A Project report on

Online Hashing with Bit Selection for Image Retrieval

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

B.Thirupathi
(19H51A0532)
K.Vamshi krishna
(19H51A0574)
G.Thriveni
(20H55A0507)

Under the esteemed guidance of

Dr. Vijaya Kumar Koppula

(Professor of CSE & Dean Academics
Dept. of CSE)

EXPLORE TO INVENT

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC & JNTUH, Approved by AICTE, Permanently Affiliated to JNTUH, Accredited by NBA.)

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Major Project Phase-II report entitled **"Online Hashing with Bit Selection for Image Retrieval"** being submitted by B.Thirupathi (19H51A0532), K.Vamshi krishna (19H51A0574), G.Thriveni (20H55A0507) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

Dr. Vijaya Kumar Koppula
Professor of CSE & Dean Academics
Dept. of CSE

Dr. Siva Skandha Sanagala
Associate Professor and HOD
Dept. of CSE

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Dr. VIJAYA KUMAR KOPPULA, Professor of CSE & Dean Academics**, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academic, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, We extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

B.Thirupathi	19H51A0532
K.Vamshi krishna	19H51A0574
G.Thriveni	20H55A0507

TABLE OF CONTENTS

List of Figures

List of Tables

ABSTRACT

Online hashing methods have been intensively investigated in semantic image retrieval due to their efficiency in learning the hash functions with one pass through the streaming data. Among the online hashing methods, those based on the target codes are usually superior to others. However, the target codes in these methods are generated heuristically in advance and cannot be learned online to capture the characteristics of the data.In this project, we propose a new online hashing method in which the target codes are constructed according to the data characteristics and are used to learn the hash functions online. By designing a metric to select the effective bits online for constructing the target codes, the learned hash functions are resistant to the bit-flipping error. At the same time, the correlation between the hash functions is also considered in the designed metric. Hence, the hash functions have low redundancy. Extensive experiments show that our method can achieve comparable or better performance than other online hashing methods on both the static database and the dynamic database.

CHAPTER 1
INTRODUCTION

CHAPTER 1

1. INTRODUCTION

1. Due to the ever-growing amount of multimedia, it is a substantial challenge to retrieve the relevant content from a large multimedia database by performing the nearest neighbor search. As the multimedia data, such as videos and images, are usually represented by high-dimensional feature descriptors conventional nearest neighbor search methods such as tree-based methods cannot deal with the high-dimensional data.

2. When new images are adding into the database, the new images entail new label classes, resulting in changes of the data distribution. When fresh data comes, it takes time to recapture the hash functions by collecting all the data. It's also prohibitively costly to learn the hash functions, because of the restricted memory space, by putting the large database at once into memory. In response to the above-mentioned problems, online hashing methods have been created to learn the hash functions online from the increasing data.

There are two-part hash techniques, update hash function and Update binary code online hashing methods. The first section focuses on learning from the data efficient hash algorithms for compact binary codes.The second portion focuses on developing a measure to determine whether all and some binary codes may be changed since the updating of the hash functions is time-consuming and unneeded.

1.1 Problem Statement

Online hashing methods have been intensively investigated in semantic image retrieval due to their efficiency in learning the hash functions with one pass through the streaming data. Among the online hashing methods, those based on the target codes are usually superior to others. However, the target codes in these methods are generated heuristically in advance and cannot be learned online to capture the characteristics of the data.

