10 Mining Object, Spatial, Multimedia, Text, and Web Data

Our previous chapters on advanced data mining discussed how to uncover knowledge from stream, time-series, sequence, graph, social network, and multirelational data. In this chapter, we examine data mining methods that handle object, spatial, multimedia, text, and Web data. These kinds of data are commonly encountered in many social, economic, scientific, engineering, and governmental applications, and pose new challenges in data mining. We first examine how to perform multidimensional analysis and descriptive mining of complex data objects in Section 10.1. We then study methods for mining spatial data (Section 10.2), multimedia data (Section 10.3), text (Section 10.4), and the World Wide Web (Section 10.5) in sequence.

10.1 Multidimensional Analysis and Descriptive Mining of Complex Data Objects

Many advanced, data-intensive applications, such as scientific research and engineering design, need to store, access, and analyze complex but relatively structured data objects. These objects cannot be represented as simple and uniformly structured records (i.e., tuples) in data relations. Such application requirements have motivated the design and development of *object-relational* and *object-oriented* database systems. Both kinds of systems deal with the efficient storage and access of vast amounts of disk-based **complex structured data objects**. These systems organize a large set of complex data objects into *classes*, which are in turn organized into *class/subclass* hierarchies. Each **object** in a class is associated with (1) an *object-identifier*, (2) a *set of attributes* that may contain sophisticated data structures, set- or list-valued data, class composition hierarchies, multimedia data, and (3) a *set of methods* that specify the computational routines or rules associated with the object class. There has been extensive research in the field of database systems on how to efficiently index, store, access, and manipulate complex objects in object-relational and object-oriented database systems. Technologies handling these issues are discussed in many books on database systems, especially on object-oriented and object-relational database systems.

One step beyond the storage and access of massive-scaled, complex object data is the systematic analysis and mining of such data. This includes two major tasks: (1) construct multidimensional data warehouses for complex object data and perform online analytical processing (OLAP) in such data warehouses, and (2) develop effective and scalable methods for mining knowledge from object databases and/or data warehouses. The second task is largely covered by the mining of specific kinds of data (such as spatial, temporal, sequence, graph- or tree-structured, text, and multimedia data), since these data form the major new kinds of complex data objects. As in Chapters 8 and 9, in this chapter we continue to study methods for mining complex data. Thus, our focus in this section will be mainly on how to construct object data warehouses and perform OLAP analysis on data warehouses for such data.

A major limitation of many commercial data warehouse and OLAP tools for multidimensional database analysis is their restriction on the allowable data types for dimensions and measures. Most data cube implementations confine dimensions to nonnumeric data, and measures to simple, aggregated values. To introduce data mining and multidimensional data analysis for complex objects, this section examines how to perform generalization on complex structured objects and construct object cubes for OLAP and mining in object databases.

To facilitate generalization and induction in object-relational and object-oriented databases, it is important to study how each component of such databases can be generalized, and how the generalized data can be used for multidimensional data analysis and data mining.

10.1.1 Generalization of Structured Data

An important feature of object-relational and object-oriented databases is their capability of storing, accessing, and modeling **complex structure-valued data**, such as set- and list-valued data and data with nested structures.

"*How can generalization be performed on such data?*" Let's start by looking at the generalization of set-valued, list-valued, and sequence-valued attributes.

A **set-valued attribute** may be of homogeneous or heterogeneous type. Typically, set-valued data can be generalized by (1) *generalization of each value in the set to its corresponding higher-level concept*, or (2) *derivation of the general behavior of the set*, such as the number of elements in the set, the types or value ranges in the set, the weighted average for numerical data, or the major clusters formed by the set. Moreover, generalization can be performed by *applying different generalization operators to explore alternative generalization paths*. In this case, the result of generalization is a heterogeneous set.

Example 10.1 **Generalization of a set-valued attribute.** Suppose that the *hobby* of a person is a set-valued attribute containing the set of values {*tennis, hockey, soccer, violin, SimCity*}. This set can be generalized to a set of high-level concepts, such as {*sports, music, computer_games*} or into the number 5 (i.e., the number of hobbies in the set). Moreover, a *count* can be associated with a generalized value to indicate how many elements are generalized to

that value, as in {*sports(3), music(1), computer_games(1)*}, where *sports(3)* indicates *three kinds of sports*, and so on. ■

A set-valued attribute may be generalized to a set-valued or a single-valued attribute; a single-valued attribute may be generalized to a set-valued attribute if the values form a lattice or "hierarchy" or if the generalization follows different paths. Further generalizations on such a generalized set-valued attribute should follow the generalization path of each value in the set.

List-valued attributes and **sequence-valued attributes** can be *generalized in a manner similar to that for set-valued attributes except that the order of the elements in the list or sequence should be preserved in the generalization*. Each value in the list can be generalized into its corresponding higher-level concept. Alternatively, a list can be generalized according to its general behavior, such as the length of the list, the type of list elements, the value range, the weighted average value for numerical data, or by dropping unimportant elements in the list. A list may be generalized into a list, a set, or a single value.

Example 10.2 **Generalization of list-valued attributes.** Consider the following list or sequence of data for a person's education record: "*((B.Sc. in Electrical Engineering, U.B.C., Dec., 1998), (M.Sc. in Computer Engineering, U. Maryland, May, 2001), (Ph.D. in Computer Science, UCLA, Aug., 2005))*". This can be generalized by dropping less important descriptions (attributes) of each tuple in the list, such as by dropping the *month* attribute to obtain "*((B.Sc., U.B.C., 1998), . . .)*", and/or by retaining only the most important tuple(s) in the list, e.g., "*(Ph.D. in Computer Science, UCLA, 2005)*". ■

A **complex structure-valued attribute** may contain sets, tuples, lists, trees, records, and their combinations, where one structure may be *nested* in another at any level. In general, a structure-valued attribute can be generalized in several ways, such as (1) generalizing each attribute in the structure while maintaining the shape of the structure, (2) flattening the structure and generalizing the flattened structure, (3) summarizing the low-level structures by high-level concepts or aggregation, and (4) returning the type or an overview of the structure.

In general, statistical analysis and cluster analysis may help toward deciding on the directions and degrees of generalization to perform, since most generalization processes are to retain main features and remove noise, outliers, or fluctuations.

10.1.2 Aggregation and Approximation in Spatial and Multimedia Data Generalization

Aggregation and approximation are another important means of generalization. They are especially useful for generalizing attributes with large sets of values, complex structures, and spatial or multimedia data.

Let's take **spatial data** as an example. We would like to generalize detailed geographic points into clustered regions, such as business, residential, industrial, or agricultural areas, according to land usage. Such generalization often requires the merge of a set of geographic areas by spatial operations, such as spatial union or spatial

clustering methods. Aggregation and approximation are important techniques for this form of generalization. In a **spatial merge**, it is necessary to not only merge the regions of similar types within the same general class but also to compute the total areas, average density, or other aggregate functions while ignoring some scattered regions with different types if they are unimportant to the study. Other spatial operators, such as *spatial-union, spatial-overlapping,* and *spatial-intersection* (which may require the merging of scattered small regions into large, clustered regions) can also use spatial aggregation and approximation as data generalization operators.

Example 10.3 **Spatial aggregation and approximation.** Suppose that we have different pieces of land for various purposes of agricultural usage, such as the planting of vegetables, grains, and fruits. These pieces can be merged or *aggregated* into one large piece of agricultural land by a spatial merge. However, such a piece of agricultural land may contain highways, houses, and small stores. If the majority of the land is used for agriculture, the scattered regions for other purposes can be ignored, and the whole region can be claimed as an agricultural area by *approximation*. ■

A **multimedia database** may contain complex texts, graphics, images, video fragments, maps, voice, music, and other forms of audio/video information. Multimedia data are typically stored as sequences of bytes with variable lengths, and segments of data are linked together or indexed in a multidimensional way for easy reference.

Generalization on multimedia data can be performed by recognition and extraction of the essential features and/or general patterns of such data. There are many ways to extract such information. For an *image*, the size, color, shape, texture, orientation, and relative positions and structures of the contained objects or regions in the image can be extracted by aggregation and/or approximation. For a segment of *music*, its melody can be summarized based on the approximate patterns that repeatedly occur in the segment, while its style can be summarized based on its tone, tempo, or the major musical instruments played. For an *article*, its abstract or general organizational structure (e.g., the table of contents, the subject and index terms that frequently occur in the article, etc.) may serve as its generalization.

In general, it is a challenging task to generalize spatial data and multimedia data in order to extract interesting knowledge implicitly stored in the data. Technologies developed in spatial databases and multimedia databases, such as spatial data accessing and analysis techniques, pattern recognition, image analysis, text analysis, content-based image/text retrieval and multidimensional indexing methods, should be integrated with data generalization and data mining techniques to achieve satisfactory results. Techniques for mining such data are further discussed in the following sections.

10.1.3 Generalization of Object Identifiers and Class/Subclass Hierarchies

"How can object identifiers be generalized?" At first glance, it may seem impossible to generalize an object identifier. It remains unchanged even after structural reorganization of the data. However, since objects in an object-oriented database are

organized into classes, which in turn are organized into class/subclass hierarchies, the generalization of an object can be performed by referring to its associated hierarchy. Thus, an object identifier can be generalized as follows. First, the object identifier is generalized to the identifier of the *lowest subclass* to which the object belongs. The identifier of this subclass can then, in turn, be generalized to a higher-level class/subclass identifier by *climbing up* the class/subclass hierarchy. Similarly, a class or a subclass can be generalized to its corresponding superclass(es) by climbing up its associated class/subclass hierarchy.

"*Can inherited properties of objects be generalized?*" Since object-oriented databases are organized into class/subclass hierarchies, some attributes or methods of an object class are not explicitly specified in the class but are inherited from higher-level classes of the object. Some object-oriented database systems allow **multiple inheritance,** where properties can be inherited from more than one superclass when the class/subclass "hierarchy" is organized in the shape of a lattice. The inherited properties of an object can be derived by query processing in the object-oriented database. From the data generalization point of view, it is unnecessary to distinguish which data are stored within the class and which are inherited from its superclass. As long as the set of relevant data are collected by query processing, the data mining process will treat the inherited data in the same manner as the data stored in the object class, and perform generalization accordingly.

Methods are an important component of object-oriented databases. They can also be inherited by objects. Many behavioral data of objects can be derived by the application of methods. Since a method is usually defined by a computational procedure/function or by a set of deduction rules, it is impossible to perform generalization on the method itself. However, generalization can be performed on *the data derived* by application of the method. That is, once the set of task-relevant data is derived by application of the method, generalization can then be performed on these data.

10.1.4 Generalization of Class Composition Hierarchies

An attribute of an object may be composed of or described by another object, some of whose attributes may be in turn composed of or described by other objects, thus forming a **class composition hierarchy**. Generalization on a class composition hierarchy can be viewed as generalization on a set of nested structured data (which are possibly infinite, if the nesting is recursive).

In principle, the reference to a composite object may traverse via a long sequence of references along the corresponding class composition hierarchy. However, in most cases, the longer the sequence of references traversed, the weaker the semantic linkage between the original object and the referenced composite object. For example, an attribute *vehicles_owned* of an object class *student* could refer to another object class *car*, which may contain an attribute *auto_dealer*, which may refer to attributes describing the dealer's *manager* and *children*. Obviously, it is unlikely that any interesting general regularities exist between a student and her car dealer's manager's children. Therefore, generalization on a class of objects should be performed on the descriptive attribute values and methods of the class, with limited reference to its closely related components

via its closely related linkages in the class composition hierarchy. That is, in order to discover interesting knowledge, generalization should be performed on the objects in the class composition hierarchy that are *closely related in semantics* to the currently focused class(es), but not on those that have only remote and rather weak semantic linkages.

10.1.5 Construction and Mining of Object Cubes

In an object database, data generalization and multidimensional analysis are not applied to individual objects but to classes of objects. Since a set of objects in a class may share many attributes and methods, and the generalization of each attribute and method may apply a sequence of generalization operators, the major issue becomes how to make the generalization processes cooperate among different attributes and methods in the class(es).

"*So, how can class-based generalization be performed for a large set of objects?*" For class-based generalization, the *attribute-oriented induction method* developed in Chapter 4 for mining characteristics of relational databases can be extended to mine data characteristics in object databases. Consider that a generalization-based data mining process can be viewed as the application of a sequence of class-based generalization operators on different attributes. Generalization can continue until the resulting class contains a small number of generalized objects that can be summarized as a concise, generalized rule in high-level terms. For efficient implementation, the generalization of multidimensional attributes of a complex object class can be performed by examining each attribute (or dimension), generalizing each attribute to simple-valued data, and constructing a multidimensional data cube, called an **object cube**. Once an object cube is constructed, multidimensional analysis and data mining can be performed on it in a manner similar to that for relational data cubes.

Notice that from the application point of view, it is not always desirable to generalize a set of values to single-valued data. Consider the attribute *keyword*, which may contain a set of keywords describing a book. It does not make much sense to generalize this set of keywords to one single value. In this context, it is difficult to construct an object cube containing the *keyword* dimension. We will address some progress in this direction in the next section when discussing spatial data cube construction. However, it remains a challenging research issue to develop techniques for handling set-valued data effectively in object cube construction and object-based multidimensional analysis.

10.1.6 Generalization-Based Mining of Plan Databases by Divide-and-Conquer

To show how generalization can play an important role in mining complex databases, we examine a case of mining significant patterns of successful actions in a plan database using a divide-and-conquer strategy.

A **plan** consists of a variable sequence of *actions*. A **plan database,** or simply a **planbase**, is a large collection of plans. **Plan mining** is the task of mining significant

patterns or knowledge from a planbase. Plan mining can be used to discover travel patterns of business passengers in an air flight database or to find significant patterns from the sequences of actions in the repair of automobiles. Plan mining is different from sequential pattern mining, where a large number of frequently occurring sequences are mined at a very detailed level. Instead, plan mining is the extraction of important or significant *generalized* (sequential) patterns from a planbase.

Let's examine the plan mining process using an air travel example.

Example 10.4 **An air flight planbase.** Suppose that the air travel planbase shown in Table 10.1 stores customer flight sequences, where each record corresponds to an *action* in a sequential database, and a *sequence* of records sharing the same plan number is considered as one plan with a sequence of actions. The columns *departure* and *arrival* specify the codes of the airports involved. Table 10.2 stores information about each airport.

There could be many patterns mined from a planbase like Table 10.1. For example, we may discover that most flights from cities in the Atlantic United States to Midwestern cities have a stopover at ORD in Chicago, which could be because ORD is the principal hub for several major airlines. Notice that the airports that act as airline hubs (such as LAX in Los Angeles, ORD in Chicago, and JFK in New York) can easily be derived from Table 10.2 based on *airport_size*. However, there could be hundreds of hubs in a travel database. Indiscriminate mining may result in a large number of "rules" that lack substantial support, without providing a clear overall picture.

Table 10.1 A database of travel plans: a travel planbase.

plan#	*action#*	*departure*	*departure_time*	*arrival*	*arrival_time*	*airline*	···
1	1	ALB	800	JFK	900	TWA	···
1	2	JFK	1000	ORD	1230	UA	···
1	3	ORD	1300	LAX	1600	UA	···
1	4	LAX	1710	SAN	1800	DAL	···
2	1	SPI	900	ORD	950	AA	···
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 10.2 An airport information table.

airport_code	*city*	*state*	*region*	*airport_size*	···
ORD	Chicago	Illinois	Mid-West	100000	···
SPI	Springfield	Illinois	Mid-West	10000	···
LAX	Los Angeles	California	Pacific	80000	···
ALB	Albany	New York	Atlantic	20000	···
⋮	⋮	⋮	⋮	⋮	⋮

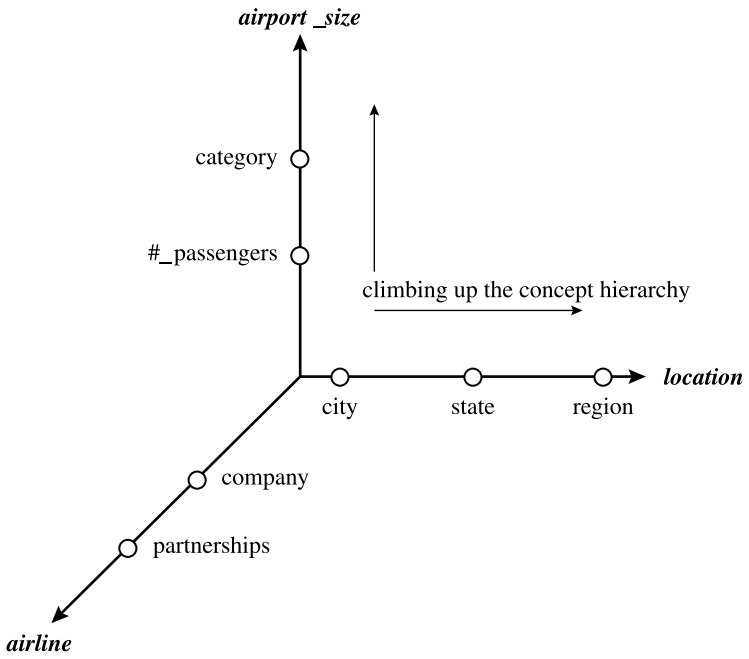

Figure 10.1 A multidimensional view of a database.

"*So, how should we go about mining a planbase?*" We would like to find a small number of general (sequential) patterns that cover a substantial portion of the plans, and then we can divide our search efforts based on such mined sequences. The key to mining such patterns is to generalize the plans in the planbase to a sufficiently high level. A multidimensional database model, such as the one shown in Figure 10.1 for the air flight planbase, can be used to facilitate such plan generalization. Since low-level information may never share enough commonality to form succinct plans, we should do the following: (1) generalize the planbase in different directions using the multidimensional model; (2) observe when the generalized plans share common, interesting, sequential patterns with substantial support; and (3) derive high-level, concise plans.

