RANKING MODEL FOR DOMAIN SPECIFIC SEARCH

PROJECT REPORT
Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in
Computer Science and Engineering
(2009-2013)

By
ARCHANA MUTHABATHULA(09241A0516)
LATHA KUNCHALA(09241A0512)
KARTHIKEYENI BURUGULA(09241A0510)
SAI PRAVEENA GUBBALA(09241A0533)

Under the guidance: P.VaraPrasadRao(Assoc.prof)

Department Of Computer Science and Engineering
Gokaraju Rangaraju Institute of Engineering and Technology
(Affiliated to Jawaharlal Nehru Technological University, Hyderabad)
2012-2013
CERTIFICATE

This is to certify that the project entitled “RANKING MODEL FOR DOMAIN SPECIFIC SEARCH” has been submitted by ARCHANA MUTHABATHULA (09241A0516), LATHA KUNCHALA (09241A0512), KARTHIKEYENI BURUGULA (09241A0510) and SAIPRAVEENA GUBBALA (09241A0533) in partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering from Jawaharlal Nehru Technological University, Hyderabad. The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

(Signature)                                             (Signature)
Project Guide:                                        Head of Department:
Prof. P.Vara Prasad Rao                                 Dr K.Anuradha
Professor                                             Professor & HOD
Dept. of Computer Science & Engg.                     Dept. of Computer Science& Engg

1. INTRODUCTION

1.1 Purpose
LEARNING to rank is a kind of learning based information retrieval techniques, specialized in learning a ranking model with some documents labeled with their relevancies to some queries, where the model is hopefully capable of ranking the documents returned to an arbitrary new query automatically. Based on various machine learning methods, e.g., Ranking SVM the learning to rank algorithms have already shown their promising performances in information retrieval, especially Web search. However, as the emergence of domain-specific search engines, more attentions have moved from the broadbased search to specific verticals, for hunting information constraint to a certain domain. Different vertical search engines deal medical search engine should clearly be specialized in terms of its topical focus, whereas a music, image or video search engine would concern only the documents in particular formats. Since currently the broad-based and vertical search engines are mostly based on text search techniques, the ranking model learned for broad-based can be utilized directly to rank the documents for the verticals. For, example, most of current image search engines only utilize the text information accompanying images as the ranking features, such as the term frequency of query word in image title, anchor text, alternative text, surrounding text, URL and so on. Therefore, Web images are actually treated as text-based documents that share similar ranking features as the document or Web page ranking, and text based ranking model can be applied here directly. However, the broad-based ranking model is built upon the data from multiple domains, and therefore cannot generalize well for a particular domain with special search intentions. In addition, the broad-based ranking model can only utilize the vertical domain’s ranking features that are same to the broadbased domain’s for ranking, while the domain-specific features, such as the content features of images, videos or music can not be utilized directly. Those features are generally important for the semantic representation of the documents and should be utilized to build a more robust ranking model for the particular vertical.

1.2 Scope

Proposed System focus whether we can adapt ranking models learned for the existing broad-based search or some verticals, to a new domain, so that the amount of
labeled data in the target domain is reduced while the performance requirement is still
guaranteed, how to adapt the ranking model effectively and efficiently and how to utilize
domain-specific features to further boost the model adaptation. The first problem is solved by
the proposed ranking adaptability measure, which quantitatively estimates whether an
existing ranking model can be adapted to the new domain, and predicts the potential
performance for the adaptation. We address the second problem from the regularization
framework and a ranking adaptation SVM algorithm is proposed. Our algorithm is a black
box ranking model adaptation, which needs only the predictions from the existing ranking
model, rather than the internal representation of the model itself or the data from the auxiliary
domains. With the black-box adaptation property, we achieved not only the flexibility but
also the efficiency. To resolve the third problem, we assume that documents similar in their
domain specific feature space should have consistent rankings.

1.3 Project Overview

The application has been identified to have the following modules:

1. Ranking Adaptation Module.
2. Explore ranking adaptability Module.
3. Ranking adaptation with domain specific search Module.

