattern search. Global search consists in retrieving images of the database globally similar to a query image. Sub-image search compares a query image with any image quadrant stored in the database whatever is the size of the compared quadrants. Pattern search retrieves images from the database having similar regions chosen in a query image. Among the approaches based on images represented by multi-level feature vectors, the proposition presented in [11, 12] allows global search. The approach of [13, 14] deals with sub-image search. In [15], a solution is presented for pattern search.

2.3 Notation
In this article, the tree nodes and the corresponding image quadrants are identified in the following way. The root node, identified by numeral 0 in this article, represents the initial quadrant containing the whole image. Numerals 0, 1, 2 or 3, following their parent node identifier 0, identify the four first level image quadrants and the four first level tree nodes. This identification is done using a Z order, which corresponds to the NW, NE, SW and SE directions. Numerals 4, following its parent node identifier 0, identifies the central quadrant and the corresponding tree node. Numerals 5 and 6 (resp. 7 and 8), following their parent node identifier 0, identify the vertical (resp. horizontal) central quadrants and the corresponding tree nodes. Recursively, sub-quadrants of an image quadrant n and children of a quadtree node n are identified by nx where x ∈ {0, 1, 2, 3, 5, 6, 7, 8}. This identification convention is used in Figure 1.

In the following, letter l represents a level in a tree. The root node is, in this article, at level 0. Letter n represents a tree node or an image quadrant identifier. Letter N represents the set of nodes (resp. quadrants) identifiers appearing in the database. In this article, we consider that all images of the database have the same tree decomposition (same numbers of nodes by level and same number of levels by tree). Letters i, j and k represent image identifiers and letter q represents the query image identifier. Table 1 (at the end of the paper) summarizes the meaning of all symbols used in this article. Two nodes (or quadrants) with the same identifier in two different trees (or images) are called homologous nodes (or homologous quadrants). This convention is used in the definition of the Δ-distance presented in the next section.

3. Δ-DISTANCE
This section presents the Δ-distance. We first proposed this distance in [22, 23] for images represented by quadtrees. In this article, our first definition of Δ-distance is generalized to any tree based on recursive decomposition of images. This generalization represents the first step of our final main purpose: to develop a CBIR system allowing to compare different Δ-based distances on the same image set and helping the user to define the right parameters depending on its image database. At the end, this system could also be used to compare similar approaches [1, 2, 5, 7, 9, 10, 12, 11, 15] by implementing their common characteristics. Indeed, until now, it is difficult to really compare the performances of these approaches because their prototypes are not always accessible and are generally based on different frameworks. This article does not deal with this desired CBIR system but gives the first bricks to develop it.

After defining the Δ-distance (see Subsection 3.1), this section points out that Δ-distance is a distance for content-based image retrieval allowing multi-level filtering (see Subsection 3.2) or region-based image queries (see Subsections 3.3 and 3.4). Existing distances used in several CBIR systems appear to be particular cases of Δ-distance, as shown out in Subsection 3.5.

3.1 General definition of Δ-distance
Let δ(i, j, n) be a normalized metric distance – δ(i, j, n) ∈ [0, 1] – between feature vectors of quadrants n of two different images i and j. δ can be any geometric distance of the Minkowski family (see [19] for more details about this family) and can be a weighted distance. Let Δ be the distance between two images i and j, represented by a multi-level feature vector. Δ-distance is defined as a weighted sum of normalized δ-distances between feature vectors stored in tree nodes n:

\[ Δ(i, j) = \frac{\sum_{n \in N} w_n δ(i, j, n)}{W} \]  

(1)

The coefficient \( w_n, w_n ≥ 0 \), represents the weight of distance δ between homologous nodes n in the Δ-distance computation. Δ-distance is normalized by the denominator \( W \) which is the sum of all weights \( w_n \) associated with the image quadrants n: \( W = \sum_{n \in N} w_n \). At least one \( w_n \) must be different from zero: \( Δ ∈ [0, 1] \). Below we only consider normalized distances.

Δ is a metric distance because it is a linear combination of metric distances (δ) between homologous tree nodes. Thus, ∀i, j and k, three different images represented by multi-level feature vectors:

- \( Δ(i, i) = 0 \) because ∀n ∈ N, \( δ(i, i, n) = 0 \)
- \( Δ(i, j) = Δ(j, i) \) because ∀n ∈ N, \( δ(i, j, n) = δ(j, i, n) \)
- \( Δ(i, j') ≤ Δ(i, j) + Δ(j, j') \) because δ is a metric distance then ∀n ∈ N, \( δ(i, i', n) ≤ δ(i, j, n) + δ(j, i', n) \) and thus \( w_n δ(i, j, n) ≤ w_n δ(i, j, n) + w_n δ(j, i', n) \) because ∀n ∈ N, \( w_n ≥ 0 \)

The Δ-distance must be used when the spatial position of features is an important criterion for the image similarity. This distance takes into account the feature vectors values and their positions in the images. Figure 3, issued from the
A global search result using a global dis-

prototype of [9], shows an example of this kind of content-

based image retrieval. The query image is on the left - see Figure 3.(a). Several images of the database result from im-
age blurring, pixelization or spreading applied to the query image. Other images correspond to the query image after a rotation. As shown in Figures 3.(b) and 3.(c), the result image ranking is different when a global distance or a -distance is used. Using a global distance, the images are compared with the query image using only their global feature vectors (9-dimensional vectors based on color moments). In this case, the rotated images appear first in the query result because they result from a rotation of the query image without any other image processing operations. Using -distance, the images are compared using their multi-level feature vectors and then comparing all homologous image quadrants. In this example, the images of the database are represented by a three-level balanced quadtree containing a 9-dimension color vectors. An image quadrant at level represents of the entire image surface. Thus each image quadrant is associated with a weight where is the level of the corresponding quadtree node. Using -distance, the processed images appear first in the query result. The rotated images appear farther because of the different location of their features comparing to the query image.

Figure 3: A global search result using a global dis-

or a -distance.

Figure 4: Using the -distance for quadtree-based multi-level filtering.

3.2 Multi-level filtering using -distance

To compute the -distance between two images represented by multi-level feature vectors, an exhaustive comparison of both trees is done. However, the -distance computation can be gradually refined by comparing both trees level by level, following a breadth first order.

Let be an approximation of -distance between multi-level feature vectors of images and . It is computed from the distances between homologous nodes appearing from root level (level 0) to level . Intuitively, the -distance is computed without taking into account details after a certain tree level , i.e. the information contained in lower-level tree nodes. Let be the set of nodes appearing at level . For any given level , in two multi-level feature vectors of images and :

\[ \Delta^{(\ell)}(i, j) = \frac{1}{W} \sum_{k=0}^{\ell} \left( \sum_{n \in \chi^{(k)}} w_n \delta(i, j, n) \right) \]  

(2)

is computed taking into account only both tree roots, i.e. the global feature vectors of images and . If all multi-level feature vectors of the database have, for example, three levels, then for all images and :

\[ \Delta^{(3)}(i, j) = \Delta^{(1)}(i, j) \]

\[ \Delta^{(\ell+1)}(i, j) = \Delta^{(\ell)}(i, j) + \sum_{n \in \chi^{(\ell+1)}} w_n \delta(i, j, n) \]

(3)

For any given level , the -distance is an increasing function of :}

\[ \Delta^{(\ell-1)}(i, j) \leq \Delta^{(\ell)}(i, j) \leq \Delta^{(\ell+1)}(i, j) \]

(4)

Using -distance, the global search of images similar to a query image is computed in several steps of filtering. Let be the set containing images from the database such that . Then, for each images in set , the -distance is computed. The result is a set containing images from the database such that . The process stops when the last level of trees is reached, i.e. when -distance is computed. Thus, the exhaustive comparison between all quadtree feature vectors is only computed for a restricted number of images, each step of the filtering process reducing the image set to be compared. Figure 4 summaries this filtering process. This figure deals with a quadtree-based multi-level filtering, but the filtering process can obviously be extended to any image recursive decomposition.

To compute a global search, the user has to specify the query image , the distance (for example the Euclidean distance ), the weights and a threshold . The query result is an ordered set of images checking (Formula 1). The first step of the filtering consists in computing . At the end of this first step, the result is a set containing images from the database such that . Then, for each images in set , the -distance is computed. The result is a set containing images from such that . The process stops when the last level of trees is reached, i.e. when -distance is computed. Thus, the exhaustive comparison between all quadtree feature vectors is only computed for a restricted number of images, each step of the filtering process reducing the image set to be compared. Figure 4 summaries this filtering process. This figure deals with a quadtree-based multi-level filtering, but the filtering process can obviously be extended to any image recursive decomposition.
of the feature vectors of its sub-quadrants. For example in [5], tree nodes store the dominant color of the corresponding quadrant. However, feature vectors are often computed from statistical methods (e.g. color moment). Tree node \( n \) can also aggregate its rooted sub-tree’s values by containing an approximation of feature vectors stored in its descendant nodes. In these cases, \( \Delta \)-distance can be approximated by associating \( w_n \geq 0 \) with all nodes \( n \) appearing at level \( \ell \) and associating null weights \( (w_n = 0) \) with the other nodes. Let \( \Delta^{(\ell)} \)-distance be this approximated \( \Delta \)-distance computed by taking only into account nodes of level \( \ell \). \( \Delta^{(\ell)} \)-distance is defined by the following formula:

\[
\Delta^{(\ell)}(i,j) = \frac{\sum_{n \in \mathcal{N}^{(\ell)}(i,j)} w_n \delta(i,j,n)}{\sum_{n \in \mathcal{N}^{(\ell)}(i,j)} w_n}
\]

(5)

In this formula, nodes \( n \) appear at level \( \ell \) in both trees of images \( i \) and \( j \) \( (n \in \mathcal{N}^{(\ell)}) \). Formula (5) preserves the previous inequality \( \Delta^{(\ell-1)}(i,j) \leq \Delta^{(\ell)}(i,j) \leq \Delta^{(\ell+1)}(i,j) \), because feature vectors stored at level \( \ell \) approximate those stored at level \( \ell + 1 \).

Figure 5 shows the experiment results obtained with the prototype of [9]. The database contains 2311 images. All images are represented by a three-level feature vectors whose nodes contain 9-dimensional color vectors based on the first third color moments. \( \Delta^{(\ell)} \)-distance is used to approximate \( \Delta \)-distance. The number of compared images decreases at each step of the filtering process. The larger the threshold \( \alpha \), the less the decrease of the number of images to be compared. With \( \alpha = 0.5 \), the first step eliminates 914 images from the result set: only 1397 images verify \( \Delta^{(0)}(q,i) \leq \alpha \). After the second filtering step (computing \( \Delta^{(1)}(q,i) \leq \alpha \)), set \( \mathcal{R}^{(1)} \) contains 611 images: 414 images do not verify \( \Delta^{(1)}(q,i) \leq \alpha \).

3.3 Pattern search using \( \Delta \)

A multi-level feature vector allows pattern search. It consists in retrieving images having regions similar to a query pattern given by the user [9]. The user selects one or several quadrants in a grid image, for example a 4 × 4 or 16 × 16 grid depending on the decomposition of the images. The selected quadrants represent a pattern \( p \) which becomes the query pattern \( q_p \). \( \Delta \)-distance can be used to compute such a pattern search associating \( w_n \geq 0 \) with all nodes \( n \) corresponding to the image quadrants selected by the user and \( w_n = 0 \) for the other quadrants. We call such a distance: \( \Delta_p(i,j) \).

Figure 6: A pattern query (adapted from [15]).

To perform pattern query, the user specifies weights \( w_n \) and a threshold \( \alpha \) for each selected quadrant. Existing image quadrants of the database can also be selected as query pattern - see the example represented in Figures 6.(a) to 6.(c), adapted from [15]. The CBIR system described in [15] detects the minimum bounding rectangle of the selected quadrants – see Figure 6.(d). The initial query \( q_p \) can be transformed into several queries, each one representing a geometric transformation (translation or rotation) of the initial minimum bounding rectangle. In [15], for example, the initial query of Figure 6.(d) is transformed into 6 queries: the initial query of Figure 6.(d) and the 5 queries of Figure 6.(c). Each query results from a translation of the selected quadrants in the initial query \( q_p \). An detailed algorithm to compute \( \Delta_p \)-distance is described in [9].

Figure 7: A pattern search using \( \Delta \)-distance.

Figure 7 shows an example of a pattern search realized by the prototype of [9]. The user wants to retrieve images from the database having NW and NE quadrants of level 1 similar to the homologous quadrants of the query image (see at the top of the figure). Weight 0.9 is associated with the query image quadrant 00 and weight 0.1 with the query image quadrant 01. The other quadrants are associated with
a null weight. In this example, no translation of the query pattern is done.

3.4 Sub-image search using $\Delta$

A multi-level feature vector also allows sub-image search. In this kind of search, a query image is compared with all image quadrants stored in the database, each quadrant being considered as an independent image. $\Delta$-distance can be used to compute such a sub-image search. In this case, $\Delta(q, i)$ is computed for all images $i$ of the database and all image quadrants of the images.

Figure 8: The first step of sub-image search: a global search.

Figure 9: The next steps of a sub-image search: a comparison between the query tree and the sub-trees representing the image quadrants.

A sub-image search is performed in several steps: as many steps as the number of levels in the image trees. The first step of a sub-image search is a global search (see Section 3.2). The second step compares the query image with all image quadrants of the first level. The last step compares the query image with the last level quadrants all images. Figures 8 and 9 summarize the sub-image search process, using a quadtree-based multi-level feature vectors — it can be generalized to any tree resulting from a recursive image decomposition. Figure 10 shows an example of a sub-image search, from the prototype of [9]. The query image is on the left. Some images of the database result from a composition of real images in order to see when a quadrant appear at different level. As shown in Figure 10, the result images are globally similar to the query image (see the first returned image) or have some quadrants similar to the query image (see the last three returned images).

Figure 10: A sub-image search using $\Delta$-distance.

3.5 Particular cases of $\Delta$-distance

Particular cases of $\Delta$-distance can be found in [1, 2, 5, 7, 9, 10, 11, 12, 13, 15, 20], depending on the weights $w_n$ and on the $\delta$-distance used to compare image quadrants $n$. In the following, due to a lack of space, particular cases presented in this article are limited to the distances used in [9, 10, 11, 12, 13, 15, 20]. For the other approaches, readers are referred to [23].

The authors of [9, 11] and [12] represent each image of the database by a full fixed-depth balanced quadtree whose nodes contain a color feature vector of the corresponding image quadrants. In [11], the root nodes contain average color vectors of the entire images and the other quadtree nodes contain color histograms. In [12], all quadtree nodes contain a color histogram. In both approaches, the distance between images is a $\Delta$-distance such that weights $w_n$ correspond to surface coefficients ($w_n = 4^{-\ell}$ for all nodes $n$ at level $\ell$) and $\delta$ is an Euclidean distance. Both approaches use a multi-filtering process, using a $\Delta^{(\ell)}$-distance. Moreover, in the DISIMA image DBMS [17], implementing the proposition of [11], sub-image queries can be answered using selected portions of the multi-scale color histograms. This kind of distance can be defined using a $\Delta_p$-distance. In [9], quadtree nodes contain a 9-dimensional color vector based on the first third color moments. This approach allows multi-filtering global search using a $\Delta^{(\ell)}$-distance and pattern or sub-image searches using a $\Delta_p$-distance.

The authors of [15] also represent each image by a full fixed-depth balanced quadtree to perform region queries. The user, after choosing an image from the database, has to select several image quadrants at a chosen level $\ell$, using a $16 \times 16$ grid. A bounding box is defined as the smallest sub-image containing the selected quadrants (see Figure 6). Then, each image of the database is compared with the initial query image and the query images obtained after a translation of the bounding box containing selected quadrants. The visual content of image quadrants is represented by a high dimensional feature vector combining color and texture feature. The authors define a distance which is a normalized linear combination of distances. This distance corresponds to a $\Delta_p$-distance. All chosen regions in the query image have the same size, thus all nodes $n$ representing the regions selected by the user in the query image have the same value of weight $w_n$. The values of weights $w_n$ are zero for the other nodes $n$ which do not represent the selected pattern.
In [13], images are represented by a tree structure, whose first level corresponds to a nona-tree, containing 9 nodes, and the last level corresponds to a quadtree, containing 16 nodes. Tree nodes store the mean and the covariance color of the corresponding image quadrants in the \( L^*a*b^* \) color space. The approach only deals with region-based image retrieval, taking into account only the leaf node distances. The distance computed is a \( \Delta \)-distance such that weights \( w_n \) are zero for all internal nodes \( n \) only leaf nodes are compared. This approach is extended in [14]. A relevance feedback process (adapting weights \( w_n \)) is proposed to improve the image search. Moreover, the tree nodes contain BIC feature vectors based on the Border/Interior pixel Classification defined in [26].

In [10], the quadtree decomposition is not applied to images but to shapes contained in the images of the database. A shape is divided into four sub-regions by two principal axes corresponding to the two eigenvectors at the center of mass of the shape. Each sub-region is subdivided into four sub-regions in the same way. The sub-division process is repeated a predetermined number of times resulting in a fixed-depth balanced quadtree whose nodes correspond to the regions of the shape derived from the above process. Four parameters, invariant to translation, rotation and scale, are calculated for the corresponding regions of each node while only two parameters are extracted from the root node. The similarity distance used to compare two shapes corresponds to a \( \Delta \)-distance where \( \delta \) is a distance \( L_1 \) (say Manhattan or city block\(^1\)) between quadtree nodes, and where weights \( w_n = 1 \) for all nodes \( n \).

In [20], a nona-tree (with only 49 nodes at level 2, redundant quadrants being stored only once) contains texture vectors. The query image is compared with all the different image quadrants stored in the database, traveling through all nona-trees. This approach deals with sub-image search using a \( \Delta_p \)-distance. When the query image is compared with an image quadrant \( n \), the compared image quadrant \( n \) is associated with weight \( w_n = 1 \) (the other quadrants being associated with a null weight). The \( \delta \)-distance used is an Euclidean distance between texture feature vectors.

Table 2, at the end of the article, summarizes these particular cases of \( \Delta \)-distance.

4. CONCLUSION AND FUTURE WORK

In this article, a generalized metric distance, \( \Delta \)-distance, is presented for images recursively decomposed into fixed-size quadrants and represented by a multi-level feature vectors. This distance allows specific content-based image retrieval queries. It allows to retrieve images globally similar to a query image, taking into account not only the visual features of the images but the location of the visual features too. Thanks to the tree representation of the images, \( \Delta \)-distance can be computed using multi-level filtering which reduces the number of images to be compared at each filtering step. \( \Delta \)-distance also allows region-based queries. In this kind of queries, the resulting images contain quadrants similar to the quadrants selected by the user in the query image (pattern search) or contain quadrants similar to the entire query image (sub-image search). Because \( \Delta \)-distance is a generalized distance, some existing CBIR systems [9, 10, 13, 11, 12, 15, 20] use particular cases of it.

The \( \Delta \)-distance computation is more time-consuming than any global distances computed between global feature vectors (for example, the color histogram of the entire image): the time used to compute \( \Delta \)-distance is proportional to the number of image quadrants resulting from the image decomposition. Thus, index structure must be used to improve the query performances. Several approaches use an index on the root of the multi-level feature vectors: a R-tree [8] in [12], a K-d-tree [3] in [15] or a structure based on extendible hashing [6] in [11]. With this index, the number of images having to be compared, quadrant by quadrant, is limited because the index clusters the images according to their global similarity. Deeper quadtree level comparisons are only performed for the images contained in the filtering result set [9]. As explained in [11], performing the filtering on the entire image database gives a result set containing false hits but no false dismissals. When a system should perform not only global search but region-based search too, an index including all nodes of the multi-level feature vectors is more efficient. Such an index structure is proposed in [9].

This article represents the first steps to develop a prototype allowing users to compare different \( \Delta \)-based distances. The purpose of such a prototype is to help users to fix the right weights \( w_n \) and to choose the \( \delta \)-distance between image quadrants, depending on the query they want and on the images of their database. In this article, \( \Delta \)-distance is only defined to compare images represented by a full balanced tree. However, such a tree could be unbalanced, for example when it results from an image segmentation or when it is used for image storage (see in [16] for a survey on quadtree uses in image domain). Particular cases of \( \Delta \)-distance can be defined for images represented by unbalanced trees [22, 23]. Our future prototype will also integrate this kind of distances.

5. ACKNOWLEDGMENTS

The authors would like to thank the referees of this article and Vincent Oria (NJIT -USA) for their helpful comments and suggestions.

6. REFERENCES


\(^1\)The sum of the absolute difference of each feature vectors stored in homologous tree nodes.


Table 1: Meaning of symbols used

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\ell$</td>
<td>a tree level</td>
</tr>
<tr>
<td>$n$</td>
<td>tree node or image quadrant identifier</td>
</tr>
<tr>
<td>$x$</td>
<td>integer, $x \in {0, 1, 2, 3, 4, 5, 6, 7, 8}$</td>
</tr>
<tr>
<td>$N$</td>
<td>set of all nodes (resp. quadrants) identifiers appearing in the image database</td>
</tr>
<tr>
<td>$N^{(\ell)}$</td>
<td>set of all nodes (resp. quadrants) identifiers appearing at level $\ell$</td>
</tr>
<tr>
<td>$i, i', j$</td>
<td>image identifiers</td>
</tr>
<tr>
<td>$q$</td>
<td>query image</td>
</tr>
<tr>
<td>$q_p$</td>
<td>query pattern (i.e. quadrants selected in an image recursively decomposed)</td>
</tr>
<tr>
<td>$\Delta(i, j)$</td>
<td>distance between multi-level feature vectors of images $i$ and $j$</td>
</tr>
<tr>
<td>$\delta(i, j, n)$</td>
<td>normalized metric distance between feature vectors of homologous quadrants $n$ in images $i$ and $j$</td>
</tr>
<tr>
<td>$w_n$</td>
<td>weight of homologous nodes (resp. quadrants) $n$ in the $\Delta$-distance computation</td>
</tr>
<tr>
<td>$W$</td>
<td>sum of all weights $w_n$ associated with the image quadrants, $W = \sum_{n \in N} w_n$</td>
</tr>
<tr>
<td>$\Delta^{(\ell)}(i, j)$</td>
<td>approximation of $\Delta$-distance computed from nodes appearing from the root level (0) until level $\ell$</td>
</tr>
<tr>
<td>$\Delta^{(\ell)}(i, j)$</td>
<td>approximation of $\Delta$-distance taking into account only nodes at level $\ell$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>similarity threshold ; images having a $\Delta$-distance below $\alpha$ are similar</td>
</tr>
<tr>
<td>$R^{(\ell)}$</td>
<td>result set of the $(\ell+1)^{th}$ step of the multi-level filtering process</td>
</tr>
<tr>
<td>$\Delta_p(i, j)$</td>
<td>$\Delta$-distance computed for a query pattern $q_p$</td>
</tr>
</tbody>
</table>

Table 2: Particular cases of $\Delta$-distance

<table>
<thead>
<tr>
<th>Ref.</th>
<th>Tree</th>
<th>Feature vectors</th>
<th>$\Delta$-distance</th>
<th>$\delta$</th>
<th>weights $w_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[9]</td>
<td>quadtree</td>
<td>color moments</td>
<td>$\Delta^{(\ell)}$ and $\Delta_p$</td>
<td>$L_2$</td>
<td>$4^{-\ell}$ for all nodes $n$ at level $\ell$</td>
</tr>
<tr>
<td>[10]</td>
<td>quadtree</td>
<td>shape feature</td>
<td>$\Delta$</td>
<td>$L_1$</td>
<td>1 for all nodes $n$</td>
</tr>
<tr>
<td>[11]</td>
<td>quadtree</td>
<td>average color in the root and color histograms in the other nodes</td>
<td>$\Delta^{(\ell)}$ and $\Delta_p$</td>
<td>$L_2$</td>
<td>$4^{-\ell}$ for all nodes $n$ at level $\ell$</td>
</tr>
<tr>
<td>[12]</td>
<td>quadtree</td>
<td>color histograms</td>
<td>$\Delta^{(\ell)}$</td>
<td>$L_2$</td>
<td>$4^{-\ell}$ for all nodes $n$ at level $\ell$</td>
</tr>
<tr>
<td>[13]</td>
<td>mixed between nona and quadtree</td>
<td>mean and covariance color</td>
<td>$\Delta$</td>
<td>$L_1$</td>
<td>$w_n = 0$ for all internal nodes $n$ and $w_n = 1$ for leaf nodes $n$</td>
</tr>
<tr>
<td>[15]$^*$</td>
<td>quadtree</td>
<td>histograms combining color and texture</td>
<td>$\Delta_p$</td>
<td>$d$</td>
<td>$w_n = 0$ for all quadrants $n$ not selected by the user</td>
</tr>
<tr>
<td></td>
<td>nona-tree</td>
<td>texture vector</td>
<td>$\Delta_p$</td>
<td>$L_2$</td>
<td>$w_n = 1$ for each compared quadrants $n$</td>
</tr>
</tbody>
</table>

$^*$In [15]: $\delta = d$, where $d$ is a specific distance based on Gaussian assumption and using the Kronecker symbol.
Effective Image and Video Mining: an Overview of Model-Based Approaches

Rokia Missaoui and Roman M. Palenichka
Dept. of Computer Science and Engineering
Université du Québec en Outaouais, Gatineau, Québec, Canada
rokia.missaoui@uqo.ca

ABSTRACT

This paper is dedicated to revisiting image and video mining techniques from the viewpoint of image modeling approaches, which constitute the theoretical basis for these techniques. The most important areas belonging to image or video mining are: image knowledge extraction, content-based image retrieval, video retrieval, video sequence analysis, change detection, model learning, as well as object recognition. Traditionally, these areas have been developed independently, and hence have not benefited from some common sense approaches which provide potentially optimal and time-efficient solutions. Two different types of input data for knowledge extraction from an image collection or video sequences are considered: original image or symbolic (model) description of the image. Several basic models are described briefly and compared with each other in order to find effective solutions for the image and video mining problems. They include feature-based models and object-related structural models for the representation of spatial and temporal entities (objects, scenes or events).

General Terms
Algorithms, design, theory.

Keywords
Image mining, video mining, content-based image retrieval, image model, pattern recognition.

1. INTRODUCTION

In the last two decades, there was a significant development in different areas dealing with image content analysis and image retrieval in large image collections and video sequences due to the tremendous progress in computer technologies, sensors engineering and other related fields. All these different tasks and techniques for information extraction from voluminous databases in the form of images can be considered as image and video mining [1, 29, 30, 42]. It originated from the general data mining paradigm and includes mostly methods for knowledge extraction and intelligent search in large collections of still images and video sequences (movies). On the one hand, image and video mining are two different sub-areas of the multimedia mining – data mining area which involves text, speech, audio, image, and video data – but dealing with similar tasks and common mining techniques as being applied to the same type of input data, i.e. images. In the case of video, we do not consider directly audio or text data and annotations incorporated into the composite video data although they can also be used for more effective multimedia mining purposes [28, 32]. Image and video mining requires a separate treatment compared to other multimedia mining techniques because image representation is different in nature and more complex compared to other multimedia sources [30, 32]. For example, text mining or speech recognition techniques cannot be easily extended to the image domain because they have no appropriate theory, computationally efficient algorithms or models to treat spatial and temporal relations between image entities (objects of interest, scenes or events).

The most important research areas in image and video mining from the application viewpoint are pattern recognition, content-based image retrieval, video retrieval, video sequence analysis, change detection, training for object recognition, and image model learning. All these techniques have been developed independently and use their own approaches although they share two main characteristics: processing of large samples of images and extraction of semantically meaningful information for the user.

According to common definitions, image and video mining deals with the learning and extraction of knowledge from images in the form of image data relationships, models, associations, and image patterns, which are not explicitly stored in the image or video collection at hand [49, 50]. Video mining is considered as an extension of still image mining by including mining of temporal image sequences. However, the video data in the general case represent the multimedia already since they include – besides the image sequence – audio data and textual annotations. However, in the context of this paper, the audio and text data are not directly
considered although their mining can help significantly in analyzing video semantics.

Image and video mining is an inter-disciplinary research area of computer science. Effective solutions of mining tasks involve methods from areas of signal processing, video processing, database management, artificial intelligence, computer vision, machine learning, visual perception, to name just a few.

We revisit the existing image and video mining techniques from the viewpoint of image modeling approaches. Here, we consider an image model as a concise and formal (symbolic) representation of original image data, which preserves its semantic content. If images represent a video sequence of events then the model has to describe additionally temporal changes (events) in the images.

The tremendous redundancy in raw image data and the drastic difference between the original image format and its semantic content requested by the user require concise content-based image description in the form of image models. Originally, images are given in an iconic form, i.e. as two-dimensional arrays of pixels. Obviously, a model representation of image data in the model-based mining methods is different from the compressed formats used in image and video compression techniques – such as the orthogonal transform-based or wavelet-based methods – to reduce huge volumes of image data [5]. The current overview principle is justified by the fact that most advanced mining techniques are based (explicitly or implicitly) on image models at different representation levels of image data. Moreover, new sophisticated models for image and video data have appeared recently, which provide effective approaches to solve the multimedia mining problems [1, 7, 24, 32]. The use of a consistent image model permits an optimal design for an image mining method and reveals ways for its performance improvement. The data mining performance depends only on the model consistency in the case of an optimally designed model-based technique.

Another major advantage of model-based methods, which operate with high-level image entities or objects, is their ability to narrow the semantic gap in image or video mining. Efforts to reduce this gap have been focused on semantic enrichment of low-level features, integration of other sources of information (e.g., audio and textual data), discovery of relevant objects using kernel-based learning techniques, and content-based navigation using generic links based on text or image features [12, 42]. Semantic enrichment is possible through additional information about users, characteristics of features or application domain using ontologies.

The proposed taxonomy of model-based image and video mining techniques is an attempt to put together different model-based methods in accordance with their relevance and effectiveness. Due to the multitude of existing methods for image and video analysis only some characteristic examples of the mining techniques are described in this paper in connection with their underlying models.

Following this brief introduction, the description of existing image and video mining techniques starts in Section 2 with different mining paradigms, which are based on input-output data types and the computational nature of dominant mining tasks. Two types of input data sets for image mining are considered: original (iconic) image or symbolic (semantic) description of the image. Taxonomy for model-based mining techniques is proposed in Section 3. Section 4 gives a brief overview of image modeling methods for concise and content-based description of image collections or video sequences for the data mining purpose. Some examples of existing techniques for effective image and video mining are described in the context of their underlying models. Concluding remarks and future trends in the model-based image mining are given in Section 5.