Let's examine this planbase. By combining tuples with the same plan number, the sequences of actions (shown in terms of airport codes) may appear as follows:

ALB - JFK - ORD - LAX - SAN

SPI - ORD - JFK - SYR

...

Table 10.3 Multidimensional generalization of a planbase.

plan#	*loc_seq*	*size_seq*	*state_seq*	*region_seq*	···
1	ALB-JFK-ORD-LAX-SAN	S-L-L-L-S	N-N-I-C-C	E-E-M-P-P	···
2	SPI-ORD-JFK-SYR	S-L-L-S	I-I-N-N	M-M-E-E	···
⋮	⋮	⋮	⋮	⋮	⋮

Table 10.4 Merging consecutive, identical actions in plans.

plan#	*size_seq*	*state_seq*	*region_seq*	···
1	S-L^+-S	N^+-I-C^+	E^+-M-P^+	···
2	S-L^+-S	I^+-N^+	M^+-E^+	···
⋮	⋮	⋮	⋮	⋮

These sequences may look very different. However, they can be generalized in multiple dimensions. When they are generalized based on the *airport_size* dimension, we observe some interesting sequential patterns, like *S-L-L-S*, where *L* represents a large airport (i.e., a hub), and *S* represents a relatively small regional airport, as shown in Table 10.3.

The generalization of a large number of air travel plans may lead to some rather general but highly regular patterns. This is often the case if the **merge** and **optional** operators are applied to the generalized sequences, where the former merges (and collapses) consecutive identical symbols into one using the transitive closure notation "+" to represent a sequence of actions of the same type, whereas the latter uses the notation "[ ]" to indicate that the object or action inside the square brackets "[ ]" is optional. Table 10.4 shows the result of applying the *merge* operator to the plans of Table 10.3.

By merging and collapsing similar actions, we can derive generalized sequential patterns, such as Pattern (10.1):

$$[S] - L^+ - [S] \qquad [98.5\%] \tag{10.1}$$

The pattern states that 98.5% of travel plans have the pattern $[S] - L^+ - [S]$, where $[S]$ indicates that action S is optional, and L^+ indicates one or more repetitions of L. In other words, the travel pattern consists of flying first from possibly a small airport, hopping through one to many large airports, and finally reaching a large (or possibly, a small) airport.

After a sequential pattern is found with sufficient support, it can be used to partition the planbase. We can then mine each partition to find common characteristics. For example, from a partitioned planbase, we may find

$$\mathit{flight}(x,y) \wedge \mathit{airport\_size}(x,S) \wedge \mathit{airport\_size}(y,L) \Rightarrow \mathit{region}(x) = \mathit{region}(y)\ [75\%], \tag{10.2}$$

which means that for a direct flight from a small airport x to a large airport y, there is a 75% probability that x and y belong to the same region. ■

This example demonstrates a *divide-and-conquer strategy*, which first finds interesting, high-level concise sequences of plans by multidimensional generalization of a planbase, and then partitions the planbase based on mined patterns to discover the corresponding characteristics of subplanbases. This mining approach can be applied to many other applications. For example, in Weblog mining, we can study general access patterns from the Web to identify popular Web portals and common paths before digging into detailed subordinate patterns.

The plan mining technique can be further developed in several aspects. For instance, a *minimum support threshold* similar to that in association rule mining can be used to determine the level of generalization and ensure that a pattern covers a sufficient number of cases. Additional operators in plan mining can be explored, such as *less_than*. Other variations include extracting associations from subsequences, or mining sequence patterns involving multidimensional attributes—for example, the patterns involving both airport size and location. Such dimension-combined mining also requires the generalization of each dimension to a high level before examination of the combined sequence patterns.

10.2 Spatial Data Mining

A **spatial database** stores a large amount of space-related data, such as maps, preprocessed remote sensing or medical imaging data, and VLSI chip layout data. Spatial databases have many features distinguishing them from relational databases. They carry topological and/or distance information, usually organized by sophisticated, multidimensional spatial indexing structures that are accessed by spatial data access methods and often require spatial reasoning, geometric computation, and spatial knowledge representation techniques.

Spatial data mining refers to the extraction of knowledge, spatial relationships, or other interesting patterns not explicitly stored in spatial databases. Such mining demands an integration of data mining with spatial database technologies. It can be used for understanding spatial data, discovering spatial relationships and relationships between spatial and nonspatial data, constructing spatial knowledge bases, reorganizing spatial databases, and optimizing spatial queries. It is expected to have wide applications in geographic information systems, geomarketing, remote sensing, image database exploration, medical imaging, navigation, traffic control, environmental studies, and many other areas where spatial data are used. A crucial challenge to spatial data mining is the exploration of *efficient* spatial data mining techniques due to the huge amount of spatial data and the complexity of spatial data types and spatial access methods.

"*What about using statistical techniques for spatial data mining?*" Statistical spatial data analysis has been a popular approach to analyzing spatial data and exploring geographic information. The term *geostatistics* is often associated with continuous geographic space,

whereas the term *spatial statistics* is often associated with discrete space. In a statistical model that handles nonspatial data, one usually assumes statistical independence among different portions of data. However, different from traditional data sets, there is no such independence among spatially distributed data because in reality, spatial objects are often interrelated, or more exactly spatially *co-located*, in the sense that *the closer the two objects are located, the more likely they share similar properties*. For example, nature resource, climate, temperature, and economic situations are likely to be similar in geographically closely located regions. People even consider this as the first law of geography: "*Everything is related to everything else, but nearby things are more related than distant things.*" Such a property of close interdependency across nearby space leads to the notion of **spatial autocorrelation**. Based on this notion, spatial statistical modeling methods have been developed with good success. Spatial data mining will further develop spatial statistical analysis methods and extend them for huge amounts of spatial data, with more emphasis on efficiency, scalability, cooperation with database and data warehouse systems, improved user interaction, and the discovery of new types of knowledge.

10.2.1 Spatial Data Cube Construction and Spatial OLAP

"*Can we construct a spatial data warehouse?*" Yes, as with relational data, we can integrate spatial data to construct a data warehouse that facilitates spatial data mining. A **spatial data warehouse** is a *subject-oriented, integrated, time-variant*, and *nonvolatile* collection of both spatial and nonspatial data in support of spatial data mining and spatial-data-related decision-making processes.

Let's look at the following example.

Example 10.5 **Spatial data cube and spatial OLAP.** There are about 3,000 weather probes distributed in British Columbia (BC), Canada, each recording daily temperature and precipitation for a designated small area and transmitting signals to a provincial weather station. With a spatial data warehouse that supports spatial OLAP, a user can view weather patterns on a map by month, by region, and by different combinations of temperature and precipitation, and can dynamically drill down or roll up along any dimension to explore desired patterns, such as "wet and hot regions in the Fraser Valley in Summer 1999." ■

There are several challenging issues regarding the construction and utilization of spatial data warehouses. The first challenge is the integration of spatial data from heterogeneous sources and systems. Spatial data are usually stored in different industry firms and government agencies using various data formats. Data formats are not only structure-specific (e.g., raster- vs. vector-based spatial data, object-oriented vs. relational models, different spatial storage and indexing structures), but also vendor-specific (e.g., ESRI, MapInfo, Intergraph). There has been a great deal of work on the integration and exchange of heterogeneous spatial data, which has paved the way for spatial data integration and spatial data warehouse construction.

The second challenge is the realization of fast and flexible on-line analytical processing in spatial data warehouses. The star schema model introduced in Chapter 3 is a good

choice for modeling spatial data warehouses because it provides a concise and organized warehouse structure and facilitates OLAP operations. However, in a spatial warehouse, both dimensions and measures may contain spatial components.

There are three types of *dimensions* in a spatial data cube:

- A **nonspatial dimension** contains only nonspatial data. Nonspatial dimensions *temperature* and *precipitation* can be constructed for the warehouse in Example 10.5, since each contains nonspatial data whose generalizations are nonspatial (such as *"hot"* for *temperature* and *"wet"* for *precipitation*).
- A **spatial-to-nonspatial dimension** is a dimension whose primitive-level data are spatial but whose generalization, starting at a certain high level, becomes nonspatial. For example, the spatial dimension *city* relays geographic data for the U.S. map. Suppose that the dimension's spatial representation of, say, Seattle is generalized to the string *"pacific_northwest."* Although *"pacific_northwest"* is a spatial concept, its representation is not spatial (since, in our example, it is a string). It therefore plays the role of a nonspatial dimension.
- A **spatial-to-spatial dimension** is a dimension whose primitive level and all of its high-level generalized data are spatial. For example, the dimension *equi_temperature_region* contains spatial data, as do all of its generalizations, such as with regions covering *0-5_degrees* (Celsius), *5-10_degrees*, and so on.

We distinguish two types of *measures* in a spatial data cube:

- A **numerical measure** contains only numerical data. For example, one measure in a spatial data warehouse could be the *monthly_revenue* of a region, so that a roll-up may compute the total revenue by year, by county, and so on. Numerical measures can be further classified into *distributive*, *algebraic*, and *holistic*, as discussed in Chapter 3.
- A **spatial measure** contains a collection of pointers to spatial objects. For example, in a generalization (or roll-up) in the spatial data cube of Example 10.5, the regions with the same range of *temperature* and *precipitation* will be grouped into the same cell, and the measure so formed contains a collection of pointers to those regions.

A nonspatial data cube contains only nonspatial dimensions and numerical measures. If a spatial data cube contains spatial dimensions but no spatial measures, its OLAP operations, such as drilling or pivoting, can be implemented in a manner similar to that for nonspatial data cubes.

"But what if I need to use spatial measures in a spatial data cube?" This notion raises some challenging issues on efficient implementation, as shown in the following example.

Example 10.6 **Numerical versus spatial measures.** A star schema for the *BC_weather* warehouse of Example 10.5 is shown in Figure 10.2. It consists of four dimensions: *region temperature*, *time*, and *precipitation*, and three measures: *region_map*, *area*, and *count*. A concept hierarchy for each dimension can be created by users or experts, or generated automatically

by data clustering analysis. Figure 10.3 presents hierarchies for each of the dimensions in the *BC_weather* warehouse.

Of the three measures, *area* and *count* are *numerical* measures that can be computed similarly as for nonspatial data cubes; *region_map* is a *spatial* measure that represents a collection of spatial pointers to the corresponding regions. Since different spatial OLAP operations result in different collections of spatial objects in *region_map*, it is a major challenge to compute the merges of a large number of regions flexibly and dynamically. For example, two different roll-ups on the BC weather map data (Figure 10.2) may produce two different generalized region maps, as shown in Figure 10.4, each being the result of merging a large number of small (probe) regions from Figure 10.2. ■

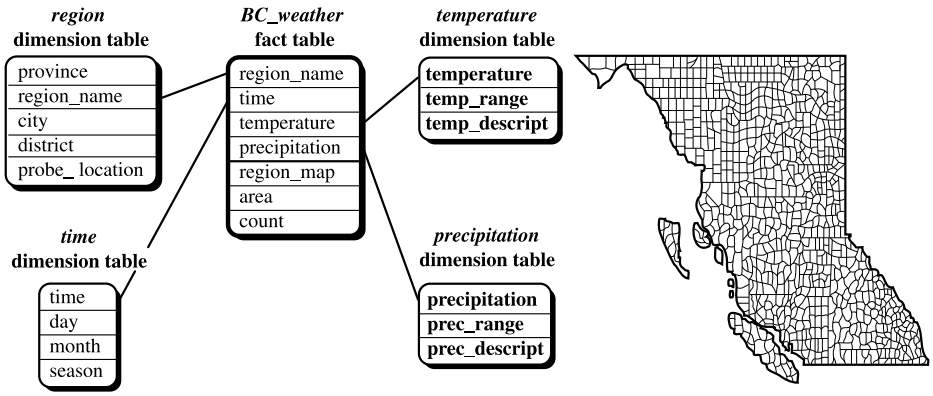

Figure 10.2 A star schema of the *BC_weather* spatial data warehouse and corresponding BC weather probes map.

region_name dimension:
probe_location < *district* < *city* < *region* < *province*

time dimension:
hour < *day* < *month* < *season*

temperature dimension:
(*cold, mild, hot*) ⊂ all(*temperature*)
(*below_−20*, −20...−11, −10...0) ⊂ *cold*
(0...10, 11...15, 16...20) ⊂ *mild*
(20...25, 26...30, 31...35, *above_35*) ⊂ *hot*

precipitation dimension:
(*dry, fair, wet*) ⊂ all(*precipitation*)
(0...0.05, 0.06...0.2) ⊂ *dry*
(0.2...0.5, 0.6...1.0, 1.1...1.5) ⊂ *fair*
(1.5...2.0, 2.1...3.0, 3.1...5.0, *above_5.0*) ⊂ *wet*

Figure 10.3 Hierarchies for each dimension of the *BC_weather* data warehouse.

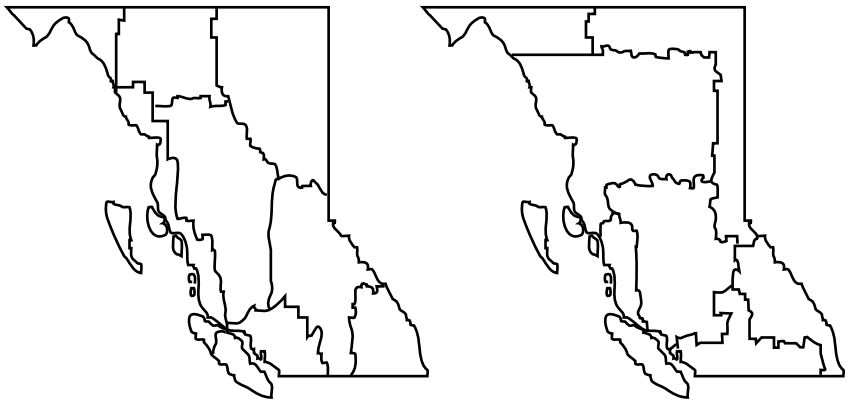

Figure 10.4 Generalized regions after different roll-up operations.

"Can we precompute all of the possible spatial merges and store them in the corresponding cuboid cells of a spatial data cube?" The answer is—probably not. Unlike a numerical measure where each aggregated value requires only a few bytes of space, a merged region map of BC may require multi-megabytes of storage. Thus, we face a dilemma in balancing the cost of on-line computation and the space overhead of storing computed measures: the substantial computation cost for on-the-fly computation of spatial aggregations calls for precomputation, yet substantial overhead for storing aggregated spatial values discourages it.

There are at least three possible choices in regard to the computation of spatial measures in spatial data cube construction:

- *Collect and store the corresponding spatial object pointers but do not perform precomputation of spatial measures in the spatial data cube.* This can be implemented by storing, in the corresponding cube cell, a pointer to a collection of spatial object pointers, and invoking and performing the spatial merge (or other computation) of the corresponding spatial objects, when necessary, on the fly. This method is a good choice if only spatial display is required (i.e., no real spatial merge has to be performed), or if there are not many regions to be merged in any pointer collection (so that the on-line merge is not very costly), or if on-line spatial merge computation is fast (recently, some efficient spatial merge methods have been developed for fast spatial OLAP). Since OLAP results are often used for on-line spatial analysis and mining, it is still recommended to precompute some of the spatially connected regions to speed up such analysis.
- *Precompute and store a rough approximation of the spatial measures in the spatial data cube.* This choice is good for a rough view or coarse estimation of spatial merge results under the assumption that it requires little storage space. For example, a **minimum bounding rectangle** (**MBR**), represented by two points, can be taken as a rough estimate

of a merged region. Such a precomputed result is small and can be presented quickly to users. If higher precision is needed for specific cells, the application can either fetch precomputed high-quality results, if available, or compute them on the fly.

- *Selectively precompute some spatial measures in the spatial data cube.* This can be a smart choice. The question becomes, "Which portion of the cube should be selected for materialization?" The selection can be performed at the cuboid level, that is, either precompute and store *each* set of mergeable spatial regions for *each* cell of a selected cuboid, or precompute none if the cuboid is not selected. Since a cuboid usually consists of a large number of spatial objects, it may involve precomputation and storage of a large number of mergeable spatial objects, some of which may be rarely used. Therefore, it is recommended to perform selection at a finer granularity level: examining each group of mergeable spatial objects in a cuboid to determine whether such a merge should be precomputed. The decision should be based on the utility (such as access frequency or access priority), shareability of merged regions, and the balanced overall cost of space and on-line computation.

With efficient implementation of spatial data cubes and spatial OLAP, generalization-based descriptive spatial mining, such as spatial characterization and discrimination, can be performed efficiently.

10.2.2 Mining Spatial Association and Co-location Patterns

Similar to the mining of association rules in transactional and relational databases, *spatial association rules* can be mined in spatial databases. A **spatial association rule** is of the form $A \Rightarrow B\ [s\%, c\%]$, where A and B are sets of spatial or nonspatial predicates, $s\%$ is the support of the rule, and $c\%$ is the confidence of the rule. For example, the following is a spatial association rule:

$$is\_a(X, \text{“}school\text{”}) \wedge close\_to(X, \text{“}sports\_center\text{”}) \Rightarrow close\_to(X, \text{“}park\text{”}) \quad [0.5\%, 80\%].$$

This rule states that 80% of schools that are close to sports centers are also close to parks, and 0.5% of the data belongs to such a case.

Various kinds of spatial predicates can constitute a spatial association rule. Examples include distance information (such as *close_to* and *far_away*), topological relations (like *intersect, overlap*, and *disjoint*), and spatial orientations (like *left_of* and *west_of*).

Since spatial association mining needs to evaluate multiple spatial relationships among a large number of spatial objects, the process could be quite costly. An interesting mining optimization method called **progressive refinement** can be adopted in spatial association analysis. The method first mines large data sets *roughly* using a fast algorithm and then improves the quality of mining in a pruned data set using a more expensive algorithm.