1. Ranking adaptation Module:

Ranking adaptation is closely related to classifier adaptation, which has shown its
effectiveness for many learning problems. Ranking adaptation is comparatively more
challenging. Unlike classifier adaptation, which mainly deals with binary targets, ranking
adaptation desires to adapt the model which is used to predict the rankings for a collection of
domains. In ranking the relevance levels between different domains are sometimes different
and need to be aligned. we can adapt ranking models learned for the existing broad-based
search or some verticals, to a new domain, so that the amount of labeled data in the target
domain is reduced while the performance requirement is still guaranteed and how to adapt the
ranking model effectively and efficiently .Then how to utilize domain-specific features to
further boost the model adaptation.
2. Explore Ranking adaptability Module:

Ranking adaptability measurement by investigating the correlation between two ranking lists of a labeled query in the target domain, i.e., the one predicted by fa and the ground-truth one labeled by human judges. Intuitively, if the two ranking lists have high positive correlation, the auxiliary ranking model fa is coincided with the distribution of the corresponding labeled data, therefore we can believe that it possesses high ranking adaptability towards the target domain, and vice versa. This is because the labeled queries are actually randomly sampled from the target domain for the model adaptation, and can reflect the distribution of the data in the target domain.

3. Ranking adaptation with domain specific search Module:

Data from different domains are also characterized by some domain-specific features, e.g., when we adopt the ranking model learned from the Web page search domain to the image search domain, the image content can provide additional information to facilitate the text based ranking model adaptation. In this section, we discuss how to utilize these domain-specific features, which are usually difficult to translate to textual representations directly, to further boost the performance of the proposed RA-SVM. The basic idea of our method is to assume that documents with similar domain-specific features should be assigned with similar ranking predictions. We name the above assumption as the consistency assumption, which implies that a robust textual ranking function should perform relevance prediction that is consistent to the domain-specific features.

4. Ranking Support Vector Machines Module:

Ranking Support Vector Machines (Ranking SVM), which is one of the most effective learning to rank algorithms, and is here employed as the basis of our proposed algorithm. the proposed RA-SVM does not need the labeled training samples from the auxiliary domain, but only its ranking model fa. Such a method is more advantageous than data based adaptation, because the training data from auxiliary domain may be missing or unavailable, for the copyright protection or privacy issue, but the ranking model is comparatively easier to obtain and access.

2. SOFTWARE DEVELOPMENT LIFE CYCLE
SDLC is a life cycle that consists of certain steps that are maintained to develop a project. By this we can easily rectify the errors that occur during execution of a project. While integrating the modules, as there should not be any problem if we follow SDLC.

2.1 TYPES OF SDLC

1. Full SDLC
2. Partial SDLC

2.1.1 FULL SDLC

Full SDLC means giving the whole project to the only one developer i.e. all the phases are completed by one developer.

2.1.2 PARTIAL SDLC

Partial SDLC means giving the project to the different developers in different modules. This may be given up to detailed design stage to one developer or it may be given from coding & testing phase to another developer.

2.2 DEVELOPMENT MODEL

Development model is the set of steps that encompasses various methods, tools and procedures to accomplish the phases in SDLC. A model is chosen based on the nature of the project, application & deliverables that are required.

2.2.1 TYPES OF DEVELOPMENT MODEL

1. Waterfall Model
2. V-Model
3. Spiral Model
2.2.1.1 WATERFALL MODEL

This life cycle model demands a systematic sequential approach to software development that begins at the customer’s software requirements & progresses through analysis, design, coding, testing & maintenance.

![Waterfall Model Diagram]

Figure 2.2.1.1: Waterfall Model

2.2.1.2 V-MODEL

In this model if there are any changes to be done in future there is no need to start from user requirements stage. We can decide the stage from which we have to start in case of any modifications.

2.2.1.3 SPIRAL MODEL

The Spiral Model for software engineering has been developed to encompass the best features of both the Classic life cycle & Prototyping, while at the same time adding a new element-risk analysis-that is missing in these paradigms.

The Spiral Model defines 4 major activities.
1. **Planning**: Determination of objectives, alternatives & constraints.
2. **Risk Analysis**: Analysis of alternatives & identification resolution of risks.
3. **Engineering**: Development of the “next level” product.
4. **Customer Evaluation**: Assessment of the results of Engineering.
2.3 METHODOLOGY IN DEVELOPING SOFTWARE PROJECT

The project titled “Automatic Image Reconstruction for Face Recognition System” is developed based on the Waterfall Model, which consists of 5 phases.

1. Analysis
2. Design
3. Coding
4. Testing
5. Maintenance

2.3.1 DESCRIPTION ABOUT PHASES

1. Analysis

Software is always a part of a larger system. The work begins by establishing requirements for all system elements and then allocating some subset of these requirements to software. This system view is essential when software must interface with other elements such as hardware, people and databases. System engineering and analysis encompasses requirements gathering at the system level with small amount of top-level design and analysis.