2. PARADIGMS OF IMAGE AND VIDEO MINING

1.1 Image data paradigms

Similar to the general data representation in the multimedia domain, two basically different types of data are involved in the mining processes: original iconic image representation and its content-based, symbolic description. The iconic representation – a two-dimensional array of pixels – includes original images which are stored in image (video) databases in standard image file formats, compressed or non-compressed [5].

A general sub-division of mining tasks into two paradigms – called image data paradigms – can be made depending on which type of image data representation (iconic or symbolic) is used for input and output data in the mining techniques. The first and generally adopted image mining paradigm considers the input as a set of images. The given set of images usually is limited to a certain class of images, representing specific semantic entities \( \{e_1, \ldots, e_M\} \), i.e., semantic objects, scenes, concepts, events, structures or shots in the case of video images. The subject of image mining also includes the determination of possible relations \( \{r_1, \ldots, r_M\} \), between the semantic entities and the attributes for the entities and their relations.

Besides the input images, a certain symbolic description of the possible content can also be available. Several mining tasks, which involve the two different types of image data, can be requested for execution. Some of the common mining tasks taken from the table in Fig. 1 are listed below.

Content-based image retrieval. An image with a specific semantic meaning is at the input of a image mining system. The task is to find the most similar image from a database or all semantically equivalent images contained in this database. Both input and output image data are in the iconic form. This is an example of intelligent image retrieval [12, 42].

Mining of temporal structures in video. Temporal structures represent video sequences of images, which have the same semantic meaning or user’s interpretation. They include video shots (video sequence obtained within a continuous camera
movement or object tracking), entire scenes, particular object movements, etc. [28, 32, 48].

Pattern (object) recognition. This is one of the main tasks in computer vision, which may involve a database of reference images representing different semantic entities (objects) and their different views. The result of the recognition is the entity (or object class number) represented by the given image or the answer "no" if the database does not contain that entity at all. The face recognition task given a database of reference images is the typical example [12, 21].

Image model description. This task consists in transforming the iconic description of a given image into a symbolic one using a particular model structure and set of its parameters. This task is used in the most model-based method for image data mining.

<table>
<thead>
<tr>
<th>Single image</th>
<th>Multiple images or video</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICONIC</td>
<td>symbolic</td>
</tr>
<tr>
<td>Symbolic</td>
<td>Iconic</td>
</tr>
<tr>
<td></td>
<td>Image pre-processing</td>
</tr>
<tr>
<td></td>
<td>Knowledge extraction, Model learning, Parameter estimation</td>
</tr>
<tr>
<td></td>
<td>Change detection, Video surveillance, Video shot and scene detection, Video navigation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input data</th>
<th>Output data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single image</td>
<td>Multiple images or video</td>
</tr>
<tr>
<td>Iconic</td>
<td>Symbolic</td>
</tr>
<tr>
<td>Knowledge extraction, Model learning, Parameter estimation</td>
<td>Change detection, Video surveillance, Video shot and scene detection, Video navigation</td>
</tr>
</tbody>
</table>

Figure 1. Image mining paradigms and examples of common tasks involving two different types of input-output image data: iconic and symbolic.

Video navigation. This is an example of knowledgeable search tasks similar to the content-based image retrieval but applied to video databases. It presents a challenging problem since it involves much larger volumes of image data and complex spatio-temporal structures of semantic entities [28, 41].

When considering the second image data paradigm – the input is a symbolic description of an entire collection of images – the common task here is the image (video) retrieval based on its symbolic description. For example, the task can also be formulated as follows: find all images which contain a bus picture. An example from video search would be the automatic selection of a news fragment from a given TV program.

Another common task is the estimation of image model parameters, model selection and validation based on a sample of images. Here, we consider mostly parametric models, i.e. the model functional (symbolic) structure is known but the parameters have to be estimated using image samples. Different techniques – considered also as image mining methods – can be used for the parameter estimation depending on the model type.

1.2 Mining task paradigms

Another two major paradigms – called mining task paradigms – emerge in image and video mining from the viewpoint of mining tasks to be executed: knowledgeable search and knowledge extraction. The first mining task paradigm is mostly represented by content-based image retrieval methods [12, 34, 42]. Image data in the iconic form or its symbolic description are supplied to the mining engine and the task is to find the most similar images to a given one or to a given symbolic description (see table in Fig. 1).

The symbolic description might be in some cases just a feature or a set of simple features of different types, e.g. color, texture, shape. It can be a verbal description such as an annotation in the form of a standard word or phrase in order to identify a particular semantic entity. The knowledge extraction paradigm considers as an input a specified sample of images – they all correspond to a specified semantic entity (set of entities) – and requires as an output a symbolic description of the image sample. The type of symbolic description depends on a given mining task and
represents semantic entities \(e_1, \ldots, e_N\) as well as relations \(r_1, \ldots, r_M\) between them.

An image model as its symbolic description may not involve directly semantic entities and relations between them since initial images are given in an iconic form. Here, image entities \(f_1, \ldots, f_N\) represent image features, feature vectors or image objects related to different image locations. For example, each image region, obtained after image segmentation, can be described by a feature vector [23, 46]. Components of the feature vectors are also called attributes, e.g., position, size, color, shape, etc. The possible relations \(r_1, \ldots, r_M\) between image features can be spatial relations between them on the image plane. For example, relative (neighborhood) positions of regions such as region adjacency can be selected as existing relations. Temporal image sequences and video data require additionally some descriptors of changes in time. In this case each image entity can be in different states corresponding to discrete time moments: \(f_1(t), \ldots, f_N(t)\). Alternatively, each entity can include attributes of changes such as object movement attributes.

The usefulness of different image modeling techniques depends on the level of image content representation. Models for low-level image description such as pixel-based or neighborhood-based representations have a limited use in image mining since they cannot convey the image semantics [15]. Since lower-level models are used in some image mining techniques, such as the image retrieval methods, the proposed taxonomy includes some of them. Besides, some methods of feature or object extraction in object-related structural models are implicitly based on such models [42]. They are considered here briefly as a computational basis for the methods of feature-based image mining.

Only higher-level models using image entities (objects) and relations between image entities are suitable to solve basic mining tasks. We call such models object-related structural models since

3. TAXONOMY OF MODEL-BASED IMAGE MINING TECHNIQUES

Since the image mining tasks are dealing with extremely redundant image data compared to their semantic meaning, the image modeling represents a natural way for the redundancy reduction. A consistent image model facilitates the development of an image or video mining method, its theoretical analysis and reveals ways for the performance improvement. In this context, consistent image modeling implies the fulfillment of three basic conditions: adequate approximation of real images (i.e., accuracy of description), conciseness (e.g., minimum parameters are involved), and generality of description (i.e., a sufficiently large category of images is represented).

An image model as its symbolic description may not involve directly semantic entities and relations between them since initial images are given in an iconic form. Here, image entities \(f_1, \ldots, f_N\) represent image features, feature vectors or image objects related to different image locations. For example, each image region, obtained after image segmentation, can be described by a feature vector [23, 46]. Components of the feature vectors are also called attributes, e.g., position, size, color, shape, etc. The possible relations \(r_1, \ldots, r_M\) between image features can be spatial relations between them on the image plane. For example, relative (neighborhood) positions of regions such as region adjacency can be selected as existing relations. Temporal image sequences and video data require additionally some descriptors of changes in time. In this case each image entity can be in different states corresponding to discrete time moments: \(f_1(t), \ldots, f_N(t)\). Alternatively, each entity can include attributes of changes such as object movement attributes.

The usefulness of different image modeling techniques depends on the level of image content representation. Models for low-level image description such as pixel-based or neighborhood-based representations have a limited use in image mining since they cannot convey the image semantics [15]. Since lower-level models are used in some image mining techniques, such as the image retrieval methods, the proposed taxonomy includes some of them. Besides, some methods of feature or object extraction in object-related structural models are implicitly based on such models [42]. They are considered here briefly as a computational basis for the methods of feature-based image mining.
they represent certain structures of objects of interest and their spatial and temporal relations.

The proposed taxonomy of model-based image and video mining techniques is shown in Fig. 3. It also includes some simple models, which are currently used to perform content-based image retrieval. This is included in order to distinguish real content-based mining from simple image feature-based techniques used at the early stage [3, 20, 42].

The block of clustering, neural networks and statistical learning methods in Fig. 3 can be classified as belonging to both two major groups since they can perform some general classification of semantic entities \(e_1, \ldots, e_n\). Only particular instances of mining methods and respective models are listed in Fig. 3 at the terminal vertices of this taxonomic network due to the multitude of existing techniques and their modifications.

A large group of model-based mining methods if not the majority rely on parametric models. It will be useful to delineate some common computational steps in these techniques since Fig. 2 explains only the general case. First, an image feature set is extracted in order to represent concisely and invariantly (to some extent, translation, rotation, or scaling) images with respect to the mining task at hand (Fig. 4). The features or feature vectors are considered as image entities \(f_1, \ldots, f_M\) to be determined. Feature-based techniques for image mining can be successfully used at this stage [21, 36, 42]. A set of possible image models or model structures has also to be identified at this stage. Second, the model parameters are estimated based on the selected model structure using an optimization criterion. Third, the optimal model structure – functional or symbolic part of the parametric model – has to be determined. It also involves relations between image entities \(f_1, \ldots, f_M\). A common practice in the existing methods is to determine \(Q\) possible model structures and to select the most appropriate one among them. Often, these two consecutive steps can be united into one by considering the same criterion to select the model and to estimate its parameters at the same time [3, 33, 47]. An illustrative example of using the flowchart in Fig. 4 for model structure selection and parameter estimation in hidden Markov models is given in Section 4.2.

4. CONTENT-BASED MODELING OF IMAGE AND VIDEO DATA

4.1 Models and methods of feature-based mining

Feature-based mining dwells on extracting different image features – global or local image intensity characteristics – from a collection of images in order to make decision about the presence of certain semantic entities or their attributes in the given images. The decision is usually made by computing a distance between the feature vectors and comparing it with a threshold. Most content-based image retrieval methods at early stage of their development are in fact feature-based image mining [14, 20, 42]. The semantic gap is persistent in these methods since their underlying image models are not semantically oriented. However, some of these models can be generalized and used as low level sub-models in the semantics-oriented mining using object-related structural models. An example of using a global feature model is the popular method of histogram-based image retrieval [14, 42].

Two types of underlying image models can be well distinguished in the local feature approach: stochastic and transformational. The stochastic modeling approach belongs to classical stochastic methods of model-based image analysis. The most important example is the Markov Random Field Model (MRFM), which takes into account statistical inter-pixel relationships in a neighborhood of each pixel [8]. Autoregressive models of different types can also be classified as random field models, which are modeling the neighboring pixel correlations as linear dependence of pixels in a neighborhood [8, 14]. Similarly, polynomial regression models can be used as a basis for feature extraction [26].

The transformational approach includes new fast methods in signal processing and image compression, such as the wavelet decomposition [12, 13, 42]. The image intensity function in the transformational approach is decomposed over a system of basis functions (wavelets), and a subset of decomposition coefficients is selected to describe the image properties. The advantage of this approach over other models used in features-based methods consists in a multi-scale image analysis and concise representation of image intensity function. However, it is still redundant and is not a semantically oriented modeling.

Although salient feature approach is not implicitly based on an image model, it represents a promising alternative and a tradeoff between the image representation adequacy and its description length [27, 38, 46]. In this approach, an image is described by a set of local features (objects) at salient locations – selected image fragments with local characteristics stable to viewing transformations and intensity changes – that can provide a concise and invariant description by a small number of local features. Salient features are currently used only in the knowledge-based search paradigm.

4.2 Hidden Markov models and Bayesian networks

The Hidden Markov Models (HMM) is a wide class of stochastic image modeling, which can be efficiently used in multimedia mining in general and in video mining in particular [22, 32, 48]. A solid motivation for video mining is that they involve the notion of state of the model usually connected with discrete time moments. They are very popular in such multimedia area as speech and music analysis and were generalized to solve some common video processing tasks. Application of HMMs to still image analysis have also shown promising results [16, 17, 34]. Their major
application tasks in video mining are: video surveillance, mining of
temporal structures, shot and scene detection in movies, video
annotation, etc.

A conventional single-level HMM is well suitable to represent
events in image sequences only on the feature level [11, 44].
Specification of a first order HMM involves the following
notions: states \( S = \{S_1, \ldots, S_N\} \) with the state transition
probabilities \( a_{ij} = P(q_{t+1} = S_j | q_t = S_i) \) as elements of the matrix \( A \), \( 1 \leq i, j \leq N \); observations \( O = \{O_1, \ldots, O_K\} \) with their probability densities
\( b_j(O) \) and the initial state probability distribution, \( \pi_i = P(q_1 = S_i), \ 1 \leq i \leq N \). Assuming Gaussian distribution laws for the
observations, this model contains a parameter set,

\[
\lambda = [\pi, A, \{\mu_j, \Sigma_j\}, 1 \leq j \leq L],
\]
to be estimated from image data, where \( \mu_j, \Sigma_j \) are the mean and
covariance of the Gaussian at state \( j \). Parameter estimation of a
HMM given \( L \) independent sequences \( \{O^{(l)}, 1 \leq l \leq L\} \) represents
itself an image mining problem since it takes as an input a video
sequence and outputs parameter values (see examples of mining
tasks in Fig. 1). An iterative statistical procedure, called EM
algorithm (Expectation-Maximization) is most often used in
practice to solve the parameter estimation problem in video mining
tasks [29].

![Figure 3. Taxonomy of image mining techniques and image models, which constitute the conceptual basis for the corresponding
techniques, with some instances of mining methods and models listed at the terminal vertices.](image)

48
In order to represent in a better way semantic structures – changes, events or object movements – in video sequences, some modifications of HMMs such as coupled HMM (CHMM) and hierarchical HMM (HHMM) have been developed. For example, a model with two coupled HMMs (by their state sets \( O_1 \) and \( O_2 \)) was successfully used for modeling human interactions in a video surveillance task [32].

Recently, a two-level HHMM was proposed for mining statistical temporal structures in video [48]. In this model, the higher-level structure elements usually correspond to semantic events \( \{q_1,\ldots,q_N\} \), while the lower-level states \( \{q_1,\ldots,q_N\} \), represent variations which can occur within the same event. The lower-level states in turn produce the observation \( O = \{O_1,\ldots,O_k\} \), i.e. computation of feature values, for example, from the raw video data.

In the context of the flowchart in Fig. 4, a modified Bayesian information criterion (BIC) was used to solve the problem of model selection in the method of HHMM [48]. This criterion is a measure of a posteriori model fitness to the training image sample corresponding to a given semantic entity [11]. The EM algorithm was used to estimate the HHMM model parameters. In this method, the feature pool contains different image features (image entities) as well as audio features. The same approach (normalized BIC criterion) was used to perform joint selection of features – feature subset and particular model. Additional use of audio features contributed to better performance when using only low-level image features. The HHMM model was tested on sports video to identify automatically different events (e.g., “break” and “play”) of soccer clips, and has shown promising results.

Bayesian networks are considered as powerful models for knowledge representation and inference in the form of network relations between entities [16, 33]. A Bayesian network is described by a directed acyclic graph where vertices correspond to random variables and the presence (or absence) of arcs indicates the dependency between the variables. An obvious advantage of this kind of models is the existence of explicit dependencies – such as cause and consequence – between the network variables, which allow semantic entities to be included explicitly in the model as well as direct knowledge inference by the Bayes’ rule.

Dynamic Bayesian networks (DBN) can be viewed as a generalization to HMMs and can be effectively used to represent temporal events in video mining methods [16]. HMMs and Bayesian networks are sometimes called graphical models.

### 4.3 Attributed graph models

In its initial form, an attributed graph model is a deterministic structural model for image content representation which allows us to describe explicitly not only image objects (entities) but also relations between them. A general graph model can be represented as an Attributed Relational Graph (ARG) [4, 9, 14]. An image content model combining object attributes along with their mutual binary relationships is represented as a complete ARG, where spatial entities and objects are represented by a set of graph vertices \( V \) each labeled with an attribute \( a \) and binary spatial relations represented as pairs of vertices \((V\times V)\) each labeled with a spatial descriptor \( r \):

\[
\text{image model } = <V, E, a, r> ,
\]

where \( V \) is the set of spatial entities; \( E \) is the set of graph edges, \( E: V \times V \rightarrow \{0;1\} \); \( a \) is the entity attribute, \( a: V \rightarrow \mathbb{A} \), \( r \) is the relation attribute \( r: V \times V \rightarrow \mathbb{R} \), where \( A \) and \( R \) are real number spaces.

The Region Adjacency Graph (RAG) model can be considered as a partial example of the ARG where the set \( V \) is composed of image homogeneous regions with respect to some local features and the set of graph edges represent the established adjacency relations between the regions [23, 47]. In the case of a RAG model, the attribute spaces \( A \) and \( R \) can be a region intensity feature space and relative position space, respectively.

![Figure 4: Basic computational steps in the model-based symbolic description of images.](image-url)

Images, which have ARGs similar to that of the query image, are selected from a given image database (Fig. 2). In the case of a classification mining task involving many classes, images of the same semantic class as a learning sample are supplied to the image mining system and a largest common sub-graph contained in all the images in that class is determined. Another option is to compute a median graph and describe semantic objects of a particular class by...
the median graph [19]. Finding the median graph is analogous to the problem of image clustering when image data are given in the form of feature vectors (see Section 4.4 for details).

For example, images are described as graphs of blob-like regions in the blob-based image description for executing image retrieval [23, 47]. Here, the images entities at the graph vertices are blobs – image regions with round or elliptic shapes extracted using an image smoothing operator combined with a thresholding.

4.4 Neural networks and statistical learning

Although these two types of methods look different, they have a common clustering nature and probabilistic foundation such as the statistical learning [46]. On the one hand, neural network operations are based on discriminating clusters of global or local image features and hence can be classified as feature-based mining. On the other hand, they allow us to perform some general classification of semantic entities \( \{e_1, \ldots, e_N\} \), and can be considered as a semantics-oriented approach to image mining. Besides, the statistical learning techniques are often used for parameter estimation in the semantics-oriented mining.

The common ground for both kinds of methods is clustering of image data into \( N \) clusters for dimensionality reduction and the possibility of image classification with semantic meaning. It is supposed that each image is represented as a vector of \( K \) image features, \( \{f_1, \ldots, f_K\} \) in \( K \) dimensional feature space, which is partitioned into \( N \) clusters representing respectively \( N \) entities.

Although clustering and neural networks are referred to as non-parametric methods, they can be considered as parametric models. The position of each cluster (mean value) and its covariance matrix (both inter-cluster and intra-cluster) can be viewed as the model parameters to be determined. In this viewpoint, the model structure is merely the \( a \ priori \) chosen number of the entities clusters. Additionally, if a distribution for each cluster is selected (e.g., a Gaussian function) then this determines the model structure besides the number of clusters. One common type of neural networks – called Self-Organizing Maps (SOM) – are currently used for content-based image indexing and retrieval [12, 21]. Another kind of networks are linear multi-layer perceptrons, which dwell on constructing hyper-planes dividing the feature clusters. In this case, the network weights \( \{w_1, \ldots, w_K\} \) represent model parameters to be learned. A newly developed learning method called Support Vector Machines (SVM) can also be considered as a statistical learning model for image data clustering and was successfully used in video mining [14, 37].

4.5 Association rule mining

This data mining technique can be classified as belonging to the group of statistical learning and neural networks because of using statistical principles. The association rule mining methods are very common in database management and other non-visual data types in the multimedia domain. The association rules establish relations between mining entities based on a training sample of images with labeled semantic entities. Relatively recently it has been investigated for image mining in two different paradigms: mining from a collection of images alone and combined collection of images and associated symbolic descriptions [10, 25, 43]. There are different types of rules to be mined in a representative sample of images or videos: associations \( (e_i \rightarrow e_j) \) rules, characterization \( (e_i \rightarrow f) \) rules, and classification \( (f_i \rightarrow e_j) \) rules, where \( e_i \) is the \( i \)-th image feature (or set of features) and \( f_i \) is the \( i \)-th semantic entity (or set of entities), and sign “\( \rightarrow \)” denotes the logical implication.

Here, the semantic entities represent classes of objects of interest with possibly associated relations. For example, the following task of the association rule mining was investigated: find from satellite images some common patterns (associations of entities) with respect to different areas which are present in given satellite images of big cities [25, 50].

Another interesting example of image mining in medical imaging using classification rules considers associations between image features from a collection of images and their semantic meaning. A common task in functional NMR (nuclear magnetic resonance) imaging is to find brain areas described by features \( \{f_1, \ldots, f_K\} \) which are responsible or involved in the specified functions of human neural system [30]. Here, the semantic entities, \( \{e_1, \ldots, e_N\} \), correspond to the investigated neural functions and activities. Association rules – as compared with HMMs and Bayesian networks – can be considered to be a simple model for image data although more complete and theoretically-founded models can be used for effective knowledge extraction. One such example, which permits data clustering and association rule mining is formal concept analysis [45].

5. CONCLUSION

State-of-the-art in image and video mining clearly indicates that solutions of dominant mining tasks are non-trivial and computationally complex procedures, and they are in confrontation with the increased demand to multimedia databases and applications.

In this paper, we have presented a brief overview of major tasks and model-based approaches to effective solutions of image and video mining problems. Different paradigms of mining tasks have been analyzed from the viewpoint of image data types. The taxonomy of image mining techniques – methods for knowledge extraction and intelligent search in collections of still images and videos – is made when considering input-output image data and dominant mining tasks. Theoretical analysis and reported experimental results indicate significant advantage of model-based techniques over conventional feature-based methods in image mining. In particular, methods based on object-related structural models have obvious advantages in terms of semantics since they rely on symbolic image representation close to the image semantics. The models with spatial and temporal relations
between entities (objects) as well as hierarchical models allow the corresponding methods to take into account local context and temporal changes. At the same time they are more effective to achieve the description conciseness and to reduce computational expenses during the implementation.

Model-based mining has been developing dynamically during the past decade. As a future trend in effective image and video mining, the use of new Bayesian models and attributed graph models is gaining more ground. Application of robust and transformation-invariant statistical procedures for parameter estimation and model structure learning in the model-based approaches looks very promising to increase their effectiveness. Reduction of computational expenses in model parameter estimation is a near-future challenge in expanding the application scope of such sophisticated statistical procedures.

6. ACKNOWLEDGMENTS

We are grateful to VRQ (Valorisation Recherche Québec) and Canadian Heritage for their financial support to CoRIMedia (Consortium de Recherche en Image et Multimédia).

7. REFERENCES


ABSTRACT

Multiple Sensor Indoor Surveillance (MSIS) is a research project at Accenture Technology Labs aimed at exploring a variety of redundant sensors in a networked environment where each sensor is giving noisy information and the goal is to coherently reason about some aspect of the environment. We describe the objectives of the project, the problems it was designed to solve and some recent results. The environment includes 32 web cameras, an infrared badge ID system, a PTZ camera, and a fingerprint reader. We discuss two concrete problems that we have tackled in this project: (1) Visualizing events detected by 32 cameras during 24 hours, and (2) Localizing people using fusion of multiple streams of noisy sensory data with the contextual and domain knowledge that is provided by both the physical constraints imposed by the local environment and by the people that are involved in the surveillance tasks. We use Self-Organizing Maps to approach the first problem and suggest a Bayesian framework for the second one. The experimental data are provided and discussed.

Categories and Subject Descriptors
1.2.6 [Artificial Intelligence]: Learning: I.5.2 [Pattern Recognition]: Design Methodology - classifier design and evaluation: I.5.5 [Pattern Recognition]: Implementation - special architectures.

General Terms
Algorithms, Experimentation.

Keywords
Multi-camera surveillance, indoor surveillance, unsupervised learning, self-organizing maps, visualization, rare event detection, Bayesian inference, people localization.

1. INTRODUCTION

The proliferation of a wide variety of sensors (video cameras, microphones, infrared badges, RFID tags, etc.) in public places such as airports, train stations, streets, parking lots, hospitals, governmental buildings, shopping malls, and homes has created the infrastructure that allows the development of security and business applications. Surveillance for threat detection, monitoring sensitive areas to detect unusual events, tracking customers in retail stores, controlling and monitoring movements of assets, and monitoring elderly and sick people at home are just some of the applications that require the ability to automatically detect, recognize and track people and other objects by analyzing multiple streams of often unreliable and poorly synchronized sensory data. A scalable and robust system built for this class of tasks should also be able to integrate this sensory data with contextual information and domain knowledge provided by both the humans as well as the physical environment to maintain a coherent and logical picture of the world over time. While video surveillance has been in use for decades, systems that can automatically detect and track people (or objects) in multiple locations using multiple streams of heterogeneous and noisy sensory data is still a great challenge and an active research area.

Since the performance of these automatic systems is not at the level at which they can work autonomously, there are a lot of human experts who are still part of the loop. It is important to develop techniques that can help human experts in this task by organizing and presenting the video surveillance data in a summarized manner, and highlighting unusual or rare events for further research by the experts. Many approaches have been proposed for object tracking [1-3] and event summarization [4, 5] in recent years. They differ in various aspects such as number of cameras used, type of cameras and their speed and resolution, type of environment (indoors or outdoors), area covered (a room or a hall, a hallway, several connected rooms, a parking lot, a highway, etc.), and location of cameras (with or without overlapping fields of view). However, the performance of most systems is still far from what is required for real-world applications. The objective of our research is to bridge the gap between the needs of practical applications and the performance of current surveillance algorithms. We seek solutions in the following directions:

- Developing a framework for logical integration of noisy sensory data from multiple heterogeneous sensory sources that combines probabilistic and knowledge-based approaches. The probabilistic part is used for object identification and tracking and the knowledge-based part is used for maintaining overall coherence of reasoning.
3. EVENT CLUSTERING AND VISUALIZATION
The described below research was inspired by the following practical problem: how to automatically summarize and visualize 24 hour video data captured by 32 cameras within a connected environment?

3.1 Data preprocessing
Our raw data are time-stamped 640 by 480 pixels JPEG images that are captured by AXIS-2100 webcams at the rate 2 - 6 Hz. A background subtraction algorithm is applied to each image to extract foreground pixels. For background modeling the system uses one dynamically selected image as the background or generates it by finding median for each pixel of a pool of images taken during some period of time. Then the following features are extracted from the foreground pixels of the image:

1. Motion features that characterize the foreground pixels’ distribution, which is calculated on an 8 by 8 grid (64 values). The value for each cell is the number of foreground pixels inside the cell divided by the cell’s area.

2. Color histogram of the foreground pixels in RGB color space (3^8*8=24 values).

This data is integrated by tick and by event. We set a tick to be a one second time interval. Tick allows us to loose synchronize the data from different cameras and other sensors and regularize frames taken with varying sampling rates. The data integration by tick and event consists of averaging both motion and color data. For visual representation of a tick a “summary” frame is created. It accumulates all foreground pixels from all images from the tick into one image. The summary frames serve as key frames for representing ticks. An event is representing visually as a sequence of tick summary frames.

For estimating boundaries of events, we used the percentage of foreground pixels in the image as an integral measure of motion ($F$-measure). If we plot $F$-measure for different cameras we would see different patterns depending on what location a camera observes. A camera can watch a hallway, a meeting room, a working space (cubicles or laboratories), a recreational area such as a coffee room, or a multi-purpose area that is used differently at different times of the day. For example, if a camera is observing a hallway then most events are people walking along the hallway, entering and leaving offices, or standing and talking to each other. In this case the $F$-plot is a number of peaks that represent short events and some trapezoidal “bumps” that correspond to longer events. Figure 1a presents a typical $F$-plot for a hallway camera.

If the camera is observing a meeting room then we have long idle periods when the room is empty that are interrupted by periods when meetings are in process. The latter have trapezoidal $F$-plots with a long base and some volatility that corresponds to movement during the meeting and some small peaks that correspond to events when participants arrive and leave the room. Figure 1b shows a sample $F$-plot for a meeting room camera.

For a recreational area camera, the events are typically longer than those for a hallway but much shorter in duration than for a
meeting room camera. They correspond to events when people enter into the area to drink coffee, have lunch, reading or to talk to each other.

If a camera watches busy working areas such as cubicles then we have events like people arriving and leaving their workplaces, sitting down and standing up, moving and communicating with each other. The F-plot for such a camera rarely goes to zero during working hours and looks like a long volatile meeting (Figure 1c). Having watched F-plots for several cameras we came to the conclusion that, first, we can use F-measure for event boundary estimation, and, second, the notion of event is a relative one. These observations encouraged us to use the wavelet decomposition of F-signal to detect events at different levels of details. We used the wavelet decomposition of levels from 2 to 5 with Haar wavelet for calculating approximation and details. Then we applied a threshold to the approximation signal that slightly exceeds the noise level to find the boundaries of major events such as meetings (mega-events), and another threshold to the absolute value of highest detail signal to find internal events (micro-events). Figure 2 presents the results of event detection for a signal of length 1024 seconds using wavelet decomposition of level 3. The top plot presents the original F-signal; the middle plot is its level 3 approximation, and the bottom plot presents its level 3 details. The boundaries of macro and micro events presented in the lower parts of the middle and bottom plots correspondingly. The boundaries of micro events are used for data integration.

After integrating the data by tick and by event, we have two sets of data for each camera. We applied the self-organizing map (SOM) approach [6] to tick/event data for clustering and visualization and use 2D rectangular maps with hexagonal elements and Gaussian neighborhood kernels.