To ensure that the pruned data set covers the complete set of answers when applying the high-quality data mining algorithms at a later stage, an important requirement for the rough mining algorithm applied in the early stage is the **superset coverage property**: that is, it preserves all of the potential answers. In other words, it should allow a *false-positive*

test, which might include some data sets that do not belong to the answer sets, but it should not allow a *false-negative test*, which might exclude some potential answers.

For mining spatial associations related to the spatial predicate *close_to*, we can first collect the candidates that pass the minimum support threshold by

- Applying certain rough spatial evaluation algorithms, for example, using an MBR structure (which registers only two spatial points rather than a set of complex polygons), and
- Evaluating the relaxed spatial predicate, *g_close_to*, which is a generalized *close_to* covering a broader context that includes *close_to, touch*, and *intersect*.

If two spatial objects are closely located, their enclosing MBRs must be closely located, matching *g_close_to*. However, the reverse is not always true: if the enclosing MBRs are closely located, the two spatial objects may or may not be located so closely. Thus, the MBR pruning is a false-positive testing tool for closeness: only those that pass the *rough* test need to be further examined using more expensive spatial computation algorithms. With this preprocessing, only the patterns that are frequent at the approximation level will need to be examined by more detailed and finer, yet more expensive, spatial computation.

Besides mining spatial association rules, one may like to identify groups of particular features that appear frequently close to each other in a geospatial map. Such a problem is essentially the problem of mining **spatial co-locations**. Finding spatial co-locations can be considered as a special case of mining spatial associations. However, based on the property of spatial autocorrelation, interesting features likely coexist in closely located regions. Thus spatial co-location can be just what one really wants to explore. Efficient methods can be developed for mining spatial co-locations by exploring the methodologies like Aprori and progressive refinement, similar to what has been done for mining spatial association rules.

10.2.3 Spatial Clustering Methods

Spatial data clustering identifies clusters, or densely populated regions, according to some distance measurement in a large, multidimensional data set. Spatial clustering methods were thoroughly studied in Chapter 7 since cluster analysis usually considers spatial data clustering in examples and applications. Therefore, readers interested in spatial clustering should refer to Chapter 7.

10.2.4 Spatial Classification and Spatial Trend Analysis

Spatial classification analyzes spatial objects to derive classification schemes in relevance to certain spatial properties, such as the *neighborhood* of a district, highway, or river.

Example 10.7 **Spatial classification.** Suppose that you would like to classify regions in a province into *rich* versus *poor* according to the average family income. In doing so, you would like to identify the important spatial-related factors that determine a region's classification.

Many properties are associated with spatial objects, such as hosting a university, containing interstate highways, being near a lake or ocean, and so on. These properties can be used for relevance analysis and to find interesting classification schemes. Such classification schemes may be represented in the form of decision trees or rules, for example, as described in Chapter 6. ■

Spatial trend analysis deals with another issue: the detection of changes and trends along a spatial dimension. Typically, trend analysis detects changes with time, such as the changes of temporal patterns in time-series data. Spatial trend analysis replaces time with space and studies the trend of nonspatial or spatial data changing with space. For example, we may observe the trend of changes in economic situation when moving away from the center of a city, or the trend of changes of the climate or vegetation with the increasing distance from an ocean. For such analyses, regression and correlation analysis methods are often applied by utilization of spatial data structures and spatial access methods.

There are also many applications where patterns are changing with *both space and time*. For example, traffic flows on highways and in cities are both time and space related. Weather patterns are also closely related to both time and space. Although there have been a few interesting studies on spatial classification and spatial trend analysis, the investigation of spatiotemporal data mining is still in its early stage. More methods and applications of spatial classification and trend analysis, especially those associated with time, need to be explored.

10.2.5 Mining Raster Databases

Spatial database systems usually handle vector data that consist of points, lines, polygons (regions), and their compositions, such as networks or partitions. Typical examples of such data include maps, design graphs, and 3-D representations of the arrangement of the chains of protein molecules. However, a huge amount of space-related data are in **digital raster (image) forms**, such as satellite images, remote sensing data, and computer tomography. It is important to explore data mining in raster or image databases. Methods for mining raster and image data are examined in the following section regarding the mining of multimedia data.

10.3 Multimedia Data Mining

"*What is a multimedia database?*" A **multimedia database system** stores and manages a large collection of *multimedia data*, such as audio, video, image, graphics, speech, text, document, and hypertext data, which contain text, text markups, and linkages. Multimedia database systems are increasingly common owing to the popular use of audio-video equipment, digital cameras, CD-ROMs, and the Internet. Typical multimedia database systems include NASA's EOS (Earth Observation System), various kinds of image and audio-video databases, and Internet databases.

In this section, our study of multimedia data mining focuses on image data mining. Mining text data and mining the World Wide Web are studied in the two subsequent

sections. Here we introduce multimedia data mining methods, including similarity search in multimedia data, multidimensional analysis, classification and prediction analysis, and mining associations in multimedia data.

10.3.1 Similarity Search in Multimedia Data

"*When searching for similarities in multimedia data, can we search on either the data description or the data content?*" That is correct. For similarity searching in multimedia data, we consider two main families of multimedia indexing and retrieval systems: (1) **description-based retrieval** systems, which build indices and perform object retrieval based on image descriptions, such as keywords, captions, size, and time of creation; and (2) **content-based retrieval** systems, which support retrieval based on the image content, such as color histogram, texture, pattern, image topology, and the shape of objects and their layouts and locations within the image. Description-based retrieval is labor-intensive if performed manually. If automated, the results are typically of poor quality. For example, the assignment of keywords to images can be a tricky and arbitrary task. Recent development of Web-based image clustering and classification methods has improved the quality of description-based Web image retrieval, because imagesurrounded text information as well as Web linkage information can be used to extract proper description and group images describing a similar theme together. Content-based retrieval uses visual features to index images and promotes object retrieval based on feature similarity, which is highly desirable in many applications.

In a content-based image retrieval system, there are often two kinds of queries: *image-sample-based queries* and *image feature specification queries*. **Image-sample-based queries** find all of the images that are similar to the given image sample. This search compares the **feature vector** (or **signature**) extracted from the sample with the feature vectors of images that have already been extracted and indexed in the image database. Based on this comparison, images that are close to the sample image are returned. **Image feature specification queries** specify or sketch image features like color, texture, or shape, which are translated into a feature vector to be matched with the feature vectors of the images in the database. Content-based retrieval has wide applications, including medical diagnosis, weather prediction, TV production, Web search engines for images, and e-commerce. Some systems, such as *QBIC* (*Query By Image Content*), support both sample-based and image feature specification queries. There are also systems that support both content-based and description-based retrieval.

Several approaches have been proposed and studied for similarity-based retrieval in image databases, based on image signature:

- **Color histogram–based signature**: In this approach, the signature of an image includes color histograms based on the color composition of an image regardless of its scale or orientation. This method does not contain any information about shape, image topology, or texture. Thus, two images with similar color composition but that contain very different shapes or textures may be identified as similar, although they could be completely unrelated semantically.

- **Multifeature composed signature**: In this approach, the signature of an image includes a composition of multiple features: color histogram, shape, image topology, and texture. The extracted image features are stored as metadata, and images are indexed based on such metadata. Often, separate distance functions can be defined for each feature and subsequently combined to derive the overall results. Multidimensional content-based search often uses one or a few probe features to search for images containing such (similar) features. It can therefore be used to search for similar images. This is the most popularly used approach in practice.
- **Wavelet-based signature**: This approach uses the dominant wavelet coefficients of an image as its signature. Wavelets capture shape, texture, and image topology information in a single unified framework.[1] This improves efficiency and reduces the need for providing multiple search primitives (unlike the second method above). However, since this method computes a single signature for an entire image, it may fail to identify images containing similar objects where the objects *differ* in location or size.
- **Wavelet-based signature with region-based granularity**: In this approach, the computation and comparison of signatures are at the granularity of regions, not the entire image. This is based on the observation that similar images may contain similar regions, but a region in one image could be a translation or scaling of a matching region in the other. Therefore, a similarity measure between the query image Q and a target image T can be defined in terms of the fraction of the area of the two images covered by matching pairs of regions from Q and T. Such a region-based similarity search can find images containing similar objects, where these objects may be translated or scaled.

10.3.2 Multidimensional Analysis of Multimedia Data

"Can we construct a data cube for multimedia data analysis?" To facilitate the multidimensional analysis of large multimedia databases, multimedia data cubes can be designed and constructed in a manner similar to that for traditional data cubes from relational data. A **multimedia data cube** can contain additional dimensions and measures for multimedia information, such as color, texture, and shape.

Let's examine a multimedia data mining system prototype called MultiMediaMiner, which extends the DBMiner system by handling multimedia data. The example database tested in the MultiMediaMiner system is constructed as follows. Each image contains two descriptors: a *feature descriptor* and a *layout descriptor*. The original image is not stored directly in the database; only its descriptors are stored. The description information encompasses fields like image file name, image URL, image type (e.g., gif, tiff, jpeg, mpeg, bmp, avi), a list of all known Web pages referring to the image (i.e., parent URLs), a list of keywords, and a thumbnail used by the user interface for image and video browsing. The **feature descriptor** is a set of vectors for each visual characteristic. The main

[1] Wavelet analysis was introduced in Section 2.5.3.

vectors are a color vector containing the color histogram quantized to 512 colors ($8 \times 8 \times 8$ for $R \times G \times B$), an MFC (Most Frequent Color) vector, and an MFO (Most Frequent Orientation) vector. The MFC and MFO contain five color centroids and five edge orientation centroids for the five most frequent colors and five most frequent orientations, respectively. The edge orientations used are 0°, 22.5°, 45°, 67.5°, 90°, and so on. The **layout descriptor** contains a color layout vector and an edge layout vector. Regardless of their original size, all images are assigned an 8×8 grid. The most frequent color for each of the 64 cells is stored in the color layout vector, and the number of edges for each orientation in each of the cells is stored in the edge layout vector. Other sizes of grids, like 4×4, 2×2, and 1×1, can easily be derived.

The *Image Excavator* component of MultiMediaMiner uses image contextual information, like HTML tags in Web pages, to derive keywords. By traversing on-line directory structures, like the Yahoo! directory, it is possible to create hierarchies of keywords mapped onto the directories in which the image was found. These graphs are used as concept hierarchies for the dimension *keyword* in the multimedia data cube.

"What kind of dimensions can a multimedia data cube have?" A multimedia data cube can have many dimensions. The following are some examples: the size of the image or video in bytes; the width and height of the frames (or pictures), constituting two dimensions; the date on which the image or video was created (or last modified); the format type of the image or video; the frame sequence duration in seconds; the image or video Internet domain; the Internet domain of pages referencing the image or video (parent URL); the keywords; a color dimension; an edge-orientation dimension; and so on. Concept hierarchies for many numerical dimensions may be automatically defined. For other dimensions, such as for Internet domains or color, predefined hierarchies may be used.

The construction of a multimedia data cube will facilitate multidimensional analysis of multimedia data primarily based on visual content, and the mining of multiple kinds of knowledge, including summarization, comparison, classification, association, and clustering. The *Classifier* module of MultiMediaMiner and its output are presented in Figure 10.5.

The multimedia data cube seems to be an interesting model for multidimensional analysis of multimedia data. However, we should note that it is difficult to implement a data cube efficiently given a large number of dimensions. This curse of dimensionality is especially serious in the case of multimedia data cubes. We may like to model color, orientation, texture, keywords, and so on, as multiple dimensions in a multimedia data cube. However, many of these attributes are set-oriented instead of single-valued. For example, one image may correspond to a set of keywords. It may contain a set of objects, each associated with a set of colors. If we use each keyword as a dimension or each detailed color as a dimension in the design of the data cube, it will create a huge number of dimensions. On the other hand, not doing so may lead to the modeling of an image at a rather rough, limited, and imprecise scale. More research is needed on how to design a multimedia data cube that may strike a balance between efficiency and the power of representation.

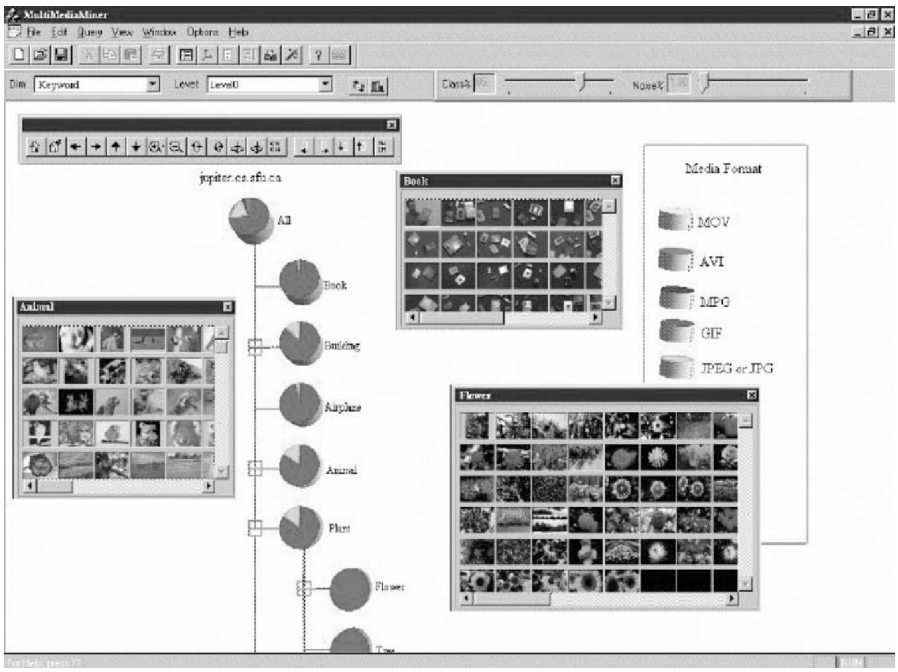

Figure 10.5 An output of the *Classifier* module of MultiMediaMiner.

10.3.3 Classification and Prediction Analysis of Multimedia Data

Classification and predictive modeling have been used for mining multimedia data, especially in scientific research, such as astronomy, seismology, and geoscientific research. In general, all of the classification methods discussed in Chapter 6 can be used in image analysis and pattern recognition. Moreover, in-depth statistical pattern analysis methods are popular for distinguishing subtle features and building high-quality models.

Example 10.8 **Classification and prediction analysis of astronomy data.** Taking sky images that have been carefully classified by astronomers as the training set, we can construct models for the recognition of galaxies, stars, and other stellar objects, based on properties like magnitudes, areas, intensity, image moments, and orientation. A large number of sky images taken by telescopes or space probes can then be tested against the constructed models in order to identify new celestial bodies. Similar studies have successfully been performed to identify volcanoes on Venus. ■

Data preprocessing is important when mining image data and can include data cleaning, data transformation, and feature extraction. Aside from standard methods used in pattern recognition, such as edge detection and Hough transformations, techniques

can be explored, such as the decomposition of images to eigenvectors or the adoption of probabilistic models to deal with uncertainty. Since the image data are often in huge volumes and may require substantial processing power, parallel and distributed processing are useful. Image data mining classification and clustering are closely linked to image analysis and scientific data mining, and thus many image analysis techniques and scientific data analysis methods can be applied to image data mining.

The popular use of the World Wide Web has made the Web a rich and gigantic repository of multimedia data. The Web not only collects a tremendous number of photos, pictures, albums, and video images in the form of on-line multimedia libraries, but also has numerous photos, pictures, animations, and other multimedia forms on almost every Web page. Such pictures and photos, surrounded by text descriptions, located at the different blocks of Web pages, or embedded inside news or text articles, may serve rather different purposes, such as forming an inseparable component of the content, serving as an advertisement, or suggesting an alternative topic. Furthermore, these Web pages are linked with other Web pages in a complicated way. Such text, image location, and Web linkage information, if used properly, may help understand the contents of the text or assist classification and clustering of images on the Web. Data mining by making good use of relative locations and linkages among images, text, blocks within a page, and page links on the Web becomes an important direction in Web data analysis, which will be further examined in Section 10.5 on Web mining.

10.3.4 Mining Associations in Multimedia Data

"*What kinds of associations can be mined in multimedia data?*" Association rules involving multimedia objects can be mined in image and video databases. At least three categories can be observed:

- **Associations between image content and nonimage content features:** A rule like "*If at least 50% of the upper part of the picture is blue, then it is likely to represent sky*" belongs to this category since it links the image content to the keyword *sky*.
- **Associations among image contents that are not related to spatial relationships:** A rule like "*If a picture contains two blue squares, then it is likely to contain one red circle as well*" belongs to this category since the associations are all regarding image contents.
- **Associations among image contents related to spatial relationships:** A rule like "*If a red triangle is between two yellow squares, then it is likely a big oval-shaped object is underneath*" belongs to this category since it associates objects in the image with spatial relationships.

To mine associations among multimedia objects, we can treat each image as a transaction and find frequently occurring patterns among different images.

"*What are the differences between mining association rules in multimedia databases versus in transaction databases?*" There are some subtle differences. First, an image may contain multiple objects, each with many features such as color, shape, texture,

keyword, and spatial location, so there could be many possible associations. In many cases, a feature may be considered as the same in two images at a certain level of resolution, but different at a finer resolution level. Therefore, it is essential to promote a **progressive resolution refinement** approach. That is, we can first mine frequently occurring patterns at a relatively rough resolution level, and then focus only on those that have passed the minimum support threshold when mining at a finer resolution level. This is because the patterns that are not frequent at a rough level cannot be frequent at finer resolution levels. Such a multiresolution mining strategy substantially reduces the overall data mining cost without loss of the quality and completeness of data mining results. This leads to an efficient methodology for mining frequent itemsets and associations in large multimedia databases.

Second, because a picture containing multiple recurrent objects is an important feature in image analysis, recurrence of the same objects should not be ignored in association analysis. For example, a picture containing two golden circles is treated quite differently from that containing only one. This is quite different from that in a transaction database, where the fact that a person buys one gallon of milk or two may often be treated the same as "*buys_milk*." Therefore, the definition of multimedia association and its measurements, such as support and confidence, should be adjusted accordingly.