2. Design

Design is concerned with identifying software components, specifying relationship among components, specifying software structure, maintaining a record of design decisions and providing blue print for the implementation phase.

**Design consists of**

Architectural Design
Detailed Design

**Architectural Design** involves identifying software components, decoupling and decomposing them into processing modules and conceptual data structures, specifying the inter connections among components.
Detailed Design is concerned with the details of “how to”: how to package the processing modules and how to implement the processing algorithms, data structures and interconnections among modules and data structures.

3. Coding

The design must be translated into a machine-readable form. If design is performed in a detailed manner, coding can be accomplished mechanistically.

4. Testing

Once code has been generated, program testing begins. The testing process focuses on the logical internals of the software, ensuring that all statements have been tested, and on the functional external that is conducting tests to uncover errors and ensure that define input will produce actual results that agree with required results.

5. Maintenance

Software will undoubtedly undergo change after it is delivered to the customer. Change will occur because errors have been encountered, because the software must be adapted to accommodate changes in its external environment or because the customer requires functional or performance enhancements.

2.4 ADVANTAGES

1. The linear sequential model is the oldest and the most widely used paradigm for software engineering.
2. It provides a template for each method (i.e. analysis, design, coding, testing & maintenance).

2.5 LIMITATIONS OF WATERFALL MODEL

Life cycle model is a valid model of the development process in situations where it is possible to write a reasonably complete set of specifications for the software product at the beginning of the life cycle.
3. SYSTEM ANALYSIS

3.1 PROBLEM DEFINITION

**Existing System:**

The existing broad-based ranking model provides a lot of common information in ranking documents only few training samples are needed to be labeled in the new domain. From the probabilistic perspective, the broad-based ranking model provides a prior knowledge, so that only a small number of labeled samples are sufficient for the target domain ranking model to achieve the same confidence. Hence, to reduce the cost for new verticals, how to adapt the auxiliary ranking models to the new target domain and make full use of their domain-specific features, turns into a pivotal problem for building effective domain-specific ranking models.

**Proposed System:**

Proposed System focus whether we can adapt ranking models learned for the existing broad-based search or some verticals, to a new domain, so that the amount of labeled data in the target domain is reduced while the performance requirement is still guaranteed, how to adapt the ranking model effectively and efficiently and how to utilize domain-specific features to further boost the model adaptation. The first problem is solved by the proposed ranking adaptability measure, which quantitatively estimates whether an existing ranking model can be adapted to the new domain, and predicts the potential performance for the adaptation. We address the second problem from the regularization framework and a ranking adaptation SVM algorithm is proposed. Our algorithm is a black box ranking model adaptation, which needs only the predictions from the existing ranking model, rather than the internal representation of the model itself or the data from the auxiliary domains. With the black-box adaptation property, we achieved not only the flexibility but also the efficiency. To resolve the third problem, we assume that documents similar in their domain specific feature space should have consistent rankings.
Advantage:
1. Model adaptation.
2. Reducing the labeling cost.
3. Reducing the computational cost.

3.2. Operating Environment:

The below are the software and the hardware requirements for this application.

**Hardware Requirements:**

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Speed</th>
<th>RAM</th>
<th>Hard Disk</th>
<th>Floppy Drive</th>
<th>Key Board</th>
<th>Mouse</th>
<th>Monitor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware</td>
<td></td>
<td>Speed</td>
<td>RAM</td>
<td>Hard Disk</td>
<td>Floppy Drive</td>
<td>Key Board</td>
<td>Monitor</td>
</tr>
<tr>
<td>Hardware</td>
<td>- Pentium</td>
<td>- 1.1 GHz</td>
<td>- 1GB</td>
<td>- 20 GB</td>
<td>- 1.44 MB</td>
<td>- Standard Windows Keyboard</td>
<td>- Two or Three Button Mouse</td>
</tr>
<tr>
<td>Hardware</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Software Requirements:**

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Technology</th>
<th>Web Technologies</th>
<th>IDE</th>
<th>Web Server</th>
<th>Database</th>
<th>Java Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>- Windows</td>
<td>- Java and J2EE</td>
<td>- My Eclipse</td>
<td>- Tomcat</td>
<td>- My SQL</td>
<td>- J2SDK1.5</td>
</tr>
</tbody>
</table>
3.3. Design and Implementation Constraints:

All modules are coded thoroughly based on requirements from software organization. The software is designed in such a way that the user can easily interact with the screen. Software is designed in such a way that it can be extended to the real time business.