![Figure 1. Foreground pixels patterns for different camera locations.](image)

![Figure 2. Event boundaries detection using wavelets.](image)
3.2 Clustering using SOM

In one-level clustering approach we use both motion and color features to build the map. For creating maps we used the SOM Toolbox for MATLAB developed at the Helsinki University of Technology [7]. The toolbox allows maps to be displayed in many ways by varying the units’ sizes and colors. The size of maps for tick data can reach 1400 units (70 by 20) and for event data – about 200 units (25 by 8). Figure 3 presents a snapshot of the visualization tool that displays the map for tick data of a camera that observes a multi-purpose area. It presents about 30,000 ticks on a 58 by 15 lattice (870 units). The size of the units reflects the number of ticks attracted by this unit (the number of hits) and ones with zero hits are not displayed. The colors show the topological similarity of the prototype vectors. This creates a visualization tool that allows exploration of the contents of each unit. However, there can be a large number of units which makes browsing very laborious. To reduce the number of units shown in the visualization, we apply the \( k \)-means algorithm for clustering map units (prototype vectors) [8]. As the number of units is about an order of magnitude smaller than the number of raw data points, we can run the \( k \)-means algorithm with different number of clusters, estimate the quality of partitioning using their Davies-Bouldin indices [9], and allow users browsing clusters of units. The Davies-Bouldin index of a partitioning \( P = (C_1, C_2, ... , C_L) \) is specified by the following formula (1).

\[
DBI(P) = \frac{1}{L} \sum_{i=1}^{L} \max_{j \neq i} \left\{ \frac{S(C_i) + S(C_j)}{D(C_i,C_j)} \right\}
\]

where \( C_i, i = 1, L \) are clusters, \( S(C) = \frac{1}{N} \sum_{i=1}^{N} \left\| x_i - c \right\| \) is the within-cluster distance of the cluster \( C \), which has elements \( x_i, k = 1, N \) and the centroid \( c = \frac{1}{N} \sum_{k=1}^{N} x_k \), and \( D(C_i,C_j) = \left\| c_i - c_j \right\| \) is the distance between clusters’ centroids (between-cluster distance).

We do \( k \)-means clustering of units for the number of clusters from 2 to 15 for event data and from 2 to 20 for tick data. When the number of clusters is large, some clusters can consist of several transitional units that have no raw data (events or ticks) associated with them. The visualization tool allows browsing both ticks and events by units and by clusters.

3.3 Two-level Clustering using SOM

In two-level clustering we explore the motion and color features sequentially. On the top level only motion features are used to create the main map. Then we use \( k \)-means clustering for the map units as we described above. Then, for each cluster we build a SOM map using color features only. Such separation of features allows better differentiation among spatial events and easier detection of unusual events. In indoor environments, where most of the moving objects are people, who usually change their clothes every day, the variance in color features is higher than the variance in the motion features. Separating motion features allows collecting them over longer periods and creating more robust classifiers. To create a classifier, we accumulated motion data for a week, built a SOM map and applied \( k \)-means clustering to its units. Each cluster is a set of weighted Gaussian kernels, which can be considered a Gaussian Mixture Model (GMM). A new piece of data can be classified using the GMM associated with each cluster and assigning the new data to the cluster with maximal probability. The same procedure can be applied to color features. Combining motion based classifiers at the top level with color based classifiers at the bottom level, we obtain a hierarchical classifier.

3.4 Finding Rare Events

For many applications, finding unusual events is of special interest. What is often difficult is determining what constitutes an unusual event since it often depends on our subjective interpretation. An event can be unusual because it happened at an unusual time or an unusual place or had unusual appearance. We can all agree that an unusual event is a rare event at a given time and space point but a rare event may not be an unusual one. For example, a person sitting in his office on weekend could be a rare but not a highly surprising event.

After thoughtful deliberation we decided to use techniques for finding rare and frequent events and then passing them onto humans to decide how unusual or usual they are. In our research we distinguish between local rare/frequent events – these are events that happened during one day – and global rare/frequent events – those that happened during longer periods of time. We also distinguish between events that happened during regular working hours and outside of them.

For visualizing local rare events we use a 3D surface that shows distances between units of the SOM map and indicates how many data points (hits) belong to each unit using markers whose size is proportional to the number of hits. A user can rotate the axes searching for “highlanders” – small unit markers that are located on the top of peaks or for “isolated villages” which are small sets of small unit markers that are located in closed mountain valleys.

For detecting global rare events the following procedure is proposed. First, the GMM classifier is applied to a new event/tick motion data. If it gives a high probability for a small cluster or for a cluster with zero data points then the system declares that a rare event of particular type is found. If the classifier gets low probabilities for all clusters then the system indicates that it is a new (and rare) spatial event. Such events are accumulated and can be used for building a new version of GMM classifier. In case when the event belongs to a moderate or frequent event cluster, the system applies the corresponding color-based GMM classifier to detect rare or new events using their color features.
3.5 Event Visualization Tool

The techniques described above have been integrated into an event visualization tool. Figure 3 shows a snapshot of the tool. The tool’s GUI consists of four panels – Data, Map, Image Browser and Unit/Cluster Summary. Using the Data panel the user selects and loads data. Currently each camera has a separate file for its tick and event data. The file contains preprocessed raw data and SOM data for chosen SOM architecture – one or two-level SOM maps. The user selects the desired map from a menu which is then displayed in the Map panel or in a separate window.

The Map panel displays the current SOM map, which could be the top level map that shows color coded units with unit sizes reflecting the number of hits, or a cluster map of different number of clusters with color coded units. The 3D surface that presents distances between units and the number of hits in each unit is displayed in a separate window. The user can explore this 3D surface to find local rare events.

When the user clicks on a unit or a cluster of units on the SOM map the contents of the unit or cluster is displayed in the Unit/Cluster Summary panel. This panel presents information about the current item (unit or cluster). It shows the number of frames in the current item, the name of image file displayed in the Image Browser panel and its time. It also has two plots. The left plot shows the distribution of original data in time. The right plot shows all data (ticks or events) that related to the current item. A small square corresponds to each piece of data. The color of the square indicates the time interval that the piece of data belongs to. When the user clicks on a square the corresponding summary frame is displayed in the Image Browser panel.

The Image Browser panel displays the visual information related to the current selected tick or event in the Unit/Cluster Summary panel. Using the browser’s control buttons the user can watch the first or last frame of the tick/event, go through the tick/event frame by frame forward and backward, and watch a slide show going in both directions. The speed of the slide show is controlled by the speed slider. Clicking on the image brings the current frame in full size into a separate window allowing the user to see details. This feature proved to be very useful for exploring busy summary images.

4. BAYESIAN FRAMEWORK FOR PEOPLE LOCALIZATION

Visualizing the information being captured by the cameras in the environment is just one aspect of our work. Another important problem that comes up in many applications (security, surveillance, retail environments, etc.) is that of people/object localization. This section presents a Bayesian framework for people localization which allows the integration of evidence of multiple sensor sources. Our task is to localize and track $N$ objects...
in a space of known geometry with stationary sensors of different kinds. The number of objects may change over time. The sensing zones for some sensors can overlap. The number of objects can change dynamically when an object arrives or leaves. We assume that there are two types of objects: known objects (employees) and unknown objects (guests or customers). The space is divided into "locations". Time is sampled into ticks. The tick duration is selected depending on the sampling frequencies of the sensors. It should be large enough to serve as a synchronization unit and small enough so that objects either stay in the same location or only transition to an adjacent one within a tick.

4.1 Sensor Streams
Each object is represented by a set of features extracted from sensor streams. We are currently using four sources of evidence.

4.1.1 Video Cameras
This is very rich data source, but requires a lot of sophisticated processing to solve problems such as background modeling, object tracking, occlusion resolution, and object recognition to extract useful information. Our system is mostly based on this source. We are using two approaches for people localization – people appearance modeling and face recognition. People appearance modeling is based on geometric (height, width) and color features. An object can have several color models – one or more for each location or even for the time of the day. Object models can be defined (through training) prior to the surveillance task or accumulated incrementally during the task. Appearance modeling works for all cameras, whereas face recognition is efficient only for some cameras where size and orientation of faces are appropriate. We use a dedicated PTZ camera that watches the main entrance to the floor for face recognition. The face recognition system uses the OpenCV algorithm [10] and tries to recognize people from a restricted list.

4.1.2 Infra-Red Badge ID System
The second source of evidence is the infra-red (IR) badge system. The system collects data from 72 readers and merges them into a database that indicates where a particular badge was sensed the last time. This source of information is not very reliable because:

1. The badge has to be in the line of sight of a reader on the ceiling. If a person puts his/her badge into a pocket, it will not be detected.

2. The orientation of the badge affects the detection. A person may be standing under an IR-reader but his badge could trigger another reader nearby depending on the orientation of the badge.

3. A person can leave his/her badge in the office or give it to another person.

4. Detection records are written to the database with a delay creating a discrepancy among different sources of evidence for fast moving objects.

4.1.3 Finger Print Reader
The third source of evidence is the fingerprint reader. This is a very reliable source, but located only at the main entrance, has a restricted number of registered users, and a person only uses it 1-2 times per day for check-in. We mostly use it for acquisition or updating of person appearance models as a person checks-in when entering the office in the morning.

4.1.4 Human Intervention
The fourth source of evidence is human intervention. People who participate in a surveillance task can interactively influence the system. They can mark an object in a camera view and associate it with a particular person which causes the system to set the probability of the person being at this location to 1 and recalculate the previous decisions by tracking the person back in time. This is a very reliable, but costly information source. We use it mostly for initializing and updating person appearance models.

4.2 Identification and Tracking of Objects
The current state of the world is specified by a probability distribution of objects being at particular locations at each time tick. Let us assume that \( P(H_i|L_j) \) are probabilities to find the object \( H_i \) at location \( L_j \). The initial (prior) distribution can be learned from data or assumed to be uniform. Each object has a set of models that are location and sensor specific. Each object has a matrix of transition probabilities \( T(H_i|j) \) that is learned from training data.

The process of identification and tracking of objects consists of the following steps:

Step 1. Data Collection and Feature Extraction
Collect data from all sensors related to the same time tick. Select data that contains information about a new "event" and extract features.

Step 2. Object Unification from Multiple Sensors
Each sensor detects "reflections" of one or more objects in its sensory field. The reflections that come from the same object are merged based on their location and sensory attributes. This gives us a unified model of how different sensors "see" the same entity. For video cameras, the blobs are first mapped into locations based on their geometric features and calibration data from the cameras. Then the blobs from different cameras that belong to the same location are assigned to the same entity based on their geometric and color features. For IR badge data, which consists of binary indicators of a badge being detected at a particular location, the system first spreads the probability to the adjacent IR locations taking into account the space geometry, and then maps IR locations into camera based locations and associates evidence with entities. The result is a set of entities \( O = \{O_r\} \) and a matrix \( W = \{w_{qk}\} k=1,K, r=1,M_r \) where \( M_r \) is the number of entities. Each \( w_{qk} \) is the membership value of \( r \)-th entity to belong to the \( k \)-th location.

Step 3. Posterior Probability Estimation
We estimate the conditional probabilities for each object being in each location during each time tick. Using the features that belong to the same entity and the object models, the conditional probability that the entity represents an object at a given location is estimated for all entities, objects and locations. The result is a sequence of probabilities \( S_q = \{Pr(z_{q1},z_{q2},...,z_{qk})\} \) associated with the entity \( O_r \), \( r=1,M_r \). Here \( R_q \) are the feature data extracted from representations of entity \( O_r \) and \( C_{oq} \) are sensors. For video cameras, the probabilities that a blob represents an object...
(person) for given cameras and locations are calculated using blob’s features and persons’ models, which are camera and location specific. Two approaches have been used for estimating conditional probabilities for camera sensors. The first one uses the distance between color histograms in the RGB space for estimating similarity to the appearance models and converts this distance into probability using a threshold obtained from training data (see [11] for more information). The second approach uses Gaussian models in RGB space for upper and lower parts of human appearance and estimates probabilities of a blob (that is represented by a set of sampled points) to belong to the models. For IR badge data the distributed to adjacent locations probabilities are used as the conditional probabilities. The output of face recognition system is also used as conditional probabilities. The fingerprint and human intervention evidence sets up the prior probabilities directly. The difference between video sensors and the other sensors is that the data from video cameras are used for every tick, but the data from the other sensors are used only when they are available. The conditional probabilities of blobs that are views of the same entity from different sensors are used for estimating posterior probabilities of an object being represented by the entity at the location using Bayes rule (2).

\[
P(H_j | O_i, L_k) = \frac{P(H_j | L_k) \cdot w_{i_k} \cdot \prod_{p(B_i_j, C_p) \in S} P(R_{B_i_j}, L_k, C_p | H_j)}{P(O_i)}
\]

where \( P(O_i) = \sum_{j=1}^{N} P(H_j | L_k) \cdot w_{i_k} \cdot \prod_{p(B_i_j, C_p) \in S} P(R_{B_i_j}, L_k, C_p | H_j) \).

Then the probabilities for entities are merged to obtain the probability \( P(H_j | L_k) \) for the object \( H_j \) being observed in location \( L_k \) using formula (3).

\[
\tilde{P}(H_j | L_k) = \frac{\sum_{j=1}^{M} P(H_j | R_{j}, L_k)}{\sum_{j=1}^{M} \sum_{i=1}^{N} P(H_j | R_{j}, L_k)}
\]

### Step 4. Update Probabilities using Transition Matrices

The locations are selected in a way that an object can either stay in the same location or move to an adjacent location during any single time tick. The specific transition probabilities among locations for a known object or generalized transition probabilities for the other objects are estimated from historical data or provided as prior knowledge by the people involved in the task. These probabilities are taken into account for re-estimating prior probabilities.

\[
P(H_j | L_k) = \frac{\left[ \sum_{j=1}^{L} P(H_j | L_k) \cdot t_{e_j}(H_j) \right] \cdot \tilde{P}(H_j | L_k)}{\sum_{j=1}^{L} \left[ \sum_{j=1}^{L} P(H_j | L_k) \cdot t_{e_j}(H_j) \right] \cdot \tilde{P}(H_j | L_k)}
\]

### Step 5. Re-Estimation

The steps 1-4 repeat for each time tick.

The sensors we consider are fixed and stationary. Knowing where the sensors are located enables us to define areas which the system should pay extra attention to as well as regions the system should ignore. For example, areas, such as working places in cubicles, armchairs in halls, doors, passages, entrances where new objects appear and disappear, etc. are more important to watch than blinking computer or TV monitors, lamps, reflective surfaces, etc. Using a layout for each camera that marks all important/unimportant areas and assigning semantics to them increases the accuracy of object localization. The inclusion of this domain knowledge is done through a Bayesian probabilistic framework by assigning prior probabilities, and fits naturally in the probabilistic framework that was described above.

The classical Bayesian approach assumes that (1) there are a constant number of mutually exclusive hypotheses, and (2) the hypotheses cover the whole decision space. In our case the situation is more dynamic – people may enter and leave the floor, and people may be “invisible” to the sensors, for example, a person has his IR badge covered and is standing in a “dead zone”, which is not observed by any camera. We extend the framework to cover these problems. The system uses two cameras that watch the elevator area and detects people who are entering or leaving the floor. If a person leaves the floor, his/her model is marked as “inactive”. If a person enters the floor, a new object and its appearance model is created and is marked as “new”. The system tracks a new object and creates models for it for other locations when it is possible (high probability of identifying the person, no occlusion, etc.). The process continues until enough data is collected.

In case when an object is temporarily invisible, the system marks it as “idle” and keeps its probability high to be in “hidden” locations that are near the location where the object has been identified last time. For example, Figure 4 shows two cameras that watch a hall. There are five observed locations (L1-L5) and three hidden locations (HL1, HL2, HL3). If an object has been seen last time at location L7 then there is high probability for the object to be in the location HL1, but the probability to be in locations HL2 and HL3 are also far from being zero and can increase with time.

For our pilot experiment we used eight video cameras integrated into four clusters. Four people served as volunteers to be localized in the experiment. Every day up to fifty people were working on the floor. The color features of most of them have been used for building the “unknown person” model. To evaluate the accuracy, we recorded four days of video data, eight hours per day. Data from one of the days was used for prior probability estimations, and the rest for testing. We used precision (P) and recall (R) as measures for evaluating the performance of the system: \( P = C / T \) and \( R = C / A \), where \( C \) is the number of events where people were correctly localized by the system, \( A \) is the number of events where people are actually visible (ground truth), and \( T \) is the number of events that the system claimed that a person was located in that location. Using only visual features, the system obtained average recall of 68.23% and precision of 59.16%. Using the domain knowledge and semantics of the sensors locations, the performance increased to average recall and precision of 87.21% and 73.55%, respectively.
5. SUMMARY

We described an approach to unsupervised classification and visualization of surveillance data captured by multiple cameras. The approach is based on self-organizing maps and enables us to efficiently locate for rare and frequent events. It also allows us to create robust classifiers to identify incoming events in real time. Experiments with the visualization tool prototype show that it is an efficient tool for humans to browse large quantities of video data that is being captured by a network of cameras.

We also describe a Bayesian framework that enables us to robustly reason from data collected from a network of various kinds of sensors. In most practical situations, sensors are producing streams of redundant, but noisy data. The probabilistic framework presented here gives us the ability to reason from this data by also incorporating the local semantics of the sensors as well as any domain knowledge that can be provided by people involved in these tasks. We believe that this framework is applicable in the larger context of creating robust and scalable systems that can reason and make inferences from different kinds of sensors that are present in the world today.

6. REFERENCES


Classify By Representative Or Associations (CBROA):
A Hybrid Approach for Image Classification

Vincent. S. Tseng  
Chon-Jei Lee  
Ja-Hwung Su  
Department of Computer Science and Information Engineering  
National Cheng Kung University, Tainan, Taiwan, R.O.C.  
Email: tsengsm@mail.ncku.edu.tw

ABSTRACT

Image classification has been an interesting research issue in multimedia content analysis due to the wide applications. In this paper, we observe that images can be classified (or annotated) in two ways: i) Classify by some main object, ii) Classify by multiple objects with their relations. These two types of images usually exist concurrently in real-life image databases. Although a number of image classification methods have been propose, they can only handle one certain type of images well and fail to deal with both types of images correctly at the same time. In this paper, we propose a hybrid image classification method, namely “CBROA” (Classify By Representative Or Associations), that can effectively classify both types of images at the same time. CBROA integrates the decision tree and association rules mining method in an adaptive manner with construction of a virtual semantic ontology. Experimental results show that CBROA outperforms other classification methods in terms of classification accuracy in classifying mixed types of images.

Keywords
Image Classification, Decision Tree, Association Rules, Data Mining.

1. INTRODUCTION

With the rapid progress of multimedia technologies, image retrieval and classification from massive amounts of multimedia contents have been a hot topic among the multimedia research and applications. Due to the variability of image data, the relevant research aim at handling the huge amount of multimedia information intelligently and automatically for providing the functions like annotation and indexing by letting the machine comprehend the human concept [10, 15]. Novel methods like Content-Based Image Retrieval (CBIR) techniques have been proposed [7, 11, 17, 19] for image retrieval. Furthermore, various methods like SVM [6, 23], k-means, decision tree and association rules-based approach [22] have been studied for image classification by utilizing the image features like Color, Shape and Texture [12, 14]. In fact, the semantics of a whole image can not be represented only by its low-level features. To get closer to human perception, approaches like [4, 5, 9, 21, 22, 24, 26] have been developed to map low-level features into high-level semantics. In general, image classification is involved with the following main issues:

(i) Extraction of image features
(ii) Organization and representation of image features
(iii) Building of effective classifier.
(iv) Semantic structure modeling

According to the underlying semantics, images can be classified (or annotated) in two ways: i) Classify by some main object, ii) Classify by multiple objects with their relations. Figure 1 (A) shows an example of the first type, where the image could be classified correctly as “dog” by identifying a representative object in the image. This can be handled well by methods like decision tree or SVM [6, 23] using the image low-level features when an image is composed of only one significant object. Unfortunately, in real-life image databases, an image may contain several significant objects that form the concept of the image together. For example, as shown in figure 1 (B), the objects “dog” and “ball” form the concept of the image as “dog playing ball”. Association mining for image classification is an effective way to deal with the second type of images [22]. For a large real-life image database, these two types of images may exist simultaneously. However, none of the existing image classification methods can correctly classify both types of images at the same time although they can handle one certain type of images well individually.

![Figure 1. Two types of images: (A) With only one representative object; (B) With multiple significant objects.](image-url)
In this paper, we propose a hybrid image classification approach, namely CBROA (Classify By Representative Or Associations) that can effectively classify mixed types of images at the same time. In the CBROA method, we address the following important issues:

1. Feature preprocessing: In low-level features of an image, there may exist some unnecessary information. Consequently, the precision of classification might decrease when we take into account redundant features. We eliminate the minor low-level features in the feature preprocessing phase by utilizing decision tree method.

2. Ontology modeling: Based on human knowledge, most semantic ontologies are defined by domain experts within a specific image database. However, the identification of an image is always limited by domain experts or the inflexible semantic ontology. This becomes additional overhead for image classification system. To resolve this problem, we propose a Virtual Semantic Ontology that can be constructed dynamically for each image database with different characteristic. The constructed ontology is used as the base for classifying the single-object image.

3. Association mining: As described previously, some image is classified by its concept formed by multiple objects with their relations. We use the association rules-based method to discover the interesting association patterns for modeling the concept of this kind of images.

4. Hybrid classification: Based on the above techniques, we propose a hybrid method for classifying mixed-types of images by integrating the decision tree and association rules mining methods. For each image, one of these two kinds of mining methods is used to determine the image class adaptively based on the properties of the image. In this way, an overall high accuracy can be achieved.

5. Empirical evaluation: Through experimental evaluation on real images, the performance of CBROA is studied and shown to outperform other methods in terms of the precision.

The rest of this paper is organized as follows: Section 2 briefly describes the previous work on image classification. Section 3 introduces the proposed method in detail. Empirical evaluations on our approach are illustrated in Section 4 and conclusions are stated in Section 5.

2. RELATED WORK

A number of methods for feature extraction and image classification have existed in the literature. Carson et al. [8] proposed a new representation for images and a system named Blobworld. Images were segmented into some sub-images called blobs. A blob is coherent in color and texture space. However, if the clustering procedure of Blobworld mis-chooses some canonical blobs, lower classification precision will be generated. Yu and Wolf [25] presented a one-dimensional Hidden Markov Model (HMM) for indoor/outdoor scene classification. At first, images were divided into several horizontal (or vertical) segments and each segment was further divided into many blocks. Color histograms of blocks were used to train HMM, and Maximum likelihood algorithm was performed to build a classifier. Aghbhari et al. [1] exploited hill-climbing and SVM (Support Vector Machine) to segment an image into several objects and performed the classification by features extracted from these objects. Another image classification approach is association mining. Association mining [2, 3] is to discover implicit relationship hidden in a mass collection of data. Such the relationships can reveal hidden linkage among objects. A well-known classification approach based on association mining is CBA [13], which generates classification rules by computing the correlations between the object association and the corresponding category. Several studies applied association mining and CBA techniques on image classification and obtained good results [1, 22, 26]. Tseng et al. [22] trained the image classifier by using multiple-level association rules discovered among image objects. They built a conceptual taxonomy and discovered classification rules from images mapped to the conceptual taxonomy. The conceptual taxonomy is constructed by several clusters from bottom to top sequentially. However, it is a challenge for most bottom-up clustering approaches to determine the appropriate cluster number since it will affect the precision of classification.

3. PROPOSED METHOD

In this section, we describe in detail the proposed approach CBROA. Figure 2 depicts the architecture of CBROA, which consists of two stages, namely training and prediction stages:

1. Training Stage: In this stage, classified images are taken as input for generating the following two classifiers:
   1. \textit{Classifier}_{CBA}: We perform decision tree (C4.5) algorithm to build a classifier with low-level features extracted from the classified images. This classifier aims at classifying images with one representative object, so we call it as the type of “Classify-By-Representative (CBR)”. Meanwhile, the Virtual Semantic Ontology (will be described in detail later) based on the trained images will also be established.
   2. \textit{Classifier}_{CBR}: This classifier is to model the image semantics by correlating the object association and the image category. The Virtual Semantic Ontology is used here to map the
image objects into possible semantics, which are input into CBA algorithm for building this classifier, namely \textit{Classifier}_{CBA}. The detail of the classifier will be described later.

II. Prediction Stage: In this stage, an unclassified image is input and its category (or say class) is predicated by using the two classifiers built in training stage.

Hence, \textit{CBROA} runs with the following three major steps:

1. Construction of Virtual Semantic Ontology: With C4.5 algorithm, the Virtual Semantic Ontology is constructed by using the classified images as input to build a classification tree. The instances of pre-clustering instance-space are first grouped into a given $k$ clusters by executing bottom-up algorithms. After grouping instance-space into $k$ clusters, bottom-up algorithms will merge the nearest clusters into a new cluster recursively by a predefine branch number until the root is generated. For the practical purpose, the number of leaf nodes, parameter $k$, acts a critical role. But for C4.5 algorithm, the number of leaf nodes is not static but dynamic.

2. Generation of Virtual Code: Each image is divided into several sub-images, called objects or blobs, by image object extraction algorithms such as Blobworld [8] and Normalized Cut [20]. Then, a virtual code that represents the main features is produced for each object by traversing its path in the Virtual Semantic Ontology. Hence, every object is specified by a Virtual Codes as shown in Figure 3. The Virtual Code will be further used by CBA classifier for classifying an image.

3. Classification by \textit{CBROA} : An unclassified image is predicted for its class based on an innovative concept – classify by \textit{classifier}_{CBR} first and re-classify by \textit{classifier}_{CBA} if the classification confidence ($C_f$) obtained by \textit{classifier}_{CBA} does not meet the expected threshold. In this way, the images with representative object can be classified well by \textit{classifier}_{CBR} and those with associative semantics can be handled correctly by \textit{classifier}_{CBA}. Consequently, an overall high precision can be obtained in classifying both types of images.

In the following sections, we describe each step in detail.

### 3.1 Extraction of Image Objects and Features

In our method, it is necessary to analyze each image to find the objects contained in it and extract features of each object. Since the issue of feature extraction is not the focus of this work, we adopt the popular public software Normalized Cut [20] to perform the task of feature extraction. For each segmented objects, we make a minimum rectangle named \textit{bounding boxes} to bound the object as shown in Figure 4.

![Figure 4. Making the minimum rectangles that bound segmented objects. (A) Original image; (B) Segmented image by Normalized Cut; (C) bounding boxes.](image)

Usually, most photographers make the representative object of a photo as large as possible. Moreover, the most meaningful objects of a photo will be also placed closed to the center of a photo. That is to say, the more central and larger a bounding box is, the more significant it is. Thus, for every bounding box, the relative importance is defined as follows:

**Definition 1.** For every bounding box, $B_1, B_2, ..., B_N$ in image $I$, let the size of region belonged to $B_i$ be $A_i$, and the distance between the center of $B_i$ and that of $I$ be $D_i$. The significance $S_i$ for $B_i$ is defined as:

$$S_i = \frac{A_i}{D_i}$$

### 3.2 Training stage

Figure 5 shows the flow diagram of training stage, which primarily concerns with the generation of two major classifiers, \textit{Classifier}_{CBB} and \textit{Classifier}_{CBA}. As shown in figure 5, soon after the images are preprocessed by C4.5, Virtual Semantic Ontology and \textit{classifier}_{CBB} are constructed. Actually, without Virtual Semantic Ontology, we cannot continue performing multilevel association mining and build \textit{classifier}_{CBA}. Hence, every sub-image is encoded by utilizing the Virtual Semantic Ontology and then the association rules are produced further. In the following, we describe the two classifiers in detail.
3.2.1 Classify By Representative – Classifier\textsubscript{CBR}

Classifier\textsubscript{CBR} is built for images that are classified based on or influenced by the meaning of the most significant bounding box. Let us illustrate this concept by the example shown in Figure 6. By pruning the blue sky, figure 6(B) is a sub-image of 6(A). Furthermore, figure 6(C) is a sub-image of 6(B) by cutting out the eagle’s body. Given figure 6(B) or 6(C), we can identify this image as an eagle because 6(B) or 6(C) contains the most significant object by human cognition. Hence, in Classifier\textsubscript{CBR}, the most significant bounding box is used to perform the classification by executing decision tree algorithm like C4.5.

![Figure 6. An image with an eagle; (A) Original image; (B) Sub-image of (A); (C) Sub-image of (B).](image)

3.2.2 Classify By Associations – Classifier\textsubscript{CBA}

In Classifier\textsubscript{CBA}, the decision is made by the relationships or co-occurrences among bounding boxes of an image. We want to get the relationship such as “if a photo contains green trees and a large blue area, this is classified as a scenic photo”. For this purpose, we have to make use of such a rule yielded by association mining approach to accomplish the classification task. In this work, we adopt the CBA method that integrates association mining and classification. For any image, we treat a sub-image as an item and every sub-image is assigned a virtual code to be used in association mining task. As depicted in Figure 8, the bounding box is a sub-image of original full image and there exists exactly one path for every input sub-image. Every bounding box could be mapped into the Virtual Semantic Ontology built by C4.5. Note that although bounding boxes may be classified as the same class, we regard them as different classes if their paths are different (such as the two X labels in Figure 8). The coding task is conducted by traversing Virtual Semantic Ontology and the path of each bounding box is recorded. In other words, by exploring the ontology, traveling paths for the sub-images of classified images are identified by specified codes. We encode the left branch with “0”, and the right branch with “1”. By doing so, not only every leaf node but also every internal node of this tree will be encoded as a serial binary code containing 0s and 1s, which is called “Virtual Code”. Consider the example shown in Figure 8, the feature of leaf node Y is “R>30, G\leq 74 and B\leq 29”, and the path of it is “right, left and left”. Therefore, the Virtual Code for node Y will be encoded as “100”.