1.2 Research Objective

BATCH-BASED HASHING

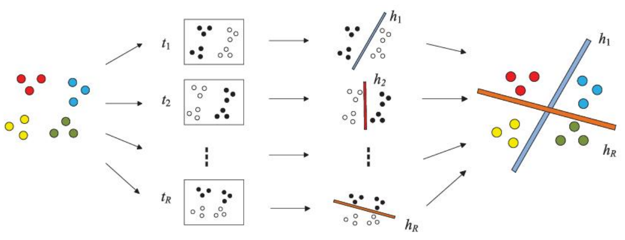
Locality sensitive hashing (LSH) is one of the representative hashing methods, which generates the hash functions by random projection. However, LSH is a data-independent hashing method and usually needs long binary codes to achieve good search accuracy. Compared to data-independent hashing methods, data-dependent hashing methods can generate compact binary codes by learning the hash functions from the data.The data-dependent hashing methods can be categorized as unsupervised hashing methods and semi-supervised hashing methods. However, the above data-dependent hashing methods are batch-based methods and cannot deal with the new data. When new data arrives, they have to re-learn the hash functions by using all the data, which is time consuming.

ZERO-SHOT HASHING

Zero-shot hashing methods attempt to address the problem of new image classes that are not seen in the training phase due to the insufficient training data. As some image classes are not covered in the training phase, the zero shot hashing methods describe the unseen images by transferring the supervised knowledge from the seen image classes to the unseen image classes There are two ways for the zero-shot hashing methods to transfer supervised knowledge. In the first way they use the class attributes as the mediator and construct a shared attribute space between the seen classes and the unseen classes. In the second way they use the word embedding technique .

to represent the class concepts, and transfer the knowledge among these classes according to the correlation between their concepts. In either way, the seen classes and the unseen classes should be highly related, and considerable auxiliary information is needed, which limits their applications.

Data Binary Bit Generation Hash Function Learning Hash Functions and Bit selection



1.3 Project Scope

By designing a metric to select the effective bits online for constructing the target codes, the learned hash functions are resistant to the bit flipping error. At the same time, the correlation between the hash functions is also considered in the designed metric. Hence, the hash functions have low redundancy .