3.4. User Interfaces

This application include GUI standards or product family style guides that are to be followed, screen layout constraints, buttons and functions that will appear on every screen, error message display standards, and so on.

3.5. User Classes and Characteristics

End user of the application is the customer or anyone who uses a certain product. The user can provide comments on any product they have used. They can search the products of any domain which are available. The admin can view the comments received and he can enhance the methods to improve the quality of the products that are falling in the corresponding domains. The admin can view the graph in terms of good comments received so far for the products of the particular domain.
4. FEASIBILITY STUDY

FEASIBILITY STUDY:

The next step in analysis is to verify the feasibility of the proposed system. “All projects are feasible given unlimited resources and infinite time”. But in reality both resources and time are scarce. Project should confirm to time bounce and should be optimal in their consumption of resources.

• Technical feasibility
• Operational feasibility
• Economic feasibility

4.1 TECHNICAL FEASIBILITY:

As we are developing this Application on Java 2 platform edition which is an open source and free of cost. Once we started developing this application in Java 2 platform edition then there is no need of purchasing any special software or application software for support.

Hardware Requirements:

<table>
<thead>
<tr>
<th>Hardware</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Speed</td>
<td>1.1 GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>1GB</td>
</tr>
<tr>
<td>Hard Disk</td>
<td>20 GB</td>
</tr>
<tr>
<td>Floppy Drive</td>
<td>1.44 MB</td>
</tr>
<tr>
<td>Key Board</td>
<td>Standard Windows Keyboard</td>
</tr>
<tr>
<td>Mouse</td>
<td>Two or Three Button Mouse</td>
</tr>
<tr>
<td>Monitor</td>
<td>SVGA</td>
</tr>
</tbody>
</table>

Software Requirements:

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Windows</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology</td>
<td>Java and J2EE</td>
</tr>
<tr>
<td>Web Technologies</td>
<td>Html, JavaScript, CSS</td>
</tr>
<tr>
<td>IDE</td>
<td>My Eclipse</td>
</tr>
<tr>
<td>Web Server</td>
<td>Tomcat</td>
</tr>
<tr>
<td>Database</td>
<td>My SQL</td>
</tr>
<tr>
<td>Java Version</td>
<td>J2SDK1.5</td>
</tr>
</tbody>
</table>
4.2 OPERATIONAL FEASIBILITY:

To determine the operational feasibility of the system we should take into consideration the awareness level of the users. Users who are using this Application don’t require much knowledge of how to use. Everything will be understood by user once he sees the application.

4.3 ECONOMIC FEASIBILITY:

To decide whether a project is economically feasible, or not we have to consider various factors as:

- Cost benefit analysis
- Long-term returns
- Maintenance costs
5. SYSTEM DESIGN

5.1 Introduction to UML:

The Unified Modeling Language (UML) is a language for specifying, visualizing and constructing the artifacts of software system as well as for business models. GRADY BOOCH, IVAR JACCOBSON and JAMES RUMBUGH are the founders of the UML. The UML notation is useful for graphically depicting Object Oriented Analysis and Object Oriented Design (OOA and OOD) modules. The unified modeling language is a standard language for specifying, Visualizing, Constructing and documenting the software system and its components.

Visualizing:

Through UML we see or visualize an existing system and ultimately we visualize how the system is going to be after implementation. Unless we think we cannot implement.UML helps to visualize how the components of the system communicate and interact with each other.

Specifying:

Specifying means building models that are precise, unambiguous and complete UML addresses the specification of all the important Analysis Design, Implementation decisions that must be made in developing and deploying a software system.

Constructing:

UML s models can be directly connected to a variety of programming language through mapping a model from UML to a programming language like Java or C++ or VB. Forward Engineering and Reverse Engineering is possible through UML.
Documenting:

The deliverables of a project apart from coding are some artifacts which are critical in controlling, measuring and communicating about a system during its development viz. Requirements, Architecture, Design, Source code, Project plans, Tests, Prototypes, Releases etc.

5.2 UML Diagram

Diagrams are graphical presentation of set of elements. Diagrams project a system, or visualize a system from different angles and perspectives.

5.2.1 Static or Structural Diagrams

Class diagram:

This shows a set of classes, interfaces, collaborations and their relationships. There are the most common diagrams in modeling the object oriented systems and are used to give the static view of a system.

Object diagram

They show set of objects and their relationships. Object diagrams represent static snapshots of instances of the things found in class diagram. These diagrams address the static design view or static process view of a system.