Third, there often exist important spatial relationships among multimedia objects, such as *above*, *beneath*, *between*, *nearby*, *left-of*, and so on. These features are very useful for exploring object associations and correlations. Spatial relationships together with other content-based multimedia features, such as color, shape, texture, and keywords, may form interesting associations. Thus, spatial data mining methods and properties of topological spatial relationships become important for multimedia mining.

10.3.5 Audio and Video Data Mining

Besides still images, an incommensurable amount of audiovisual information is becoming available in digital form, in digital archives, on the World Wide Web, in broadcast data streams, and in personal and professional databases. This amount is rapidly growing. There are great demands for effective content-based retrieval and data mining methods for audio and video data. Typical examples include searching for and multimedia editing of particular video clips in a TV studio, detecting suspicious persons or scenes in surveillance videos, searching for particular events in a personal multimedia repository such as MyLifeBits, discovering patterns and outliers in weather radar recordings, and finding a particular melody or tune in your MP3 audio album.

To facilitate the recording, search, and analysis of audio and video information from multimedia data, industry and standardization committees have made great strides toward developing a set of standards for multimedia information description and compression. For example, MPEG-*k* (developed by MPEG: *Moving Picture Experts Group*) and JPEG are typical video compression schemes. The most recently released MPEG-7, formally named "*Multimedia Content Description Interface*," is a standard for describing the multimedia content data. It supports some degree of interpretation of the information meaning, which can be passed onto, or accessed by, a device or a computer.

MPEG-7 is not aimed at any one application in particular; rather, the elements that MPEG-7 standardizes support as broad a range of applications as possible. The audiovisual data description in MPEG-7 includes still pictures, video, graphics, audio, speech, three-dimensional models, and information about how these data elements are combined in the multimedia presentation.

The MPEG committee standardizes the following elements in MPEG-7: (1) a set of *descriptors*, where each descriptor defines the syntax and semantics of a feature, such as color, shape, texture, image topology, motion, or title; (2) a set of *descriptor schemes*, where each scheme specifies the structure and semantics of the relationships between its components (descriptors or description schemes); (3) a set of *coding schemes* for the descriptors, and (4) a *description definition language* (DDL) to specify schemes and descriptors. Such standardization greatly facilitates content-based video retrieval and video data mining.

It is unrealistic to treat a video clip as a long sequence of individual still pictures and analyze each picture since there are too many pictures, and most adjacent images could be rather similar. In order to capture the story or event structure of a video, it is better to treat each video clip as a collection of actions and events in time and first temporarily segment them into video shots. A *shot* is a group of frames or pictures where the video content from one frame to the adjacent ones does not change abruptly. Moreover, the most representative frame in a video shot is considered the *key frame* of the shot. Each key frame can be analyzed using the image feature extraction and analysis methods studied above in the content-based image retrieval. The sequence of key frames will then be used to define the sequence of the events happening in the video clip. Thus the detection of shots and the extraction of key frames from video clips become the essential tasks in video processing and mining.

Video data mining is still in its infancy. There are still a lot of research issues to be solved before it becomes general practice. Similarity-based preprocessing, compression, indexing and retrieval, information extraction, redundancy removal, frequent pattern discovery, classification, clustering, and trend and outlier detection are important data mining tasks in this domain.

10.4 Text Mining

Most previous studies of data mining have focused on structured data, such as relational, transactional, and data warehouse data. However, in reality, a substantial portion of the available information is stored in **text databases** (or **document databases**), which consist of large collections of documents from various sources, such as news articles, research papers, books, digital libraries, e-mail messages, and Web pages. Text databases are rapidly growing due to the increasing amount of information available in electronic form, such as electronic publications, various kinds of electronic documents, e-mail, and the World Wide Web (which can also be viewed as a huge, interconnected, dynamic text database). Nowadays most of the information in government, industry, business, and other institutions are stored electronically, in the form of text databases.

Data stored in most text databases are *semistructured data* in that they are neither completely unstructured nor completely structured. For example, a document may contain a few structured fields, such as *title*, *authors*, *publication_date*, *category*, and so on, but also contain some largely unstructured text components, such as *abstract* and *contents*. There have been a great deal of studies on the modeling and implementation of semistructured data in recent database research. Moreover, information retrieval techniques, such as text indexing methods, have been developed to handle unstructured documents.

Traditional information retrieval techniques become inadequate for the increasingly vast amounts of text data. Typically, only a small fraction of the many available documents will be relevant to a given individual user. Without knowing what could be in the documents, it is difficult to formulate effective queries for analyzing and extracting useful information from the data. Users need tools to compare different documents, rank the importance and relevance of the documents, or find patterns and trends across multiple documents. Thus, text mining has become an increasingly popular and essential theme in data mining.

10.4.1 Text Data Analysis and Information Retrieval

"*What is information retrieval?*" **Information retrieval** (**IR**) is a field that has been developing in parallel with database systems for many years. Unlike the field of database systems, which has focused on query and transaction processing of structured data, information retrieval is concerned with the organization and retrieval of information from a large number of text-based documents. Since information retrieval and database systems each handle different kinds of data, some database system problems are usually not present in information retrieval systems, such as concurrency control, recovery, transaction management, and update. Also, some common information retrieval problems are usually not encountered in traditional database systems, such as unstructured documents, approximate search based on keywords, and the notion of relevance.

Due to the abundance of text information, information retrieval has found many applications. There exist many information retrieval systems, such as on-line library catalog systems, on-line document management systems, and the more recently developed Web search engines.

A typical information retrieval problem is to locate relevant documents in a document collection based on a user's query, which is often some keywords describing an information need, although it could also be an example relevant document. In such a search problem, a user takes the initiative to "pull" the relevant information out from the collection; this is most appropriate when a user has some ad hoc (i.e., short-term) information need, such as finding information to buy a used car. When a user has a long-term information need (e.g., a researcher's interests), a retrieval system may also take the initiative to "push" any newly arrived information item to a user if the item is judged as being relevant to the user's information need. Such an information access process is called *information filtering*, and the corresponding systems are often called *filtering systems* or *recommender systems*. From a technical viewpoint, however, search and

filtering share many common techniques. Below we briefly discuss the major techniques in information retrieval with a focus on search techniques.

Basic Measures for Text Retrieval: Precision and Recall

"*Suppose that a text retrieval system has just retrieved a number of documents for me based on my input in the form of a query. How can we assess how accurate or correct the system was?*" Let the set of documents relevant to a query be denoted as $\{Relevant\}$, and the set of documents retrieved be denoted as $\{Retrieved\}$. The set of documents that are both relevant and retrieved is denoted as $\{Relevant\} \cap \{Retrieved\}$, as shown in the Venn diagram of Figure 10.6. There are two basic measures for assessing the quality of text retrieval:

- **Precision:** This is the percentage of retrieved documents that are in fact relevant to the query (i.e., "correct" responses). It is formally defined as

$$precision = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Retrieved\}|}.$$

- **Recall:** This is the percentage of documents that are relevant to the query and were, in fact, retrieved. It is formally defined as

$$recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|}.$$

An information retrieval system often needs to trade off recall for precision or vice versa. One commonly used trade-off is the **F-score**, which is defined as the harmonic mean of recall and precision:

$$F\_score = \frac{recall \times precision}{(recall + precision)/2}.$$

The harmonic mean discourages a system that sacrifices one measure for another too drastically.

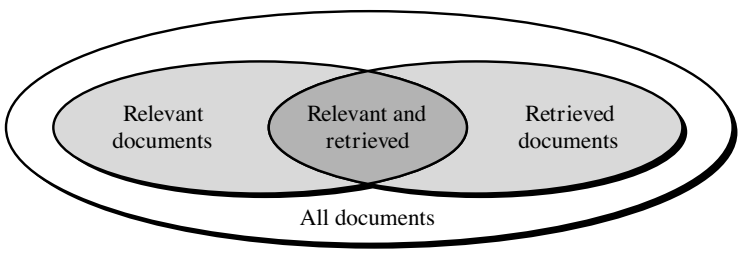

Figure 10.6 Relationship between the set of relevant documents and the set of retrieved documents.

Precision, recall, and F-score are the basic measures of a retrieved set of documents. These three measures are not directly useful for comparing two ranked lists of documents because they are not sensitive to the internal ranking of the documents in a retrieved set. In order to measure the quality of a ranked list of documents, it is common to compute an average of precisions at all the ranks where a new relevant document is returned. It is also common to plot a graph of precisions at many different levels of recall; a higher curve represents a better-quality information retrieval system. For more details about these measures, readers may consult an information retrieval textbook, such as [BYRN99].

Text Retrieval Methods

"*What methods are there for information retrieval?*" Broadly speaking, retrieval methods fall into two categories: They generally either view the retrieval problem as a *document selection problem* or as a *document ranking problem*.

In **document selection** methods, the query is regarded as specifying constraints for selecting relevant documents. A typical method of this category is the **Boolean retrieval model**, in which a document is represented by a set of keywords and a user provides a Boolean expression of keywords, such as "*car* and *repair shops,*" "*tea* or *coffee,*" or "*database systems* but not *Oracle.*" The retrieval system would take such a Boolean query and return documents that satisfy the Boolean expression. Because of the difficulty in prescribing a user's information need exactly with a Boolean query, the Boolean retrieval method generally only works well when the user knows a lot about the document collection and can formulate a good query in this way.

Document ranking methods use the query to rank all documents in the order of relevance. For ordinary users and exploratory queries, these methods are more appropriate than document selection methods. Most modern information retrieval systems present a ranked list of documents in response to a user's keyword query. There are many different ranking methods based on a large spectrum of mathematical foundations, including algebra, logic, probability, and statistics. The common intuition behind all of these methods is that we may match the keywords in a query with those in the documents and score each document based on how well it matches the query. The goal is to approximate the *degree of relevance* of a document with a score computed based on information such as the frequency of words in the document and the whole collection. Notice that it is inherently difficult to provide a precise measure of the degree of relevance between a set of keywords. For example, it is difficult to quantify the distance between *data mining* and *data analysis*. Comprehensive empirical evaluation is thus essential for validating any retrieval method.

A detailed discussion of all of these retrieval methods is clearly out of the scope of this book. Following we briefly discuss the most popular approach—the *vector space model*. For other models, readers may refer to information retrieval textbooks, as referenced in the bibliographic notes. Although we focus on the vector space model, some steps discussed are not specific to this particular approach.

The basic idea of the vector space model is the following: We represent a document and a query both as vectors in a high-dimensional space corresponding to all the

keywords and use an appropriate similarity measure to compute the similarity between the query vector and the document vector. The similarity values can then be used for ranking documents.

"*How do we tokenize text?*" The first step in most retrieval systems is to identify keywords for representing documents, a preprocessing step often called **tokenization**. To avoid indexing useless words, a text retrieval system often associates a *stop list* with a set of documents. A **stop list** is a set of words that are deemed "irrelevant." For example, *a*, *the*, *of*, *for*, *with*, and so on are **stop words,** even though they may appear frequently. Stop lists may vary per document set. For example, *database systems* could be an important keyword in a newspaper. However, it may be considered as a stop word in a set of research papers presented in a database systems conference.

A group of different words may share the same **word stem**. A text retrieval system needs to identify groups of words where the words in a group are small syntactic variants of one another and collect only the common word stem per group. For example, the group of words *drug, drugged*, and *drugs*, share a common word stem, *drug*, and can be viewed as different occurrences of the same word.

"*How can we model a document to facilitate information retrieval?*" Starting with a set of d documents and a set of t terms, we can model each document as a vector v in the t dimensional space $\mathcal{R}^t$, which is why this method is called the **vector-space model**. Let the **term frequency** be the number of occurrences of term t in the document d, that is, $freq(d,t)$. The (weighted) **term-frequency matrix** $TF(d,t)$ measures the association of a term t with respect to the given document d: it is generally defined as 0 if the document does not contain the term, and nonzero otherwise. There are many ways to define the term-weighting for the nonzero entries in such a vector. For example, we can simply set $TF(d,t) = 1$ if the term t occurs in the document d, or use the term frequency $freq(d,t)$, or the **relative term frequency**, that is, the term frequency versus the total number of occurrences of all the terms in the document. There are also other ways to normalize the term frequency. For example, the Cornell SMART system uses the following formula to compute the (normalized) term frequency:

$$TF(d,t) = \begin{cases} 0 & \text{if } freq(d,t) = 0 \\ 1 + \log(1 + \log(freq(d,t))) & otherwise. \end{cases} \tag{10.3}$$

Besides the term frequency measure, there is another important measure, called **inverse document frequency (IDF)**, that represents the scaling factor, or the importance, of a term t. If a term t occurs in many documents, its importance will be scaled down due to its reduced discriminative power. For example, the term *database systems* may likely be less important if it occurs in many research papers in a database system conference. According to the same Cornell SMART system, $IDF(t)$ is defined by the following formula:

$$IDF(t) = \log \frac{1 + |d|}{|d_t|}, \tag{10.4}$$

where d is the document collection, and d_t is the set of documents containing term t. If $|d_t| \ll |d|$, the term t will have a large IDF scaling factor and vice versa.

In a complete vector-space model, TF and IDF are combined together, which forms the **TF-IDF measure**:

$$TF\text{-}IDF(d,t) = TF(d,t) \times IDF(t). \tag{10.5}$$

Let us examine how to compute similarity among a set of documents based on the notions of term frequency and inverse document frequency.

Example 10.9 **Term frequency and inverse document frequency.** Table 10.5 shows a term frequency matrix where each row represents a document vector, each column represents a term, and each entry registers $freq(d_i, t_j)$, the number of occurrences of term t_j in document d_i. Based on this table we can calculate the TF-IDF value of a term in a document. For example, for t_6 in d_4, we have

$$TF(d_4,t_6) = 1 + \log(1 + \log(15)) = 1.3377$$

$$IDF(t_6) = \log \frac{1+5}{3} = 0.301.$$

■

Therefore,

$$TF\text{-}IDF(d_4,t_6) = 1.3377 \times 0.301 = 0.403$$

"*How can we determine if two documents are similar?*" Since similar documents are expected to have similar relative term frequencies, we can measure the similarity among a set of documents or between a document and a query (often defined as a set of keywords), based on similar relative term occurrences in the frequency table. Many metrics have been proposed for measuring document similarity based on relative term occurrences or document vectors. A representative metric is the **cosine measure**, defined as follows. Let v_1 and v_2 be two document vectors. Their cosine similarity is defined as

$$sim(v_1, v_2) = \frac{v_1 \cdot v_2}{|v_1||v_2|}, \tag{10.6}$$

where the inner product $v_1 \cdot v_2$ is the standard vector dot product, defined as $\Sigma_{i=1}^{t} v_{1i} v_{2i}$, and the norm $|v_1|$ in the denominator is defined as $|v_1| = \sqrt{v_1 \cdot v_1}$.

Table 10.5 A term frequency matrix showing the frequency of terms per document.

document/term	t_1	t_2	t_3	t_4	t_5	t_6	t_7
d_1	0	4	10	8	0	5	0
d_2	5	19	7	16	0	0	32
d_3	15	0	0	4	9	0	17
d_4	22	3	12	0	5	15	0
d_5	0	7	0	9	2	4	12

Text Indexing Techniques

There are several popular text retrieval indexing techniques, including *inverted indices* and *signature files*.

An **inverted index** is an index structure that maintains two hash indexed or B+-tree indexed tables: *document_table* and *term_table*, where

- *document_table* consists of a set of document records, each containing two fields: *doc_id* and *posting_list*, where *posting_list* is a list of terms (or pointers to terms) that occur in the document, sorted according to some relevance measure.
- *term_table* consists of a set of term records, each containing two fields: *term_id* and *posting_list*, where *posting_list* specifies a list of document identifiers in which the term appears.

With such organization, it is easy to answer queries like "*Find all of the documents associated with a given set of terms*," or "*Find all of the terms associated with a given set of documents*." For example, to find all of the documents associated with a set of terms, we can first find a list of document identifiers in *term_table* for each term, and then intersect them to obtain the set of relevant documents. Inverted indices are widely used in industry. They are easy to implement. The *posting_list*s could be rather long, making the storage requirement quite large. They are easy to implement, but are not satisfactory at handling *synonymy* (where two very different words can have the same meaning) and *polysemy* (where an individual word may have many meanings).

A **signature file** is a file that stores a *signature* record for each document in the database. Each signature has a fixed size of b bits representing terms. A simple encoding scheme goes as follows. Each bit of a document signature is initialized to 0. A bit is set to 1 if the term it represents appears in the document. A signature S_1 matches another signature S_2 if each bit that is set in signature S_2 is also set in S_1. Since there are usually more terms than available bits, multiple terms may be mapped into the same bit. Such multiple-to-one mappings make the search expensive because a document that matches the signature of a query does not necessarily contain the set of keywords of the query. The document has to be retrieved, parsed, stemmed, and checked. Improvements can be made by first performing frequency analysis, stemming, and by filtering stop words, and then using a hashing technique and superimposed coding technique to encode the list of terms into bit representation. Nevertheless, the problem of multiple-to-one mappings still exists, which is the major disadvantage of this approach.

Readers can refer to [WMB99] for more detailed discussion of indexing techniques, including how to compress an index.

Query Processing Techniques

Once an inverted index is created for a document collection, a retrieval system can answer a keyword query quickly by looking up which documents contain the query keywords. Specifically, we will maintain a score accumulator for each document and update these

accumulators as we go through each query term. For each query term, we will fetch all of the documents that match the term and increase their scores. More sophisticated query processing techniques are discussed in [WMB99].