CHAPTER 2
BACKGROUND WORK

CHAPTER 2

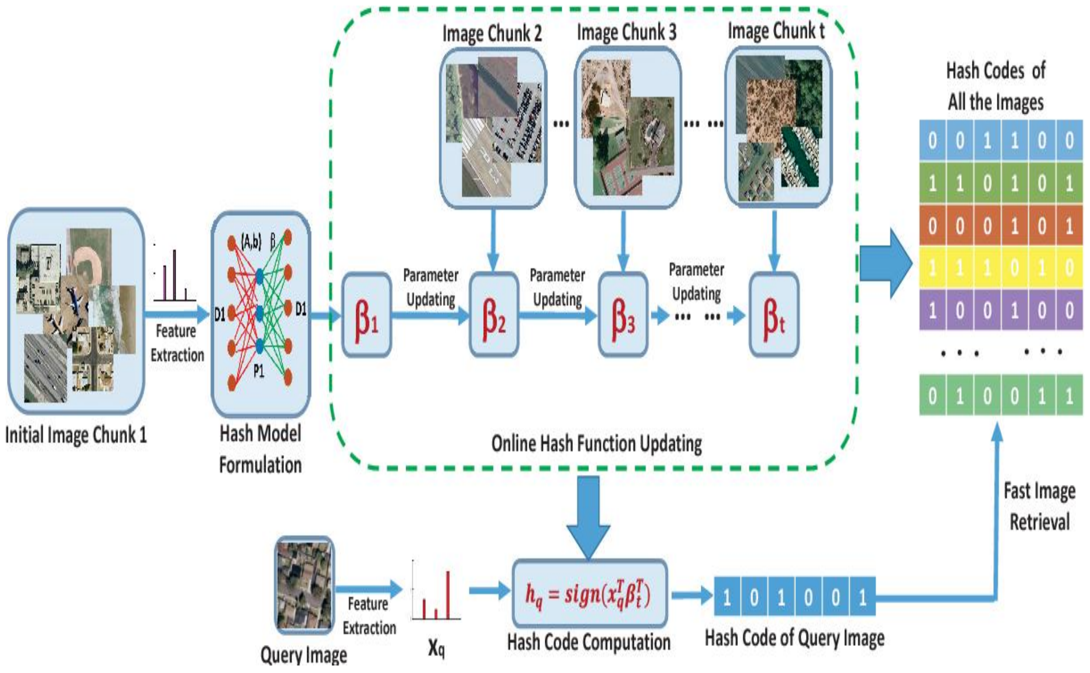
BACKGROUND WORK

2.1. Online Hashing for Scalable Remote Sensing Image Retrieval

2.1.1. Introduction

The rapid development of satellite and aerial vehicle technologies, we have entered an era of remote sensing (RS) big data. Automatic knowledge discovery from massive RS data has become increasingly urgent. Among emerging RS big data mining efforts, large-scale RS image retrieval has attracted an increasing amount of research interest due to its broad applications in the RS research community. For example, a fast and accurate retrieval of similar satellite cloud images can provide valuable judging information for short-term weather forecasting. Besides, in the disaster rescue scenario, a fast rescue and optimal resources allocating also depend on the real-time and precise retrieval strategies for the photographs of disaster area.

RS image retrieval systems, RS image retrieval mainly relied on manual tags in terms of sensor types, waveband information, and geographical locations of remote sensing images. However, the manual generation of tags is quite time consuming and becomes especially prohibitive when the volume of remote sensing images is oversized. As an effective method to manage a large number of images, content-based image retrieval (CBIR) can retrieve the interesting images according to their visual content. In recent years, content-based RS image retrieval has been comprehensively studied, in which the similarity of RS images is measured by different kinds of visual descriptors. More , local invariant, morphological, textural, and data-driven features have been evaluated in terms of content-based RS image retrieval tasks. To further improve image retrieval performance levels, proposed a multiple feature-based remote sensing image retrieval approach by combining handcrafted features and data-driven features via unsupervised feature learning.



Merits-

Online Hashing for Scalable Remote Sensing Image Retrieval offers the following advantages:

- Efficient and adapts well to long code lengths.
- Effective compact hashing.
- Less memory consumption and calculation cost.

Demerits-

Online Hashing for Scalable Remote Sensing Image Retrieval offers the following disadvantages:

- Training speed very slow Difficult to optimize.
- Slower because of the extraction of SIFT descriptors.