Component Diagram

They show the organizations and dependencies among a set of components. They are related to class diagrams in that a component typically maps to one or more classes, interfaces, or collaborations.
Component Diagram

It shows the configuration of runtime processing nodes and the components that live on them. Deployment diagrams address the static deployment view of an architecture. They are related to component diagrams in that a node typically encloses one or more components.

5.2.2. Dynamic or behavioral diagrams

Sequence diagram & collaboration diagram:

These two diagrams are semantically same i.e. the dynamics of a system can be modeled using one diagram and transform it to the other kind of diagram without loss of information. Both form the, Interaction diagram.

UseCase Diagram:

![UseCase Diagram for Admin]

Fig 5.2.2.1- Usecase diagram for the admin
Fig 5.2.2.2- Use case diagram for the user
Sequence diagram:

Sequence diagram is an interaction diagram which focuses on the time ordering of messages it shows a set of objects and messages exchange between these objects. This diagram illustrates the dynamic view of a system.

Fig 5.2.2.3-Sequence diagram for the admin
Collaboration diagram:

This diagram is an interaction diagram that stresses or emphasizes the structural organization of the objects that send and receive messages. It shows a set of objects, links between objects and messages send and received by those objects. There are used to illustrate the dynamic view of a system.

Activity Diagram:

Activity diagram shows the flow from one activity to another within a system. The activities may be sequential or branching objects that act and are acted upon. These also show the dynamic view of the system.
Fig 5.2.2.5-Activity diagram for the admin
Fig 5.2.2.6- Activity diagram for the user
CONTROL FLOW:

The state diagram gives the control flow of the system. The various steps are validation, uploading, downloading, viewing and deleting.

DATABASE DESIGN:

The below tables are used for this application

Table_5.2.2.1: Admin

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Field Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>Varchar2</td>
<td>255</td>
</tr>
<tr>
<td>Password</td>
<td>Varchar2</td>
<td>255</td>
</tr>
</tbody>
</table>

Table_5.2.2.2: cmtcount

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Field Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Rate</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Count</td>
<td>Int</td>
<td>11</td>
</tr>
</tbody>
</table>

Table_5.2.2.3: comment

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Field Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Feedback</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Rate</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Comment</td>
<td>Varchar</td>
<td>255</td>
</tr>
</tbody>
</table>

Table_5.2.2.4: reg

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Field Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>Int</td>
<td>255</td>
</tr>
<tr>
<td>Name</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Pass</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Email</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Phone</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Date</td>
<td>Varchar</td>
<td>255</td>
</tr>
</tbody>
</table>

Table_5.2.2.5: upload

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Field Type</th>
<th>Field Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image</td>
<td>Longblob</td>
<td></td>
</tr>
<tr>
<td>Domain</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Subdomain</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Title</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Types</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Descrip</td>
<td>Varchar</td>
<td>255</td>
</tr>
<tr>
<td>Countg</td>
<td>Int</td>
<td>11</td>
</tr>
<tr>
<td>Countb</td>
<td>Int</td>
<td>11</td>
</tr>
</tbody>
</table>
6. IMPLEMENTATION

6.1 OVERVIEW OF SOFTWARE DEVELOPMENT TOOLS

Software:

The life-cycle paradigm demands a systematic, sequential approach to software development that begins at the system level and progresses through requirements analysis, design, coding, testing and maintenance.

a) Analysis Phase:

The analysis phase consists of two sub phases: planning and requirements definition. During planning the activities that are performed are - understand the customer’s problem, developing a recommended solution. Requirements definition is concerned with identifying the basic functions of a software component in a hardware/software/people system.

b) Design Phase:

Design isss concerned with identifying software components, specifying relationships among components, maintaining a record of design decisions. Design consists of architectural design and detailed design.

i) Architectural Design involves identifying the software components, decoupling and decomposing them in to processing modules and conceptual data structures and specifying the interconnection between the components.

ii) Detailed Design is concerned with the details of how to package the processing modules and how to implement the processing algorithms, data structures and interconnection between them.

c) Implementation Phase:

The implementation phase of software development involves translation of design specifications source code and debugging, documentation and unit testing of the source code.
d) Testing Phase:

It involves two kinds of testing:

i) In integration testing the individual program units or programs are integrated and tested.

ii) Acceptance Testing involves planning and execution of various types of tests in order to demonstrate that the implemented software satisfies the stated requirements.
7. Technology review

7.1. JAVA and its FEATURES

Java was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan at SUN Microsystems Incorporation in the year 1991. It took 18 months to develop the 1st working version. This language was initially called “OAK”, but was renamed “JAVA” in 1995, many more contributed to the design and evolution of the language.