When examples of relevant documents are available, the system can learn from such examples to improve retrieval performance. This is called *relevance feedback* and has proven to be effective in improving retrieval performance. When we do not have such relevant examples, a system can *assume* the top few retrieved documents in some initial retrieval results to be relevant and extract more related keywords to expand a query. Such feedback is called *pseudo-feedback* or *blind feedback* and is essentially a process of mining useful keywords from the top retrieved documents. Pseudo-feedback also often leads to improved retrieval performance.

One major limitation of many existing retrieval methods is that they are based on exact keyword matching. However, due to the complexity of natural languages, keyword-based retrieval can encounter two major difficulties. The first is the **synonymy problem**: two words with identical or similar meanings may have very different surface forms. For example, a user's query may use the word "automobile," but a relevant document may use "vehicle" instead of "automobile." The second is the **polysemy problem**: the same keyword, such as *mining*, or *Java*, may mean different things in different contexts.

We now discuss some advanced techniques that can help solve these problems as well as reduce the index size.

10.4.2 Dimensionality Reduction for Text

With the similarity metrics introduced in Section 10.4.1, we can construct similarity-based indices on text documents. Text-based queries can then be represented as vectors, which can be used to search for their nearest neighbors in a document collection. However, for any nontrivial document database, the number of terms T and the number of documents D are usually quite large. Such high dimensionality leads to the problem of inefficient computation, since the resulting frequency table will have size $T \times D$. Furthermore, the high dimensionality also leads to very sparse vectors and increases the difficulty in detecting and exploiting the relationships among terms (e.g., synonymy). To overcome these problems, dimensionality reduction techniques such as *latent semantic indexing*, *probabilistic latent semantic analysis*, and *locality preserving indexing* can be used.

We now briefly introduce these methods. To explain the basic idea beneath latent semantic indexing and locality preserving indexing, we need to use some matrix and vector notations. In the following part, we use $\boldsymbol{x_1}, \ldots, \boldsymbol{x_{tn}} \in \mathbb{R}^m$ to represent the n documents with m features (words). They can be represented as a term-document matrix $X = [\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}]$.

Latent Semantic Indexing

Latent semantic indexing (LSI) is one of the most popular algorithms for document dimensionality reduction. It is fundamentally based on SVD (singular value

decomposition). Suppose the *rank* of the term-document X is r, then LSI decomposes X using SVD as follows:

$$X = U\Sigma V^T, \tag{10.7}$$

where $\Sigma = diag(\sigma_1, \ldots, \sigma_r)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are the singular values of X, $U = [\boldsymbol{a_1}, \ldots, \boldsymbol{a_r}]$ and $\boldsymbol{a}_i$ is called the *left singular vector*, and $V = [\boldsymbol{v_1}, \ldots, \boldsymbol{v_r}]$, and $\boldsymbol{v}_i$ is called the *right singular vector*. LSI uses the first k vectors in U as the transformation matrix to embed the original documents into a k-dimensional subspace. It can be easily checked that the column vectors of U are the eigenvectors of XX^T. The basic idea of LSI is to extract the most representative features, and at the same time the reconstruction error can be minimized. Let $\boldsymbol{a}$ be the transformation vector. The objective function of LSI can be stated as follows:

$$\boldsymbol{a}_{opt} = \arg\min_{\boldsymbol{a}} \|X - \boldsymbol{a}\boldsymbol{a}^T X\|^2 = \arg\max_{\boldsymbol{a}} \boldsymbol{a}^T XX^T \boldsymbol{a}, \tag{10.8}$$

with the constraint,

$$\boldsymbol{a}^T \boldsymbol{a} = 1. \tag{10.9}$$

Since XX^T is symmetric, the basis functions of LSI are orthogonal.

Locality Preserving Indexing

Different from LSI, which aims to extract the most representative features, Locality Preserving Indexing (LPI) aims to extract the most discriminative features. The basic idea of LPI is to preserve the locality information (i.e., if two documents are near each other in the original document space, LPI tries to keep these two documents close together in the reduced dimensionality space). Since the neighboring documents (data points in high-dimensional space) probably relate to the same topic, LPI is able to map the documents related to the same semantics as close to each other as possible.

Given the document set $\boldsymbol{x}_1, \ldots, \boldsymbol{x_n} \in \mathbb{R}^m$, LPI constructs a similarity matrix $S \in \mathbb{R}^{n \times n}$. The transformation vectors of LPI can be obtained by solving the following minimization problem:

$$\boldsymbol{a}_{opt} = \arg\min_{\boldsymbol{a}} \sum_{i,j} \left(\boldsymbol{a}^T \boldsymbol{x}_i - \boldsymbol{a}^T \boldsymbol{x}_j\right)^2 S_{ij} = \arg\min_{\boldsymbol{a}} \boldsymbol{a}^T XLX^T \boldsymbol{a}, \tag{10.10}$$

with the constraint,

$$\boldsymbol{a}^T XDX^T \boldsymbol{a} = 1, \tag{10.11}$$

where $L = D - S$ is the *Graph Laplacian* and $D_{ii} = \sum_j S_{ij}$. D_{ii} measures the local density around $\boldsymbol{x}_i$. LPI constructs the similarity matrix S as

$$S_{ij} = \begin{cases} \frac{\boldsymbol{x}_i^T \boldsymbol{x}_j}{\|\boldsymbol{x}_i^T \boldsymbol{x}_j\|}, & \text{if } \boldsymbol{x}_i \text{ is among the } p \text{ nearest neighbors of } \boldsymbol{x}_j \\ & \text{or } \boldsymbol{x}_j \text{ is among the } p \text{ nearest neighbors of } \boldsymbol{x}_i \\ 0, & \text{otherwise.} \end{cases} \tag{10.12}$$

Thus, the objective function in LPI incurs a heavy penalty if neighboring points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are mapped far apart. Therefore, minimizing it is an attempt to ensure that if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are

"close" then y_i ($=\boldsymbol{a}^T\boldsymbol{x}_i$) and y_j ($=\boldsymbol{a}^T\boldsymbol{x}_j$) are close as well. Finally, the basis functions of LPI are the eigenvectors associated with the smallest eigenvalues of the following generalized eigen-problem:

$$XLX^T\boldsymbol{a} = \lambda XDX^T\boldsymbol{a}. \tag{10.13}$$

LSI aims to find the best subspace approximation to the original document space in the sense of minimizing the global reconstruction error. In other words, LSI seeks to uncover the most representative features. LPI aims to discover the local geometrical structure of the document space. Since the neighboring documents (data points in high-dimensional space) probably relate to the same topic, LPI can have more discriminating power than LSI. Theoretical analysis of LPI shows that LPI is an unsupervised approximation of the supervised Linear Discriminant Analysis (LDA). Therefore, for document clustering and document classification, we might expect LPI to have better performance than LSI. This was confirmed empirically.

Probabilistic Latent Semantic Indexing

The probabilistic latent semantic indexing (PLSI) method is similar to LSI, but achieves dimensionality reduction through a probabilistic mixture model. Specifically, we assume there are k latent common themes in the document collection, and each is characterized by a multinomial word distribution. A document is regarded as a sample of a mixture model with these theme models as components. We fit such a mixture model to all the documents, and the obtained k component multinomial models can be regarded as defining k new semantic dimensions. The mixing weights of a document can be used as a new representation of the document in the low latent semantic dimensions.

Formally, let $C = \{d_1, d_2, \ldots, d_n\}$ be a collection of n documents. Let $\theta_1, \ldots, \theta_k$ be k theme multinomial distributions. A word w in document d_i is regarded as a sample of the following mixture model.

$$p_{d_i}(w) = \sum_{j=1}^{k} [\pi_{d_i,j} p(w|\theta_j)] \tag{10.14}$$

where $\pi_{d_i,j}$ is a document-specific mixing weight for the j-th aspect theme, and $\sum_{j=1}^{k} \pi_{d_i,j} = 1$.

The log-likelihood of the collection C is

$$\log p(C|\Lambda) = \sum_{i=1}^{n} \sum_{w \in V} [c(w, d_i) \log(\sum_{j=1}^{k} (\pi_{d_i,j} p(w|\theta_j)))], \tag{10.15}$$

where V is the set of all the words (i.e., vocabulary), $c(w, d_i)$ is the count of word w in document d_i, and $\Lambda = (\{\theta_j, \{\pi_{d_i,j}\}_{i=1}^{n}\}_{j=1}^{k})$ is the set of all the theme model parameters.

The model can be estimated using the Expectation-Maximization (EM) algorithm (Chapter 7), which computes the following maximum likelihood estimate:

$$\hat{\Lambda} = argmax_{\Lambda} \log p(C|\Lambda). \tag{10.16}$$

Once the model is estimated, $\theta_1, \ldots, \theta_k$ define k new semantic dimensions and $\pi_{d_i,j}$ gives a representation of d_i in this low-dimension space.

10.4.3 Text Mining Approaches

There are many approaches to text mining, which can be classified from different perspectives, based on the inputs taken in the text mining system and the data mining tasks to be performed. In general, the major approaches, based on the kinds of data they take as input, are: (1) the **keyword-based approach**, where the input is a set of keywords or terms in the documents, (2) the **tagging approach**, where the input is a set of tags, and (3) the **information-extraction approach**, which inputs semantic information, such as events, facts, or entities uncovered by information extraction. A simple keyword-based approach may only discover relationships at a relatively shallow level, such as rediscovery of compound nouns (e.g., "database" and "systems") or co-occurring patterns with less significance (e.g., "terrorist" and "explosion"). It may not bring much deep understanding to the text. The tagging approach may rely on tags obtained by *manual tagging* (which is costly and is unfeasible for large collections of documents) or by some *automated categorization algorithm* (which may process a relatively small set of tags and require defining the categories beforehand). The information-extraction approach is more advanced and may lead to the discovery of some deep knowledge, but it requires semantic analysis of text by natural language understanding and machine learning methods. This is a challenging knowledge discovery task.

Various text mining tasks can be performed on the extracted keywords, tags, or semantic information. These include document clustering, classification, information extraction, association analysis, and trend analysis. We examine a few such tasks in the following discussion.

Keyword-Based Association Analysis

"*What is keyword-based association analysis?*" Such analysis collects sets of keywords or terms that occur frequently together and then finds the association or correlation relationships among them.

Like most of the analyses in text databases, association analysis first preprocesses the text data by parsing, stemming, removing stop words, and so on, and then evokes association mining algorithms. In a document database, each document can be viewed as a transaction, while a set of keywords in the document can be considered as a set of items in the transaction. That is, the database is in the format

$$\{document\_id, a\_set\_of\_keywords\}.$$

The problem of keyword association mining in document databases is thereby mapped to item association mining in transaction databases, where many interesting methods have been developed, as described in Chapter 5.

Notice that a set of frequently occurring consecutive or closely located keywords may form a *term* or a *phrase*. The association mining process can help detect **compound associations**, that is, domain-dependent terms or phrases, such as [Stanford, University] or [U.S., President, George W. Bush], or **noncompound associations**, such as [dollars,

shares, exchange, total, commission, stake, securities]. Mining based on these associations is referred to as "*term-level* association mining" (as opposed to mining on individual words). Term recognition and term-level association mining enjoy two advantages in text analysis: (1) terms and phrases are automatically tagged so there is no need for human effort in tagging documents; and (2) the number of meaningless results is greatly reduced, as is the execution time of the mining algorithms.

With such term and phrase recognition, term-level mining can be evoked to find associations among a set of detected terms and keywords. Some users may like to find associations between pairs of keywords or terms from a given set of keywords or phrases, whereas others may wish to find the maximal set of terms occurring together. Therefore, based on user mining requirements, standard association mining or max-pattern mining algorithms may be evoked.

Document Classification Analysis

Automated document classification is an important text mining task because, with the existence of a tremendous number of on-line documents, it is tedious yet essential to be able to automatically organize such documents into classes to facilitate document retrieval and subsequent analysis. Document classification has been used in automated topic tagging (i.e., assigning labels to documents), topic directory construction, identification of the document writing styles (which may help narrow down the possible authors of anonymous documents), and classifying the purposes of hyperlinks associated with a set of documents.

"*How can automated document classification be performed?*" A general procedure is as follows: First, a set of preclassified documents is taken as the training set. The training set is then analyzed in order to derive a classification scheme. Such a classification scheme often needs to be refined with a testing process. The so-derived classification scheme can be used for classification of other on-line documents.

This process appears similar to the classification of relational data. However, there is a fundamental difference. Relational data are well structured: each tuple is defined by a set of attribute-value pairs. For example, in the tuple {*sunny*, *warm*, *dry*, *not_windy*, *play_tennis*}, the value "*sunny*" corresponds to the attribute *weather_outlook*, "*warm*" corresponds to the attribute *temperature*, and so on. The classification analysis decides which set of attribute-value pairs has the greatest discriminating power in determining whether a person is going to play tennis. On the other hand, document databases are not structured according to attribute-value pairs. That is, a set of keywords associated with a set of documents is not organized into a fixed set of attributes or dimensions. If we view each distinct keyword, term, or feature in the document as a dimension, there may be thousands of dimensions in a set of documents. Therefore, commonly used relational data-oriented classification methods, such as decision tree analysis, may not be effective for the classification of document databases.

Based on our study of a wide spectrum of classification methods in Chapter 6, here we examine a few typical classification methods that have been used successfully in text

classification. These include nearest-neighbor classification, feature selection methods, Bayesian classification, support vector machines, and association-based classification.

According to the vector-space model, two documents are similar if they share similar document vectors. This model motivates the construction of the ***k*-nearest-neighbor classifier**, based on the intuition that similar documents are expected to be assigned the same class label. We can simply index all of the training documents, each associated with its corresponding class label. When a test document is submitted, we can treat it as a query to the IR system and retrieve from the training set k documents that are most similar to the query, where k is a tunable constant. The class label of the test document can be determined based on the class label distribution of its k nearest neighbors. Such class label distribution can also be refined, such as based on weighted counts instead of raw counts, or setting aside a portion of labeled documents for validation. By tuning k and incorporating the suggested refinements, this kind of classifier can achieve accuracy comparable with the best classifier. However, since the method needs nontrivial space to store (possibly redundant) training information and additional time for inverted index lookup, it has additional space and time overhead in comparison with other kinds of classifiers.

The vector-space model may assign large weight to rare items disregarding its class distribution characteristics. Such rare items may lead to ineffective classification. Let's examine an example in the TF-IDF measure computation. Suppose there are two terms t_1 and t_2 in two classes C_1 and C_2, each having 100 training documents. Term t_1 occurs in five documents in each class (i.e., 5% of the overall corpus), but t_2 occurs in 20 documents in class C_1 only (i.e., 10% of the overall corpus). Term t_1 will have a higher TF-IDF value because it is rarer, but it is obvious t_2 has stronger discriminative power in this case. A **feature selection**[2] process can be used to remove terms in the training documents that are statistically uncorrelated with the class labels. This will reduce the set of terms to be used in classification, thus improving both efficiency and accuracy.

After feature selection, which removes nonfeature terms, the resulting "cleansed" training documents can be used for effective classification. **Bayesian classification** is one of several popular techniques that can be used for effective document classification. Since document classification can be viewed as the calculation of the statistical distribution of documents in specific classes, a Bayesian classifier first trains the model by calculating a generative document distribution $P(d|c)$ to each class c of document d and then tests which class is most likely to generate the test document. Since both methods handle high-dimensional data sets, they can be used for effective document classification. Other classification methods have also been used in documentation classification. For example, if we represent classes by numbers and construct a direct mapping function from term space to the class variable, **support vector machines** can be used to perform effective classification since they work well in high-dimensional space. The *least-square linear regression method* is also used as a method for discriminative classification.

[2] Feature (or attribute) selection is described in Chapter 2.

Finally, we introduce **association-based classification**, which classifies documents based on a set of associated, frequently occurring text patterns. Notice that very frequent terms are likely poor discriminators. Thus only those terms that are not very frequent and that have good discriminative power will be used in document classification. Such an association-based classification method proceeds as follows: First, keywords and terms can be extracted by information retrieval and simple association analysis techniques. Second, concept hierarchies of keywords and terms can be obtained using available term classes, such as WordNet, or relying on expert knowledge, or some keyword classification systems. Documents in the training set can also be classified into class hierarchies. A term association mining method can then be applied to discover sets of associated terms that can be used to maximally distinguish one class of documents from others. This derives a set of association rules associated with each document class. Such classification rules can be ordered based on their discriminative power and occurrence frequency, and used to classify new documents. Such kind of association-based document classifier has been proven effective.

For Web document classification, the Web page linkage information can be used to further assist the identification of document classes. Web linkage analysis methods are discussed in Section 10.5.

Document Clustering Analysis

Document clustering is one of the most crucial techniques for organizing documents in an unsupervised manner. When documents are represented as term vectors, the clustering methods described in Chapter 7 can be applied. However, the document space is always of very high dimensionality, ranging from several hundreds to thousands. Due to the *curse of dimensionality*, it makes sense to first project the documents into a lower-dimensional subspace in which the semantic structure of the document space becomes clear. In the low-dimensional semantic space, the traditional clustering algorithms can then be applied. To this end, spectral clustering, mixture model clustering, clustering using Latent Semantic Indexing, and clustering using Locality Preserving Indexing are the most well-known techniques. We discuss each of these methods here.

The **spectral clustering method** first performs spectral embedding (dimensionality reduction) on the original data, and then applies the traditional clustering algorithm (e.g., k-means) on the reduced document space. Recently, work on spectral clustering shows its capability to handle highly nonlinear data (the data space has high curvature at every local area). Its strong connections to differential geometry make it capable of discovering the manifold structure of the document space. One major drawback of these spectral clustering algorithms might be that they use the nonlinear embedding (dimensionality reduction), which is only defined on "training" data. They have to use all of the data points to learn the embedding. When the data set is very large, it is computationally expensive to learn such an embedding. This restricts the application of spectral clustering on large data sets.