![Figure 7. A decision tree established by color feature (R, G, B).](image)

![Figure 7. The flow diagram of training stage.](image)

![Figure 8. A decision tree encoded with 0s and 1s.](image)

The virtual code indicates the hierarchical characteristic. Moreover, we employ the information of the structure to calculate the classification reliability.
**Observation 1.** Let \( I \) be the set of instances in a node (internal or leaf) belonging to the tree structure, \( I_{1} \) be the instances split left from \( I \), \( I_{2} \) be the instances split right from \( I \), and \(|I|\) be the amount of instances of the node. We get that

\[
\begin{align*}
& I_{0} \neq \phi, I_{1} \neq \phi \\
& I_{0} \cap I_{1} = \phi \\
& |I| = |I_{0}| + |I_{1}| \\
& I_{0} \subset I \\
& I_{1} \subset I
\end{align*}
\]

For example, the images classified into \( 010100110 \) are a subset of \( 01010011 \), exclusive to \( 0100011 \), and organize all the images in \( 0100011 \) with \( 01010011 \). Not only the classification of leaf nodes can be done by voting method, it is also applicable for internal nodes. For example, in real world semantic ontology, the DOG and CAT classes are sub-class of ANIMAL class. However, this kind of semantic ontology is not always available or suitable for any image database. Although we can not assign a real world class label to every node, we assign a unique code to each of them. The so called Virtual Code means a virtual class label assigned by the tree structure, and the tree structure is called a “Virtual Semantic Ontology”. For association mining, a Virtual Code could be viewed as an item and the grouped Virtual Codes (sorted by the significance) belonging to an image could be viewed as a transaction (The meaning of transaction could be referenced from [2], and the transaction length is the number of bounding boxes of an image in CBROA). Then, the classification rules are generated by executing cross-level CBA from the image transactions. For example, an association rule

\[
\{010100110, 0100011, 1001101\} \Rightarrow X
\]

indicates that an image containing \( 010100110, 0100011 \) and \( 1001101 \) always belongs to class \( X \). More precisely, we can further infer that the association rule

\[
\{010*\ldots, 0100011, 1001*\ldots\} \Rightarrow X
\]

indicates that the image belongs to a specific class \( X \) because it contains \( 010*\ldots \), \( 0100011 \) and \( 1001*\ldots \) bounding boxes. For \( 010*\ldots \), “010” is the leading bits and the “*” means the branches below the leading bit would not be considered. In other words, without considering the whole virtual code “010*\ldots”, we can identify only by leading bit “010” in this case. In addition, we sort the bounding boxes by their significances to make the rules carry semantics. Some examples are like “The image is assigned to a class \( X \) because its most significant bounding box contains Virtual Code 010*\ldots”, the second significant one contains 0100011 and the third significant one contains 1001*\ldots”. Finally, the class is resulted from more than one element of transactions. Furthermore, we set the minimum confidence for CBA (minconfidence) to be 0 to retrieve all possible rules.

As shown in Figure 5, given the training images, Virtual Semantic Ontology and classifier are first produced by C4.5. Next, they would be segmented into several bounding boxes and be extracted for the features. From these bounding boxes, the special significance of each bounding box is soon calculated. Next, the bounding boxes are sorted by their Significances and encoded by traversing the virtual ontology. Table 1 shows an image transaction table that contains four training images A, B, C, D, and the virtual code with respect to bounding boxes segmented by Normalized Cut.

<table>
<thead>
<tr>
<th>Classified Image</th>
<th>Image Class</th>
<th>bounding boxes Virtual Code</th>
<th>bounding boxes class</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Y</td>
<td>1st : 11</td>
<td>Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2nd : 000</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3rd : 101</td>
<td>Z</td>
</tr>
<tr>
<td>B</td>
<td>X</td>
<td>1st : 01</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2nd : 001</td>
<td>Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3rd : 11</td>
<td>Z</td>
</tr>
<tr>
<td>C</td>
<td>Z</td>
<td>1st : 000</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2nd : 11</td>
<td>Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3rd : 11</td>
<td>Z</td>
</tr>
</tbody>
</table>

Let \( T \) be a database of transactions, the support for a rule is defined to be the fraction of transactions in \( T \) that satisfy the union of items in the consequent and antecedent of the rule. Generate all combinations of items that have fractional transaction support above a certain threshold, called minsupport (minimum support), and call those combinations large itemsets [2]. If the minsupport (a minimum support for CBA) be 0.5, the large itemsets will be generated as shown in Table 2. Note that all codes marked with double quotes are Virtual Classifications in Classification column.

<table>
<thead>
<tr>
<th>Significance</th>
<th>Large Item Set</th>
<th>Support</th>
<th>Virtual Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>1</td>
<td>2/4</td>
<td>“1”</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>2/4</td>
<td>“11”</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>2/4</td>
<td>“0”</td>
</tr>
<tr>
<td>2nd</td>
<td>0</td>
<td>3/4</td>
<td>“0”</td>
</tr>
<tr>
<td></td>
<td>001</td>
<td>2/4</td>
<td>“001”</td>
</tr>
<tr>
<td>3rd</td>
<td>1</td>
<td>3/4</td>
<td>“1”</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>2/4</td>
<td>“11”</td>
</tr>
</tbody>
</table>
Since \( \text{minconfidence}_{CBA} \) is 0, the association rules will be generated from the large itemsets with different significances. Table 3 is an example for generated association rules when the first virtual classification is 1.

Table 3. Association Rules.

<table>
<thead>
<tr>
<th>1st Virtual Classification</th>
<th>2nd Virtual Classification</th>
<th>3rd Virtual Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;0&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;0&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;1&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;1&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;1&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;0&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;0&quot;</td>
</tr>
<tr>
<td>&quot;1&quot;</td>
<td>&quot;0&quot;</td>
<td>&quot;0&quot;</td>
</tr>
</tbody>
</table>

Moreover, we can obtain a classification class for every association rule by using voting approach. For example, the association rule \{*, *, 1\} are matched by A \{11, 000, 101\}, Y class), B \{01, 001, 11\}, X class), and D \{000, 11, 11\}, Y class), so a classification rule \{*, *, 1\} \(\Rightarrow\) Y is generated. According to these association rules, the \(\text{classifier}_{CBR} \) will be achieved successfully in the prediction stage.

3.3 CBROA Algorithm

In this section, we describe how to make the classification decision between \(\text{classifier}_{CBR} \) and \(\text{classifier}_{CBA} \) in the prediction stage. Basically, we perform \(\text{classifier}_{CBR} \) first to see if it meets the pre-specified threshold for confidence. If not, \(\text{classifier}_{CBA} \) is called subsequently for re-classification.

Definition 2. Let \( L \) be the instances of a leaf node, \( L_j \) be the instances of its parent node, and \( L_{\text{incorrect}} \) be the instances that are classified incorrectly in \( L \). The confidence measurement of a leaf node \( L \) is defined as

\[
C_L = \frac{|L| - |L_{\text{incorrect}}|}{|L|} = 1 - \frac{|L_{\text{incorrect}}|}{|L|}
\]

Assume that the confidence threshold is set as \( C_T \), what we do is to conduct the classification according to the comparison between \( C_L \) and \( C_T \). If \( C_L \) is larger than \( C_T \), we select the classification result by \(\text{classifier}_{CBR} \); otherwise the one by \(\text{classifier}_{CBA} \) is chosen.

Figure 9 shows the \(\text{CBROA} \) algorithm (with a subroutine called Classification returning the class label according to the classifiers and bounding boxes), and the flow of prediction is as shown in Figure 10. For an unclassified image, \(\text{CBROA} \) starts with the segmentation of the input image (Function \(\text{CBROA} \), line 1). Next, by applying the most significant bounding box with \(\text{classifier}_{CBR} \), we can get the specific class (Function \(\text{CBROA} \), line 4). To be continued, \( C_L \), the confidence of most significant bounding box classification, is used to decide whether this classification is strong enough (Function \(\text{CBROA} \), line 6). If \( C_{\text{CLASS}} \) is less than \( C_T \), the \(\text{CLASS} \) will be replaced by \(\text{classifier}_{CBA} \) (Function \(\text{CBROA} \), line 7).

Algorithm: CBROA

Input: An unclassified image \( I \), and \( C_T \)

Output: The class label \(\text{CLASS} \) for the unclassified image \( I \)

1. Perform \(\text{NCut} \)
2. \( B=\{b_1,\ldots,b_n\} \mid b_i \) is a bounding box of \( I, 1 \leq i \leq n \}
3. Rank \( b_i \) by \( S_i \) /* \( b_1 \) is the most significant one
4. \( \text{CLASS} = \text{Classification} (\text{Classifier}_{CBR}, b_1) \)
5. Calculate \( C_L \)
6. if \( C_L < C_T \) then
7. \(\text{CLASS} = \text{Classification} (\text{Classifier}_{CBA}, B)\)
8. endif
9. Return \(\text{CLASS} \)

Function: Classification

Input: Classifier, and the sorted Bounding Box

Output: A class label

1. if \(\text{Classifier} == \text{Classifier}_{CBR} \) then
2. return \( C4.5(b_1) \)
3. elseif \(\text{Classifier} == \text{Classifier}_{CBA} \) then
4. return \(\text{CBA}(b_1,\ldots,b_n) \)
5. endif

Figure 9. The \(\text{CBROA} \) algorithm.
4. EXPERIMENTAL EVALUATION

In this section, we evaluate the performance of CBROA by comparing with C4.5, SVM, and the method in [22]. The experiments are implemented in Java on a Pentium-4 1.6GHz personal computer with 2.0GB RAM running on Windows 2003 Server.

4.1 Experimental Data

The image collection in our experiments is from Corel Gallery Professional Image Library which contains many types of images including signs, backgrounds, icons and so on. We selected 5000 images from one million images with nine classes. The classes and corresponding amount of images in database are shown in Table 4.

<table>
<thead>
<tr>
<th>Image Class</th>
<th>Amount</th>
<th>Description</th>
<th>Image Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animal</td>
<td>1100</td>
<td>dog, bird, and fish, etc.</td>
<td>192<em>128 or 128</em>192</td>
</tr>
<tr>
<td>Artifact</td>
<td>400</td>
<td>all kinds of artifacts</td>
<td></td>
</tr>
<tr>
<td>Food</td>
<td>300</td>
<td>all kinds of food</td>
<td></td>
</tr>
<tr>
<td>Insect</td>
<td>100</td>
<td>all kinds of insect</td>
<td></td>
</tr>
<tr>
<td>People</td>
<td>800</td>
<td>all kinds of people</td>
<td></td>
</tr>
<tr>
<td>Plant</td>
<td>600</td>
<td>tree, and flower, etc.</td>
<td></td>
</tr>
<tr>
<td>Scenic</td>
<td>400</td>
<td>mountain, cloud, and sea, etc.</td>
<td></td>
</tr>
<tr>
<td>Structure</td>
<td>700</td>
<td>building</td>
<td></td>
</tr>
<tr>
<td>Transportation</td>
<td>600</td>
<td>car, airplane, and boat, etc.</td>
<td></td>
</tr>
</tbody>
</table>

There are 5000 images in our image database, and is no overlap between above classes.

The size of each image is either 192*128 or 128*192 pixels. We extract 6 low-level features, namely Color Structure, Color Layout, Region Shape, Scalable Color, Homogeneous Texture, and Edge Histogram. We thus adopt a precision described in the following. Let D be the set of all images to be predicted and C be the set of images that are predicted correctly. The precision is defined as below:

\[
\text{precision} = \frac{C}{D}
\]

Precision represents the ratio between the cardinality of correctly classified images and the cardinality of classified images. For example, from ten predicting images, if eight images are correctly classified from the ten predicted images, the Precision is 8/10 = 80%. We randomly select 30% of the 5000 images for training, and the other images are for testing the prediction accuracy.

4.2 Parameters for Normalized Cut

The parameter \(N\), which is the quantity of the segmentations for an image, should be pre-defined for Normalized Cut method. In segmentation task, we set it as 4, 8, and 16 and pick the best setting for evaluating our experiments. Because an image is a transaction for classifier_{CBA}, a segmentation of a predicting image represents an item included in the transaction. According to the evaluation of preliminary experiments, \(N\) is set to 8 by us.

4.3 Effects of Varying Number of Bounding Boxes and minsupport_{CBA}

After choosing \(N\), we evaluate the precision for CBA corresponding to different number of bounding boxes \((n)\) per image and minsupport_{CBA}. Although \(N\) objects are extracted per image, some of the objects may not be significant. Note that the parameter \(n\) \((n \leq N)\) decides the top \(n\) significant of \(N\) bounding boxes. In other words, we take the 1st, 2nd, ..., \(n\)th significant bounding boxes into consideration with CBA. As illustrated in Figure 11, the best precision appears when \(n = 4\) and the minsupport_{CBA} is 0.1. Note that we set minconfidence_{CBA} as 0 in this research because we want to retrieve all possible rule patterns.

![Figure 11. The precision for classifier_{CBA} under different number of bounding boxes (n) and minsupport_{CBA}.](image-url)
4.4 Overall Comparisons

After evaluating the accuracy under different number of bounding boxes \((n)\) and minimum support \((\text{minsupport}_{\text{CBA}})\) setting, we further compared CBROA with C4.5, SVM and the multilevel association rules (denoted as MLA) proposed in [22]. Since the confidence threshold \((C_T)\) is designed for CBROA, it is meaningless to other algorithms. That is the reason why the precision values for other approaches keep still. By doing so, we could illustrate the precision between any other algorithms in different confidence threshold values.

The following observations are made in Figure 12:

1. Setting \(n\) as 4 and \(C_T\) as 0.5 results in best performance for CBROA.

2. With the best setting, CBROA outperforms all other algorithms substantially in terms of classification precision. It achieves improvements on MLA [22], C4.5 and SVM by about 10%, 30% and 40%, respectively.

We could explain and discuss the experiment result by the Venn diagram as shown in Figure 13. Let \(U\) be the set of all image instances (● in Figure 13 denotes an image instance), \(\alpha\) be the set of image instances that could be classified correctly by C4.5, and \(\beta\) be the set of image instances that could be classified correctly by CBA. Therefore, the closed area I=\(U-(\alpha+\beta)\) indicates the set of instances that can not be classified correctly by either C4.5 or CBA. The area II=\(\alpha-\beta\) means the set of instances that could be classified correctly only by CBA. The area IV=\(\alpha\cap\beta\) means the set of instances that could be classified correctly by both C4.5 and CBA. The area II will be an incorrect classification if we classified it with CBA and the classification on area I and IV will not affect the classification precision no mater by C4.5 or CBA. The mission of confidence threshold \((C_T)\) for CBROA is to determine the proper classifier to promote the precision of image classification throughout the judgment of the specific area an instance should belong to.

5. CONCLUSIONS

In this paper, we proposed a hybrid method named “CBROA” to accomplish image classification by integrating decision tree and association rules mining, which forms the CBR and CBA classifiers, respectively. In CBROA, we conquer the requirement of ontology by constructing the Virtual Semantic Ontology. Because of the characteristic of CBR and CBA, CBROA can perform the classification of two types of images well. This resolves the problem of existing classification algorithms that can perform well only for some type of images. The experimental results show that CBROA outperforms other classification methods like C4.5 and SVM substantially in terms of classification accuracy.

We will explore the following issues in future research: First, the accuracy of the segmentation is the critical factor for the precision in our work. We shall integrate other more effective methods for improving further the accuracy. Second, in the experiments, a good confidence threshold \((C_T)\) has great effect on the precision of our CBROA algorithm. We will further investigate the best confidence threshold setting.

ACKNOWLEDGEMENT

This research was supported by Ministry of Economic Affairs, R.O.C., under grant no. 92-EC-17-A-02-51-024

REFERENCES


OCRS: An Interactive Object-based Image Clustering and Retrieval System

Chengcui Zhang  
Department of Computer and Information Sciences,  
University of Alabama at Birmingham  
Birmingham, AL 35294-1170, U.S.A.  
(+1)205-934-8606  
zhang@cis.uab.edu

Xin Chen  
Department of Computer and Information Sciences,  
University of Alabama at Birmingham  
Birmingham, AL 35294-1170, U.S.A.  
(+1)205-934-8669  
chenxin@cis.uab.edu

ABSTRACT
In this paper, we propose an Interactive Object-based Image Clustering and Retrieval System (OCRS). The system incorporates two major modules: Preprocessing and Object-based Image Retrieval. In preprocessing, we use WavSeg to segment images into meaningful semantic regions (image objects). This is an area where a huge number of image regions are involved. Therefore, we propose a Genetic Algorithm based algorithm to cluster these images objects and thus reduce the search space for image retrieval. In learning and retrieval module, Diverse Density is adopted to analyze user’s interest and generate the initial hypothesis which provides a prototype for later learning and retrieval. Relevance Feedback technique is incorporated to provide progressive guidance to the learning process. In interacting with user, we propose to use One-Class Support Vector Machine (SVM) to learn user’s interest and refine the returned result. Performance is evaluated on a large image database and the effectiveness of our retrieval algorithm is demonstrated through comparative studies.

Categories and Subject Descriptors
H.2.8 [Database Management]: Database Applications – Multimedia Data Mining.

General Terms
Algorithms, Experimentation.

Keywords
Object-based CBIR.

1. INTRODUCTION
With the rapid increase of various digital image data, image retrieval has drawn attention of many researchers in the computer vision and the database communities. However, the current state-of-art technologies are facing two main problems: 1) semantic gap between low level features and high level concept; 2) curse of dimensionality. This paper aims to build a framework to alleviate the above problems especially for object-based image retrieval.

For “semantic gap” problem, Relevance feedback (RF) technique is widely used to incorporate the user’s concept with the learning process [4][6]. As a supervised learning technique, it has been shown to significantly increase the retrieval accuracy. However, most of the existing RF-based approaches consider each image as a whole, which is represented by a vector of N dimensional image features. However, user’s query interest is often just one part of the query image i.e. a region in the image that has an obvious semantic meaning. Therefore, rather than viewing each image as a whole, it is more reasonable to view it as a set of semantic regions. In this context, the goal of image retrieval is to find the semantic region(s) of the user’s interest. Since each image is composed of several regions and each region can be taken as an instance, region-based CBIR is then transformed into a Multiple Instance Learning (MIL) problem. Maron et al. applied MIL into natural scene image classification [2]. Each image is viewed as a bag of semantic regions (instances). In the scenario of MIL, the labels of individual instances in the training data are not available, instead the bags are labeled. When applied to RF-based image retrieval, this corresponds to the scenario that the user gives feedback on the whole image (bag) although he/she may be only interested in a specific region (instance) of that image. The goal of MIL is to obtain a hypothesis from the training examples that generates labels for unseen bags (images) based on the user’s interest in a specific region.

In order to support region-based image retrieval, we need to divide each image into several semantic regions. Instead of viewing each image as a whole, we thus examine region similarity during image retrieval. However, this further increases the search space by a factor of 4~6. Clustering is a process of grouping a set of physical or abstract objects into classes based on some similarity criteria. Given the huge amount of semantic regions in this problem, we first preprocess image regions by dividing them into clusters. In this way the search space can be reduced to a few clusters that are relevant to the query region. K-means is a traditional clustering method and has been widely used in image clustering such as [14] [15]. However, it is incapable of finding non-convex clusters and tends to fall into local optimum especially when the number of data objects is large. In contrast, Genetic algorithm [9] is known for its robustness and ability to approximate global optimum. In this study, we adapted it to suit our needs of clustering image regions.

After clustering, our proposed system applies Diverse Density (DD) as proposed within the framework of MIL by Maron et al. [2] to learn the region of interest from users’ relevance feedback.
on the whole image and tells the system to shift its focus of attention to that region. In [1], Zhang et al. further extends Maron’s work by incorporating EM (Expectation-Maximization) algorithm in finding the maximum DD point. We adopt Zhang’s method because it is less sensitive to the dimension of feature space and scales up well. We take the output of DD as our initial hypothesis of user’s interest and continue the relevance feedback with our kernel learning algorithm.

Chen et al. [12] proposed a Support Vector Machine (SVM) based algorithm for Content-based Image Retrieval. In Chen’s method, the problem falls into two class classification solved by standard SVM. However, we consider grouping all negative regions into one class somewhat inappropriate. Therefore, after initial analysis of user’s interest by DD, our proposed learning algorithm concentrates on those positive images and uses the learned region-of-interest to evaluate all the other images in the image database. The motivation comes from the fact that positive samples are all alike, while negative samples are each bad in their own way. In other words, instead of building models for both positive class and negative class, it makes more sense to assume that all positive regions are in one class while the negative regions are outliers of the positive class. Therefore, we applied One-Class Support Vector Machine (SVM) [3] to solve the MIL problem in CBIR. Chen et al. [7] and Gondra [11] use One-Class SVM in image retrieval but, again, it is applied to the image as a whole. In our approach, One-Class SVM is used to model the non-linear distribution of image regions and to separate the learned regions from negative ones. Each region of the test images is given a score by the evaluation function built from the model. The higher the score, the more similar it is to the region of interest. The images with the highest scores are returned to the user as query results. However, the critical issue here is how to transform the traditional SVM learning, in which labeled training instances are readily available, to a MIL learning problem where only the labels of bags (e.g. images with positive/negative feedbacks) are available. In this study, we proposed a method to solve the aforementioned problem and our experiments show that high retrieval accuracy can be achieved usually within 4 iterations.

In Section 2, we present an overview of our OCRS system. The preprocessing module is presented in Section 3 which involves segmentation and clustering. The detailed learning and retrieval approach is discussed in Section 4. In Section 5, system performance evaluation with experimental results is presented. Section 6 concludes the paper.

2. SYSTEM ARCHITECTURE

Figure 1 shows the architecture of our system. In preprocessing module, images are segmented into semantic regions, with each represented by a 19-feature vector. A Genetic Algorithm based clustering method is then implemented to cluster these image segments into clusters so that similar image segments are grouped together.

In initial query, the system first gets the user’s query. However, at this point, the system has no clue to the user’s interested semantic region. Therefore, a simple Euclidean based similarity comparison is performed to retrieve the initial query results to the user. After the initial query, the user gives feedback to the retrieved images and these feedbacks are examined by Diverse Density trying to analyze user’s interest. The output of Diverse Density algorithm is the initial input of One-Class Support Vector Machine (SVM) based algorithm which learns from these feedbacks and starts another round of retrieval. In each round, the refined retrieval result is provided to the user for feedback. One-Class SVM studies these feedbacks and builds a model for future retrieval.

3. PREPROCESSING

3.1 Segmentation

3.1.1 WavSeg Image Segmentation

Instead of manually dividing each image into many overlapping regions [5], in this study, we propose to use a fast yet effective image segmentation method called WavSeg [13] to partition the images. In Wavseg, a wavelet analysis in concert with the SPCPE algorithm [16] is used to segment an image into regions.

By using wavelet transform and choosing proper wavelets (Daubechies wavelets), the high-frequency components will disappear in larger scale sub bands and therefore, the potential regions will become clearly evident. In our experiments, the images are pre-processed by Daubechies wavelet transform because it is proven to be suitable for image analysis. The decomposition level is 1. Then by grouping the salient points from each channel, an initial coarse partition can be obtained and passed as the input to the SPCPE segmentation algorithm. Actually, even the coarse initial partition generated by wavelet transform is much closer to some global minima in SPCPE than a random initial partition, which means a better initial partition will lead to better segmentation results. In addition, wavelet transform can produce other useful features such as texture features in addition to extracting the region-of-interest within one entry scanning through the image data. Based on our initial testing results, the wavelet based SPCPE segmentation framework (WavSeg) outperforms the random initial partition based SPCPE algorithm in average. It is worth pointing out that WavSeg is fast. The processing time for a 240×384 image is only about 0.33 second in average.
3.1.2 Region Feature Extraction

Both the local color and local texture features are extracted for each image region. For color features, HSV color space and its variants are proven to be particularly amenable to color image analysis. Therefore, we quantize the color space using color categorization based on H S V value ranges. Twelve representative colors are identified. They are black, white, red, red-yellow, yellow, yellow-green, green, green-blue, blue, blue-purple, purple, and purple-red. The Hue is divided into five main color slices and five transition color slices. Each transition color slice such as yellow-green is considered in both adjacent main color slices. We disregard the difference between the bright chromatic colors and the chromatic colors. Each transition color slice is treated as a separate category instead of being combined into both adjacent main color slices. A new category “gray” is added so that there are totally thirteen color features for each image region in our method.

For texture features, one-level wavelet transformation using Daubechies wavelets are used to generate four subbands of the original image. They include the horizontal detail sub-image, the vertical detail sub-image, and the diagonal detail sub-image. For the wavelet coefficients in each of the above three subbands, the mean and variance values are collected respectively. Therefore, totally six texture features are generated for each image region in our method.

The thirteen color features and six texture features of each region are extracted after image segmentation. Thus, for each image, the number of its objects (regions) is equal to the number of regions within that image. Each object has nineteen features.

3.2 Clustering

3.2.1 Overview of Genetic Algorithm

The basic idea of Genetic Algorithm originates from the theory of evolution -- “survival of the fittest”. It was formally introduced in the 1970s by John Holland [9]. It is less susceptible to getting “stuck” at local optima. The overview of genetic algorithm is shown in Figure 2.

The algorithm starts with randomly generating the initial population (possible solutions to a real-world problem). In order to be understood in genetic world, the possible solutions to a real world problem are first encoded. Each solution forms a chromosome. A population is a group of chromosomes. From the first generation, these chromosomes are first evaluated. Then they are operated by three genetic operators: Selection, Crossover and Mutation and generate the next generation. The next generation of chromosomes is again evaluated. An objective function is used in evaluation which measures the fitness of each individual solution (chromosome). This accomplishes the evolution of the first generation. Genetic algorithm then starts to run the next generation and goes through the above-mentioned process again until an optimal solution is found.

The thirteen color features and six texture features of each region are extracted after image segmentation. Thus, for each image, the number of its objects (regions) is equal to the number of regions within that image. Each object has nineteen features.

3.2.2 Genetic Algorithm Design for Image Region Clustering

The objective of image region clustering is to find the optimal combination that minimizes the function below:

$$F(R) = \sum_{i=1}^{n} \sum_{j=1}^{k} d(p_i, rep[p_i, R_j])$$  \hspace{1cm} (1)

$p_i$ is an image region in the cluster $R_j$ which is represented by a representative image region $rep[p_i, R_j]$. $n$ is the total number of image regions and $k$ is the number of clusters. The value of $k$ is determined experimentally as there is no prior knowledge about how many clusters are there. A too large $k$ value would result in over-clustering and increase the number of false negatives, while a too small $k$ value would not help much in reducing the search space. According to our experiment, in which there are 10,000 images with 49,584 regions, we divide the entire set of image regions into 100 clusters since it results in a good balance between accuracy and efficiency. $d$ is some distance measure. In this study, we use the Euclidean distance. Equation 1 is the objective function in our algorithm. The goal is to find its minimum.

In image region clustering, the target is to group semantic image regions into clusters according to their similarities. The above-mentioned representative regions are actually centroids clusters. Therefore, a feasible solution to a clustering problem would be a set of centroids. To encode it, we give each region an ID: 1, 2, ..., $n$ ($n$ is an integer). The centroids are represented by their IDs in the chromosome.

![Figure 3. An Example of Chromosome](image)

Figure 3 is an example of a chromosome. In this chromosome, each integer is a gene in genetic world which corresponds to the ID of a centroid image region in the real world.
The initial size of population is set to \( l \) which is 50 in this study. For each chromosome we randomly generate \( k \) genes, which are actually \( k \) integers between 1 and \( n \) (the number of image regions). These \( k \) genes correspond to the representative image region for each of the \( k \) clusters. We then calculate the inverse values of the objective function for these chromosomes: \( f_1, f_2, \ldots, f_l \). The fitness of each individual chromosome is computed according to Equation (2).

\[
\text{Fit}_i = \frac{f_i}{\sum_{i=1}^{l} f_i}
\]

With the first generation, “evolution” begins. In each generation, the whole population goes through three operators: Selection, Recombination and Mutation.

1) **Selection**: There are many kinds of selection operations. We use a Roulette to simulate the selection as shown in Figure 4. For each chromosome we compute its fitness according to Equation (2). Two chromosomes from the population are then randomly selected. The higher the fitness the higher the chance a chromosome is selected. This mechanism is like a rotating roulette as shown in Figure 4. \( C_1, C_2 \ldots \) are chromosomes. The area each chromosome occupies is determined by its fitness. Therefore, chromosomes with higher fitness values would have more chances to be selected in each rotation. We select \( l \) pairs of chromosomes and feed them into the next step.

2) **Recombination**: In this step, the recombination operator proposed in [10] is used instead of a simple crossover. Given a pair of chromosomes \( C_1 \) and \( C_2 \), we use recombination operator to generate their child chromosome \( C_0 \) one gene at a time. Each gene in \( C_0 \) is either in \( C_1 \) or \( C_2 \) or both and is not repetitive of other genes in \( C_0 \).

3) **Mutation**: In order to obtain high diversity of the genes, a “newly-born” child chromosome may mutate one of its genes to a random integer between 1 and \( n \). However, this mutation is operated at a very low frequency.

At the end of the process, the chromosome with highest fitness in the last population is selected to be a feasible solution. This chromosome is decoded and we have the centroids of clusters as our final output.

\[\text{Figure 4. Roulette}\]

4. **LEARNING AND RETRIEVAL FRAMEWORK**

4.1 **User Interest Analysis**

In this study, we assume that user is only interested in one semantic region of the query image. The goal is to retrieve those images that contain similar semantic regions. In the initial query, user identifies a query image. At this point, no information is provided as to which specific part of the image is of the user’s interest. Therefore, for all the images in the image database, we compute the Euclidean distances between the regions of these images and the regions of the query image. The distance between the query image and an image in the image database is represented by the smallest pair-wise distance of their regions. The “representative” distances are then sorted in an ascending order and the top 30 images are returned to the user for feedback.

The user identifies a returned image as “positive” if it is of his/her interest; otherwise the user labels it “negative”. With this information at hand, our next step is to estimate the user’s interest i.e. which specific region of the query image user is interested in. We apply Diverse Density (DD) algorithm to accomplish this goal. Diverse Density was first proposed by Maron and Lozano-Pérez in the framework of Multiple Instance Learning [2].

In Multiple Instance Learning (MIL), the label of an individual instance (object) is unknown. Only the label of a set of instances is available, which is called the label of a bag. MIL needs to map an instance to its label according to the information learned from the bag labels. In Content-based Image Retrieval, we have two types of labels – Positive and Negative. Each image is considered a bag of semantic regions (instances). When supplying feedback to the retrieved images, the label of each retrieved image, i.e. bag label, is available. However, the label of each semantic regions in that image bag is still unknown because the user only gives feedback to an image as a whole, not to individual semantic regions in that image. The goal of MIL is to estimate the labels (similarity scores) of the test image regions/instances based on the learned information from the labeled images/bags in the training set.

With Diverse Density approach, an objective function called DD function is defined to measure the co-occurrence of similar instances from different bags (images) with the same label. The target of DD is to find a point which is the closest to all the positive images and farthest from all the negative images. The framework of DD by Maron and Lozano-Pérez [2] is briefly explained below.

We denote the positive bags as \( B_{1+}, B_{2+}, \ldots, B_{n+} \) and the negative bags as \( B_{1-}, B_{2-}, \ldots, B_{n-} \). The \( j^{th} \) instance of bag \( B_{i+} \) is represented as \( B_{ij+} \), while the \( j^{th} \) instance of bag \( B_{i-} \) is written as \( B_{ij-} \). Each bag may contain any number of instances, but every instance must be represented by a \( k \) dimensional vector where \( k \) is a constant.