7.1.1. Java Overview:

Java is a powerful but lean object-oriented programming language. It has generated a lot of excitement because it makes it possible to program for Internet by creating Applets. Programs that can be embedded in web page. The context of an applet can be an animation with sound, an interactive game or a ticker tape. With constantly updated stock prices. Applets can be just little decorations to liven up web page, or they can be serious applications like Word processor or Spreadsheet.

But Java is more than a programming language for writing Applets. It is being used more and more for writing standalone applications as well. It is becoming so popular that many people believe it will become standard language for both general purpose and Internet programming.
Java is simple, elegant, and powerful and easy-to-use.
Java is actually a platform consisting of 3 components:
Java Programming Language.
Java Library of Classes and Interfaces.
Java Virtual Machine

7.1.2. Java is portable

One of the biggest advantages Java offers is that it is portable. An application written in Java will run on all the major platforms. Any computer with a Java-based browser can run the applications or Applets written in the Java-Programming-Language. A programmer no longer has to write one program to run on a Macintosh, another program to run on a
Windows-machine still another to run on a UNIX-machine and so on. In other words, with Java developers write their programs only once. The Virtual Machine is what gives Java is cross platform capabilities. Rather being compiled into machine language, which is different for each OS’s and computer architecture, Java code is compiled into Byte codes. With other languages, the program code is compiled into a language that the computer can understand.

7.1.3. Java is Object-Oriented

The Java programming language is OBJECT-ORIENTED, which makes program design focus on what you are dealing with, rather than on how you are going to do something. This makes it more useful for programming in sophisticated projects, because one can break the things into understandable components. A big benefit is that these components can then be reused.

The class paradigm allows one to encapsulate data so that specific data values are those using the data cannot see the function implementation. Encapsulation makes it possible to make the changes in code without breaking other programs that use that code. If for example, the implementation of a function is changed, the change is invisible to any programmer who invokes that function, and does not affect his/her program, except hopefully to improve it.

7.2. JAVA DEVELOPMENT ENVIRONMENT

To code, edit, debug and test the java programs, one needs to have a java development environment. At the minimum this will consists of a java compiler interpreter and applet viewer where applets can be tested. Sun’s java development kit (JDK) latest version is 2.2 can be freely downloaded from the Internet. Java compiler is available on DOS, Win95, WIN’NT, Solaris and MAC etc.

Data flow diagram is a graphical tool used to describe analyze the movement of data through a system manual or automated including the processes, stores of data, and delays in the system.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto
standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE. There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn’t change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn’t as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the
quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

**JDBC**

In an effort to set an independent database standard API for Java, Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or drivers. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

**7.3 HTML**

HyperText Markup Language (HTML) is the main markup language for creating web pages and other information that can be displayed in a web browser. HTML is written in the form of HTML elements consisting of tags enclosed in angle brackets (like <html>), within the web page content. HTML tags most commonly come in pairs like <h1> and </h1>, although some tags, known as empty elements, are unpaired, for example <img>. The first tag in a pair is the start tag, the second tag is the end tag (they are also called
opening tags and closing tags). In between these tags web designers can add text, tags, comments and other types of text-based content.

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages.

Web browsers can also refer to Cascading Style Sheets (CSS) to define the appearance and layout of text and other material. The W3C, maintainer of both the HTML and the CSS standards, encourages the use of CSS over explicit presentational HTML markup

7.4 JSP:

JavaServer Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems, JSP is similar to PHP, but it uses the Java programming language.

Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSPs are translated into servlets at runtime; each JSP's servlet is cached and re-used until the original JSP is modified.

JSP can be used independently or as the view component of a server-side model–view–controller design, normally with JavaBeans as the model and Java servlets (or a framework such as Apache Struts) as the controller. This is a type of Model 2 architecture.

JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, use Java bytecode rather than a native software format. Like any other Java program, they must be executed within a Java virtual machine (JVM) that integrates with the server's host operating system to provide an abstract platform-neutral environment.

JSPs are usually used to deliver HTML and XML documents, but through the use of OutputStream, they can deliver other types of data as well.
The Web container creates JSP implicit objects like pageContext, servletContext, session, request & response.

JSP pages use several delimiters for scripting functions. The most basic is `<% ... %>`, which encloses a JSP scriptlet. A scriptlet is a fragment of Java code that is run when the user requests the page. Other common delimiters include `<%= ... %>` for expressions, where the value of the expression is placed into the page delivered to the user, and directives, denoted with `<%@ ... %>`.