The **mixture model clustering method** models the text data with a mixture model, often involving multinomial component models. Clustering involves two steps: (1) estimating

the model parameters based on the text data and any additional prior knowledge, and (2) inferring the clusters based on the estimated model parameters. Depending on how the mixture model is defined, these methods can cluster words and documents at the same time. Probabilistic Latent Semantic Analysis (PLSA) and Latent Dirichlet Allocation (LDA) are two examples of such techniques. One potential advantage of such clustering methods is that the clusters can be designed to facilitate comparative analysis of documents.

The **Latent Semantic Indexing** (**LSI**) and **Locality Preserving Indexing** (**LPI**) methods introduced in Section 10.4.2 are linear dimensionality reduction methods. We can acquire the transformation vectors (*embedding function*) in LSI and LPI. Such embedding functions are defined everywhere; thus, we can use part of the data to learn the embedding function and embed all of the data to low-dimensional space. With this trick, clustering using LSI and LPI can handle large document data corpus.

As discussed in the previous section, LSI aims to find the best subspace approximation to the original document space in the sense of minimizing the *global* reconstruction error. In other words, LSI seeks to uncover the most representative features rather than the most discriminative features for document representation. Therefore, LSI might not be optimal in discriminating documents with different semantics, which is the ultimate goal of clustering. LPI aims to discover the *local* geometrical structure and can have more discriminating power. Experiments show that for clustering, LPI as a dimensionality reduction method is more suitable than LSI. Compared with LSI and LPI, the PLSI method reveals the latent semantic dimensions in a more interpretable way and can easily be extended to incorporate any prior knowledge or preferences about clustering.

10.5 Mining the World Wide Web

The World Wide Web serves as a huge, widely distributed, global information service center for news, advertisements, consumer information, financial management, education, government, e-commerce, and many other information services. The Web also contains a rich and dynamic collection of hyperlink information and Web page access and usage information, providing rich sources for data mining. However, based on the following observations, the Web also poses great challenges for effective resource and knowledge discovery.

- *The Web seems to be too huge for effective data warehousing and data mining*. The size of the Web is in the order of hundreds of terabytes and is still growing rapidly. Many organizations and societies place most of their public-accessible information on the Web. It is barely possible to set up a data warehouse to replicate, store, or integrate all of the data on the Web.[3]

[3]There have been efforts to store or integrate all of the data on the Web. For example, a huge Internet archive can be accessed at *www.archive.org*.

- *The complexity of Web pages is far greater than that of any traditional text document collection.* Web pages lack a unifying structure. They contain far more authoring style and content variations than any set of books or other traditional text-based documents. The Web is considered a huge digital library; however, the tremendous number of documents in this library are not arranged according to any particular sorted order. There is no index by category, nor by title, author, cover page, table of contents, and so on. It can be very challenging to search for the information you desire in such a library!
- *The Web is a highly dynamic information source.* Not only does the Web grow rapidly, but its information is also constantly updated. News, stock markets, weather, sports, shopping, company advertisements, and numerous other Web pages are updated regularly on the Web. Linkage information and access records are also updated frequently.
- *The Web serves a broad diversity of user communities.* The Internet currently connects more than 100 million workstations, and its user community is still rapidly expanding. Users may have very different backgrounds, interests, and usage purposes. Most users may not have good knowledge of the structure of the information network and may not be aware of the heavy cost of a particular search. They can easily get lost by groping in the "darkness" of the network, or become bored by taking many access "hops" and waiting impatiently for a piece of information.
- *Only a small portion of the information on the Web is truly relevant or useful.* It is said that 99% of the Web information is useless to 99% of Web users. Although this may not seem obvious, it is true that a particular person is generally interested in only a tiny portion of the Web, while the rest of the Web contains information that is uninteresting to the user and may swamp desired search results. How can the portion of the Web that is truly relevant to your interest be determined? How can we find high-quality Web pages on a specified topic?

These challenges have promoted research into efficient and effective discovery and use of resources on the Internet.

There are many index-based **Web search engines**. These search the Web, index Web pages, and build and store huge keyword-based indices that help locate sets of Web pages containing certain keywords. With such search engines, an experienced user may be able to quickly locate documents by providing a set of tightly constrained keywords and phrases. However, a simple keyword-based search engine suffers from several deficiencies. First, a topic of any breadth can easily contain hundreds of thousands of documents. This can lead to a huge number of document entries returned by a search engine, many of which are only marginally relevant to the topic or may contain materials of poor quality. Second, many documents that are highly relevant to a topic may not contain keywords defining them. This is referred to as the *polysemy* problem, discussed in the previous section on text mining. For example, the keyword *Java* may refer to the Java programming language, or an island in Indonesia, or brewed coffee. As another example, a search based on the keyword *search engine* may not find even the most popular Web

search engines like Google, Yahoo!, AltaVista, or America Online if these services do not claim to be search engines on their Web pages. This indicates that a simple keyword-based Web search engine is not sufficient for Web resource discovery.

"If a keyword-based Web search engine is not sufficient for Web resource discovery, how can we even think of doing Web mining?" Compared with keyword-based Web search, **Web mining** is a more challenging task that searches for Web structures, ranks the importance of Web contents, discovers the regularity and dynamics of Web contents, and mines Web access patterns. However, Web mining can be used to substantially enhance the power of a Web search engine since Web mining may identify authoritative Web pages, classify Web documents, and resolve many ambiguities and subtleties raised in keyword-based Web search. In general, Web mining tasks can be classified into three categories: *Web content mining*, *Web structure mining*, and *Web usage mining*. Alternatively, Web structures can be treated as a part of Web contents so that Web mining can instead be simply classified into *Web content mining* and *Web usage mining*.

In the following subsections, we discuss several important issues related to Web mining: *mining the Web page layout structure* (Section 10.5.1), *mining the Web's link structures* (Section 10.5.2), *mining multimedia data on the Web* (Section 10.5.3), *automatic classification of Web documents* (Section 10.5.4), and *Weblog mining* (Section 10.5.5).

10.5.1 Mining the Web Page Layout Structure

Compared with traditional plain text, a Web page has more structure. Web pages are also regarded as semi-structured data. The basic structure of a Web page is its DOM[4] (Document Object Model) structure. The **DOM structure** of a Web page is a tree structure, where every HTML tag in the page corresponds to a node in the DOM tree. The Web page can be segmented by some predefined structural tags. Useful tags include ⟨P⟩ (paragraph), ⟨TABLE⟩ (table), ⟨UL⟩ (list), ⟨H1⟩ ∼ ⟨H6⟩ (heading), etc. Thus the DOM structure can be used to facilitate information extraction.

Unfortunately, due to the flexibility of HTML syntax, many Web pages do not obey the W3C HTML specifications, which may result in errors in the DOM tree structure. Moreover, the DOM tree was initially introduced for presentation in the browser rather than description of the semantic structure of the Web page. For example, even though two nodes in the DOM tree have the same parent, the two nodes might not be more semantically related to each other than to other nodes. Figure 10.7 shows an example page.[5] Figure 10.7(a) shows part of the HTML source (we only keep the backbone code), and Figure 10.7(b) shows the DOM tree of the page. Although we have surrounding description text for each image, the DOM tree structure fails to correctly identify the semantic relationships between different parts.

In the sense of human perception, people always view a Web page as different semantic objects rather than as a single object. Some research efforts show that users

[4] *www.w3c.org/DOM*

[5] *http://yahooligans.yahoo.com/content/ecards/content/ecards/category?c=133&g=16*

```
<tr>
<td><img src="t1.jpg"></td><td><img src="t2.jpg"></td>
<td><img src="t3.jpg"></td><td><img src="t4.jpg"></td>
</tr>
<tr>
<td>Timber Wolf</td><td>Giraffes</td>
<td>Elephant Sunrise</td><td>Prowling Fox</td>
</tr>
```

(a) Part of HTML source (only keep the backbone)

(b) The DOM tree structure (The picture area and caption area are two different TR nodes)

Figure 10.7 The HTML source and DOM tree structure of a sample page. It is difficult to extract the correct semantic content structure of the page.

always expect that certain functional parts of a Web page (e.g., navigational links or an advertisement bar) appear at certain positions on the page. Actually, when a Web page is presented to the user, the spatial and visual cues can help the user unconsciously divide the Web page into several semantic parts. Therefore, it is possible to automatically segment the Web pages by using the spatial and visual cues. Based on this observation, we can develop algorithms to extract the Web page content structure based on spatial and visual information.

Here, we introduce an algorithm called **VIsion-based Page Segmentation** (**VIPS**). VIPS aims to extract the semantic structure of a Web page based on its visual presentation. Such semantic structure is a tree structure: each node in the tree corresponds to a block. Each node will be assigned a value (Degree of Coherence) to indicate how coherent is the content in the block based on visual perception. The VIPS algorithm makes full use of the page layout feature. It first extracts all of the suitable blocks from the HTML DOM tree, and then it finds the separators between these blocks. Here separators denote the horizontal or vertical lines in a Web page that visually cross with no blocks. Based on these separators, the semantic tree of the Web page is constructed. A Web page can be represented as a set of blocks (leaf nodes of the semantic tree). Compared with DOM-based methods, the segments obtained by VIPS are more semantically aggregated. Noisy information, such as navigation, advertisement, and decoration can be easily removed because these elements are often placed in certain positions on a page. Contents with different topics are distinguished as separate blocks. Figure 10.8 illustrates the procedure of VIPS algorithm, and Figure 10.9 shows the partition result of the same page as in Figure 10.7.

10.5.2 Mining the Web's Link Structures to Identify Authoritative Web Pages

"*What is meant by authoritative Web pages?*" Suppose you would like to search for Web pages relating to a given topic, such as financial investing. In addition to retrieving pages that are relevant, you also hope that the pages retrieved will be of high quality, or *authoritative* on the topic.

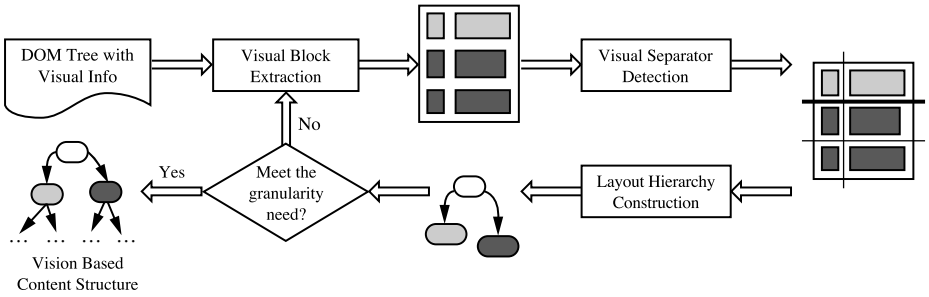

Figure 10.8 The process flow of vision-based page segmentation algorithm.

Figure 10.9 Partition using VIPS (The image with their surrounding text are accurately identified)

"*But how can a search engine automatically identify authoritative Web pages for my topic?*" Interestingly, the secrecy of authority is hiding in Web page linkages. The Web consists not only of pages, but also of *hyperlinks* pointing from one page to another. These hyperlinks contain an enormous amount of latent human annotation that can help automatically infer the notion of authority. When an author of a Web page creates a hyperlink pointing to another Web page, this can be considered as the author's endorsement of the other page. The collective endorsement of a given page by different authors on the Web may indicate the importance of the page and may naturally lead to the discovery of authoritative Web pages. Therefore, the tremendous amount of Web linkage information provides rich information about the relevance, the quality, and the structure of the Web's contents, and thus is a rich source for Web mining.

This idea has motivated some interesting studies on mining authoritative pages on the Web. In the 1970s, researchers in information retrieval proposed methods of using citations among journal articles to evaluate the quality of research papers. However, unlike journal citations, the Web linkage structure has some unique features. First, not every hyperlink represents the endorsement we seek. Some links are created for other purposes, such as for navigation or for paid advertisements. Yet overall, if the majority of

hyperlinks are for endorsement, then the collective opinion will still dominate. Second, for commercial or competitive interests, one authority will seldom have its Web page point to its rival authorities in the same field. For example, *Coca-Cola* may prefer not to endorse its competitor *Pepsi* by not linking to *Pepsi*'s Web pages. Third, authoritative pages are seldom particularly descriptive. For example, the main Web page of Yahoo! may not contain the explicit self-description "*Web search engine.*"

These properties of Web link structures have led researchers to consider another important category of Web pages called a *hub*. A **hub** is one or a set of Web pages that provides collections of links to authorities. Hub pages may not be prominent, or there may exist few links pointing to them; however, they provide links to a collection of prominent sites on a common topic. Such pages could be lists of recommended links on individual home pages, such as recommended reference sites from a course home page, or professionally assembled resource lists on commercial sites. Hub pages play the role of implicitly conferring authorities on a focused topic. In general, a good hub is a page that points to many good authorities; a good authority is a page pointed to by many good hubs. Such a mutual reinforcement relationship between hubs and authorities helps the mining of authoritative Web pages and automated discovery of high-quality Web structures and resources.

"*So, how can we use hub pages to find authoritative pages?*" An algorithm using hubs, called **HITS** (Hyperlink-Induced Topic Search), was developed as follows. First, HITS uses the query terms to collect a starting set of, say, 200 pages from an index-based search engine. These pages form the **root set**. Since many of these pages are presumably relevant to the search topic, some of them should contain links to most of the prominent authorities. Therefore, the root set can be expanded into a **base set** by including all of the pages that the root-set pages link to and all of the pages that link to a page in the root set, up to a designated size cutoff such as 1,000 to 5,000 pages (to be included in the base set).

Second, a weight-propagation phase is initiated. This iterative process determines numerical estimates of hub and authority weights. Notice that links between two pages with the same Web domain (i.e., sharing the same first level in their URLs) often serve as a navigation function and thus do not confer authority. Such links are excluded from the weight-propagation analysis.

We first associate a non-negative **authority weight**, a_p, and a non-negative **hub weight**, h_p, with each page p in the base set, and initialize all a and h values to a uniform constant. The weights are normalized and an invariant is maintained that the squares of all weights sum to 1. The authority and hub weights are updated based on the following equations:

$$a_p = \Sigma_{(q \text{ such that } q \rightarrow p)} \, h_q \tag{10.17}$$

$$h_p = \Sigma_{(q \text{ such that } q \leftarrow p)} \, a_q \tag{10.18}$$

Equation (10.17) implies that if a page is pointed to by many good hubs, its authority weight should increase (i.e., it is the sum of the current hub weights of all of the pages pointing to it). Equation (10.18) implies that if a page is pointing to many good authorities, its hub weight should increase (i.e., it is the sum of the current authority weights of all of the pages it points to).

These equations can be written in matrix form as follows. Let us number the pages $\{1, 2, \ldots, n\}$ and define their **adjacency matrix** A to be an $n \times n$ matrix where $A(i, j)$ is 1 if page i links to page j, or 0 otherwise. Similarly, we define the **authority weight vector** $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$, and the **hub weight vector** $\boldsymbol{h} = (h_1, h_2, \ldots, h_n)$. Thus, we have

$$\boldsymbol{h} = A \cdot \boldsymbol{a} \tag{10.19}$$

$$\boldsymbol{a} = A^T \cdot \boldsymbol{h}, \tag{10.20}$$

where A^T is the transposition of matrix A. Unfolding these two equations k times, we have

$$\boldsymbol{h} = A \cdot \boldsymbol{a} = AA^T\boldsymbol{h} = (AA^T)\boldsymbol{h} = (AA^T)^2\boldsymbol{h} = \cdots = (AA^T)^k\boldsymbol{h} \tag{10.21}$$

$$\boldsymbol{a} = A^T \cdot \boldsymbol{h} = A^TA\boldsymbol{a} = (A^TA)\boldsymbol{a} = (A^TA)^2\boldsymbol{a} = \cdots = (A^TA)^k\boldsymbol{a}. \tag{10.22}$$

According to linear algebra, these two sequences of iterations, when normalized, converge to the principal eigenvectors of AA^T and A^TA, respectively. This also proves that the authority and hub weights are intrinsic features of the linked pages collected and are not influenced by the initial weight settings.

Finally, the HITS algorithm outputs a short list of the pages with large hub weights, and the pages with large authority weights for the given search topic. Many experiments have shown that HITS provides surprisingly good search results for a wide range of queries.

Although relying extensively on links can lead to encouraging results, the method may encounter some difficulties by ignoring textual contexts. For example, HITS sometimes drifts when hubs contain multiple topics. It may also cause "topic hijacking" when many pages from a single website point to the same single popular site, giving the site too large a share of the authority weight. Such problems can be overcome by replacing the sums of Equations (10.17) and (10.18) with weighted sums, scaling down the weights of multiple links from within the same site, using *anchor text* (the text surrounding hyperlink definitions in Web pages) to adjust the weight of the links along which authority is propagated, and breaking large hub pages into smaller units.

Google's PageRank algorithm is based on a similar principle. By analyzing Web links and textual context information, it has been reported that such systems can achieve better-quality search results than those generated by term-index engines like AltaVista and those created by human ontologists such as at Yahoo!.