Different semantic concepts may share some common low-level features, but not all \( k \) dimensions contribute equally. Therefore, given a semantic concept, greater weights shall be assigned to relevant features as compared to less relevant features. For example, color and texture features shall be assigned greater weights for “grass” compared to shape features. We denote this weight vector \( a = \{a_1, a_2, \ldots, a_k\} \).
For any point \( p = \{p_1, p_2, \ldots, p_k\} \) in the feature space, the Diverse Density is defined by the probability of it being our target point, given all the positive and negative bags. So the point we are looking for is the one that maximizes the probability below.

\[
\text{Argmax}_{h} \prod_{i} P(p|B_i^+) \prod_{i} P(p|B_i^-) \tag{3}
\]

Assuming a uniform prior over the concept location \( P(p) \) and conditional independence of the bags given the target concept \( p \), the above function equals to

\[
\text{Argmax} \prod_{i} P(p|B_i^+) \prod_{i} P(p|B_i^-) \tag{4}
\]

The following model was introduced by Maron and Lozano-Pérez for estimating hypothesis \( h = \{p_1, \ldots, p_w, a_1, \ldots, a_k\} \).

\[
P(B_j \mid p) = \exp\left(-\sum_{i} \alpha_k (B_{ijk} - p_k)^2\right) \tag{5}
\]

The goal is to find such a hypothesis \( h \) such that the above function reaches its maximum. We apply EM (Expectation-Maximization) algorithm as proposed by Zhang et al. [1]. EM starts with an initial hypothesis \( h \), and then repeatedly performs E-step and M-step. In E-step, the current hypothesis \( h \) is used to pick one instance from each bag which is most likely to be the one responsible for the label of the bag. In M-step, a two step gradient ascent search (Quasi-Newton search) is performed on DD algorithm to find a new \( h' \) that maximizes the above function. In the new iteration \( h' \) replaces \( h \). The loop continues until the algorithm converges.

\[
\text{Figure 5. Image Region Clusters}
\]

We then use the final result \( h \), the point of the user’s interest, to find the cluster this point \( h \) belongs to. Hence all the other image regions in this cluster can be located. However, we cannot simply reduce the search space to this cluster alone because it is not a rare case that a particular region is closer to some regions in another cluster than some regions within the same cluster. This situation is illustrated in Figure 5. Suppose the query image region is a “horse” region in Image B. It is in the same cluster with the “castle” region of Image C because these two regions share similar low level features. However, Image B is conceptually closer to Image A whose “horse” region is in another cluster. Therefore, in our system, we choose three clusters whose centroids are the closest to the query region. As an image is composed of several semantic regions, it can fall into any cluster that has at least one of its semantic regions. We then group all the images that have at least one semantic region fall into the three clusters mentioned above and take it as the reduced search space for a given query region. The effectiveness of this reduction is presented in Section 5.

\[
\text{Figure 6. One Class Classifications}
\]

4.2 Learning and Retrieval

As mentioned above, DD algorithm is applied to analyze the user’s interest after the initial query. Yet due to the large amount of noise in the image data set, we cannot guarantee that the user’s interest is exactly \( h \). Instead, it is taken as our initial hypothesis and the system continues interacting with user to collect more feedbacks. The output of DD is a group of instances, one from each image, that contribute most to the image (bag) label. Specifically, in the output of DD, instances that come from positive bags are positive instances. Because of these instances, their corresponding bags are labeled positive. We then construct the training sample set according to this output of DD. This is then fed into One-Class Support Vector Machine as the initial training sample set, which further learns and models user’s interest and refines the retrieval result in the following iterations.

One-Class classification is a kind of unsupervised learning mechanism. It tries to assess whether a test point is likely to belong to the distribution underlying the training data. In our case, the training set is composed of positive samples only. Figure 6 shows how positive image regions are all alike and should be in one class while it is inappropriate to group negative image regions into a single class. In Figure 6, we use images segmented by Blobworld [17] as an example in which image regions are outlined by red lines. Suppose the user’s interest is a “horse” object, then ideally positive image regions shall be those “horse” regions. However, negative image regions can be anything other than “horse”. As shown in Figure 6, negative image regions can be “flower”, “wall”, and “sea”, etc.
One-Class SVM has so far been studied in the context of SVMs [3]. The objective is to create a binary-valued function that is positive in those regions of input space where the data predominantly lies and negative elsewhere. The idea is to model the dense region as a “ball”. In our problem, positive instances are inside the “ball” and negative instances are outside. If the origin of the “ball” is the point \( p \) and the radius is \( r \), a point \( p' \), in this case an instance (image region) represented by an 19-dimension feature vector, is inside the “ball” iff \( \| p_i - p \| \leq r \). This “ball” is actually a hyper-sphere. The goal is to keep this hyper-sphere as “pure” as possible and include most of the positive objects. Since this involves a non-linear distribution in the original space, the strategy of Schölkopf’s One-Class SVM is first to do a mapping \( \theta \) to transform the data into a feature space \( F \) corresponding to the kernel \( K \):  
\[
\theta(p_1) \cdot \theta(p_2) = K(p_1, p_2)
\]
where \( p_1 \) and \( p_2 \) are two data points. In this study, we choose to use Radial Basis Function (RBF) Machine below.

\[
K(p_1, p_2) = \exp(-\| p_1 - p_2 \|^2 / 2\sigma^2)
\]

Mathematically, One-Class SVM solves the following quadratic problem:

\[
\min \frac{1}{2} \| u \|_2^2 - u \rho + \sum_{i=1}^{n} \xi_i
\]

subject to

\[
(w \cdot \theta(p_i)) \geq \rho - \xi_i, \quad \xi_i \geq 0 \quad \text{and} \quad i = 1,...,n
\]

where \( \xi_i \) is the slack variable, and \( u \in (0,1) \) is a parameter that controls the trade off between maximizing the distance from the origin and containing most of the data in the region created by the hyper-sphere and corresponds to the ratio of “outliers” in the training dataset. If \( w \) and \( \rho \) are a solution to this problem, then the decision function is \( f(x) = \text{sign}(w \cdot \theta(p) - \rho) \) and it will be 1 for most examples \( p_i \) contained in the training set.

Some images may actually contain more than one positive region. Therefore, we cannot assume that only one region in each image is positive. Suppose the number of positive images is \( n \) and the number of all semantic regions in the training set is \( N \). Then the ratio of “outliers” in the training set is set to:

\[
u = 1 - \left( \frac{n}{N} + z \right)
\]

\( z \) is a small number used to adjust the \( u \) in order to alleviate the above mentioned problem. Our experiment results show that \( z = 0.01 \) is a reasonable value.

The training set as well as the parameter \( u \) are fed into One-Class SVM to obtain \( w \) and \( \rho \), which are used to calculate the value of the decision function for the test data, i.e. all the image regions in the database. Each image region will be assigned a “score” by \( w \cdot \theta(p) - \rho \) in the decision function. The higher the score, the more likely this region is in the positive class. The similarity score of each image is then set to the highest score of all its regions.

5. System Performance Evaluation

The experiment is conducted on a Corel image database consisting of 10,000 images from 100 categories. After segmentation, there are in total 49,584 image segments. We tested the system performance under different clustering schemes by dividing the entire set of image regions into 40 to 150 clusters. Each time we increase the number of clusters by 10 and find that when the number of clusters \( k=100 \), the result is most reasonable in terms of the balance between accuracy and reduction of search space. After the initial query, according to the hypothesis generated by DD, we pull out the three closest clusters as the reduced search space. All the images that have at least one segment fall into these three clusters are identified and fed into the learned One-Class SVM for classification. Sixty images are randomly chosen from 20 categories as the query images. According to our experiment, the search space, in terms of the number of images in the three candidate clusters, is reduced to 13.4% of the original search space (10,000 images) on average.

We compare the performance of our system with two other relevance feedback algorithms: 1) Neural Network based Multiple Instance Learning (MIL) algorithm with relevance feedback [8]; 2) General feature re-weighting algorithm [4] with relevance feedback. For the latter, both Euclidean and Manhattan distances are tested.

Five rounds of relevance feedback are performed for each query image - Initial (no feedback), First, Second, Third, and Fourth. The accuracy rates with different scopes, i.e. the percentage of positive images within the top 6, 12, 18, 24 and 30 retrieved images, are calculated. Figure 7 shows the result after the Fourth Query. “BP” is the Neural Network based MIL which uses both positive and negative examples. “RF_E” is feature re-weighting method with Euclidean Distance while “RF_M” uses Manhattan Distance. “OCRS” is the proposed system.

![Fourth Query Result](image)

Figure 7. Retrieval Accuracy after the Fourth Query

It can be gleaned from Figure 7 that while the search space is substantially reduced, the accuracy of the proposed framework still outperforms all the other 3 algorithms. It also can be seen that the Neural Network based MIL (BP) shows a better result than
that of general feature re-weighting algorithm after 4 rounds of learning. In addition, the performance of RF_E using Euclidean Distance is slightly better than that of RF_M which uses Manhattan Distance.

Figures 8 and 9 show the first and the fourth Query results of “OCRS”, respectively, given the query image on the upper left corner of the interface. In this example, “horse” is the user’s interest. It can be seen that only several “horse” images are found correctly initially whereas more “horse” images are found after the fourth round of iteration.

It is worth mentioning that, the number of positive images increases steadily through each iteration. Figure 10 gives a concrete view as to how accuracy rates of our algorithm increases across 5 iterations.

6. CONCLUSIONS
In this paper, we proposed a framework, OCRS, for single region based image retrieval. OCRS strives to solve two crucial problems in this area, i.e. time complexity due to the huge amount of high-dimensionality data; semantic gap between low level features and human subjecitivity. Specifically in preprocessing, a Genetic Algorithm based clustering mechanism is proposed to reduce the search space. An efficient image segmentation algorithm -- WavSeg is implemented to divide an image into semantic regions. We then adopt Diverse Density to do the initial analysis of user’s interest. As initial hypothesis, the output of DD is fed into One-Class SVM in the image retrieval phase. The advantage of our algorithm is that it targets image region retrieval instead of the whole image, which is more reasonable since the user is often interested in only one region in the image. The proposed work also transforms the One-Class SVM learning for region-based image retrieval into a Multiple Instance Learning problem. In addition, due to the robustness of Genetic Algorithm in approximating global optima and the generality of One-Class SVM, the proposed system has proved to be effective in better identifying the user’s real need and removing the noise data.

7. ACKNOWLEDGMENTS
The work of Chengcui Zhang was supported in part by SBE-0245090 and the UAB ADVANCE program of the Office for the Advancement of Women in Science and Engineering.

8. REFERENCES


A Framework for a Video Analysis Tool for Suspicious Event Detection

Gal Lavee
The University of Texas at Dallas
800 West Campbell Road
Richardson, Texas 75080
001-972-883-2111
Gxl034000@utdallas.edu

Latifur Khan
The University of Texas at Dallas
800 West Campbell Road
Richardson, Texas 75080
001-972-883-2111
lkhan@utdallas.edu

Bhavani Thuraisingham
The University of Texas at Dallas
800 West Campbell Road
Richardson, Texas 75080
001-972-883-2111
bhavani.thuraisingham@utdallas.edu

ABSTRACT

This paper proposes a framework to aid video analysts in machine learning algorithms to learn the event representation. Experiments are used as a lead to a discussion on the most optimal future experiment to refine this technique is proposed. These events seen by a human observer are discussed. An evaluation of low-level machine readable features of video data and high-level video. Ideas and techniques for closing the semantic gap between video data that exists in today’s world of omnipresent surveillance detecting suspicious activity within the tremendous amounts of video sequences. We proposed and implemented a framework for a suspicious event detection system based on this technique. The design and implementation details of this framework are given in section 2. A preliminary evaluation of this system’s event classification component’s performance is given in section 3. In section 4 we seek to determine an optimal learning algorithm for correctly classifying new events based on previously known manually (or automatically) labeled examples and propose an experiment to that end. In section 5 we present our findings and discuss advantages and disadvantages of each of the proposed techniques. In section 6 we summarize the paper and discuss future directions.

This security guard (or other video analyst) could be helped a great deal if there existed a computer system which could learn events based on training examples and use this definition in conjunction with some configuration parameters to differentiate between “suspicious” and “normal” behavior. An interface to point out areas of particular interest in offline analysis could then be developed. At that time the analyst can exercise his judgment as to whether action needs to be taken based on this highlighted video segment.

Intrinsically tied to this work is the problem of bridging the semantic gap between the low-level features a machine sees when a video is input into it (e.g. color, texture, shape) and the high-level semantic concepts (or events) a human being sees when looking at a video clip (e.g. presentation, newscast, boxing match). Any approach to solving this problem must consider how low-level features will be extracted, how events should be represented and how unlabeled events will be classified. The first step towards solving this problem is to develop a robust event representation that characterizes the event (a series of frames in a video sequence) such that it can be meaningfully contrasted with other events. [1, 3, 4, 6]

The event representation technique suggested in our paper is based on the paper by Zelnik-Manor et al. [1]. This method takes advantage of the fact that the same event captured by different camera configurations looks identical when projected on to the temporal dimension. Similar events which differ slightly in scale, translation, rotation and even color (different clothing) are classified as the same event. This characteristic allows us to cluster a video into event-specific segments, devise an event comparison method, and define “suspicious” activity by example. We proposed and implemented a framework for a suspicious event detection system based on this technique. The design and implementation details of this framework are given in section 2. A preliminary evaluation of this system’s event classification component’s performance is given in section 3. In section 4 we seek to determine an optimal learning algorithm for correctly classifying new events based on previously known manually (or automatically) labeled examples and propose an experiment to that end. In section 5 we present our findings and discuss advantages and disadvantages of each of the proposed techniques. In section 6 we summarize the paper and discuss future directions.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MDM/KDD 2005 Chicago, August 21, Chicago, Illinois, USA
Copyright 2005 ACM -- MDM 2005 - 1-59593-216-X...$5.00.
2. OUR APPROACH

2.1 System Design

Our proposed system takes a new unlabeled video sequence and multiple labeled video sequences representing different types of possible events. It produces a visualization of the content in the unlabeled video sequence as output to the user. The user can adapt this visualization according to their preferences (i.e. what type of event they consider to be suspicious) using the Video Analysis Tool interface. Figure 1 shows a block diagram of the system design.

The new video sequence is read in and stored as a matrix of RGB values over time (width*height*3*number of frames). This phase can be thought of as the extraction of low-level features. An event representation (see below) for several overlapping subsequences is generated for use in event detection. These newly generated events are compared to a set of predefined events using the event comparison (distance) function defined below. The events are then classified by use of the nearest neighbor algorithm.

Once each of the overlapping events has a label the system generates a summary of events contained within the new video sequences and encodes it in an XML document. This document is used in conjunction with user input to provide the appropriate visualization of the video content.

2.1 Event Representation

This representation model effectively measures the quantity and type of changes occurring within a scene under the assumption that similar events (e.g. running, walking, waving) will have similar distributions of change for each of the three dimensions considered. A video event is represented as a set of x, y and t intensity gradient histograms over several temporal scales. Temporal scales are used to model the granularity of motion in an event (e.g. in a high-granularity temporal scale local motion like the swaying of the arms while walking will be captured while a low-granularity temporal scale will capture the overall motion of the subject across the frame). In our example we take four different temporal scales each with half the number of frames as the one preceding it. Temporal scale 1 is usually composed of all frames in the video sequence. For example, if Video Sequence V1 is 120 frames long then the intensity value matrix of V1

- at temporal Scale 1 includes frames 1,2,…..120 (120 frames)
- at temporal Scale 2 includes frames 1,3,5…..120 (60 frames)
- at temporal Scale 3 includes frames 1,5,9…..120 (30 frames)
- at temporal Scale 4 includes frames 1,9,17,…120 (15 frames)

Once we obtain these 3D intensity value matrices for each temporal scale we are able to calculate the intensity gradient for each of the three directions. Intensity gradient values measure the change in the intensity in each of the three dimensions (x,y,t). Histograms of these values represent the distribution of change throughout the scene. A complete representation of an event is twelve of these 256 bin histograms (one for each of the (x,y,t) dimensions over 4 temporal scales). The gradient for each dimension is computed by averaging the difference between current value and next/previous value on the appropriate dimension. Figure 2 illustrates this process for a 2-dimensional matrix.

2.2 Extracting the Event Representation

After reading in the video segment representing the event and extracting the 3D intensity matrix we compute the gradient of these values in the x, y and t directions for each of the four temporal scales. We then transform these values by taking their absolute value and normalizing the gradient vector to a length of 1. Before the histograms are computed (x,y,t) points where the t gradient is not above a certain threshold (these points are less relevant to the event) are taken out of consideration.

Figure 2 illustrates the calculation of the x and y gradient matrix from the original intensity value matrix a. Because our data is three dimensional we can extended this method to calculate the t gradient matrix as well.

Histograms are computed by taking the range of these 12 normalized gradient matrices (x,y,t * 4 temporal scales) and dividing it into 256 equal bins. A count of the gradient values that fall in each one of these bins is the histogram value at that bin’s.
index. The y axis corresponds to the number of pixels whose gradient values fall into a particular bin and the x axis corresponds to the bin index. Each bin represents an equal interval of intensity gradient values. The cumulative interval of the bins spans the entire range of intensity gradient values. These histograms are, thus, fairly simple to plot, visualize and compare.

The histograms of longer length video segments will have larger magnitude peaks and thus a larger area. In order to determine whether they represent the same event (to determine whether they represent the same event) we will need to normalize this area to equal 1.

After normalizing the area of the histogram the final step is smoothing the values. This is done to minimize the effect that local mismatches have on the comparison function. We do this by walking down the histogram and setting each bin value to the average value of the bins in its neighborhood (usually 5 bins on either side). While this method produces some artifacts such as a gentle slope towards the gradient extremes, it has proved to be effective in improving the comparison measure of important regions within the histograms.

### 2.3 Event Comparison

Once we have a 12 histogram representation of a video segment we are tasked with comparing it to the representation of others to determine if the two represent similar high-level semantic concepts (events). For this purpose a simple squared distance measure equation has been devised:

$$D^2 = \frac{1}{3L} \sum_{k,i} \left[ h_{1k}^i(i) - h_{2k}^i(i) \right]^2$$

where $L$ is the number of temporal scales (4 in our experiments) and $h(i)$ is the value of the histogram bin at index i. $k$ represents the different dimensions (x,y,t) and 1 represents the 4 different temporal scales.

This measure averages the difference between each bin in the histogram of each dimension of each temporal scale over the total number of histograms. This produces a number that indicates how close the two compared events are to one another. The lower this number is the closer the two events are. A distance of zero results when an event is compared to itself.

This comparison function enables the use of the nearest neighbor classification algorithm to label an unknown video sequence with the label of the known event closest to it (in $D^2$ distance).

### 2.2 Event Detection

Having a good event representation and event comparison function allows us to begin to detect and classify events in a new video sequence. However, the process of determining which video frames constitute an event and how to separate adjacent events in a video sequence still requires definition. The event detection problem is, thus, defined in two different ways in this paper. The first is the simplified problem. In this problem we are given a video sequence that we know to contain a single event. We are to classify this event using our representation scheme and comparison function. In the second, more complex, interpretation of the problem we are given a video sequence containing an unknown number of events and are to determine which events occur and where (what frame numbers) within the video they occur.

The first step in our approach to both versions of this problem is generating a set of video segments (and their histogram representations) and manually classifying these. One or more of these “Classifiers”, as we have dubbed them, exists for each possible event in the scene. To improve accuracy multiple classifiers representing the same type of event should be different lengths, occur in different geographical regions of the scene and mirror each other’s direction of motion.

In the “simple”, one event per video segment, problem a new event is classified using the label of the classifier with the closest similarity to itself (the smallest distance value).

The broader problem, where an event can occur at any point within the video sequence, is a bit more challenging. To solve this problem we use a sliding window through the video sequence to capture fixed length events. In our experiment the window length was 25 frames with a skip interval of 8 frames. This means that each of the video windows created overlaps several other windows. A histogram event representation is generated for each of these windows and compared to the classifiers. Similar to the above problem each window is labeled according to the classifier it is most similar too. However, if this minimum distance is above a certain threshold (i.e. we cannot say this event is similar to any of the classifiers) the window is labeled as unknown. Each frame is then labeled with the mode classification of all the windows which contain it. This allows us to derive a frame by frame description/annotation of the events occurring within a given video segment. Figure 3 demonstrates this concept.

This paper explores this framework component further in later sections. We assess the performance of the techniques discussed above in section 3 and propose an experiment to optimize efficiency in section 4.

### 2.3 XML Video annotation

Once a video sequence undergoes the event detection described in the previous section the events contained within it are stored in XML format. Because this format is easily machine readable the analyst is now able to sort through the video data much more efficiently (using the Video Analysis Tool). XML is also human-readable and thus manual viewing of event-content summary may also yield a good description of the video segment in question. The XML document contains a reference to the video data file, video segment specific attributes (such as video length and file format), and data on each of the events occurring within the video segment. Video events are modeled as complex elements and include such sub-elements as event type, event starting point and event duration.

This XML document annotation might on some future date be replaced by a more robust computer-understandable format (for example: the VEML video event ontology language). This will enable a deeper level of event representation within video segments by making use of relationship cardinality, context definition and other features available within an ontology language.
Figure 3 illustrates the method used to detect events in a video sequence with an unknown number of events. Fixed length overlapping windows are compared to “classifier” events. Their event type classification is then used to determine the event type of individual frames.

2.4 A Video Analysis Tool

The availability of machine-readable annotation documents, whether these are in XML or a specialized video content ontology language format, is a big step towards the bridging of the semantic gap. A video analysis tool that takes this kind of annotation as input and organizes the corresponding video segment accordingly is certainly conceivable. This kind of utility could function as an aid to a surveillance analyst searching for “Suspicious” events within a stream of video data.

This activity of interest may be defined dynamically by the analyst during the running of the utility. The definition would then be compared to the video annotation and similar frames could be flagged for further analysis. Alternately, the analyst could define what is considered to be normal behavior and any annotated event deviating from this norm would be flagged.

3. PRELIMINARY EVALUATION OF CLASSIFICATION TECHNIQUE

We evaluated our classification technique with two different experiments. For both of these experiments we provided the system with manually labeled video clips (and their event representations) of three types (walking, running, and waving). These “classifier” video clips all portrayed the same actor on the same background (i.e. only the actions were variable). “Classifier” clips were of variable length, directionality of the event (e.g. left-to-right, right-to-left), and geographic location (e.g. waving occurred in different parts of the scene). We used Matlab [5] to generate the gradient intensity values over four temporal scales. From the resulting matrices we constructed the histogram representation for each event. We used the histogram distance function (section 2) to compare event representations.

The first experimental problem we defined was to recognize and classify events irrespective of direction (right-to-left, left-to-right) and with reduced sensitivity to spatial variations (Clothing). The main assumption made in this experiment is that unlabeled video sequences are known to contain only one event. The unlabeled events used for testing the system showed the same actor performing the same type of events on the same background as in the “classifier” videos, while wearing a different outfit or “disguise” to try and fool the system. A representation of each of these new “disguised” events is generated and compared to each of the classifiers using the comparison function described above.

The new event is then classified using the label of the classifier with the closest similarity to itself (the smallest distance value). We contrasted this classification with the “truth” (manual labeling) of the “disguised” events to evaluate the accuracy of the classification scheme.

The analyst would then consider all flagged areas to determine if action is required. A color coated scroll bar (with video segments of interest indicated by different colors) may be used as a graphical interface for this task. The analyst may then further refine the search parameters by marking false and true positives. Figure 4 shows a screen shot from a prototype of such a tool.
Table 1
Comparison of various “Disguised” event representation to the “classifier” events. Minimum distance is highlighted.

<table>
<thead>
<tr>
<th>Disguise</th>
<th>Disguise</th>
<th>Disguise</th>
<th>Disguise</th>
<th>Disguise</th>
<th>Disguise</th>
<th>Disguise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>walking1</td>
<td>walking2</td>
<td>running1</td>
<td>running2</td>
<td>running3</td>
<td>waving1</td>
</tr>
<tr>
<td>Walking</td>
<td>0.97653</td>
<td>0.948</td>
<td>1.411</td>
<td>1.3543</td>
<td>1.3049</td>
<td>13.646</td>
</tr>
<tr>
<td>walking2</td>
<td>0.45154</td>
<td>0.38097</td>
<td>1.3841</td>
<td>1.1909</td>
<td>1.0021</td>
<td>13.113</td>
</tr>
<tr>
<td>walking3</td>
<td>0.59608</td>
<td>0.53852</td>
<td>1.0637</td>
<td>1.0071</td>
<td>0.88092</td>
<td>13.452</td>
</tr>
<tr>
<td>running1</td>
<td>1.5476</td>
<td>1.9412</td>
<td>0.56724</td>
<td>0.61541</td>
<td>0.8114</td>
<td>18.615</td>
</tr>
<tr>
<td>running2</td>
<td>1.4633</td>
<td>1.844</td>
<td>0.97417</td>
<td>0.95833</td>
<td>1.1042</td>
<td>19.592</td>
</tr>
<tr>
<td>running3</td>
<td>1.5724</td>
<td>1.8711</td>
<td>0.93587</td>
<td>0.94227</td>
<td>1.1189</td>
<td>18.621</td>
</tr>
<tr>
<td>running4</td>
<td>1.5406</td>
<td>1.9673</td>
<td>1.0957</td>
<td>0.93731</td>
<td>1.0902</td>
<td>20.239</td>
</tr>
</tbody>
</table>

Table 1 shows the distance between each of the “disguised” events (columns) and the “classifier” events (rows). Highlighted cells indicate the minimum distance “classifier” event, whose label the system will use to label the “disguised” event.

This method yielded 100% Precision (i.e. all disguised events were classified correctly). These results are, of course, not necessarily representative of the general event detection problem. Future evaluation with more event types, more varied data and a larger set of training and testing data is needed (section 4).

The second experiment evaluated the scenario where an unlabeled video can contain any number of events. The goal of this experiment was to generate an accurate description of the high-level events within the unlabeled video. A sliding window was used to capture overlapping events of a fixed length (25 frames). Each event’s starting point was 8 frames separated from the previous event.

Each of these window events is then classified in the same manner as the first experiment. The combined labeling of overlapping windows is used to determine the labeling for each video frame.

Figure 5 illustrates the minimum distance graph used to generate the video labeling. When watching the video sequence along with annotated event labeling it was observed to be similar to human perception. This is a subjective measure of performance and we have not yet developed a reliable metric for measuring the accuracy of such a video description. One possible option is to manually label each frame in the video sequence and compare this labeling to the automatic labeling frame by frame. This, however, is labor intensive and sensitive to the subjective perception of humans during the labeling.

4. THE PROPOSED EXPERIMENT
We propose an experiment utilizing the event representation in [1] to determine the machine learning algorithm that will achieve optimal results in classifying unlabeled video segments based on the training data. As a preprocessing step we consider each of the 1024 (256*4) histogram bins and their value as a feature, resulting in a 1024 dimension feature vector (the traditional input for machine learning algorithms). We can further use techniques such as principal component analysis to reduce the dimensionality; however, in this experiment, we will use the raw feature vector.

4.1 The Data
Each classifier will be fed a video sequence of length one to ten seconds featuring one of three events (walking, running, and waving). Approximately 20 video clips will be collected and used to test each algorithm. The event videos will each contain one actor against a (relatively) motionless background performing one of the three actions. The various actors in the event videos will be of different sizes, genders and ethnicities and will be wearing different clothing. Left to right as well as right to left examples of “directional” events (e.g. walking, running) will be included in the video data. Different backgrounds will also be used throughout.
the various clips. Video data will be collected using a Canon Z100 camcorder.

4.2 Methods
- Video data will be processed into four histogram representation using Matlab
- Histograms will be reduced into feature vector
- Feature vectors of 19 (out of 20) clips will be used to train classifier (leave one out cross validation)
- Feature vector of remaining clip will be used to test the classifier
- Output label produced by classifier will be compared against ground truth (manual labeling)
- Repeat these steps for each of the video clips
- Calculate precision of classifier (correct classifications/all examples)
- Calculate the confusion matrix of classifier (between events)

4.3 The Classifiers
The machine learning algorithms to be evaluated as classifiers in this experiment are as follows:
- Nearest Neighbor Algorithm (Histogram Distance Function) - This is the classification algorithm proposed in [1] by Zelnik-Manor et al. A preliminary evaluation of its performance is detailed in the previous section.
- Nearest Neighbor Algorithm (Euclidean Distance) – This algorithm computes Euclidean distance between all feature vectors and selects the nearest Euclidean neighbor.
- Neural Networks- A network of perceptrons containing 1024 inputs 1 output and multiple hidden layer nodes. Trained through the backpropagation algorithm.
- Decision Tree- This algorithm constructs a series of decision points based on features with minimal entropy.

5. DISCUSSION
As is shown in section 3 (albeit on a minimal dataset) the Nearest Neighbor classification algorithm using the histogram distance measure proposed in [1] performs very well in this kind of experiment. It will be interesting to see whether this performance level is maintained as the dataset size is increased and the training clips are more variable (i.e. different actors and backgrounds).

Another interesting question is whether comparing histograms as feature vectors (rather than taking the average difference of corresponding bins) will degrade performance. Highly dimensional data can sometimes be problematic for decision tree algorithms, as individual features have only a small contribution to the classification of the data. Neural Networks, by contrast, have shown to be quite successful in recognizing patterns from high dimensional examples; however, it is not known if our limited dataset will suffice to properly train the network. Other classifying algorithms, not discussed in the proposed experiment such as support vector machines and naïve Bayesian might also be used. However, for these to work some extra processing steps are necessary (defining a reliable probability distribution for Bayesian Networks and atomic concepts for support vector machines). The adaptation of the video data representation discussed in this paper for classification of video events using these methods is an interesting area for further exploration.

Refining the event classification component of our proposed framework for a Video Analysis Tool for Suspicious Event Detection will increase efficiency and thus improve the productivity of the video analyst.

6. SUMMARY AND DIRECTION
In this paper we proposed a framework for a video analysis tool for suspicious event detection. This tool is designed to reduce the demanding task of manually sorting through hours of surveillance video sequentially to determine if suspicious activity has occurred. We discussed the ideas behind the various components of this framework in detail as well as some of the implementation specifics.

We presented the results of some preliminary experiments we conducted to gauge the potential of this schema. The results of these experiments were promising but more in-depth analysis must be conducted before passing judgment on the representation, learning and classification schemes discussed in this paper. For this reason, we proposed an experiment that would contrast the performance of different learning algorithms and determine if we can improve upon the current technique. We provided some hypothesis on the results of such an experiment and contrast the strengths and weaknesses of the various algorithms.