Java code is not required to be complete or self-contained within its scriptlet element block, but can straddle markup content providing the page as a whole is syntactically correct. For example, any Java if/for/while blocks opened in one scriptlet element must be correctly closed in a later element for the page to successfully compile. Markup which falls inside a split block of code is subject to that code, so markup inside an if block will only appear in the output when the if condition evaluates to true; likewise, markup inside a loop construct may appear multiple times in the output depending upon how many times the loop body runs.
8. SYSTEM TESTING

The software testing process starts once the program is written and the documentation and related data structures are designed. Software testing is essential for correcting errors and improving the quality of the software system. Without proper testing or with incomplete testing, the program or the project is said to be incomplete.

8.1 UNIT TESTING

In this, the programs that made up the system were tested. This is also called as program testing. This level of testing focuses on the modules, independently of one another. The purpose of unit testing is to determine the correct working of the individual modules. For unit testing, we first adopted the code testing strategy, which examined the logic of program. During the development process itself all the syntax errors etc. got rooted out. For this we developed test case that results in executing every instruction in the program or module i.e. every path through program was tested. (Test cases are data chosen at random to check every possible branch after all the loops.).

Unit testing involves a precise definition of test cases, testing criteria, and management of test cases. The main purpose behind testing is to find errors. This level of testing focuses on the modules, independently of one another.

Testing means to check weather system meets user requirements about:

Data entry

In Employee Resource Information System the data entry is done through the forms like Item Employee Details, Applicant details, client details, Etc

Error handling
In this system we have tried to handle all the errors that are occurred while running the forms. The common errors we saw are reading the empty record and displaying a compiler message, etc.

**Data access protection**

In Employee Resource Information System data is stored in tables. There is no need to use any RDBMS for the system since there is for the user to incorporate any changes himself. On it contains tables, which are used for storing the data about the details of Employee details, applicant details, client details etc.

For Testing we used Top-Down design a decomposition process which focuses as the flow of control, at latter strategies concern itself with code production. The first step is to study the overall aspects of the tasks at hand and break it into a number of independent modules. The second step is to break one of these modules further into independent sub modules. One of the important features is that at each level the details at lower levels are hidden.

**8.2 INTEGRATION TESTING**

In this the different modules of a system are integrated using an integration plan. The integration plan specifies the steps and the order in which modules are combined to realize the full system. After each integration step, the partially integrated system is tested. The primary objective of integration testing is to test the module interface.

An important factor that guides the integration plan is the module dependency graph. The module dependency graph denotes the order in which different modules call each other. A structure chart is a form of a module dependency graph. Thus, by examining the structure chart the integration plan can be developed based on any of the following approaches:

- Big-bang approach.
- Top-down approach.
- Bottom-up approach.
- Mixed approach.
Bottom-up Integration Testing

In this approach, each subsystem is tested separately and then the full system is tested. A subsystem might consist of many modules, which communicate among each other through well-defined interfaces. The primary purpose of testing each subsystem is to test the interfaces among various modules making up the subsystem. Both control and data interface is tested. A principal advantage of bottom-up integration testing is that several disjoint subsystems can be tested simultaneously. A disadvantage of bottom-up testing is the complexity that occurs when the system is made up of large number of small subsystems.

In Employee Resource Information System, we have tested all the individual programs first and after having successful results in the individual program testing we moved further for the integration. We have combined some programs and then tested it, after having good results; we have combined all the programs together and started for system testing.

8.3 SYSTEM TESTING

Once we are satisfied that all the modules work well in themselves and there are no problems, we do in to how the system will work or perform once all the modules are put together. The main objective is to find discrepancies between the system and its original objective, current specifications, and system documentation. Analysts try to find moulds that have been designed with different specifications, which could cause incompatibility. At this stage the system is used experimentally to ensure that all the requirements of the user are fulfilled. At this point of the testing takes place at different levels so as to ensure that the system is free from failure. Testing is mostly performed by persons who have never worked with the system before, so that the feedback we get is free from bias.

Testing is vital to success of the system. System testing makes a logical assumption that whether all parts of the system are correct. Initially the system was given to the user for entry validation was provided at each and every stage. So, that the user is not allowed to enter unrelated data. The training is given to user about how to make an entry.
While implementing the system it was observed that the user was initially resisting the change, however the system being the need of the hour and user friendly, the fear was overcome. Entering live data of the past months records was little tedious, prior to the actual day to day transactions.
The best test made on the system was whether it produces the correct outputs. All the outputs were checked out and were found to be correct. Feedback sessions were conducted and the suggested changes given by the user were made before the acceptance test. Finally the system is being accepted and made to run with live data.
System tests are designed to validate a fully developed system with a view to assuring that it meets its requirements.
There are three main kinds of system testing:
Alpha Testing.
Beta Testing.
Acceptance Testing.