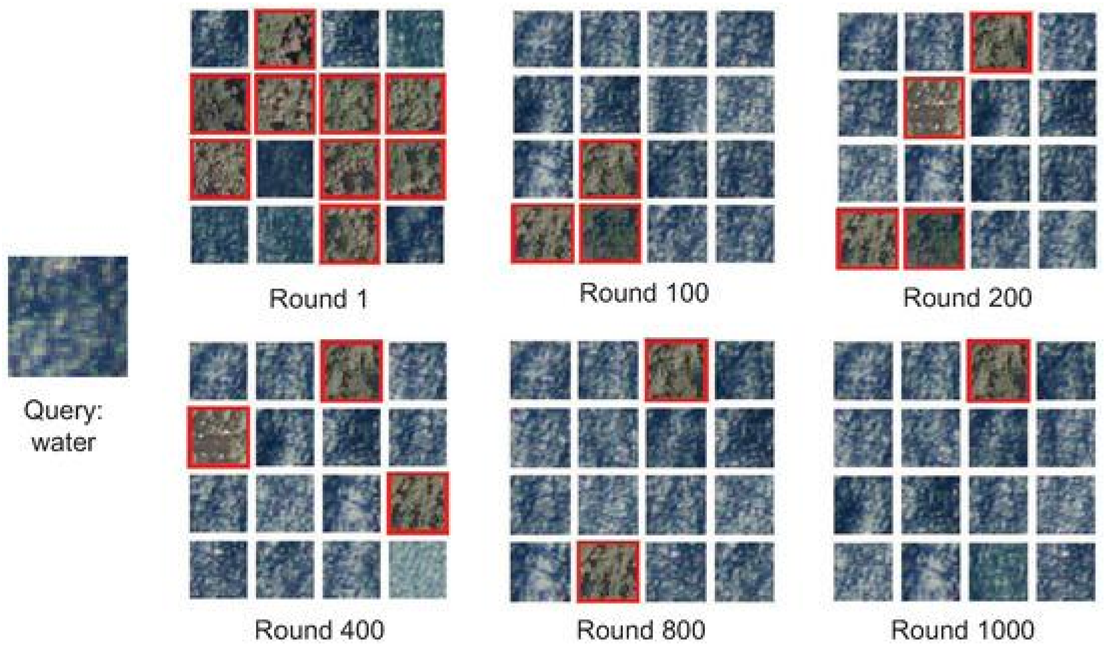
Challenges-

1. hashing approach for real-world large-scale remote sensing image retrieval.
2. state-of-the-art for online hashing.

2.1.2 Implementation of Online Hashing for Scalable Remote Sensing Image Retrieval

```
1:Input: Streaming image data chunk D1,D2,…,Dk, code length r
2:Output: Hash codes H for all the images
3:Randomly generate a projection matrix A∈Rd×r and a bias row vector b∈Rr
4:Compute P1 by P1=g(DT1· A+1nb)
5:Compute Q1=PT1P1 and β^1=Q−11PT1DT1
6:fori=2:k do
7:Compute Pi by Pi=g(DTi· A+1nb)
8:Update Qi with Equation (8)
9:Update β^i with Equation (9)
10:end
For
11:Compute the hash codes H for the whole database X=[D1,D2,…,Dk] by H=sign(XTβ^Tk).
```

Output:

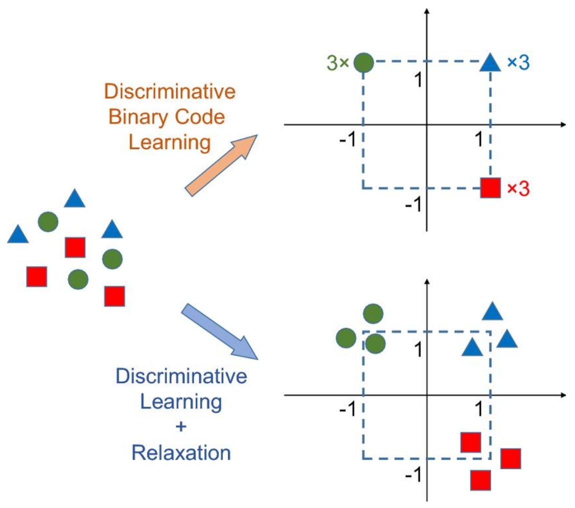

Round 1
Round 100
Round 200
Query:
water
Round 400
Round 800
Round 1000

2.2 Scalable Supervised Online Hashing For Image Retrieval

2.2.1 Introduction

A new loss function is introduced into the supervised online hashing model by constructing a similarity measure between the new data stream and the existing data. Large-scale pairwise similarity calculations are replaced by calculated intermediate variables, which effectively saves calculation costs. The introduction of intermediate variables also effectively alleviates the phenomenon of data imbalance. A discrete optimization algorithm is designed that makes it possible to achieve online discrete optimization.

In the online hashing method, the data will reform along with the change of time, and the model should be updated continuously to cater to the new data. However, the hash code of the historical data stored in the training set is not obtained by mapping using the latest hash function, but received by mapping the hash function at a certain time in the past. Since the hash mapping functions of the data are inconsistent, this may result in a decrease in search accuracy. Therefore, in the research of online hashing methods, it is necessary to balance the influence of historical data and new data on model updates to ensure the accuracy of hash search. Briefly, online hash work mainly focuses on exploring and solving three problems.


Discriminative Binary Code Learning
Discriminative Learning + Relaxation
3×
×3
×3
1
-1
1
-1

In view of the broad application prospect of hash method in real life, domestic and foreign experts have made a lot of efforts and contributions. Next, we divide the hashing method into offline hashing and online hashing according to the incremental update of the hash model when the research data change dynamically.