In the continuation of this research path we will carry out the proposed experiments and report on the results and how they conform to our hypothesis. The addition of an access control component to restrict access to video content based on semantic events to our framework is another area that we are interested in exploring further.

7. REFERENCES
ABSTRACT
Automated surveillance methods frequently rely on algorithms which detect the presence of suspicious keywords and topics within messages to properly flag suspicious content for review. However, subsequent messages based on the original may not carry the same characteristics that were initially detected. A correlation algorithm is necessary to find such messages and thread a conversation together. Within this document, we propose such an algorithm and an experiment with which to test it.

General Terms
Algorithms, experimentation.

Keywords
Singular value decomposition, noise filtering, text correlation, surveillance, threat detection.

1. INTRODUCTION
One of the biggest challenges in automated message surveillance is the recognition of messages containing suspicious content. A classic approach to this problem is constructing a set of keywords (i.e., ‘bomb’, ‘nuclear’). In the event that a communiqué contains one or more of these words, the message is flagged as suspicious for further review.

However, there are two drawbacks to this particular approach. First, it is reasonable to assume that such relatively static keywords will not always be present in messages that would otherwise warrant suspicion. Second, there is little guarantee that a sufficiently intelligent individual will not recognize such surveillance is in place and, instead, use substitute words in place of known keywords.

David Skillicorn of Queens University has suggested a different approach in his work on the Enron e-mail dataset [1]. In his work, he outlines a method for using singular value decomposition (SVD) in the interest of recognizing trends in such topics as e-mail and social networks. Within his work, he constructed a matrix composed of word frequencies associated with each message, ranked by global word rank. We believe that this work can be utilized in our system by extrapolating some of the techniques he used and applying them to a real-time message monitoring system.

We propose an experiment that makes use of the word frequency techniques outlined in Mr. Skillicorn’s work. By using the Enron e-mail dataset and altering the role of his use of the SVD, we believe we can achieve a high degree of message correlation accuracy based on the intersection of uncommon words across messages.

2. BACKGROUND
The ultimate goal of our research is to construct a system that can determine the presence of anomalous or suspicious messages within typical communication traffic [2]. Our areas of research include social network analysis, text processing, and result filtering. This experiment will determine the effectiveness of singular value decomposition as applied as a text filtering technique in our system.

One particular approach we use in our work in message surveillance involves the recognition of social context. Given a message that was deemed suspicious by a detection technique, we assume there is an increased probability that future messages bearing similar characteristics to the detected message warrant further suspicion. We track this through the generation and passing of tokens carrying these characteristics in a model derived from observed social interaction within the message-passing network. Currently, these tokens are triggered by the presence of two or more keywords from a list tailored to our dataset. By using the techniques described by David Skillicorn, we believe it is possible to enhance the token passing methods of our approach.
The Enron e-mail dataset was originally made available by the Federal Energy Regulatory Commission during its investigation of the company. It was purchased by MIT for use in data analysis, and is currently available in both raw and compiled database forms from Carnegie Mellon University, courtesy of William Cohen. This dataset is useful to e-mail surveillance research as it is a freely available collection of e-mail that has been generated from a 'real world' scenario. After pruning and restructuring for consistency, the overall corpus is composed of 250,000 e-mails that span a five-year period.

Each e-mail remains in its' original raw ASCII-encoded text format, conforming to the RFC 2822 standard. The simplicity of this specification and the data it requires for proper mail transport assists with analysis in a number of ways. First, every e-mail must contain a unique identification number assigned by the originating server. Second, each message must have a well-defined origin and destination address, with the exception of any group or mass-mailing aliases. Third, every message has an explicit date stamp that reflects when it was originally sent. Finally, the body text of each e-mail is free of any escape sequences or special characters. With the exception of HTML-encoded e-mail, this greatly simplifies any preprocessing necessary for text parsing.

The content of the Enron dataset is its' most useful aspect. We understand that, at some point in time, e-mail began circulating that eventually led to the investigation of the entire corporation. This has been documented by countless news reports, committees, and even other research groups. Coupled with the official announcement on January 9th, 2002 that the United States Department of Justice was beginning it's official inquiry, we have definitive moments and individuals that can be scrutinized to determine the overall effectiveness of our experiment.

Singular value decomposition is an important tool in statistical analysis. Given a \(mn\) matrix \(A\) of real or complex numbers, we can decompose it into a series of component matrices: \(U\), \(\Sigma\) and \(V^T\). \(U\) represents the patterns among values contained among the objects. In this experiment, the objects are the messages. \(V\) embodies the patterns among the attributes of each object. This will be the patterns among the ranks of words within the messages. The \(\Sigma\) matrix is a diagonal matrix confirming to the dimensions \(n \times n\) that stores the singular values of \(A\). Essentially, the most 'interesting' parts of the original matrix are made evident [4].

![Figure 2. Correlation of a single message across the decomposition](image)

Once the SVD process has been carried out, the resulting matrices can be used for a variety of purposes. One particular way to use the results is the elimination of deviations from the patterns to filter out noise. We define noise among our messages as misspelled words, accidentally insertion of punctuation, and anything else a user may accidentally insert within their message that cannot be removed automatically within preprocessing in a timely and effective manner. Other uses include object correlation, signal processing, and even text retrieval [3]. Most of the uses for SVD stem from the fact that the components, once modified, can be used to build a new matrix that contains only the most 'interesting' qualities of the original.

### 3. PROPOSED APPROACH

Our existing system [2] has been outfitted with a new detection module based on the SVD method. The algorithm within the module requires a ranked word database, one or more messages that share the same token, and a set of messages that have originated from them representing recent traffic. A matrix is constructed based on the word content of the messages, which is decomposed according to the SVD method. A threshold is set for the noise, and the results are analyzed for word correlation. If a message in the recent traffic set has a strong correlation with one or more of the token holding messages, the token is passed to all recipients of the message.

We have made a number of assumptions based on the characteristics of our dataset. First, we assume that the content among messages is not obscured through covert communication techniques, such as encryption or word replacement. Second, we assume that on any given topic there are a number of words that are rarely used outside of the text. This includes subjects, terminology, and any form of slang. Finally, we assume that subsequent conversations based on an original message will not always carry the same unique nouns that would otherwise make detection simple. This is made simply to assume that a basic anomaly detection algorithm is insufficient, necessitating the use of a correlation algorithm.

To determine the rank of a particular word, a running word rank database is kept. The 5,000 most common words within the Brown Corpus of Standard American English are used to seed the database. As words are extracted from messages, each word and the number of times it occurred within the message is either
inserted into the database if the word is new or used to update the existing word rank.

The matrix constructed is created based on the number of messages involved and the words present. Each message represents a column, while each row represents a word. Thus, each cell represents the number of times a particular word occurs within a particular message. The columns are ordered from most common (left) to least common (right). Mathematically, this is represented as follows:

\[
\begin{align*}
c_{ij} &= count(w_i, m_j) \\
W &= M_1 \cup M_2 \ldots \cup M_t, \\
w_i &\in W
\end{align*}
\]

where \( W \) represents the set of all words that occur in the union of the word sets for each message \( M \). The function \( count \) returns the number of times a particular word \( w_i \) occurs within a message \( m_j \). Note that \( M_j \) is derived from the words in \( m_j \); however, \( m_j \) may contain the same word more than once while \( M_j \) is a list that strictly represents all words only once.

Once the SVD technique has been used on the messages, the noise is removed through the use of a threshold ( \( \sigma = 0.25 \)). After the singular value dimensions falling below this threshold are removed, the matrix is re-constructed from the components. The resulting matrix should assist in simplifying the amount of data that must be analyzed. The correlation of any two messages is based on a total score. Each word that is found in both messages is given a rating according to the following equation:

\[
\text{score}(w_i) = \frac{\text{count}(w_i, m_j) + \text{count}(w_i, m_k)}{\text{rank}(w_i)}
\]

where \( w_i \) is the word that occurs in messages \( m_j \) and \( m_k \). The \( count \) function returns the number of occurrences of the word within a supplied message. The \( rank \) function is a cumulative count of the occurrence of the word \( w_i \) in all messages up to this point. This equation is designed to place emphasis on words that occur less frequently others. It is assumed that rare words that occur in both messages are more likely to be good candidates for a topical match. Note that \( s_i \) is naturally normalized to one. Given that the rank of a word is adjusted before these messages are processed with this equation, the maximum score will be one. In order to determine if two messages match, the following equation is used:

\[
\sum_{w_i \in W \cap W_j} \text{score}(w_i) > \alpha
\]

where \( W_j \) and \( W_i \) are the set of all words contained within \( m_j \) and \( m_i \), respectively, and \( \alpha \) represents a predetermined threshold that determines the minimum. In theory, this should be some value that is determined by statistical analysis of known correlating messages. For our experiment, we assume \( \alpha = 4.00 \).

4. FUTURE DIRECTIONS

Once this experiment has been run successfully, an adequate infrastructure will exist to fully exploit the potential of singular value decomposition. After the noise has been eliminated, we could use the patterns to observe topical correlation between messages. Given that our overall system includes social network analysis, it may become possible to determine suspicious messages as those that deviate from the patterns without the use of suspicious keywords.

Another approach this could be used for is bridging potential gaps in our analysis due to users that use another form of communication (i.e. the telephone). The pattern analysis techniques could allow us to potentially determine when messages exchanged independently are intrinsically linked due to other social connections. Should there be a significant clustering among these conversations, it may even be possible to derive the presence of social groups [5, 6].

There are a number of alternative methods that decompose a matrix into components. One such technique is Principle Component Analysis, a technique that yields similar results to the SVD by attempting to capture the majority of variability in the data. If we construct our code in an abstract fashion, we will be able to use different approaches to the same matrix to enhance the overall effectiveness of our system [4].

Beyond text processing, we are currently investigating the use of matrix decomposition as applied to graph theory. Given a matrix composed of nodes in a graph and the values which represent the strength of the unidirectional links between them, other communication patterns could be discovered.

4. REFERENCES


A Similarity Measure for Motion Stream Segmentation and Recognition*

Chuanjun Li B. Prabhakaran
Department of Computer Science
The University of Texas at Dallas, Richardson, TX 75083
{chuanjun, prab}@utdallas.edu

ABSTRACT

Recognition of motion streams such as data streams generated by different sign languages or various captured human body motions requires a high performance similarity measure. The motion streams have multiple attributes, and motion patterns in the streams can have different lengths from those of isolated motion patterns and different attributes can have different temporal shifts and variations. To address these issues, this paper proposes a similarity measure based on singular value decomposition (SVD) of motion matrices. Eigenvector differences weighed by the corresponding eigenvalues are considered for the proposed similarity measure. Experiments with general hand gestures and human motion streams show that the proposed similarity measure gives good performance for recognizing motion patterns in the motion streams in real time.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications – Data Mining

General Terms: Algorithm

Keywords: Pattern recognition, gesture, data streams, segmentation, singular value decomposition.

1. INTRODUCTION

Motion streams can be generated by continuously performed sign language words [14] or captured human body motions such as various dances. Captured human motions can be applied to the movie and computer game industries by reconstructing various motions from video sequences [10] or images [15] or from motions captured by motion capture systems [4]. Recognizing motion patterns in the streams with unsupervised methods requires no training process, and is very convenient when new motions are expected to be added to the known pattern pools. A similarity measure with good performance is thus necessary for segmenting and recognizing the motion streams. Such a similarity measure needs to address some new challenges posed by real world motion streams: first, the motion patterns have dozens of attributes, and similar patterns can have different lengths due to different motion durations; second, different attributes of similar motions have different variations and different temporal shifts due to motion variations; and finally, motion streams are continuous, and there are no obvious "pauses" between neighboring motions in a stream. A good similarity measure not only needs to capture the similarity of complete motion patterns, but also needs to capture the differences between complete motion patterns and incomplete motion patterns or sub-patterns in order to segment a stream for motion recognition.

As the main contribution of this paper, we propose a similarity measure to address the above issues. The proposed similarity measure is defined based on singular value decomposition of the motion matrices. The first few eigenvectors are compared for capturing the similarity of two matrices, and the inner products of the eigenvectors are given different weights for their different contributions. We propose to use only the eigenvalues corresponding to the involved eigenvectors of the two motion matrices as weights. This simple and intuitive weighing strategy gives the same importance to eigenvalues of the two matrices. We also show that the 95% variance rule for choosing the number of eigenvectors [13] is not sufficient for recognizing both isolated patterns and motion streams. Our experiments demonstrate that at least the first 6 eigenvectors need to be considered for motion streams of either 22 attribute or 54 attributes, and the first 6 eigenvalues accounts for more than 99.5% of the total variance in the motion matrices.

2. RELATED WORK

Multi-attribute pattern similarity search, especially in continuous motion streams, has been widely studied for sign language recognition and for motion synthesis in computer animation. The recognition methods usually include template matching by distance measures and hidden Markov models (HMM).

Template matching by using similarity/distance measures has been employed for multi-attribute pattern recognition. Joint angles are extracted in [11] as features to represent different human body static poses for the Mahalanobis distance measure of two joint angle features. Similarly, momentum, kinetic energy and force are constructed in [2, 5] as activity measure and prediction of gesture boundaries for various segments of the human body, and the Mahalanobis distance function of two composite features are solved by dynamic programming.
Similarity measures are defined for multi-attribute data in [6, 12, 16] based on principal component analysis (PCA). Inner products or angular differences of principal components (PCs) are considered for similarity measure definitions, with different weighted strategies for different PCs. Equal weights are considered for different combinations of PCs in [6], giving different PCs equal contributions to the similarity measure. The similarity measure in [12] takes the minimum of two weighted sums of PC inner products, and the two sums are respectively weighted by different weights. A global weight vector is obtained by taking into account all available isolated motion patterns in [16], and this weight vector is used for specifying different contributions from different PC inner products to the similarity measure Eros. The dominating first PC and a normalized eigenvalue vector are considered in [7, 8] for pattern recognition. In contrast, this paper propose to consider the first few PCs, and the angular differences or inner products of different PCs are weighted by different weights which depends on the data variances along the corresponding PCs.

The HMM technique has been widely used for sign language recognition, and different recognition rates have been reported for different sign languages and different feature selection approaches. Starner et al. [14] achieved 92% and 98% word accuracy respectively for two systems, the first of the systems used a camera mounted on a desk and the second one used a camera in a user’s cap for extracting features as the input of HMM. Similarly Liang and Ouyoung [9] used HMM for postures, orientations and motion primitives as features extracted from continuous Taiwan sign language streams and an average 80.4% recognition rate was achieved. In contrast, the approach proposed in this paper is an unsupervised approach, and no training as required for HMM recognizers is needed.

3. SIMILARITY MEASURE FOR MOTION STREAM RECOGNITION

The joint positional coordinates or joint angular values of a subject in motion can be represented by a matrix: the columns or attributes of the matrix are for different joints, and the rows or frames of the matrix are for different time instants. Similarity of two motions is the similarity of the resulting motion matrices, which have the same number of attributes or columns, and yet can have different number of rows due to different motion durations. To capture the similarity of two matrices of different lengths, we propose to apply singular value decomposition (SVD) to the motion matrices in order to capture the similarity of the matrix geometric structures. Hence we briefly present SVD and its associated properties below before proposing the similarity measure based on SVD in this section.

3.1 Singular Value Decomposition

The geometric structure of a matrix can be revealed by the SVD of the matrix. As shown in [3], any real \( m \times n \) matrix \( A \) can be decomposed into \( A = U \Sigma V^T \), where \( U = [u_1, u_2, \ldots, u_m] \in \mathbb{R}^{m \times m} \) and \( V = [v_1, v_2, \ldots, v_n] \in \mathbb{R}^{n \times n} \) are two orthogonal matrices, and \( \Sigma \) is a diagonal matrix with diagonal entries being the singular values of \( A \): \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min(m,n)} \geq 0 \). Column vectors \( u_i \) and \( v_i \) are the \( i^{th} \) left and right singular vectors of \( A \), respectively.

It can be shown that the right singular vectors of the symmetric \( n \times n \) matrix \( A = A^T A \) are identical to the corresponding right singular vectors of \( A \), referred to as eigenvectors of \( M \). The singular values of \( M \), or eigenvalues of \( M \), are squares of the corresponding singular values of \( A \). The eigenvector with the largest eigenvalue gives the first principal component. The eigenvector with the second largest eigenvalue is the second principal component and so on.

3.2 Similarity Measure

Since SVD exposes the geometric structure of a matrix, it can be used for capturing the similarity of two matrices. We can compute the SVD of \( M = A^T A \) instead of computing the SVD of \( A \) to save computational time. The reasons are that the eigenvectors of \( M \) are identical to the corresponding right singular vectors of \( A \), the eigenvalues of \( M \) are the squares of the corresponding singular values of \( A \), and SVD takes \( O(n^3) \) time for the \( m \times n \) and takes \( O(mn^2) \) time with a large constant for the \( m \times n \) and usually \( m > n \).

Ideally, if two motions are similar, their corresponding eigenvectors should be parallel to each other, and their corresponding eigenvalues should also be proportional to each other. This is because the eigenvectors are the corresponding principal components, and the eigenvalues reflect the variances of the matrix data along the corresponding principal components. But due to motion variations, all corresponding eigenvectors cannot be parallel as shown in Figure 1. The parallelness or angular differences of two eigenvectors can be described by the absolute value of their inner products: \( \|u_i \cdot v_i\| = \|u_i\| \cdot \|v_i\| \cdot \cos \theta = \|u_i\| \cdot \|v_i\| \cdot \frac{\|u_i\| \cdot \|v_i\|}{\|u_i\| \cdot \|v_i\|} \), where \( u_i = v_i = 1 \). We consider the absolute value of the inner products because eigenvectors can have different signs as shown in [8].

Since eigenvalues are numerically related to the variances of the matrix data along the associated eigenvectors, the importance of the eigenvector parallelness can be described by the corresponding eigenvalues. Hence, eigenvalues are to be used to give different weights to different eigenvector pairs. Figure 2 shows that the first eigenvalues are the dominating components of all the eigenvalues, and other eigenvalues become smaller and smaller and approach zero. As the eigenvalues are close to zero, their corresponding eigenvectors can be very different even if two matrices are similar. Hence not all the eigenvectors need to be incorporated into the similarity measure.

Since two matrices have two eigenvalues for the corresponding eigenvector pair, these two eigenvalues should have equal contributions or weights to the eigenvector parallelness. In addition, the similarity measure of two matrices should be independent to other matrices, hence only eigenvectors and eigenvalues of the two matrices should be considered.

Based on the above discussions, we propose the following similarity measure for two matrices \( Q \) and \( P \):

\[
\Psi(Q, P) = \frac{1}{2} \sum_{i=1}^{k} ((\sigma_i/\lambda_i) \sum_{i=1}^{n} \sum_{i=1}^{n} \lambda_i |u_i \cdot v_i|)
\]

where \( \sigma_i \) and \( \lambda_i \) are the \( i^{th} \) eigenvalues corresponding to the \( i^{th} \) eigenvectors \( u_i \) and \( v_i \) of square matrices of \( Q \) and \( P \), respectively, and \( 1 < k < n \). Integer \( k \) determines how many eigenvectors are considered and it depends on the number of attributes \( n \) of motion matrices. Experiments with hand gesture motions (\( n = 22 \)) and human body motions (\( n = 22 \)) show that \( k \) can range from 2 to 5.
In Section 4 show that \( k = 6 \) is large enough without loss of pattern recognition accuracy in streams. We refer to this non-metric similarity measure as \( k \) Weighted Angular Similarity (kWAS), which captures the angular similarities of the first \( k \) corresponding eigenvector pairs weighted by the corresponding eigenvalues.

It can be easily verified that the value of kWAS ranges over \([0,1]\). When all corresponding eigenvectors are normal to each other, the similarity measure will be zero, and when two matrices are identical, the similarity measure approaches the maximum value one if \( k \) approaches \( n \).

### 3.3 Stream Segmentation Algorithm

In order to recognize motion streams, we assume one motion in a stream has a minimum length \( l \) and a maximum length \( L \). The following steps can be applied to incrementally segment a stream for motion recognition:

1. SVD is applied to all isolated motion patterns \( P \) to obtain their eigenvectors and eigenvalues. Let \( \delta \) be the incremented stream length for segmentation, and let \( L \) be the location for segmentation. Initially \( L = l \).

2. Starting from the beginning of the stream or the end of the previously recognized motion, segment the stream at location \( L \). Compute the eigenvectors and eigenvalues of the motion segment \( Q \).

3. Compute kWAS between \( Q \) and all motion patterns \( P \). Update \( \Psi_{\text{max}} \) to be the highest similarity after the previous motion’s recognition.

4. If \( L + \delta < L \), update \( L = L + \delta \) and go to step 2. Otherwise, the segment corresponding to \( \Psi_{\text{max}} \) is recognized to be the motion pattern which gives the highest similarity \( \Psi_{\text{max}} \), update \( L = l \) starting from the end of the last recognized motion pattern and go to step 2.

### 4. PERFORMANCE EVALUATION

This section evaluates experimentally the performances of the similarity measure kWAS proposed in this paper. It has been shown in [16] that Eros [16] outperforms other similarity measures mentioned in Section 2 except MAS [8]. Hence in this section, we compare the performances of the proposed kWAS with Eros and MAS for recognizing similar isolated motion patterns and for segmenting and recognizing motion streams from hand gesture capturing CyberGlove and human body motion capture system.

#### 4.1 Data Generation

A similarity measure should be able to be used not only for recognizing isolated patterns with high accuracy, but also for recognizing patterns in continuous motions or motion streams. Recognizing motion streams is more challenging than recognizing isolated patterns. This is because many very similar motion segments or sub-patterns needs to be compared in order to find appropriate segmentation locations, and a similarity measure should capture the difference between a complete motion or pattern and its sub-patterns. Hence, both isolated motion patterns and motion streams were generated for evaluating the performance of kWAS. Two data sources are considered for data generation: a CyberGlove for capturing hand gestures and a Vicon motion capture system for capturing human body motions.

#### 4.1.1 CyberGlove Data

A CyberGlove is a fully instrumented data glove that provides 22 sensors for measuring hand joint angular values to capture motions of a hand, such as American Sign Language (ASL) words for hearing impaired. The data for a hand gesture contain 22 angular values for each time instant/frame, one value for a joint of one degree of freedom. The motion data are extracted at around 120 frames per second. Data matrices thus have 22 attributes for the CyberGlove motions.

One hundred and ten different isolated motions were generated as motion patterns, and each motion was repeated for three times, resulting in 330 isolated hand gesture motions. Some motions have semantic meanings. For example,
the motion for BUS as shown in Table 1 is for the ASL sign "bus". Yet for segmentation and recognition, we only require that each individual motion be different from others, and thus some motions are general motions, and do not have any particular semantic meanings, such as the THUMBUP motion in Table 1.

The following 18 motions shown in Table 1 were used to generate continuous motions or streams. Twenty four different motion streams were generated for segmentation and recognition purpose. There are 5 to 10 motions in a stream and 150 motions in total in 24 streams, with 6.25 motions in a stream on average. It should be noted that variable-length transitional noises occur between successive motions in the generated streams.

**Table 1: Individual motions used for streams**

| 35 60 70 80 90 BUS GOODBYE | HALF IDIOM JAR JUICE KENNEL KNEE | MILK TV SCISSOR SPREAD THUMBUP |

### 4.1.2 Motion Capture Data

The motion capture data come from various motions captured collectively by using 16 Vicon cameras and the Vicon iQ Workstation software. A dancer wears a suit of non-reflective material and 44 markers are attached to the body suit. After system calibration and subject calibration, global coordinates and rotation angles of 19 joints/segments can be obtained at about 120 frames per second for any motion. Similarity of patterns with global 3D positional data can be disguised by different locations, orientations or different paths of motion execution as illustrated in Figure 3(a). Since two patterns are similar to each other because of similar relative positions of corresponding body segments at corresponding time, and the relative positions of different segments are independent of locations or orientations of the body, we can transform the global position data into local position data as follows.

Let $X_p, Y_p, Z_p$ be the global coordinates of one point on pelvis, the selected origin of the "moving" local coordinate system, and $\alpha, \beta, \gamma$ be the rotation angles of the pelvis segment relative to the global coordinate system axes, respectively. The translation matrix is $T$ as follows:

$$T = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
-X_p & -Y_p & -Z_p & 1
\end{pmatrix}$$

The rotation matrix $R = R_x \times R_y \times R_z$, where

$$R_x = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & \cos\alpha & -\sin\alpha & 0 \\
0 & \sin\alpha & \cos\alpha & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}$$

$$R_y = \begin{pmatrix}
\cos\beta & 0 & \sin\beta & 0 \\
0 & 1 & 0 & 0 \\
-\sin\beta & 0 & \cos\beta & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}$$

$$R_z = \begin{pmatrix}
\cos\gamma & -\sin\gamma & 0 & 0 \\
\sin\gamma & \cos\gamma & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}$$

Let $X, Y, Z$ be the global coordinates of one point on any segments, and $x, y, z$ be the corresponding transformed local coordinates. $x, y$ and $z$ can be computed as follows:

$$[x \ y \ z] = [X \ Y \ Z \ 1] \times T \times R$$

The transformed data are positions of different segments relative to a moving coordinate system with the origin at some fixed point of the body, for example the pelvis. The moving coordinate system is not necessarily aligned with the global system, and it can rotate with the body. So data transformation includes both translation and rotation, and the transformed data would be translation and rotation invariant as shown in Figure 3(b). The coordinates of the origin pelvis are not included, thus the transformed matrices have 54 columns.

Sixty two isolated motions including Taiqi, Indian dances, and western dances were performed for generating motion capture data, and each motion was repeated 5 times, yielding 310 isolated human motions. Every repeated motion has a different location and different durations, and can face different orientations. Twenty three motion streams were generated for segmentation. There are 3 to 5 motions in a stream, and 93 motions in total in 23 streams, with 4.0 motions in a stream on average.

### 4.2 Performance of kWAS for Capturing Similarities and Segmenting Streams

We first apply kWAS to isolated motion patterns to show that the proposed similarity measure kWAS can capture the similarities of isolated motion patterns. Then kWAS is applied to motion streams for segmenting streams and recognizing motion patterns in the streams. We experimented with different $k$ values in order to find out the smallest $k$ without loss of good performance.

Figure 2 shows the accumulated eigenvalue percentages averaged on 330 hand gestures and 310 human motions, respectively. Although the first two eigenvalues account for
Figure 4: Recognition rate of similar CyberGlove motion patterns. When \( k = 3 \), \( kWAS \) can find the most similar motions for about 99.7% of 330 motions, and can find the second most similar motions for 97.5% of the them.

Figure 5: Recognition rate of similar captured motion patterns. When \( k = 5 \), by using \( kWAS \), the most similar motions of all 310 motions can be found, and the second most similar motions of 99.8% of the 310 motions can also be found.

more than 95% of the respective sums of all eigenvalues, considering only the first two eigenvectors for \( kWAS \) is not sufficient as shown in Figure 4 and Figure 5. For CyberGlove data with 22 attributes, \( kWAS \) with \( k = 3 \) gives the same performance as \( kWAS \) with \( k = 22 \), and for motion capture data with 54 attributes, \( kWAS \) with \( k = 5 \) gives the same performance as \( kWAS \) with \( k = 54 \). Figure 4 and Figure 5 illustrate that \( kWAS \) can be used for finding similar motion patterns and outperforms \( MAS \) and \( Eros \) for both hand gesture and human body motion data.

The steps in Section 3.3 are used for segmenting streams and recognizing motions in streams. The recognition accuracy as defined in [14] is used for motion stream recognition. The motion recognition accuracies are shown in Table 2. For both CyberGlove motion and captured motion data, \( k = 6 \) is used for \( kWAS \), which gives the same accuracy as \( k = 22 \) for CyberGlove data and \( k = 54 \) for motion capture data, respectively.

Figure 6 shows the time taken for updating the candidate segment, including updating the matrix, computing the SVD of the updated matrix, and computing the similarities of the segment and all motion patterns. The code implemented in C++ was run on one 2.70 GHz Intel processor of a GenuineIntel Linux box. There are 22 attributes for the CyberGlove streams, and 54 attributes for the captured motion streams. Hence updating captured motion segments takes longer than updating CyberGlove motion segments as shown in Figure 6. The time required by \( kWAS \) is close to the time required by \( MAS \), and is less than half of the time taken by using \( Eros \).

Table 2: Stream Pattern Recognition Accuracy (%)

<table>
<thead>
<tr>
<th>Similarity Measures</th>
<th>CyberGlove Streams</th>
<th>Motion Capture Streams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eros</td>
<td>68.7</td>
<td>78.5</td>
</tr>
<tr>
<td>MAS</td>
<td>93.3</td>
<td>78.5</td>
</tr>
<tr>
<td>( kWAS ) (k=6)</td>
<td>94.0</td>
<td>94.6</td>
</tr>
</tbody>
</table>

4.3 Discussions

\( kWAS \) captures the similarity of square matrices of two matrices \( P \) and \( Q \), yet the temporal order of pattern execution is not revealed in the square matrices. As shown in [7], two matrices with the identical row vectors in different orders have identical eigenvectors and identical eigenvalues. If different temporal orders of pattern execution yield patterns with different semantic meanings, we need to further consider the temporal execution order, which is not reflected in the eigenvectors and eigenvalues and has not been considered previously in [6, 12, 16].

Since the first eigenvectors are close or parallel for similar patterns, we can project pattern \( A \) onto its first eigenvector \( u_1 \) by \( Au_1 \). Then similar patterns would have similar projections (called projection vectors hereafter), showing similar temporal execution orders while the projection variations for each pattern can be maximized. The pattern projection vectors can be compared by computing their dynamic time warping (DTW) distances, for DTW can align sequences of different lengths and can be solved easily by dynamic programming [1]. Incorporating temporal order information into the similarity measure can be done as for \( MAS \) in [7] if motion temporal execution orders cause motion pattern ambiguity to \( kWAS \).

5. CONCLUSIONS

This paper has proposed a similarity measure \( kWAS \) for motion stream segmentation and motion pattern recognition. \( kWAS \) considers the first few \( k \) eigenvectors and computes their angular similarities/differences, and weighs contributions of different eigenvector pairs by their correspond-
ing eigenvalues. Eigenvalues from two motion matrices are given equal importance to the weights. Experiments with CyberGlove hand gesture streams and captured human body motions such as Taiqi and dances show that kWAS can recognize 100% most similar isolated patterns and can recognize 94% motion patterns in continuous motion streams.