**Alpha Testing** This refers to the system testing that is carried out by the test team with the organization.

**Beta Testing** This refers to the system testing that is performed by a select group of friendly customers.

**Acceptance Testing** This refers to the system testing that is performed by the customer to determine whether or not to accept the delivery of the system.

### 8.4 USER ACCEPTANCE TESTING

Acceptance testing involves planning and execution of functional test, performance tests and stress tests to verify that the Project **Automatic Image Reconstruction for Face Recognition System** satisfies its requirements.
Acceptance tests are typically performed by the quality assurance and or customer organizations. Depending on local circumstances, the development group may or may not be involved in acceptance testing. In addition to, functional and performance tests, stress tests are performed to determine the limitations of the system. Acceptance tests will incorporate test cases developed during unit testing and integration testing. Additional test cases are added to achieve the desired level of functional, performance, and stress testing of the entire
Tools of special importance during acceptance testing include a test coverage analyzer, a timing analyzer, and a coding standard checker.

**Test coverage analyzer** records the control paths followed for each test case. The cumulative record is used to establish the extent of test coverage obtained during acceptance testing. Without this tool, it is impossible to establish the extent of test coverage obtained.

**Timing analyzer** reports the time spent in various regions of the source code under different test cases. It is not unusual for a program to spend 80 to 90 percent of execution time in 20 percent or less of code. These regions of the code are areas to concentrate on to improve system performance.

Certain coding standards are often stated in the product requirements. Coding standards may be project-related, customer-related, developer-related, or language-related. Manual inspection is usually not an adequate mechanism for detecting violations of coding standards. Static analyzers and standards checkers can be used to inspect code for departures from standards and guidelines. All the coding standards and Acceptance tests, unit testing and integration testing cases are added to achieve the desired level of functional, performance, and stress testing of the Warehouse Management System and every care is taken so that the user has no problem with the project. The entire project is menu driven and it is extremely user friendly and does not require any computer skills to use it.
9. SCREENSHOTS

9.1 Home Page of the Application

![Home Page of the Application](image1)

We can adapt an existing ranking model to a new domain, so that the amount of labeled data and the training cost is reduced while the performance is still guaranteed. Our algorithm only requires the predictions from the existing ranking module, rather than their internal representations or the data from auxiliary domain. In addition, we assume that documents similar in the domain-specific feature space should have consistent rankings, and add some constraints to control the margin.

Figure 9.1

9.2 Admin login Page

![Admin login Page](image2)

Figure 9.2
9.3 Admin inserts the domain details

Figure 9.3

9.4 New User Registration page

Figure 9.4
9.5 User registers into the website

![Figure 9.5](image1)

9.6 User login into the website

![Figure 9.6](image2)
9.7 User searches the domain item

![Image of a search engine interface]

Figure 9.7

9.8 User views the details of the domain item

![Image of a search engine interface with search results]

electronic: mobile
Figure 9.8
9.9 User posting the comments

![Image of user posting comments]

Figure 9.9

9.10 Admin views the comments

![Image of admin viewing comments]

Figure 9.10
9.11 Admin views the feedback

![Figure 9.11](image)

9.12 Admin views the users registered

![Figure 9.12](image)
9.13 Admin views the graphical representation

**Figure 9.13**
10. CONCLUSION AND FUTURE SCOPE

10.1 CONCLUSION

Here, the system is implemented fulfilling all the client requirements. The interfaces designed for the system are very user friendly and attractive. It has successfully implemented the operations of an organization like creating the various domains and uploading the images. The user can register and comment on the domains inserted by the admin by the search option. The comments of the users are viewed by the admin and they are differentiated based on the good and bad comments. He can view the description of the comments received and they can perform the ways to increase their product efficiency. The user can provide their comments but cannot see the comments received before which are given by other users. Only the admin can view them.

10.2 Future Scope

Further the application can be designed in such a way that the comments can be received through the mobile SMS and the users can view all the comments received so far for the products and the user can also view the graphical representation of the ranked items.
REFERENCES

http://en.wikipedia.org/wiki/Evolutionary_data_mining
http://ieeeprojects.ellamey.com/datamining.php
http://www.google.co.in/
http://www.roseindia.com/
http://www.java2s.com/