The above link analysis algorithms are based on the following two assumptions. First, links convey human endorsement. That is, if there exists a link from page A to page B and these two pages are authored by different people, then the link implies that the author of page A found page B valuable. Thus the importance of a page can be propagated to those pages it links to. Second, pages that are co-cited by a certain page are likely related to the same topic. However, these two assumptions may not hold in many cases. A typical example is the Web page at *http://news.yahoo.com* (Figure 10.10), which contains multiple semantics (marked with rectangles with different colors) and many links only for navigation and advertisement (the left region). In this case, the importance of each page may be miscalculated by PageRank, and *topic drift* may occur in HITS when the popular

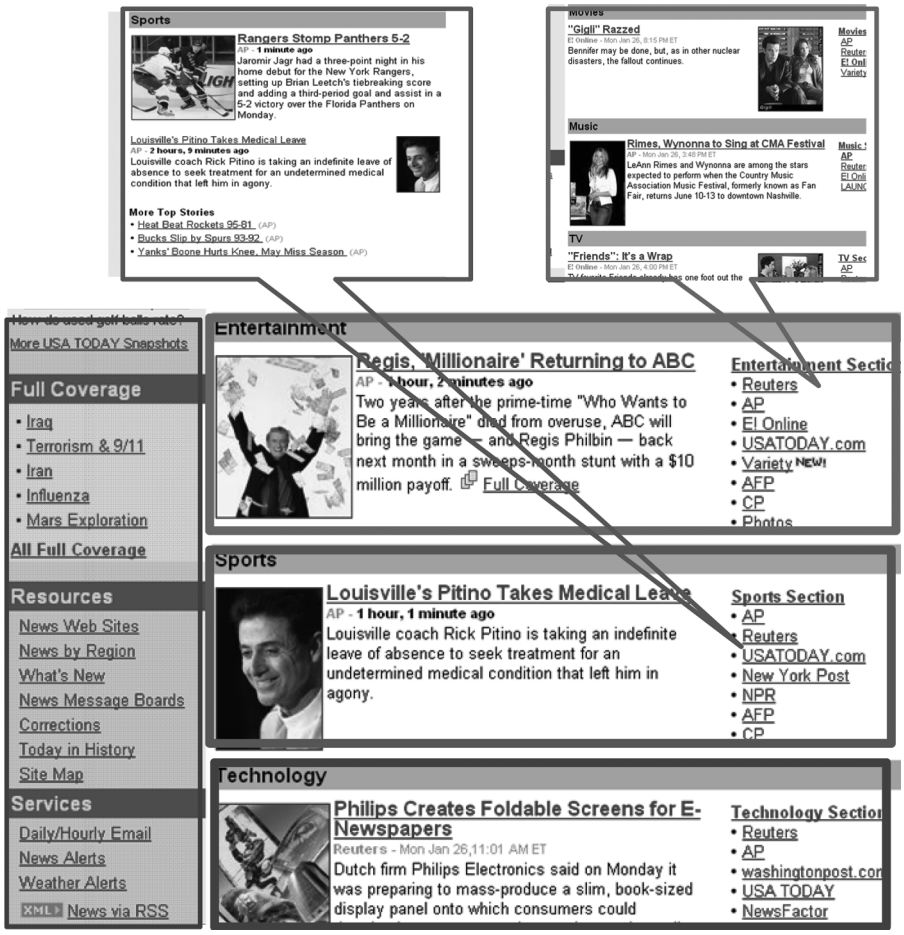

Figure 10.10 Part of a sample Web page (*news.yahoo.com*). Clearly, this page is made up of different semantic blocks (with different color rectangles). Different blocks have different importances in the page. The links in different blocks point to the pages with different topics.

sites such as Web search engines are so close to any topic, and thus are ranked at the top regardless of the topics.

These two problems are caused by the fact that a single Web page often contains multiple semantics, and the different parts of the Web page have different importance in that page. Thus, from the perspective of semantics, a Web page should not be the smallest unit. The hyperlinks contained in different semantic blocks usually point to the pages of different topics. Naturally, it is more reasonable to regard the semantic blocks as the smallest units of information.

By using the VIPS algorithm introduced in Section 10.5.1, we can extract *page-to-block* and *block-to-page* relationships and then construct a page graph and a block graph. Based on this graph model, the new link analysis algorithms are capable of discovering the intrinsic semantic structure of the Web. The above two assumptions become reasonable in block-level link analysis algorithms. Thus, the new algorithms can improve the performance of search in Web context.

The graph model in **block-level link analysis** is induced from two kinds of relationships, that is, *block-to-page* (link structure) and *page-to-block* (page layout).

The **block-to-page relationship** is obtained from link analysis. Because a Web page generally contains several semantic blocks, different blocks are related to different topics. Therefore, it might be more reasonable to consider the hyperlinks from block to page, rather than from page to page. Let Z denote the block-to-page matrix with dimension $n \times k$. Z can be formally defined as follows:

$$Z_{ij} = \begin{cases} 1/s_i, & \text{if there is a link from block } i \text{ to page } j \\ 0, & \text{otherwise,} \end{cases} \tag{10.23}$$

where s_i is the number of pages to which block i links. Z_{ij} can also be viewed as a probability of jumping from block i to page j. The block-to-page relationship gives a more accurate and robust representation of the link structures of the Web.

The **page-to-block relationships** are obtained from page layout analysis. Let X denote the page-to-block matrix with dimension $k \times n$. As we have described, each Web page can be segmented into blocks. Thus, X can be naturally defined as follows:

$$X_{ij} = \begin{cases} f_{p_i}(b_j), & \text{if } b_j \in p_i \\ 0, & \text{otherwise,} \end{cases} \tag{10.24}$$

where f is a function that assigns to every block b in page p an importance value. Specifically, the bigger $f_p(b)$ is, the more important the block b is. Function f is empirically defined below,

$$f_p(b) = \alpha \times \frac{\text{the size of block } b}{\text{the distance between the center of } b \text{ and the center of the screen}}, \tag{10.25}$$

where α is a normalization factor to make the sum of $f_p(b)$ to be 1, that is,

$$\sum_{b \in p} f_p(b) = 1$$

Note that $f_p(b)$ can also be viewed as a probability that the user is focused on the block b when viewing the page p. Some more sophisticated definitions of f can be formulated by considering the background color, fonts, and so on. Also, f can be learned from some prelabeled data (the importance value of the blocks can be defined by people) as a regression problem by using learning algorithms, such as support vector machines and neural networks.

Based on the *block-to-page* and *page-to-block* relations, a new Web page graph that incorporates the block importance information can be defined as

$$W_{\mathcal{P}} = XZ, \tag{10.26}$$

where X is a $k \times n$ page-to-block matrix, and Z is a $n \times k$ block-to-page matrix. Thus $W_{\mathcal{P}}$ is a $k \times k$ page-to-page matrix.

The block-level PageRank can be calculated on the new Web page graph. Experiments have shown the powerfulness of block-level link analysis.

10.5.3 Mining Multimedia Data on the Web

A huge amount of multimedia data are available on the Web in different forms. These include video, audio, images, pictures, and graphs. There is an increasing demand for effective methods for organizing and retrieving such multimedia data.

Compared with the general-purpose multimedia data mining, the multimedia data on the Web bear many different properties. Web-based multimedia data are embedded on the Web page and are associated with text and link information. These texts and links can also be regarded as features of the multimedia data. Using some Web page layout mining techniques (like VIPS), a Web page can be partitioned into a set of *semantic blocks*. Thus, the block that contains multimedia data can be regarded as a whole. Searching and organizing the Web multimedia data can be referred to as searching and organizing the multimedia blocks.

Let's consider Web images as an example. Figures 10.7 and 10.9 already show that VIPS can help identify the surrounding text for Web images. Such surrounding text provides a textual description of Web images and can be used to build an image index. The Web image search problem can then be partially completed using traditional text search techniques. Many commercial Web image search engines, such as Google and Yahoo!, use such approaches.

The block-level link analysis technique described in Section 10.5.2 can be used to organize Web images. In particular, the image graph deduced from block-level link analysis can be used to achieve high-quality Web image clustering results.

To construct a Web-image graph, in addition to the *block-to-page* and *page-to-block* relations, we need to consider a new relation: **block-to-image** relation. Let Y denote the block-to-image matrix with dimension $n \times m$. For each image, at least one block contains this image. Thus, Y can be simply defined below:

$$Y_{ij} = \begin{cases} 1/s_i, & \text{if } I_j \in b_i \\ 0, & \text{otherwise,} \end{cases} \tag{10.27}$$

where s_i is the number of images contained in the image block b_i.

Now we first construct the block graph from which the image graph can be further induced. In block-level link analysis, the block graph is defined as:

$$W_{\mathcal{B}} = (1-t)ZX + tD^{-1}U, \tag{10.28}$$

where t is a suitable constant. D is a diagonal matrix, $D_{ii} = \sum_j U_{ij}$. U_{ij} is 0 if block i and block j are contained in two different Web pages; otherwise, it is set to the *DOC* (*degree of coherence*, a property of the block, which is the result of the VIPS algorithm) value of the smallest block containing both block i and block j. It is easy to check that the sum of each row of $D^{-1}U$ is 1. Thus, $W_{\mathcal{B}}$ can be viewed as a probability transition matrix such that $W_{\mathcal{B}}(a,b)$ is the probability of jumping from block a to block b.

Once the block graph is obtained, the image graph can be constructed correspondingly by noticing the fact that every image is contained in at least one block. In this way, the weight matrix of the image graph can be naturally defined as follows:

$$W_I = Y^T W_{\mathcal{B}} Y, \tag{10.29}$$

where W_I is an $m \times m$ matrix. If two images i and j are in the same block, say b, then $W_I(i,j) = W_{\mathcal{B}}(b,b) = 0$. However, the images in the same block are supposed to be semantically related. Thus, we get a new definition as follows:

$$W_I = tD^{-1}Y^T Y + (1-t)Y^T W_{\mathcal{B}} Y, \tag{10.30}$$

where t is a suitable constant, and D is a diagonal matrix, $D_{ii} = \sum_j (Y^T Y)_{ij}$.

Such an image graph can better reflect the semantic relationships between the images. With this image graph, clustering and embedding can be naturally acquired. Figure 10.11(a) shows the embedding results of 1,710 images from the Yahooligans website.[6] Each data point represents an image. Each color stands for a semantic class. Clearly, the image data set was accurately clustered into six categories. Some example images of these six categories (i.e., mammal, fish, reptile, bird, amphibian, and insect) are shown in Figure 10.12.

If we use traditional link analysis methods that consider hyperlinks from page to page, the 2-D embedding result is shown in Figure 10.11(b). As can be seen, the six categories were mixed together and can hardly be separated. This comparison shows that the image graph model deduced from block-level link analysis is more powerful than traditional methods as to describing the intrinsic semantic relationships between WWW images.

10.5.4 Automatic Classification of Web Documents

In the automatic classification of Web documents, each document is assigned a class label from a set of predefined topic categories, based on a set of examples of preclassified documents. For example, Yahoo!'s taxonomy and its associated documents can be used as training and test sets in order to derive a Web document classification scheme. This scheme may then be used to classify new Web documents by assigning categories from the same taxonomy.

Keyword-based document classification methods were discussed in Section 10.4.3, as well as keyword-based association analysis. These methods can be used for Web document classification. Such a term-based classification scheme has shown good results

[6] *www.yahooligans.com/content/animals*

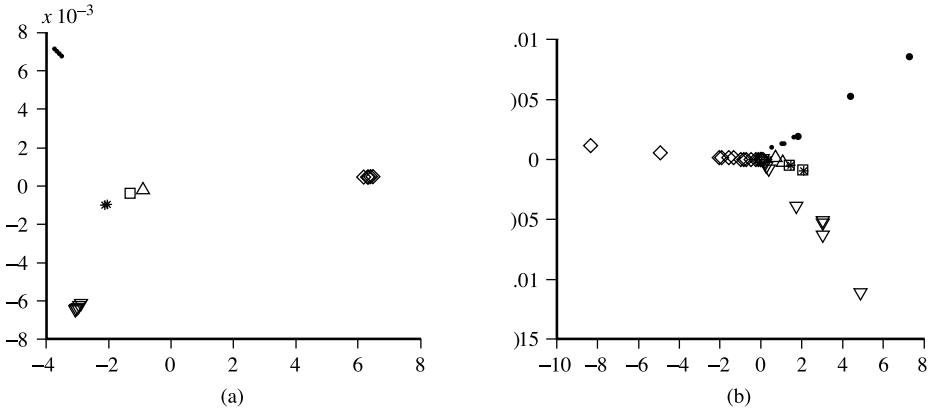

Figure 10.11 2-D embedding of the WWW images. (a) The image graph is constructed using block-level link analysis. Each color (shape) represents a semantic category. Clearly, they are well separated. (b) The image graph was constructed based on traditional perspective that the hyperlinks are considered from pages to pages. The image graph was induced from the page-to-page and page-to-image relationships.

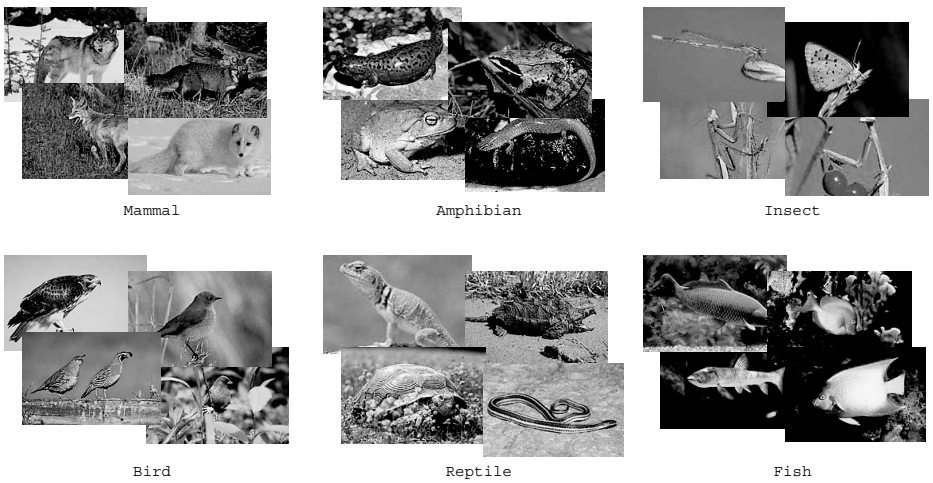

Figure 10.12 Six image categories.

in Web page classification. However, because a Web page may contain multiple themes, advertisement, and navigation information, *block-based page content analysis* may play an important role in construction of high-quality classification models. Moreover, because hyperlinks contain high-quality semantic clues to a page's topic, it is beneficial to make

good use of such semantic information in order to achieve even better accuracy than pure keyword-based classification. Note that because the hyperlinks surrounding a document may be quite noisy, naïve use of terms in a document's hyperlink neighborhood can even *degrade* accuracy. The use of block-based Web linkage analysis as introduced in the previous subsections will reduce such noise and enhance the quality of Web document classification.

There have been extensive research activities on the construction and use of **the semantic Web**, a Web information infrastructure that is expected to bring structure to the Web based on the semantic meaning of the contents of Web pages. Web document classification by Web mining will help in the automatic extraction of the semantic meaning of Web pages and build up ontology for the semantic Web. Conversely, the semantic Web, if successfully constructed, will greatly help automated Web document classification as well.

10.5.5 Web Usage Mining

"*What is Web usage mining?*" Besides mining Web contents and Web linkage structures, another important task for Web mining is **Web usage mining**, which mines Weblog records to discover user access patterns of Web pages. Analyzing and exploring regularities in Weblog records can identify potential customers for electronic commerce, enhance the quality and delivery of Internet information services to the end user, and improve Web server system performance.

A Web server usually registers a (Web) log entry, or **Weblog entry**, for every access of a Web page. It includes the URL requested, the IP address from which the request originated, and a timestamp. For Web-based e-commerce servers, a huge number of Web access log records are being collected. Popular websites may register Weblog records in the order of hundreds of megabytes every day. Weblog databases provide rich information about Web dynamics. Thus it is important to develop sophisticated Weblog mining techniques.

In developing techniques for Web usage mining, we may consider the following. First, although it is encouraging and exciting to imagine the various potential applications of Weblog file analysis, it is important to know that the success of such applications depends on what and how much valid and reliable knowledge can be discovered from the large raw log data. Often, raw Weblog data need to be *cleaned, condensed, and transformed* in order to retrieve and analyze significant and useful information. In principle, these preprocessing methods are similar to those discussed in Chapter 2, although Weblog customized preprocessing is often needed.

Second, with the available URL, time, IP address, and Web page content information, a multidimensional view can be constructed on the Weblog database, and *multidimensional OLAP analysis* can be performed to find the top N users, top N accessed Web pages, most frequently accessed time periods, and so on, which will help discover potential customers, users, markets, and others.

Third, *data mining* can be performed on Weblog records to find association patterns, sequential patterns, and trends of Web accessing. For Web access pattern mining, it is

often necessary to take further measures to obtain additional information of user traversal to facilitate detailed Weblog analysis. Such additional information may include user-browsing sequences of the Web pages in the Web server buffer.

With the use of such Weblog files, studies have been conducted on analyzing system performance, improving system design by Web caching, Web page prefetching, and Web page swapping; understanding the nature of Web traffic; and understanding user reaction and motivation. For example, some studies have proposed *adaptive sites*: websites that improve themselves by learning from user access patterns. Weblog analysis may also help build customized Web services for individual users.

Because Weblog data provide information about what kind of users will access what kind of Web pages, Weblog information can be integrated with Web content and Web linkage structure mining to help Web page ranking, Web document classification, and the construction of a multilayered Web information base as well. A particularly interesting application of Web usage mining is to mine a user's interaction history and search context on the client side to extract useful information for improving the ranking accuracy for the given user. For example, if a user submits a keyword query "Java" to a search engine, and then selects "Java programming language" from the returned entries for viewing, the system can infer that the displayed snippet for this Web page is interesting to the user. It can then raise the rank of pages similar to "Java programming language" and avoid presenting distracting pages about "Java Island." Hence the quality of search is improved, because search is contextualized and personalized.