6. REFERENCES


Analyzing user’s behavior on a video database

Sylvain Mongy
LIFL - UMR CNRS 8022
Univ. de Lille1, Bât. M3 59655
Villeneuve d’Ascq Cedex – FRANCE
+33 (0) 3 20 33 55 17
mongy@lifl.fr

Fatma Bouali
STID
25-27 rue du Maréchal Foch 59100
Roubaix - FRANCE
fatma.bouali@univ-lille2.fr

Chabane Djeraba
LIFL - UMR CNRS 8022
Univ. de Lille1, Bât. M3 59655
Villeneuve d’Ascq Cedex – FRANCE
+33 (0) 3 20 33 55 64
djeraba@lifl.fr

ABSTRACT
The analysis of user behaviors in large video databases is an emergent problem. The growing importance of video in everyday life (ex. Movie production) is bound to the importance of video usage. In order to cope with the abundance of available videos, users of these videos need intelligent software systems that fully utilize the rich source information hidden in user behaviors on large video data bases to retrieve and navigate through videos. In this paper, we present a framework for video usage mining to generate user profiles on a video search engine in the context of movie production. We suggest a two levels model based approach for modeling user behaviors on a video search engine. The first level aims at modeling and clustering user behavior on a single video sequence (intra video behavior), the second one aims at modeling and clustering user behavior on a set of video sequences (inter video behavior). Based on this representation we have developed a two phase clustering algorithm that fits these data.

Categories
G.3 [Time series analysis]
I.5.3 [Clustering]

General Terms
Algorithms, Human Factors.

Keywords
Video data mining, User behavior analysis, Hierarchical clustering, Markovian model.

1. INTRODUCTION
With the fast development in video capture, storage and distribution technologies, digital videos are more accessible than ever. The amount of these archives is soaring. To deal with it, video usage mining, which aims at analyzing user behaviors on a set of video data, is one of the key technologies to create suitable tools to help people browsing and searching the large amount of video data. Indeed, as in web mining field the extracted information will enable to improve video access.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

In this paper, we present a framework (see Figure 1) that combines intra-video usage mining and inter-video usage mining to generate user profiles on a video search engine in the context of movie production. Specifically, we have borrowed the idea of navigation history from web browsers used in web usage mining, and suggest a new approach that defines two types of log from the log data gathered on a video search engine. The first one concerns the way a user views a video sequence (play, pause, forward.…. and can be called intra-video usage mining. At this level we define the “video sequence viewing” as a behavior unit. The second type of log tracks the transitions between each video sequence viewing. This part gathers requests, results and successive viewed sequences. At this higher level, as in web mining we introduce a “session” like a behavior unit.

This paper is organized as follows. In section 2, we present the related work in video usage mining. Section 3 presents our two levels model based approach for modeling users’ behaviors on a video search engine. Finally, section 4 describes the evaluation of the technique on some test datasets, and section 5 gives the conclusion and some future directions.

Figure 1. Overview of the video usage mining framework
2. RELATED WORK
In the absence of any prior survey of video usage mining, the closest related work can be classified into roughly two types.

The first type of work concerns the analysis of user behaviors without considering the video content. These works report on statistics of user behavior and frequency counts of video access. For example, [3] analyzes the student usage of an educational multimedia system. This analysis is based on the student personality types. Indeed, the learning needs and expectations depend on the characteristics of the student personality type. To achieve this, the authors have developed a program that extracts the student’s actions on the multimedia system and profiles what each user did each time he has used the system. These user profiles include the following statistics: number of video viewing sessions, total seconds spent viewing videos; number of video viewing sessions that lasted more than 20 minutes, average duration of a video viewing session, average number commands per minute during video viewing sessions, forward transitions, backward transitions, forward jumps and jump ratio. While being based on the statistics collected on each type of students, they analyze how the learning multimedia system can be improved to remedy its shortcomings.

[1] presents an analysis of trace data obtained from user access on videos on the web. The authors examine properties such as how user requests vary on a day to day, and whether video accesses exhibit any temporal properties. They propose to benefit from these properties to design the multimedia systems such as web video proxy caches, and video servers. For example the analysis revealed that users preview the initial portion of a video to find out if they are interested. If they like it, they continue watching, otherwise they stop it. This pattern suggests that caching the first several minutes of video data should improve access performance.

The second type of work relates to the behavior analysis on a single video.

[6] presents a framework that combines video content analysis and user log mining to generate a video summary. They develop a video browsing and summarization system that is based on previous viewers browsing log to facilitate future viewers. They adopt the link analysis technique used in web mining, and propose a concept of ShotRank that measures the importance of each video shot, the user behavior is simulated with an Interest-guided Walk model, and the probability of a shot being visited is taken as an indication of the importance of that shot. The resulting ShotRank is used to organize the presentation of video shots and generate video skims.

The lack in the previous work is to correlate general behavior of the users with their behavior on each of the videos. They do not take into account actions done during a video viewing while considering navigation between video sequences. In short, these works are rather distant from our context. The navigation and research concepts in a large video data base are missing. Moreover, there are neither standards nor benchmarks on video log data.

Two important points differentiate our approach of present works. First, there are no tools working on usage of complete video database exploration. The only works we have referenced for the field of video analysis consider only a video at once. The nearest techniques are finally the one concerning Web Usage Mining. But
the log data are here more complete and we will be able to fully exploit these ones.

Secondly, we have developed a clustering technique that fits our data. Indeed, many Web Usage Mining techniques are based on distance-based clustering algorithms and neighborhood comparison. This leads to results that are hard to analyze. In such approaches, two sessions are associated to the same cluster if they are connected by a chain of very close neighbors, even if they are completely different. We introduce here a model to represent cluster that gathered information given by every element, each of these elements corresponding to this model.

3. PROPOSED APPROACH

3.1 Context
One of the needs of the professional users of Companies of audio-visual sector is to be able to find existing video sequences in order to re-use them in the creation of new films. Our approach is based on the use of a well suited video search engine (see Figure 2). Our tool is a classical browser for finding video in huge databases. Researches are executed on content-based indexing. Much hidden information can be extracted from the usage and used to improve the closeness between requests and videos returned by the search engine.

To achieve this task, we first need to define what a usage of a video search engine is. Such a behavior can be divided into three parts. – 1° Request creation: the user defines its search attributes and values. – 2° Result set exploitation: found sequences are presented to the user. They are ordered by an attribute-based confidence value. – 3° Selected sequences viewing: the user is viewing sequences he is interested in. This viewing is achieved with a video browser offering usual functions (play, pause, forward, rewind, stop, jump).

Groups of viewed sequences form sessions. They correspond to a visit of a user. They are compound of several searches and video sequences viewing.

3.2 Gathering data
All of these data are collected and written into log files. In order to create these files, we define a XML-based language. A session is gathered as followed. The first part contains the request executed and the list of video sequences returned. The second one logs the viewing of sequences.

Like web logfile, our video logfile traces the actions of users. To extract sessions, we have developed an XSLT (eXtensible Stylesheet Language Transformation) converter [4]. This converter extracts and regroups sessions from this logfile in XML format. The following part of the paper explains how we propose to model a video session.

3.3 Modeling user’s behavior: a two-level-based model
From the log data gathered previously, we generate two models to represent the user’s behavior. The first one concerns the way a user views a video sequence (play, pause, forward…). At this level we define the “video sequence viewing” as a behavior unit. The second one tracks the transitions between each video sequence viewing. This part gathers requests, results and successive viewed sequences. At this higher level, we introduce a “session” as a behavior unit.

Presently our work is only based on sequences. We do not take into account the information given by the requests. This will be further upgraded.

A session is a list of viewed video sequences. The particularity and the interest of the video log data will be the ability to define the importance of each sequence in each session. More than a simple weight, comparable to time comparison in web mining [5], we will here characterize several behavior types (complete viewing, overview, open-close, precise scene viewing). Based on these behaviors, viewing will be precisely defined and then we will be able to know which has been the use of a video in a session (whether if it is the most important or if it is just the result of a bad search).

3.3.1 Modeling and clustering intra video user’s behavior
An intra video user’s behavior is modeled by a first order non-hidden Markovian model (see Figure 3). This model represents the probability to execute an action each second of the viewing of a video. Each vertex represents one of the actions proposed to the user while viewing. For the first version of our tool, these actions are play, pause, forward, rewind, jump, stop. For example, the edge from play to pause means that when a user is playing a video, there is a probability of 8% that he executes a pause the next second.

This model is fully determined by the following parameters:
- \( V_i \) the vertices. \( N \) is set of the actions proposed to the user during a viewing. For the first version of our tool, we have set \( N = \{ \text{play}, \text{pause}, \text{forward}, \text{rewind}, \text{jump}, \text{stop} \} \).
- \( I_i \) the probability of starting in state \( i \). Here, and
- \( A_{ij} \) the transition probability from a state \( V_i \) to \( V_j \) during the next second. This discretization of time (taking the second for unit) is interesting because it considers time without any additional parameter.

Its limited complexity will allow us to propose an effective clustering method of these behaviors.

![Figure 3. Video sequence viewing behavior](image)
stays in use of probability instead of distance to associate viewings to clusters. We calculate the probability that a viewing has been generated by models. We then associate the viewing to the cluster with the highest probability.

Such algorithm can be split in three phases: initialization, expectation, maximization.

- **Initialization phase.** The initialization phase, like for K-Means, appears hard. Indeed it is important to define appropriate models to start the clustering. Here, we have some knowledge about the looked for clusters. First, Vi and Iii are already defined. The remaining problem is to determine the Aij. Even if data are varied, the ways of watching a video is quite limited and constant regarding to the type of video type. After some tests on real datasets, we are able to well define the initial models of the clusters, different enough from one to another and approximately corresponding to the resulted models.

- **Expectation phase.** For each sequence \( e = (e_1, \ldots, e_l) \) belonging to \( N \), of length \( l \), for each cluster determined by a model \( k \), we calculate the probability that \( k \) has generated \( e \). \( e \) is associated to the cluster with the highest probability.

- **Maximization phase.** Each model \( k \) representing a cluster \( c \) of size \( m \) is updated with respect to the data belonging to \( c \) \( \{e_1 \ldots e_m\} \). This update corresponds to count each transition in each element \( e_i \) and attributes these counts to the transition probability of the model. For each cluster \( c \), probabilities are updated this way.

Based on these discovered models, we create a vector of behavior for each viewing. This vector corresponds to the probabilities that the viewing has been generated by each model.

3.3.2 Modeling and clustering inter video user’s behavior

From the initial dataset and the vector created with the intra video clustering, we construct a sequential representation of the sessions. A session is a time-ordered sequence of the viewed video. Each viewing is characterized by a couple of the unique identifier of the video and the vector of behavior connected to it. Based on this representation of sessions, we have developed a clustering algorithm that fills the following requirements: - any element belonging to a cluster has a common part with any other element of the cluster. - The generality level of produced clusters relies on the definition of some parameters given by the user. These ones are presented next.

These requirements lead us to define the representation of a cluster this way: a cluster \( c \) is represented by a set of \( S \) sessions \( sc \) of minimal length \( l \). A session \( s \) is attributed to a cluster if it matches at least \( p \) of the \( S \) sessions. The session \( s \) matches \( sc \) if \( sc \) is a subsequence extracted from \( s \) (see Figure 4).

\[
\text{IsSubsequence}(s_1 \ldots s_n), (s'_1 \ldots s'_{n'}) = \exists i \leq n / s'_1 = s_i
\]

\[
\text{IsSubsequence}(s_1 \ldots s_n), (s'_1 \ldots s'_{n'})
\]

**Figure 4. The subsequence formula**

This way, we ensure the homogeneity of clusters and the fact that there is a common factor between any elements of a cluster.

Hence, we avoid obtaining clusters composed of fully different elements, connected by a chain of next neighbors generally produced by distance-based clustering techniques [2]. The minimum length of a representative sequence and the number of sequences needed to model a cluster are given by the analyst to allow him to retrieve clusters of the required homogeneity.

The clustering algorithm itself is a classical hierarchical clustering algorithm. It starts with considering little groups of sessions as clusters and iteratively merge the two nearest clusters. The algorithm ends when the required level of homogeneity has been reached.

4. Experimental Results

This part will point out the two following abilities of our technique compared to usual approaches. First, we will see how the analysis of the intra-video behavior allows a division of groups of sessions that are composed of the same video but viewed in a different manner. Then, we will introduce the advantage of describing a cluster by a group of session compared to a simple subsequence extraction.

4.1 Creation of the test datasets

Due to the lack of precise data concerning video extraction, we have conducted our tests on generated datasets. The creation of the test datasets is divided into two phases. First, we have created intra-video behavior models. We have defined four typical behaviors (complete play, quick overview, partial play of a part, quick closure). Future experiments on real data will allow us to fully determine these ones based on the use of the search engine. Based on these models, we have randomly generated behavior vectors for each viewing of the video sessions. Secondly, to create the video sessions, we have defined source of clusters with a set of video identifiers sequences. For each cluster, we have created session by randomly merge these sequences. Then we have added in each generated session about 5 to 20% of noise by adding into sequences viewing identifiers that have no link with the content of the clusters.

Finally, we have generated different test datasets composed of 2000 sessions. Each session is composed of 5 to 20 viewings that are linked to behaviors of 10 to 20 basic actions (play, pause...). The test files are then composed of around 100,000 to 800,000 basic actions.

In the following, we present two executions corresponding to predefined scenarios that are pointing out the abilities of our technique. Let us consider a video database of natural videos containing videos of mountains and volcanoes.
4.2 Exploiting the intra-video behavior

This scenario is based on the following assumption. The video database is not correctly indexed and many videos dealing with volcanoes are indexed like mountain video.

We have generated two clusters. The first one corresponds to a search on volcanoes and gives only a unique video completely viewed. The second one is the result of a search on mountains and every video are viewed completely (see Figure 6).

With a classical approach, that not take into account intra-video behavior, the two clusters are not discovered and the result is a unique cluster ((1, 2, 3, 4, 5)).

With our double-level approach, the technique is able to discover that the use of videos has been different and the two clusters corresponding to the two searches are discovered. For any value less than 6 of the minimum length of the representative sequence, clusters ((1, 2, 3, 4, 5, 6)) and ((1, 2, 3, 4, 5, 10, 10)) are returned.

<table>
<thead>
<tr>
<th>Cluster id</th>
<th>Base model</th>
<th>minLength = 2 or 3</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (301, 302, 303, 304)</td>
<td>(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (301, 302, 303, 304)</td>
<td>(18, 73, 29, 41)</td>
</tr>
<tr>
<td>2</td>
<td>(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (401, 402, 403, 404)</td>
<td>(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (401, 402, 403, 404)</td>
<td>(18, 73, 29, 41)</td>
</tr>
<tr>
<td>3</td>
<td>(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (501, 502, 503, 504)</td>
<td>(18, 73, 29, 41) (17, 25, 12, 19, 87, 10) (129, 2, 73, 32, 91) (501, 502, 503, 504)</td>
<td>(18, 73, 29, 41)</td>
</tr>
<tr>
<td>4</td>
<td>(18, 73, 29, 41) (2, 3, 4, 8)</td>
<td>(18, 73, 29, 41) (2, 3, 4, 8)</td>
<td>(18, 73, 29, 41)</td>
</tr>
</tbody>
</table>

Figure 5. Clustering results for the second dataset

4.3 Multiple subsequence cluster modeling

The following experiment points out the advantage of modeling clusters with a set of sequences instead of a unique sequence. We have generated data corresponding to 4 clusters. Three of them are about ski place and are composed of 1 specific sequence and 3 others shared: Mountain, corresponding to the subsequence (18, 73, 29, 41); Snowboard (17, 25, 12, 19, 87, 10); Ski (129, 2, 73, 32, 91). The last cluster is composed of a search on mountain too (18, 73, 29, 41) and an other one on trekking (2, 3, 4, 8). For this set, every video has been completely viewed (see Figure 5).

Setting the minimum quantity of representative sequences to 2 or 3, sessions corresponding to any of the three first clusters are merged to form a cluster corresponding to “sessions dealing with ski place”. With a value of 4, this cluster is split and each source cluster is discovered. For this value, the cluster corresponding to mountain and trekking is correctly analyzed and not merged with the other data. But if we use the value of 1, leading to a classical subsequence extraction, all of these data are merged in a unique cluster and the difference between ski places and mountain trekking is not detected by the clustering.

5. Future work and research

In this paper, we propose a two levels model based approach for modeling user behaviors on a video search engine. The first level aims at modeling and clustering user behavior on a single video sequence (intra video behavior), the second one aims at modeling and clustering user behavior on a set of video sequences (inter video behavior). Based on this representation we have developed a two phase clustering algorithm that fits these data. We have showed that our approach is able to: 1° differentiate sessions dealing with the same videos but in different manners. 2° discover clusters that are not detected by basic subsequence approaches.

The main remaining work is to validate our technique on real datasets in a context of movie production. We have to ensure that results are still interesting on huge video database.

An other objective is to model clusters using frequent itemsets instead of subsequence and compare results to measure the impact of the sequential representation.

6. References


[6] Bin Yu, Wei-Ying Ma, Klara Nahrstedt, Hong-Jiang Zhang: Video summarization based on user log enhanced link analysis. MM'03, November 2-8 2003, Berkeley CA, USA.
Seeing and Reading Red: Hue and Color-word Correlation in Images and Attendant Text on the WWW

Shawn Newsam
School of Engineering
University of California at Merced
Merced, CA 95340
snewsam@ucmerced.edu

ABSTRACT
This work represents an initial investigation into determining whether correlations actually exist between metadata and content descriptors in multimedia datasets. We provide a quantitative method for evaluating whether the hue of images on the WWW is correlated with the occurrence of color-words in metadata such as URLs, image names, and attendant text. It turns out that such a correlation does exist: the likelihood that a particular color appears in an image whose URL, name, and/or attendant text contains the corresponding color-word is generally at least twice the likelihood that the color appears in a randomly chosen image on the WWW. While this finding might not be significant in and of itself, it represents an initial step towards quantitatively establishing that other, perhaps more useful correlations exist. These correlations form the basis for exciting novel approaches that leverage semi-supervised datasets, such as the WWW, to overcome the semantic gap that has hampered progress in multimedia information retrieval for some time now.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval]: clustering; information filtering; retrieval models.

General Terms
Algorithms.

Keywords
Multimedia and metadata data mining, semi-supervised learning, image content descriptors.

1. INTRODUCTION
The impasse presented by the proverbial semantic gap has hampered progress in multimedia information retrieval over the past several years. The early successes that promised access to multimedia data based solely on its content without manual annotation have failed to develop into usable systems. Content descriptors that can be automatically extracted from images, such as color and texture, provide limited high-level information and usually only in highly constrained situations. Focus needs to be shifted away from developing new content descriptors to investigating novel ways in which metadata that is available without manual annotation can aid multimedia information retrieval. This work stipulates that one promising approach to addressing the semantic gap is to leverage the semi-supervised multimedia datasets that are appearing in our information society.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MDM/KDD 2005 Chicago, August 21, Chicago, Illinois, USA

Early investigations into leveraging semi-supervised datasets are encouraging. Examples include:

- Searching for images on the world wide web (WWW) using image URLs, image names, image ALT tags and attendant text on the webpage containing the images. This includes commercial systems such as Google’s Image Search [4] and Yahoo’s Image Search [5].
- Searching for images on the WWW using a combination of textual information and content descriptors [10].
- Searching for images on the WWW using category structure, textual information, and content descriptors [8][1][9][2].
- Using annotated stock photography or art collections to learn statistical word-descriptor correlations to perform auto-annotation or auto-illustration [6][7].

These investigations suggest that using available metadata, such as URLs, annotations, etc., instead of or in addition to content descriptors results in better performance than solely relying on content descriptors alone. This, in turn, suggests that the metadata and image content are correlated, but improved performance is only anecdotal evidence of such a correlation. The work presented in this paper represents initial investigations into whether such correlations actually exist. We focus on the particular question of whether the colors or hues of images on the WWW are correlated with the occurrence of color-words, such as red or blue, in the URLs, image names, and attendant text corresponding to the images. The existence of such a correlation might not have direct application, such as searching for images on the
WWW, but serves to provide initial insight into the nature of other correlations that might prove more useful.

2. THE APPROACH

In summary, the objective is to estimate the likelihood that a particular color appears in an image on the WWW given that the corresponding color-word occurs in the metadata associated with the image. This metadata includes the image URL, the image name, and the text, if any, that appears in attendance with the image on the webpage. This section describes the steps taken towards this end. First, nearly 2 million images and their metadata are downloaded from the WWW. Next, hue histograms are computed for each image. Finally, the conditional hue probabilities are estimated by averaging the hue histograms over subsets of images corresponding to different hypotheses, such as the word red occurring in the attendant text. The conditional color likelihoods are then computed based on these conditional probabilities.

2.1 Acquiring the Images and Attendant Text

The first step is to acquire a sizable set of images and their metadata from the WWW. This data was generously provided by Till Quack from the Cortina content-based web retrieval dataset [2][9]. The Cortina dataset was collected using WWW Uniform Resource Locators (URLs) from the DMOZ open directory project [3]. The entire DMOZ directory can be downloaded as a Resource Description Framework file which can be parsed for URLs. Only URLs from the Shopping and Recreation categories were used in creating the Cortina dataset. These URLs were used to locate webpages containing images. The URL for each image was then stored in a database along with other key information extracted from the webpage. This includes up to 20 words above and 30 words below the image. These word sets constitute the attendant text utilized in this paper.

The Cortina dataset also includes two color descriptors for each image. We decided, however, that these descriptors were not appropriate for our analysis and so we downloaded each image again using the URLs in the Cortina dataset to extract our own color descriptor termed hue histograms. Only 1,751,578 of the 2 million image URLs obtained from the Cortina dataset were still valid. Of these, 933,875 were located on webpages that contained attendant text.

2.2 Hue Histograms

Our analysis required a compact characterization of the color distribution of each image. Rather than use a traditional multi-dimensional histogram, we derived a one-dimensional histogram of just the hue values of the pixels in an image. The pixel values are first transformed from the red-green-blue (RGB) colorspace to a hue-lightness-saturation (HLS) colorspace. This hue channel is similar to an angle on the color-wheel in that its range from 0 to 360 corresponds to the colors red, orange, yellow, etc., through pink, and then back to red again. The hue histograms are computed by binning the hue channel into 360 one-degree intervals.

The HLS colorspace presented some expected problems. First, even very dark (low lightness value) or very light (high lightness value) colors have associated hue values, the colors corresponding to the hue values are not really perceivable. To deal with this we added two additional histogram bins, one for pixels with lightness values below an empirically chosen threshold and another for pixels with lightness values above an empirically chosen threshold. These bins correspond to the "colors" black and white, respectively. The second problem is that grey-ish colors also have an associated hue. This was dealt with by adding a third histogram bin for pixels with saturation values below another empirically chosen threshold.

The complete hue histograms contain 363 bins. Three for black, white, and grey, and 360 for the hue values at one-degree intervals. Table 1 indicates the central hue values for the seven colors analyzed: red, orange, yellow, green, blue, purple, and pink. It also indicates the range of hues for each color. The ranges for adjacent colors overlap by 10 degrees to account for the ambiguous regions between colors.

Finally, the hue histograms are normalized so that they sum to one. This accounts for the different image sizes and makes the histograms interpretable as estimates of probability density functions (PDFs).

2.3 Conditional Hue PDFs

The conditional hue PDF’s are estimated by averaging the hue histograms corresponding to a particular hypothesis. For example, if \( \text{hist}_i(h) \) is the value of the hue histogram for image \( i \) at hue \( h \) then the conditional hue PDF for the hypothesis of "all images whose attendant text contains the word red," is estimated as:

\[
p(h|\text{red} \in \text{text}) = \frac{1}{\#(i : \text{red} \in \text{text})} \sum_{i, \text{red} \in \text{text}} \text{hist}_i(h). \quad (1)
\]

2.4 Conditional Color Likelihoods

Finally, the conditional hue PDF’s, such as \( p(h|\text{red} \in \text{text}) \), are used to compute the conditional likelihoods that colors occur in sets of images satisfying hypotheses. For example, the likelihood that the color blue occurs in images whose attendant text contains the color-word blue is computed as:

\[
p(\text{blue} \in \text{image}|\text{blue} \in \text{text}) = \sum_{h=1}^{275} p(h|\text{blue} \in \text{text}). \quad (2)
\]

The bounds on the summation for each of the seven color considered are listed in Table 1. These conditional color likelihoods can be used to compare different hypotheses, such as color-words appearing in attendant text, URLs, image names, etc., with each other as well as with baseline hypotheses such as the color blue occurring in any image on
the WWW which can be computed as:

\[ p(\text{blue} \in \text{image}|\text{all images}) = \sum_{h=175}^{275} p(h|\text{all images}) . \] (3)

3. EXPERIMENTS AND RESULTS

The conditional hue PDFs and conditional color likelihoods are computed for a variety of hypotheses as well as the baseline hypothesis. This section describes these hypotheses and their results.

3.1 Hypotheses

The conditional hue PDFs and conditional color likelihoods are computed for a total of eight hypotheses:

all This is the baseline of all 1,751,578 images.

all with text The 933,875 images with attendant text (many images appear alone on webpages).

color-word \in text A color-word occurs in the attendant text for an image.

color-word \in URL A color-word occurs in the URL for an image.

color-word \in image name A color-word occurs in the image name. The image name is considered as the substring in the URL after the rightmost backslash.

color-word \in text,URL Conjunction of two hypotheses above.

color-word \in text,image name Conjunction of two hypotheses above.

only color-word \in text A color-word occurs in the attendant text for an image without any other color-words.

The color-words considered are red, orange, yellow, green, blue, purple, and pink. The meaning of occurrence varies by hypothesis above. In the case of attendant text, the color-word must occur as a separate word; that is, it must be preceded and followed by a non-alphabetic character. This constraint is not enforced for occurrence in a URL or image name. Thus, the color-word red is considered to occur in the URL http://threddies.com/images/tinysageside.jpg and the image name redruff,computer.jpg. This likely introduces noise but considerably more complex string matching algorithms would be required to identify only those URLs and image names that contain the color-word red as a “separate word” as in the attendant text.

3.2 Conditional Hue PDF Results

The conditional hue PDFs are shown for several hypothesis in Figure 1. Note that the PDFs are plotted only for hue values corresponding to the 360 color-bins in the hue histograms. The three bins corresponding to black, white and grey have been left out since their magnitudes are generally much larger and their values are not informative for our analysis. Thus, the PDFs as plotted do not necessarily sum to one. Note, also, that the PDFs have been smoothed by averaging the values over five-degree intervals.

Figures 1(a) through 1(f) show the PDF for the proposed hypothesis with a solid line, and the PDF for the baseline hypothesis (all images) with a dashed line for comparison.

The first thing to note from Figure 1 is that the baseline PDF is not uniform. There appear to be distinct peaks around the red-orange and green-blue regions. Second is that there is approximately only a 30% probability that a pixel is not black, white, or grey. The images corresponding to the baseline hypothesis are approximately 16% black, 30% white, and 25% grey.

The PDFs in Figure 1 all correspond to hypotheses relating to the color-word red. Thus, it is significant that they all exhibit the following noteworthy characteristics. First, they are all greater than the baseline PDF for hue values corresponding to red. Second, they are generally all less than the baseline PDF for other hue values. This indicates that there is a correlation between the color-word red occurring in the attendant text, URL and/or image name, and the color red appearing in the image.

The conditional hue probabilities for the other color are similar to those for red but space restrictions prevent them from being included.

3.3 Conditional Color Likelihood Results

The conditional color likelihoods allow a more quantitative comparison of the hypotheses. Table 2 shows the conditional color likelihoods for a number of hypotheses. The rows represent the hypotheses, such as the color-word red occurring in both the attendant text and URL for an image, and the columns represent the color-word occurring in the image. The maximum for each column is shown in bold. There is also a column indicating the number of images that satisfy the hypothesis.

The results in Table 2 again demonstrate there is a correlation between a color-word occurring in attendant text, URL and/or image name, and the color appearing in an image. The correlations are strongest for the color-word occurring in the image name but the correlations for the other hypotheses are not significantly weaker. In general, the likelihood that a color appears in an image given the corresponding color-word occurs in the attendant text, URL and/or image name is at least twice the baseline likelihood.

The following specific observations can be made from Table 2:

- The hypotheses for the color-word appearing in the image name maximizes the conditional color likelihoods of all hypothesis for orange, yellow, green, blue, and purple, and there is the correct correspondence between the color-word and color; i.e., \( p(\text{orange}\mid \text{image name}) \) is maximal.
- The hypothesis for the color-word red appearing in both the text and image name maximizes the likelihood that red appears in the image.
- The hypothesis for the color-word pink appearing in both the text and URL maximizes the likelihood that
pink appears in the image.

- In all cases but two, the conditional color likelihood for a hypothesis is maximum for the correct color. I.e., \(p(\text{red} | \text{red} \in \text{text})\) is greater than \(p(\text{red} | \text{color-word} \in \text{text})\) for any other color-word. The exceptions are \(p(\text{red} | \text{color-word} \in \text{URL})\) and \(p(\text{red} | \text{color-word} \in \text{image name})\).

- The conditional likelihoods “bleed” into adjacent colors. Not only are the conditional likelihoods corresponding to the correct color-words large but so are the conditional likelihoods for adjacent colors on the hue axis.

4. DISCUSSION

This work represents an initial investigation into determining whether correlations actually exist between metadata and content descriptors in multimedia datasets. We provide a quantitative method for evaluating whether the hue of images on the WWW is correlated with the occurrence of color-words in metadata such as URLs, image names, and attendant text. It turns out that such a correlation does exist: the likelihood that a particular color appears in an image whose URL, name, or attendant text contains the corresponding color-word is generally at least twice the likelihood of the color appears in a randomly chosen image on the WWW. As pointed out in the Introduction, this finding might not be significant in and of itself, but represents an initial step towards quantitatively establishing that other, perhaps more useful correlations exist. These correlations form the basis for exciting novel approaches that leverage semi-supervised datasets, such as the WWW, to overcome the semantic gap that has hampered progress in multimedia information retrieval for some time now.

As this is only an initial investigation, there are plenty of directions for this work to proceed in. Establishing quantitative ways to evaluate correlations between higher-level textual concepts and image content would be very useful for designing the learning algorithms for tasks such as retrieval, classification, and auto-annotation.

5. ACKNOWLEDGMENTS

This work was performed in part under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

6. REFERENCES

http://vision.ece.ucsb.edu/multimedia/search.html.


Figure 1: Conditional hue PDFs for hypotheses corresponding to (a) the word red appearing in text, (b) the word red appearing in the URL, (c) the word red appearing in the image name, (d) the word red appearing in both the URL and text, (e) the word red appearing in both the image name and text, and (f) the word red appearing alone in the text. The solid line shows the PDF for the hypothesis. The dashed line shows the baseline PDF computed over all images. The number of images satisfying the hypothesis is indicated in the title in parentheses.
| event          | # images | p(red|e) | p(orange|e) | p(yellow|e) | p(green|e) | p(blue|e) | p(purple|e) | p(pink|e) |
|----------------|----------|-------|-----------|-----------|-----------|---------|-----------|---------|
| all            | 1751578  | 0.0724| 0.1309    | 0.0481    | 0.0181    | 0.0703  | 0.0059    | 0.0149  |
| all with text  | 933875   | 0.0710| 0.1292    | 0.0476    | 0.0182    | 0.0691  | 0.0056    | 0.0147  |
| red ∈ text     | 2108     | 0.1881| 0.1159    | 0.0334    | 0.0161    | 0.0434  | 0.0052    | 0.0379  |
| orange ∈ text  | 365      | 0.1257| 0.2153    | 0.0472    | 0.0134    | 0.0417  | 0.0039    | 0.0195  |
| yellow ∈ text  | 808      | 0.0447| 0.1774    | 0.1298    | 0.0136    | 0.0392  | 0.0040    | 0.0097  |
| green ∈ text   | 1333     | 0.0517| 0.1199    | 0.0858    | 0.0708    | 0.0569  | 0.0035    | 0.0091  |
| blue ∈ text    | 2568     | 0.0503| 0.0882    | 0.0323    | 0.0155    | 0.1588  | 0.0056    | 0.0121  |
| purple ∈ text  | 404      | 0.0555| 0.0754    | 0.0283    | 0.0155    | 0.1117  | 0.0694    | 0.0591  |
| pink ∈ text    | 801      | 0.1383| 0.0915    | 0.0281    | 0.0113    | 0.0368  | 0.0188    | 0.0902  |
| red ∈ URL      | 34994    | 0.0861| 0.1003    | 0.0336    | 0.0125    | 0.0416  | 0.0039    | 0.0179  |
| orange ∈ URL   | 1422     | 0.1676| 0.2742    | 0.0494    | 0.0152    | 0.0508  | 0.0074    | 0.0141  |
| yellow ∈ URL   | 2658     | 0.0545| 0.2018    | 0.1675    | 0.0161    | 0.0593  | 0.0035    | 0.0100  |
| green ∈ URL    | 7734     | 0.0545| 0.1151    | 0.0752    | 0.0678    | 0.0557  | 0.0043    | 0.0112  |
| blue ∈ URL     | 12771    | 0.0462| 0.0915    | 0.0366    | 0.0183    | 0.1756  | 0.0060    | 0.0106  |
| purple ∈ URL   | 2569     | 0.0516| 0.0767    | 0.0256    | 0.0122    | 0.0939  | 0.0597    | 0.0304  |
| pink ∈ URL     | 3381     | 0.1322| 0.1039    | 0.0292    | 0.0114    | 0.0417  | 0.0181    | 0.0967  |
| red ∈ iname    | 11685    | 0.1679| 0.1222    | 0.0401    | 0.0148    | 0.0501  | 0.0059    | 0.0375  |
| orange ∈ iname | 1161     | 0.1794| 0.2980    | 0.0523    | 0.0136    | 0.0399  | 0.0039    | 0.0115  |
| yellow ∈ iname | 1953     | 0.0453| 0.2169    | 0.2036    | 0.0157    | 0.0518  | 0.0030    | 0.0089  |
| green ∈ iname  | 4200     | 0.0418| 0.0953    | 0.0826    | 0.0994    | 0.0542  | 0.0038    | 0.0088  |
| blue ∈ iname   | 8758     | 0.0392| 0.0786    | 0.0302    | 0.0182    | 0.2141  | 0.0062    | 0.0099  |
| purple ∈ iname | 1499     | 0.0453| 0.0732    | 0.0293    | 0.0126    | 0.1099  | 0.0920    | 0.0405  |
| pink ∈ iname   | 2813     | 0.1366| 0.0924    | 0.0261    | 0.0111    | 0.0365  | 0.0200    | 0.1118  |
| red ∈ text,URL | 706      | 0.2149| 0.1076    | 0.0352    | 0.0196    | 0.0456  | 0.0074    | 0.0489  |
| orange ∈ text,URL | 104   | 0.1566| 0.2740    | 0.0606    | 0.0103    | 0.0545  | 0.0093    | 0.0150  |
| yellow ∈ text,URL | 191 | 0.0337| 0.1859    | 0.1887    | 0.0164    | 0.0454  | 0.0039    | 0.0097  |
| green ∈ text,URL | 437  | 0.0545| 0.1151    | 0.0752    | 0.0678    | 0.0557  | 0.0043    | 0.0112  |
| blue ∈ text,URL | 804  | 0.0487| 0.0960    | 0.0340    | 0.0198    | 0.1876  | 0.0075    | 0.0150  |
| purple ∈ text,URL | 146 | 0.0381| 0.0624    | 0.0298    | 0.0188    | 0.1471  | 0.0839    | 0.0374  |
| pink ∈ text,URL | 250  | 0.1570| 0.0784    | 0.0197    | 0.0137    | 0.0322  | 0.0219    | 0.1338  |
| red ∈ text,iname | 663  | 0.2187| 0.1061    | 0.0351    | 0.0200    | 0.0445  | 0.0076    | 0.0509  |
| orange ∈ text,iname | 99   | 0.1583| 0.2842    | 0.0635    | 0.0092    | 0.0512  | 0.0098    | 0.0154  |
| yellow ∈ text,iname | 184 | 0.0343| 0.1871    | 0.1896    | 0.0167    | 0.0453  | 0.0041    | 0.0099  |
| green ∈ text,iname | 389  | 0.0428| 0.0914    | 0.0786    | 0.0864    | 0.0741  | 0.0080    | 0.0074  |
| blue ∈ text,iname | 722  | 0.0476| 0.0944    | 0.0346    | 0.0172    | 0.1940  | 0.0070    | 0.0147  |
| purple ∈ text,iname | 138 | 0.0394| 0.0650    | 0.0298    | 0.0186    | 0.1543  | 0.0882    | 0.0393  |
| pink ∈ text,iname | 249  | 0.1569| 0.0787    | 0.0198    | 0.0137    | 0.0323  | 0.0220    | 0.1333  |

Table 2: Conditional color likelihoods for different hypotheses. The rows represent the hypotheses, and the columns represent the color likelihood conditioned on a hypothesis. There is also a column indicating the number of images that satisfy the hypothesis.
Learning Semantics-Preserving Distance Metrics for Clustering Graphical Data

Aporna S. Varde\textsuperscript{1,2} Elke A. Rundensteiner\textsuperscript{1} Carolina Ruiz\textsuperscript{1} Mohammed Maniruzzaman\textsuperscript{2} Richard D. Sisson Jr.\textsuperscript{2,3}

\textsuperscript{1}Department of Computer Science
\textsuperscript{2}Center for Heat Treating Excellence
\textsuperscript{3}Department of Mechanical Engineering
Worcester Polytechnic Institute (WPI)
Worcester, MA 01609. USA
(aparna | rundenst | ruiz | maniruzz | sisson)@wpi.edu

ABSTRACT
In mining graphical data the default Euclidean distance is often used as a notion of similarity. However this does not adequately capture semantics in our targeted domains, having graphical representations depicting results of scientific experiments. It is seldom known a priori what other distance metric best preserves semantics. This motivates the need to learn such a metric. A technique called LearnMet is proposed here to learn a domain-specific distance metric for graphical representations. Input to LearnMet is a training set of correct clusters of such graphs. LearnMet iteratively compares these correct clusters with those obtained from an arbitrary but fixed clustering algorithm. In the first iteration a guessed metric is used for clustering. This metric is then refined using the error between the obtained and correct clusters until the error is below a given threshold. LearnMet is evaluated rigorously in the Heat Treating domain which motivated this research. Clusters obtained using the learned metric and clusters obtained using Euclidean distance are both compared against the correct clusters over a separate test set. Our results show that the learned metric provides better clusters.

Categories and Subject Descriptors
I.2.6 [Artificial Intelligence]: Learning – Parameter learning.

General Terms
Algorithms, Experimentation.

Keywords
Distance metric, clustering, semantic graphical mining.

1. INTRODUCTION
Experimental results in scientific domains are often represented graphically. Such graphs depict the functional behavior of process parameters hence incorporating semantics. In this paper, we use the term “graph” to mean such a graphical representation. In mining graphs with techniques such as clustering \cite{9} the measure for comparison is typically Euclidean distance \cite[see e.g. 7]{7} which often poses problems. For example, in the domain of Heat Treating of Materials which motivated this work, graphs are used to plot the heat transfer versus temperature during the rapid cooling of a material \cite[see e.g. 3]{3}. Critical regions on these graphs represent significant physical phenomena in the domain. Any algorithm (based for instance on Euclidean distance) that considers two graphs as similar \cite[relative to other graphs]{} although their critical regions differ is considered semantically incorrect \cite[3,14]{3,14}. Likewise in several scientific domains, there could be significant features on graphs. Knowledge of these features and their relative importance may at best be available in a subjective form, but not as a metric. This motivates the development of a technique to learn distance metrics that capture the semantics of the graphs.

Hinneburg et al \cite{6} propose a learning method to find the relative importance of dimensions for n-dimensional objects. However, their focus is on dimensionality reduction and not on domain semantics. In \cite{16} they learn which type of position-based distance is applicable for the given data starting from the formula of Mahalanobis distance. However they do not deal with graphical data and semantics. Keim et al. \cite{8} overview various distance types for similarity search over multimedia databases. However no single distance measure encompassing several types is proposed. Linear regression \cite{1} and neural networks \cite{2} could possibly be used for learning a domain-specific distance metric for graphs. However these techniques do not achieve accuracy acceptable in our targeted domains \cite{14}. Genetic algorithms \cite{5} if used for feature selection in graphs also give the problem of insufficient accuracy. This is due to lack of domain knowledge \cite[14]{14}. Fourier transforms \cite{4} if used to represent the graphs do not preserve the critical regions in the domain due to the nature of the transform \cite{14}. Hence they are not accurate enough in capturing semantics. Accuracy in this context is measured by evaluating the effectiveness of a given metric in mining unseen data \cite{14}.

We propose an approach called LearnMet to learn a distance metric for graphs incorporating domain semantics. The input to LearnMet is a training set with correct (i.e., given by domain experts) clusters of graphs over a subset of the experimental data. The steps of our approach are: (1) guess initial metric guided by domain knowledge; (2) use that metric for clustering with an arbitrary but fixed clustering algorithm; (3) evaluate accuracy of obtained clusters by comparing them with correct clusters; (4) adjust metric based on error between obtained and correct clusters, if error is below threshold or if execution times out then terminate and go to step (5), else go to step (2); and (5) once

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MDM/KDD 2005 Chicago, August 21, Chicago, Illinois, USA
Copyright 2005 ACM -- MDM 2005 - 1-59593-216-X...$5.00.
terminated, output the metric giving error below threshold, or minimum error so far as the learned metric. LearnMet is evaluated using a distinct test set of correct clusters of graphs provided by experts. The learned metric is used to cluster the graphs in the test set. The obtained clusters are compared with correct clusters in the test set. The closer the obtained clusters match the correct clusters, the lower the error. The clusters obtained using the default notion of Euclidean distance, are also compared with the correct clusters. The difference in the error denotes the effectiveness of the learned metric. The LearnMet metric consistently outperformed the Euclidean metric in our experiments. LearnMet is also evaluated by integrating it with the AutoDomainMine system designed for computational estimation of process parameters in scientific experiments [13]. This is explained in Section 4.

2. DISTANCE METRICS

We describe distance metric types relevant to our domains. Let \( A (A_1, A_2, ..., A_n) \) and \( B (B_1, B_2, ..., B_n) \) be two \( n \)-dimensional objects.

**Position-based Distances:** They refer to absolute position of objects [see e.g. 7]. Examples are:

- **Euclidean Distance:** As-the-crow-flies distance between objects.

\[
D_{\text{Euclidean}} (A, B) = \sqrt{\sum_{i=1}^{n} (A_i - B_i)^2}
\]

- **Manhattan Distance:** City-block distance between objects.

\[
D_{\text{Manhattan}} (A, B) = \sum_{i=1}^{n} |A_i - B_i|
\]

**Statistical Distances:** This refers to distances based on statistical features [see e.g. 11]. Examples are:

- **Mean Distance:** Distance between mean values of the objects:

\[
D_{\text{Mean}} (A, B) = |\text{Mean}(A) - \text{Mean}(B)|
\]

- **Maximum Distance:** Distance between maximum values:

\[
D_{\text{Max}} (A, B) = |\text{Max}(A) - \text{Max}(B)|
\]

In addition to these, we introduce the concept of Critical Distance for graphs as applicable to our targeted domains [14].

**Critical Distances:** Given graphs \( A \) and \( B \), a Critical Distance is a distance metric between critical regions of \( A \) and \( B \) where a critical region represents the occurrence of a significant physical phenomenon. They are calculated in a domain-specific manner.

3. THE LEARNMET STRATEGY

To give details of LearnMet, a distance metric is first defined.

**Distance Metric in LearnMet:** A LearnMet distance metric \( D \) is a weighted sum of components, where each component can be a position-based, a statistical, or a critical distance metric. The weight of each component is a numerical value indicating its relative importance in the domain.

Thus a LearnMet distance metric is of the form

\[
D = w_1 D_{C_1} + ... + w_m D_{C_m}
\]

where each \( D_{C_i} \) is a component, \( W_i \) is its weight, and \( m \) is number of components. As required in our application domain, \( D \) should be a metric so that clustering algorithms requiring the notion of similarity to be a distance metric can be used; indexing structures such as \( B^+ \) trees for metrics can be applied; and pruning in similarity search can be performed using triangle inequality. Conditions for \( D \) to be a metric are stated as a theorem below.

**Theorem 1:** If each component \( D_{C_i} \) is a distance metric and each weight \( w_i \geq 0 \) then \( D = \sum_{i=1}^{m} w_i D_{C_i} \) is a distance metric, i.e., it satisfies the metric properties.

The proof of this theorem is straightforward and can be found in [14]. In our targeted domains, conditions in Theorem 1 are satisfied. Since our graphs involve interval-scaled variables and the fundamental distance types applicable to these are metrics [see e.g. 7], this is sufficient to say that each \( D_{C_i} \) is a metric. Also, we consider only non-negative weights.

The LearnMet steps are discussed in the subsections below.

3.1 Initial Metric Step

Domain experts are asked to identify components (i.e., distance metrics) applicable to the graphs that will serve as building blocks for the learning of a new metric. If the experts know the relative importance of the components, this information is used to assign initial weights. An Initial Weight Heuristic is proposed.

**Initial Weight Heuristic:** Assign initial weights to the components in the LearnMet distance metric based on the relative importance of the components on the graphs in the domain.
Let TP, TN, FP and FN denote the number of true positive, true negative, false positive and false negative pairs respectively. Also let SR denote the Success Rate and FR = (1 – SR) denote the Failure Rate, as defined below:

\[
SR = \frac{TP + TN}{TP + FP + TN + FN} \quad \text{and} \quad FR = \frac{FP + FN}{TP + FP + TN + FN}
\]

### 3.2 Clustering Step

Using D as the distance metric, k clusters are constructed using an arbitrary but fixed clustering algorithm (e.g., k-means [10]), where k is the number of clusters in the training set.

**1: INITIAL METRIC STEP**
- GIVEN: DOMAIN EXPERT INPUT ON DISTANCE TYPES
- FOR EACH DISTANCE TYPE ASSIGN A COMPONENT TO “D”
- IF RELATIVE IMPORTANCE OF COMPONENTS IS AVAILABLE THEN USE “INITIAL WEIGHT HEURISTIC”
- ELSE ASSIGN RANDOM WEIGHTS TO COMPONENTS

**2: CLUSTERING STEP**
- GIVEN: A TRAINING SET CONSISTING OF A COLLECTION OF GRAPHS AND A CORRECT k-CLUSTERING OF THEM
- SELECT AN ARBITRARY BUT FIXED CLUSTERING ALGORITHM
- SET NUMBER OF CLUSTERS TO k (CONSTANT)
- CLUSTER GRAPHS USING \( D = w_1Dc_1 + ... + w_mDc_m \)

### 3.3 Cluster Evaluation Step

The clusters obtained from the algorithm, i.e., the “predicted” clusters, are evaluated by comparing them with correct clusters in the training set, i.e., the “actual” clusters. An example of predicted and actual clusters is shown in Figure 2.

Ideally, the predicted clusters should match the actual clusters. Any difference between predicted and actual clusters is considered an error. To compute this error, we consider pairs of graphs and introduce the following notation.

**True/False Positive/Negative Pairs of Graphs:** Given a pair of graphs I and J, we say that:

- \((I,J)\) is a True Positive (TP) pair if I and J are in the same actual cluster and in the same predicted cluster.
- \((I,J)\) is a True Negative (TN) pair if I and J are in different actual clusters and in different predicted clusters.
- \((I,J)\) is a False Positive (FP) pair if I and J are in different actual clusters but in the same predicted cluster.
- \((I,J)\) is a False Negative (FN) pair if I and J are in the same actual cluster but in different predicted clusters.

Figure 2 includes examples of each of these kinds of pairs: \((g_a,g_a)\) is a true positive pair; \((g_b,g_d)\) is a true negative pair; \((g_c,g_d)\) is a false positive pair; and \((g_e,g_d)\) is a false negative pair. The error measure of interest to us is failure rate which is defined below.

**Success and Failure Rates:** Let TP, TN, FP and FN denote the number of true positive, true negative, false positive and false negative pairs respectively. Also let SR denote the Success Rate and FR = (1 – SR) denote the Failure Rate, as defined below:

\[
SR = \frac{TP + TN}{TP + FP + TN + FN} \quad \text{and} \quad FR = \frac{FP + FN}{TP + FP + TN + FN}
\]

In our domain, false positives and false negatives are equally undesirable. Hence, our definition of failure rate weighs them equally.

Given a number \(G\) of graphs in the training set, the total number of pairs of graphs is \(C_G = \binom{G}{2}(G - 2)\). Thus, for 25 graphs there are 300 pairs, for 50 graphs, 1225 pairs, etc. We define an *epoch* in LearnMet as one run of all its steps. That is, a complete training cycle.

*Overfitting:* To avoid overfitting in LearnMet, we use an approach analogous to incremental gradient descent [2, 14]. Instead of using all pairs of graphs for evaluation, a subset of pairs is used called *ppe* or *pairs per epoch*. In each epoch, a distinct combination of pairs is used for evaluation and weight adjustments. Thus there is enough randomization in every epoch. If *ppe* = 25, then we have a total of \(300C_{25} = 1.95 \times 10^6\) distinct pairs for learning [11, 14]. Thus in each epoch 25 distinct pairs can be used. This still gives a large number of epochs with distinct pairs for learning. This incremental approach reduces the time complexity of the algorithm and helps avoid overfitting. Determining the best *ppe* value is an optimization problem. Also in LearnMet, the random seed is altered in the clustering algorithm in different epochs as an additional method to avoid overfitting.

Ideally, the error i.e., failure rate in an epoch should be zero. However, in practice a *domain-specific error threshold* "t" is used.

*Error Threshold:* A domain-specific error threshold "t" is the extent of error allowed per epoch in the domain, where error is measured by failure rate.

**Distance between a Pair of Graphs:** The distance \(D(g_a,g_b)\) between a pair of graphs \(g_a\) and \(g_b\) is the weighted sum of components in the graphs using metric \(D\).

Thus, \(D(g_a,g_b) = w_1Dc_1(g_a,g_b) + ... + w_mDc_m(g_a,g_b)\)

Given this, consider FN pairs, e.g., \((g_a,g_b)\) and \((g_c,g_d)\). These pairs are in the same actual cluster. However they are predicted to be in different clusters. Since predicted clusters are obtained with the metric \(D\), the (average) distance \(D(g_a,g_b)\) for these pairs is greater than it should be. Conversely, for FP pairs in different actual, same predicted clusters, e.g., \((g_b,g_d)\), the (average) distance \(D(g_a,g_b)\) is smaller than it should be.

![Figure 2: Predicted and Actual Clusters](image)
Average FN and FP Distances $D_{FN}$ and $D_{FP}$:

\[
D_{FN} = \frac{1}{|FN|} \sum_{i=1}^{FN} D(g_{x_i} g_{y_i}) \quad \text{where } (g_{x_i} g_{y_i}) \text{ denotes FN pairs.}
\]

\[
D_{FP} = \frac{1}{|FP|} \sum_{i=1}^{FP} D(g_{x_i} g_{y_i}) \quad \text{where } (g_{x_i} g_{y_i}) \text{ denotes FP pairs.}
\]

3. Cluster Evaluation Step

- SET “ppe” TO THE DESIRED NUMBER OF PAIRS OF GRAPHS FROM THE TRAINING DATASET TO BE CONSIDERED IN AN EPOCH
- RANDOMLY SELECT ppe PAIRS OF GRAPHS
- SET THE ERROR THRESHOLD “t”
- IDENTIFY THE “TP”, “TN”, “FP”, “FN” FROM “ppe” PAIRS
- CALCULATE FAILURE RATE “FR”
- IF (FR < t) THEN RETURN “CLUSTERING IS ACCURATE”
- ELSE CALCULATE $D_{FN}$, $D_{FP}$

3.4 Weight Adjustment Step

If the result of the evaluation does not indicate that the clustering is accurate, then the distances $D_{FN}$ and $D_{FP}$ are used to make weight adjustments to reduce the error in clustering. Consider the error in FN pairs. To reduce the average error $D_{FN}$, the weights of one or more components in the metric used to calculate the distance in the present epoch is decreased. For this we propose the FN Heuristic.

**FN Heuristic:** Decrease the weights of the components in the metric $D$ in proportion to their contributions to the distance $D_{FN}$. That is, for each component:

\[
w_i' = w_i - \frac{D_{FN}c}{D_{FN}} \quad \text{where } D_{FN}c = \frac{1}{|FN|} \sum_{i=1}^{FN} Dc_i(g_{x_i} g_{y_i})
\]

Conversely, consider FP pairs. To reduce their error, we increase $D_{FP}$. This is done by increasing the weights of one or more components in the metric using the FP Heuristic.

**FP Heuristic:** Increase the weights of the components in the metric $D$ in proportion to their contributions to the distance $D_{FP}$. That is, for each component:

\[
w_i'' = w_i + \frac{D_{FP}c}{D_{FP}} \quad \text{where } D_{FP}c = \frac{1}{|FP|} \sum_{i=1}^{FP} Dc_i(g_{x_i} g_{y_i})
\]

Combining these two adjustments:

**Weight Adjustment Heuristic:**

\[
w_i''' = \min(0, w_i - FN(D_{FN}c - D_{FN}) + FP(D_{FP}c - D_{FP}))
\]

Thus $D'''$ is obtained after weight adjustments is likely to minimize the error due to the FN and FP type pairs. If the weight of a component becomes negative, it is converted to zero as we consider only non-negative weights. Clustering is done with this new metric. If the resulting error is below the threshold, then a confirmatory test using the same metric to cluster for 2 more epochs is performed, and then this step is complete.

4. Weight Adjustment Step

- IF CLUSTERING IS ACCURATE OR MAX EPOCHS REACHED
  THEN GO TO “5: Final Metric Step”
- ELSE APPLY WEIGHT ADJUSTMENT HEURISTIC TO GET D'''
  GO TO “2: CLUSTERING STEP”

3.5 Final Metric Step

If the weight adjustment terminates because the error is below the threshold then the metric in the last epoch is considered accurate and it is returned as output. However, if termination occurs because the maximum number of epochs is reached, then the most reasonable metric to be output is the one corresponding to the epoch with the minimum error among all epochs.

5. Final Metric Step

- IF (FR < t) THEN RETURN METRIC D
- ELSE FIND EPOCH WITH MINIMUM Failure Rate
  RETURN CORRESPONDING METRIC D

Convergence: LearnMet is not guaranteed to converge or to yield an optimal distance metric. However, thorough experimental evaluation in our application domain has shown consistent convergence to errors below the required threshold.

4. Experimental Evaluation

4.1 Evaluation of LearnMet

Experimental Parameters: A training set of 300 pairs of graphs in Heat Treating is obtained from correct clusters of 25 graphs given by experts. A distinct test set of 300 pairs of graphs is derived from 25 graphs given by experts. We select “ppe = 25” which yields $^{300}C_{25} = 1.95 \times 10^{36}$ total distinct combinations of pairs. Thus 25 distinct pairs are used in each epoch. Experts give an error threshold of 10%, i.e., 0.1 for estimation. We use the same threshold for clustering. Initial components in the metric are given by experts. Two distinct assignments of initial weights are given by two different experts [14]. The corresponding two metrics are denoted by DE1 and DE2 respectively. A third initial metric EQU is obtained by assigning equal weights to all components. Several experiments are run by assigning random weights to components in the initial metric [14]. We present two experiments with randomly generated metrics called RND1 and RND2. See Table 1.

**Table 1: Initial Metrics in LearnMet Experiments**

<table>
<thead>
<tr>
<th>EXPT</th>
<th>INITIAL METRIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>DE1</td>
<td>5D_{Ox} + 2D_{f} + 5D_{l} + 1D_{f} + 3D_{ip}</td>
</tr>
<tr>
<td>DE2</td>
<td>5D_{Ox} + 1.5D_{f} + 4D_{l} + 3D_{ip}</td>
</tr>
<tr>
<td>EQU</td>
<td>1D_{Ox} + 1D_{f} + 1D_{l} + 1D_{ip}</td>
</tr>
<tr>
<td>RND1</td>
<td>6D_{Ox} + 2.75D_{f} + 3.25D_{l} + 3.25D_{ip} + 1.85D_{ip}</td>
</tr>
<tr>
<td>RND2</td>
<td>4D_{Ox} + 3.2D_{f} + 4.3D_{l} + 1.86D_{ip} + 2.95D_{ip}</td>
</tr>
</tbody>
</table>

**Table 2: Learned Metrics and Number Epochs to Learn**

<table>
<thead>
<tr>
<th>EXPT</th>
<th>LEARNED METRIC</th>
<th>EPOCHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DE1</td>
<td>4.068D_{Ox} + 1.774D_{f} + 2.842D_{l} + 1.962D_{ip} + 3.023D_{ip}</td>
<td></td>
</tr>
<tr>
<td>DE2</td>
<td>4.298D_{Ox} + 1.611D_{f} + 2.998D_{l} + 2.277D_{ip} + 2.92D_{ip}</td>
<td></td>
</tr>
<tr>
<td>EQU</td>
<td>4.057D_{Ox} + 1.865D_{f} + 2.774D_{l} + 2.325D_{ip} + 2.84D_{ip}</td>
<td></td>
</tr>
<tr>
<td>RND1</td>
<td>4.130D_{Ox} + 1.734D_{f} + 2.851D_{l} + 2.111D_{ip} + 3.140D_{ip}</td>
<td></td>
</tr>
<tr>
<td>RND2</td>
<td>4.220D_{Ox} + 1.817D_{f} + 2.796D_{l} + 2.086D_{ip} + 3.086D_{ip}</td>
<td></td>
</tr>
</tbody>
</table>
Observations during Training: Table 2 shows the metric learned in each experiment with number of epochs. Figures 3 to 7 depict the behavior of LearnMet during training. Experiments EQU, RND1 and RND2 take longer to converge than DE1 and DE2. However, they all converge to approximately the same D.

Observations during Testing: The learned metric in each experiment is used to cluster graphs in the test set. These clusters are compared with correct clusters over the test set. Euclidean distance (ED) is also used to cluster the graphs and the clusters are compared with the correct clusters. The observations for all the experiments are shown in Figure 8. This figure depicts the accuracy (success rate) for each metric over the test set. Clustering accuracies of the learned metrics are higher.

4.2 Evaluation with System Integration

Purpose of Integration: LearnMet has been developed mainly for the AutoDomainMine system, which performs computational estimation of process parameters [13]. AutoDomainMine estimates the graphical result of a scientific experiment given its input conditions, using existing data to predict future trends. This technique clusters graphs from existing experiments and sends clustering output to a decision tree classifier e.g., ID3/J4.8 [12,15] to learn the clustering criteria. For each graph, the input conditions of its corresponding experiment and the cluster in which it was placed are used to construct the decision tree. The tree identifies the combination of input conditions that characterize each cluster. A representative graph is also selected per cluster. When input conditions of a new experiment are submitted, the tree is traversed to predict the cluster. The representative graph of that cluster is the estimated graph for that experiment.

Figure 9a: Evaluation with AutoDomainMine Step 1
Evaluating LearnMet with AutoDomainMine: LearnMet is evaluated by measuring the accuracy of the AutoDomainMine estimation with and without the learned metrics. This process is illustrated in Figures 9a and 9b. The estimation obtained from clustering using the learned metrics is compared with that from clustering using Euclidean distance. Another criterion for comparison is which is called the AutoDomainMine metric denoted as $ADM_{[13,14]}$.

Observations with AutoDomainMine: The average estimation accuracy over 10 experiments, each using 10-fold cross validation, is shown in Figure 10. Accuracy with each metric output from LearnMet is higher than that with Euclidean distance. Accuracies with the learned metrics are also higher than the accuracy with the AutoDomainMine metric.

5. CONCLUSIONS
A technique called LearnMet is proposed to learn a domain-specific distance metric for mining graphs. LearnMet compares clusters of graphs obtained from a state-of-the-art clustering algorithm with correct clusters given by experts. An initial metric is guessed and refined with every round of clustering to give a final metric with error below a threshold. LearnMet is evaluated rigorously in the Heat Treating domain that inspired its development. Additional evaluations will be conducted in related domains. Ongoing research includes determining a good number of pairs per epoch; using normalized weights; considering scaling factors in weight adjustments; refining thresholds; and assigning suitable components to the initial metric without expert input.

ACKNOWLEDGMENTS
This work is supported by the Center for Heat Treating Excellence and by DOE – ITP Award DE-FC-07-01ID14197.

REFERENCES