10.6 Summary

- Vast amounts of data are stored in various complex forms, such as structured or unstructured, hypertext, and multimedia. Thus, **mining complex types of data**, including *object data, spatial data, multimedia data, text data*, and *Web data*, has become an increasingly important task in data mining.
- Multidimensional analysis and data mining can be performed in **object-relational and object-oriented databases**, by (1) *class-based generalization of complex objects*, including set-valued, list-valued, and other sophisticated types of data, class/subclass hierarchies, and class composition hierarchies; (2) constructing *object data cubes*; and (3) performing *generalization-based mining*. A **plan database** can be mined by a generalization-based, divide-and-conquer approach in order to find interesting general patterns at different levels of abstraction.
- **Spatial data mining** is the discovery of interesting patterns from large geospatial databases. *Spatial data cubes* that contain spatial dimensions and measures can be constructed. *Spatial OLAP* can be implemented to facilitate *multidimensional spatial data analysis*. Spatial data mining includes *mining spatial association and co-location patterns*, *clustering*, *classification*, and *spatial trend and outlier analysis*.
- **Multimedia data mining** is the discovery of interesting patterns from multimedia databases that store and manage large collections of multimedia objects, including

audio data, image data, video data, sequence data, and hypertext data containing text, text markups, and linkages. Issues in multimedia data mining include *content-based retrieval and similarity search*, and *generalization and multidimensional analysis*. Multimedia data cubes contain additional dimensions and measures for multimedia information. Other topics in multimedia mining include *classification and prediction analysis*, *mining associations*, and *audio and video data mining*.

- A substantial portion of the available information is stored in text or document databases that consist of large collections of documents, such as news articles, technical papers, books, digital libraries, e-mail messages, and Web pages. Text information retrieval and data mining has thus become increasingly important. *Precision, recall*, and *the F-score* are three based measures from Information Retrieval (IR). Various **text retrieval methods** have been developed. These typically either focus on *document selection* (where the query is regarded as providing constraints) or *document ranking* (where the query is used to rank documents in order of relevance). The *vector-space model* is a popular example of the latter kind. Latex Sementic Indexing (LSI), Locality Preserving Indexing (LPI), and Probabilistic LSI can be used for **text dimensionality reduction**. **Text mining** goes one step beyond keyword-based and similarity-based information retrieval and discovers knowledge from semistructured text data using methods such as *keyword-based association analysis*, *document classification*, and *document clustering*.
- The World Wide Web serves as a huge, widely distributed, global information service center for news, advertisements, consumer information, financial management, education, government, e-commerce, and many other services. It also contains a rich and dynamic collection of hyperlink information, and access and usage information, providing rich sources for data mining. **Web mining** includes mining *Web linkage structures, Web contents*, and *Web access patterns*. This involves mining the *Web page layout structure*, mining the *Web's link structures* to identify *authoritative Web pages*, mining *multimedia data* on the Web, *automatic classification of Web documents*, and *Web usage mining*.

Exercises

10.1 An *object cube* can be constructed by generalization of an object-oriented or object-relational database into relatively structured data before performing multidimensional analysis. Because a set of complex data objects or properties can be generalized in multiple directions and thus derive multiple generalized features, such generalization may lead to a high-dimensional, but rather sparse (generalized) "object cube." Discuss how to perform effective online analytical processing in such an object cube.

10.2 A *heterogeneous database system* consists of multiple database systems that are defined independently, but that need to exchange and transform information among themselves and answer local and global queries. Discuss how to process a descriptive mining query in such a system using a generalization-based approach.

10.3 A *plan database* consists of a set of action sequences, such as legs of connecting flights, which can be generalized to find generalized sequence plans. Similarly, a *structure database* may consists of a set of structures, such as trees or graphs, which may also be generalized to find generalized structures. Outline a scalable method that may effectively perform such generalized structure mining.

10.4 Suppose that a city transportation department would like to perform *data analysis on highway traffic* for the planning of highway construction based on the city traffic data collected at different hours every day.

(a) Design a spatial data warehouse that stores the highway traffic information so that people can easily see the average and peak time traffic flow by highway, by time of day, and by weekdays, and the traffic situation when a major accident occurs.

(b) What information can we mine from such a spatial data warehouse to help city planners?

(c) This data warehouse contains both spatial and temporal data. Propose one mining technique that can efficiently mine interesting patterns from such a spatiotemporal data warehouse.

10.5 *Spatial association mining* can be implemented in at least two ways: (1) dynamic computation of spatial association relationships among different spatial objects, based on the mining query, and (2) precomputation of spatial distances between spatial objects, where the association mining is based on such precomputed results. Discuss (1) how to implement each approach efficiently and (2) which approach is preferable under what situation.

10.6 Traffic situations are often auto-correlated: the congestion at one highway intersection may trigger the congestion in nearby highway segments after a short period of time. Suppose we are given highway traffic history data in Chicago, including road construction segment, traffic speed associated with highway segment, direction, time, and so on. Moreover, we are given weather conditions by weather bureau in Chicago. Design a data mining method to find high-quality *spatiotemporal association rules* that may guide us to predict what could be the expected traffic situation at a given highway location.

10.7 *Similarity search in multimedia* has been a major theme in developing multimedia data retrieval systems. However, many *multimedia data mining* methods are based on the analysis of isolated simple multimedia features, such as color, shape, description, keywords, and so on.

(a) Can you show that an integration of similarity-based search with data mining may bring important progress in multimedia data mining? You may take any one mining task as an example, such as multidimensional analysis, classification, association, or clustering.

(b) Outline an implementation technique that applies a similarity-based search method to enhance the quality of clustering in multimedia data.

10.8 It is challenging but important to *discover unusual events from video data* in real time or in a very short time frame. An example is the detection of an explosion near a bus stop or a car collision at a highway junction. Outline a *video data mining* method that can be used for this purpose.

10.9 *Precision* and *recall* are two essential quality measures of an information retrieval system.

(a) Explain why it is the usual practice to trade one measure for the other. Explain why the F-score is a good measure for this purpose.

(b) Illustrate the methods that may effectively improve the F-score in an information retrieval system.

10.10 *TF-IDF* has been used as an effective measure in document classification.

(a) Give one example to show that TF-IDF may not be always a good measure in document classification.

(b) Define another measure that may overcome this difficulty.

10.11 An *e-mail database* is a database that stores a large number of electronic mail (e-mail) messages. It can be viewed as a semistructured database consisting mainly of text data. Discuss the following.

(a) How can such an e-mail database be structured so as to facilitate multidimensional search, such as by sender, by receiver, by subject, and by time?

(b) What can be mined from such an e-mail database?

(c) Suppose you have roughly classified a set of your previous e-mail messages as *junk, unimportant, normal,* or *important.* Describe how a data mining system may take this as the training set to automatically classify new e-mail messages or unclassified ones.

10.12 Junk e-mail is one of the most annoying things in Web-based business or personal communications. Design an effective scheme (which may consist of a set of methods) that can be used to *filter out junk e-mails* effectively and discuss how such methods should be evolved along with time.

10.13 It is difficult to construct a global data warehouse for the World Wide Web due to its dynamic nature and the huge amounts of data stored in it. However, it is still interesting and useful to construct data warehouses for summarized, localized, multidimensional information on the Internet. Suppose that an Internet information service company would like to set up an Internet-based data warehouse to help tourists choose local hotels and restaurants.

(a) Can you design a Web-based tourist data warehouse that would facilitate such a service?

(b) Suppose each hotel and/or restaurant contains a Web page of its own. Discuss how to locate such Web pages and what methods should be used to extract information from these Web pages in order to populate your Web-based tourist data warehouse.

(c) Discuss how to implement a mining method that may provide additional associated information, such as *"90% of customers who stay at the Downtown Hilton dine at the Emperor Garden Restaurant at least twice,"* each time a search returns a new Web page.

10.14 Each scientific or engineering discipline has its own subject index classification standard that is often used for *classifying documents* in its discipline.

(a) Design a Web document classification method that can take such a subject index to classify a set of Web documents automatically.
(b) Discuss how to use Web linkage information to improve the quality of such classification.
(c) Discuss how to use Web usage information to improve the quality of such classification.

10.15 It is interesting to *cluster* a large set of Web pages based on their similarity.

(a) Discuss what should be the similarity measure in such cluster analysis.
(b) Discuss how the block-level analysis may influence the clustering results and how to develop an efficient algorithms based on this philosophy.
(c) Since different users may like to cluster a set of Web pages differently, discuss how a user may interact with a system to influence the final clustering results, and how such a mechanism can be developed systematically.

10.16 Weblog records provide rich *Web usage information* for data mining.

(a) Mining Weblog access sequences may help prefetch certain Web pages into a Web server buffer, such as those pages that are likely to be requested in the next several clicks. Design an efficient implementation method that may help mining such access sequences.
(b) Mining Weblog access records can help cluster users into separate groups to facilitate customized marketing. Discuss how to develop an efficient implementation method that may help user clustering.

Bibliographic Notes

Mining complex types of data has been a fast-developing, popular research field, with many research papers and tutorials appearing in conferences and journals on data mining and database systems. This chapter covers a few important themes, including multidimensional analysis and mining of complex data objects, spatial data mining, multimedia data mining, text mining, and Web mining.

Zaniolo, Ceri, Faloutsos, et al. [ZCF+97] present a systematic introduction of advanced database systems for handling complex types of data. For multidimensional analysis and mining of complex data objects, Han, Nishio, Kawano, and Wang [HNKW98]

proposed a method for the design and construction of object cubes by multidimensional generalization and its use for mining complex types of data in object-oriented and object-relational databases. A method for the construction of multiple-layered databases by generalization-based data mining techniques for handling semantic heterogeneity was proposed by Han, Ng, Fu, and Dao [HNFD98]. Zaki, Lesh, and Ogihara worked out a system called PlanMine, which applies sequence mining for plan failures [ZLO98]. A generalization-based method for mining plan databases by divide-and-conquer was proposed by Han, Yang, and Kim [HYK99].

Geospatial database systems and spatial data mining have been studied extensively. Some introductory materials about spatial database can be found in Maguire, Goodchild, and Rhind [MGR92], Güting [Gue94], Egenhofer [Ege89], Shekhar, Chawla, Ravada, et al. [SCR+99], Rigaux, Scholl, and Voisard [RSV01], and Shekhar and Chawla [SC03]. For geospatial data mining, a comprehensive survey on spatial data mining methods can be found in Ester, Kriegel, and Sander [EKS97] and Shekhar and Chawla [SC03]. A collection of research contributions on geographic data mining and knowledge discovery are in Miller and Han [MH01b]. Lu, Han, and Ooi [LHO93] proposed a generalization-based spatial data mining method by attribute-oriented induction. Ng and Han [NH94] proposed performing descriptive spatial data analysis based on clustering results instead of on predefined concept hierarchies. Zhou, Truffet, and Han proposed efficient polygon amalgamation methods for on-line multidimensional spatial analysis and spatial data mining [ZTH99]. Stefanovic, Han, and Koperski [SHK00] studied the problems associated with the design and construction of spatial data cubes. Koperski and Han [KH95] proposed a progressive refinement method for mining spatial association rules. Knorr and Ng [KN96] presented a method for mining aggregate proximity relationships and commonalities in spatial databases. Spatial classification and trend analysis methods have been developed by Ester, Kriegel, Sander, and Xu [EKSX97] and Ester, Frommelt, Kriegel, and Sander [EFKS98]. A two-step method for classification of spatial data was proposed by Koperski, Han, and Stefanovic [KHS98].

Spatial clustering is a highly active area of recent research into geospatial data mining. For a detailed list of references on spatial clustering methods, please see the bibliographic notes of Chapter 7. A spatial data mining system prototype, GeoMiner, was developed by Han, Koperski, and Stefanovic [HKS97]. Methods for mining spatiotemporal patterns have been studied by Tsoukatos and Gunopulos [TG01], Hadjieleftheriou, Kollios, Gunopulos, and Tsotras [HKGT03], and Mamoulis, Cao, Kollios, Hadjieleftheriou, et al. [MCK+04]. Mining spatiotemporal information related to moving objects have been studied by Vlachos, Gunopulos, and Kollios [VGK02] and Tao, Faloutsos, Papadias, and Liu [TFPL04]. A bibliography of temporal, spatial, and spatiotemporal data mining research was compiled by Roddick, Hornsby, and Spiliopoulou [RHS01].

Multimedia data mining has deep roots in image processing and pattern recognition, which has been studied extensively in computer science, with many textbooks published, such as Gonzalez and Woods [GW02], Russ [Rus02], and Duda, Hart, and Stork [DHS01]. The theory and practice of multimedia database systems have been introduced in many textbooks and surveys, including Subramanian [Sub98], Yu and Meng [YM97], Perner [Per02], and Mitra and Acharya [MA03]. The IBM QBIC

(Query by Image and Video Content) system was introduced by Flickner, Sawhney, Niblack, Ashley, et al. [FSN+95]. Faloutsos and Lin [FL95] developed a fast algorithm, FastMap, for indexing, data mining, and visualization of traditional and multimedia datasets. Natsev, Rastogi, and Shim [NRS99] developed WALRUS, a similarity retrieval algorithm for image databases that explores wavelet-based signatures with region-based granularity. Fayyad and Smyth [FS93] developed a classification method to analyze high-resolution radar images for identification of volcanoes on Venus. Fayyad, Djorgovski, and Weir [FDW96] applied decision tree methods to the classification of galaxies, stars, and other stellar objects in the Palomar Observatory Sky Survey (POSS-II) project. Stolorz and Dean [SD96] developed Quakefinder, a data mining system for detecting earthquakes from remote sensing imagery. Zaïane, Han, and Zhu [ZHZ00] proposed a progressive deepening method for mining object and feature associations in large multimedia databases. A multimedia data mining system prototype, MultiMediaMiner, was developed by Zaïane, Han, Li, et al. [ZHL+98] as an extension of the DBMiner system proposed by Han, Fu, Wang, et al. [HFW+96]. An overview of image mining methods is performed by Hsu, Lee, and Zhang [HLZ02].

Text data analysis has been studied extensively in information retrieval, with many good textbooks and survey articles, such as Salton and McGill [SM83], Faloutsos [Fal85], Salton [Sal89], van Rijsbergen [vR90], Yu and Meng [YM97], Raghavan [Rag97], Subramanian [Sub98], Baeza-Yates and Riberio-Neto [BYRN99], Kleinberg and Tomkins [KT99], Berry [Ber03], and Weiss, Indurkhya, Zhang, and Damerau [WIZD04]. The technical linkage between information filtering and information retrieval was addressed by Belkin and Croft [BC92]. The *latent semantic indexing* method for document similarity analysis was developed by Deerwester, Dumais, Furnas, et al. [DDF+90]. The *probabilistic latent semantic analysis* method was introduced to information retrieval by Hofmann [Hof98]. The *locality preserving indexing* method for document representation was developed by He, Cai, Liu, and Ma [HCLM04]. The use of signature files is described in Tsichritzis and Christodoulakis [TC83]. Feldman and Hirsh [FH98] studied methods for mining association rules in text databases. Method for automated document classification has been studied by many researchers, such as Wang, Zhou, and Liew [WZL99], Nigam, McCallum, Thrun, and Mitchell [NMTM00] and Joachims [Joa01]. An overview of text classification is given by Sebastiani [Seb02]. Document clustering by *Probabilistic Latent Semantic Analysis (PLSA)* was introduced by Hofmann [Hof98] and that using the *Latent Dirichlet Allocation (LDA)* method was proposed by Blei, Ng, and Jordan [BNJ03]. Zhai, Velivelli, and Yu [ZVY04] studied using such clustering methods to facilitate comparative analysis of documents. A comprehensive study of using dimensionality reduction methods for document clustering can be found in Cai, He, and Han [CHH05].

Web mining started in recent years together with the development of Web search engines and Web information service systems. There has been a great deal of work on Web data modeling and Web query systems, such as W3QS by Konopnicki and Shmueli [KS95], WebSQL by Mendelzon, Mihaila, and Milo [MMM97], Lorel by Abitboul, Quass, McHugh, et al. [AQM+97], Weblog by Lakshmanan, Sadri, and Subramanian [LSS96], WebOQL by Arocena and Mendelzon [AM98], and NiagraCQ by

Chen, DeWitt, Tian, and Wang [CDTW00]. Florescu, Levy, and Mendelzon [FLM98] presented a comprehensive overview of research on Web databases. An introduction to the the semantic Web was presented by Berners-Lee, Hendler, and Lassila [BLHL01].

Chakrabarti [Cha02] presented a comprehensive coverage of data mining for hypertext and the Web. Mining the Web's link structures to recognize authoritative Web pages was introduced by Chakrabarti, Dom, Kumar, et al. [CDK+99] and Kleinberg and Tomkins [KT99]. The HITS algorithm was developed by Kleinberg [Kle99]. The PageRank algorithm was developed by Brin and Page [BP98b]. Embley, Jiang, and Ng [EJN99] developed some heuristic rules based on the DOM structure to discover record boundaries within a page, which assist data extraction from the Web page. Wong and Fu [WF00] defined tag types for page segmentation and gave a label to each part of the Web page for assisting classification. Chakrabarti et al. [Cha01, CJT01] addressed the fine-grained topic distillation and disaggregated hubs into regions by analyzing DOM structure as well as intrapage text distribution. Lin and Ho [LH02] considered ⟨TABLE⟩ tag and its offspring as a content block and used an entropy-based approach to discover informative ones. Bar-Yossef and Rajagopalan [BYR02] proposed the template detection problem and presented an algorithm based on the DOM structure and the link information. Cai et al. [CYWM03, CHWM04] proposed the Vision-based Page Segmentation algorithm and developed the block-level link analysis techniques. They have also successfully applied the block-level link analysis on Web search [CYWM04] and Web image organizing and mining [CHM+04, CHL+04].

Web page classification was studied by Chakrabarti, Dom, and Indyk [CDI98] and Wang, Zhou, and Liew [WZL99]. A multilayer database approach for constructing a Web warehouse was studied by Zaïane and Han [ZH95]. Web usage mining has been promoted and implemented by many industry firms. Automatic construction of adaptive websites based on learning from Weblog user access patterns was proposed by Perkowitz and Etzioni [PE99]. The use of Weblog access patterns for exploring Web usability was studied by Tauscher and Greenberg [TG97]. A research prototype system, WebLogMiner, was reported by Zaïane, Xin, and Han [ZXH98]. Srivastava, Cooley, Deshpande, and Tan [SCDT00] presented a survey of Web usage mining and its applications. Shen, Tan, and Zhai used Weblog search history to facilitate context-sensitive information retrieval and personalized Web search [STZ05].