Merits:

- Efficient hash functions.
- Higher retrieval accuracy.
- Fast ANN search.

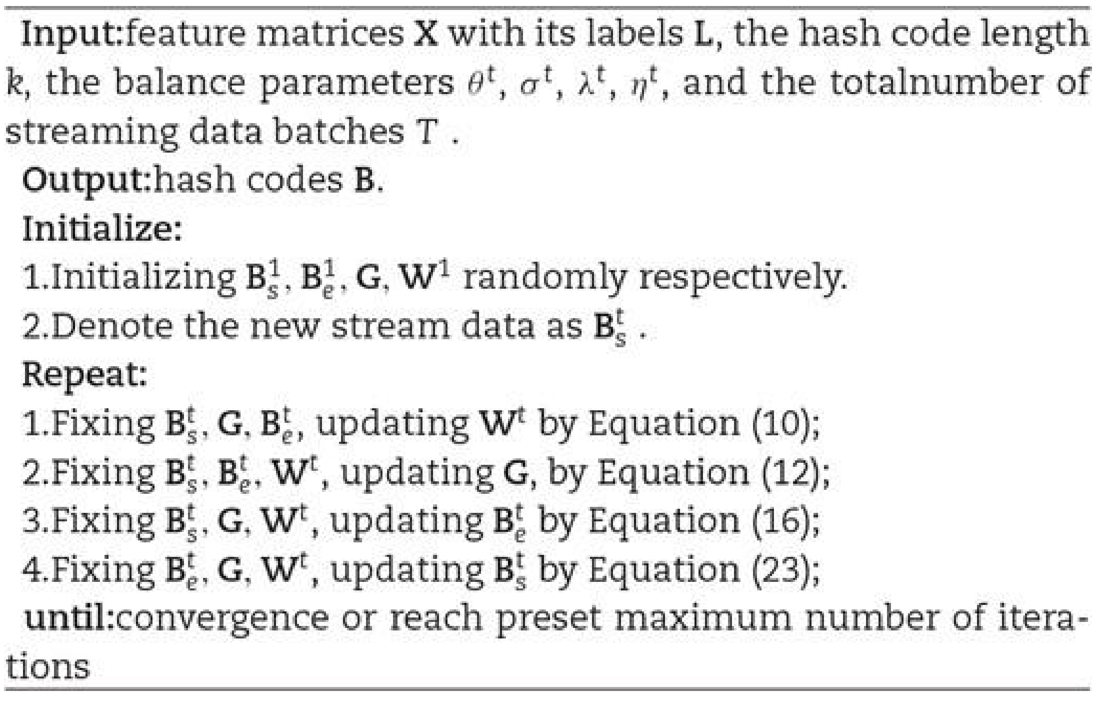
Demerits

- Not sufficient for high - dimensional descriptors.
- Not use all training points due to the complexity.
- Unsatisfactory performance in real-world applications.

Challenges:

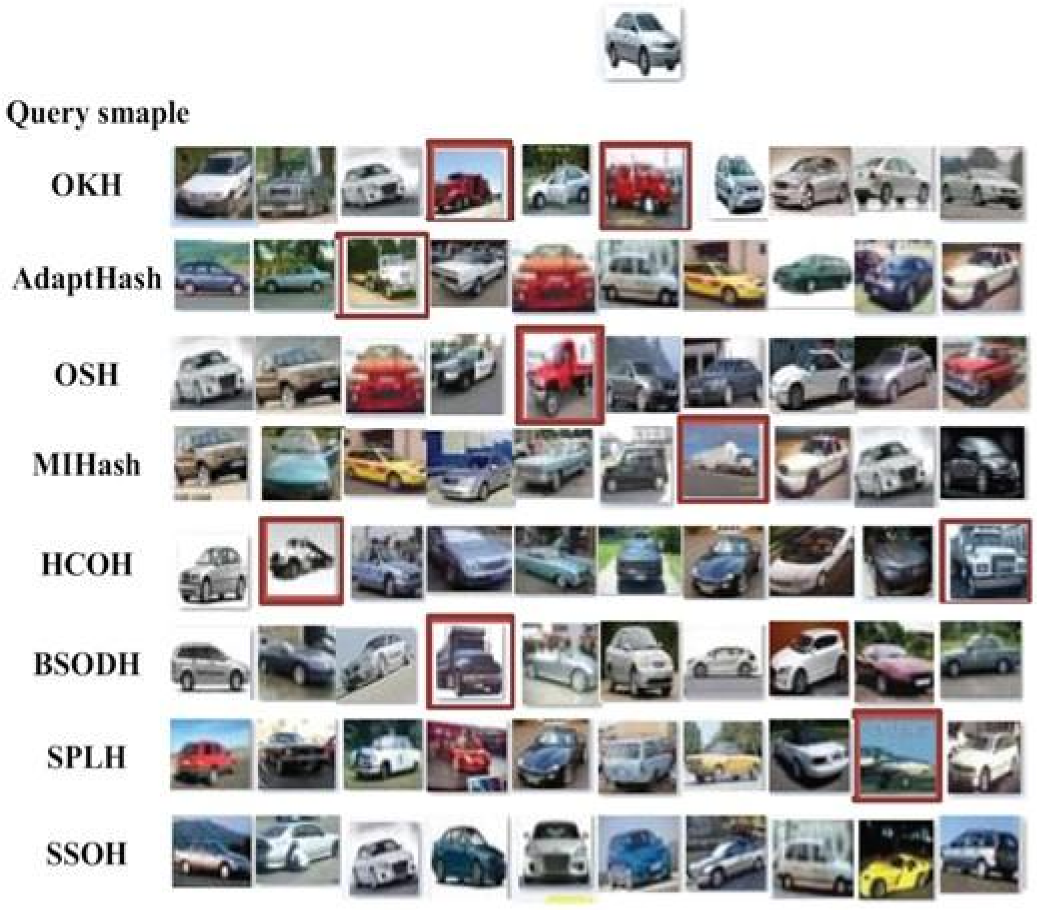
1. Retrieve all correct images based on the query image.
2. It reflects the differences in the retrieval results of each method to a certain extent.

2.2.2 Implementation of Scalable Supervised Online Hashing For Image Retrieval



```
Input: feature matrices $\mathbf{X}$ with its labels $\mathbf{L}$, the hash code length $k$, the balance parameters $\theta^t$, $\sigma^t$, $\lambda^t$, $\eta^t$, and the totalnumber of streaming data batches $T$ .
Output: hash codes $\mathbf{B}$.
Initialize:
  1.Initializing $\mathbf{B}_s^1$, $\mathbf{B}_e^1$, $\mathbf{G}$, $\mathbf{W}^1$ randomly respectively.
  2.Denote the new stream data as $\mathbf{B}_s^t$ .
Repeat:
  1.Fixing $\mathbf{B}_s^t$, $\mathbf{G}$, $\mathbf{B}_e^t$, updating $\mathbf{W}^t$ by Equation (10);
  2.Fixing $\mathbf{B}_s^t$, $\mathbf{B}_e^t$, $\mathbf{W}^t$, updating $\mathbf{G}$, by Equation (12);
  3.Fixing $\mathbf{B}_s^t$, $\mathbf{G}$, $\mathbf{W}^t$, updating $\mathbf{B}_e^t$ by Equation (16);
  4.Fixing $\mathbf{B}_e^t$, $\mathbf{G}$, $\mathbf{W}^t$, updating $\mathbf{B}_s^t$ by Equation (23);
until: convergence or reach preset maximum number of itera-tions
```

Output:

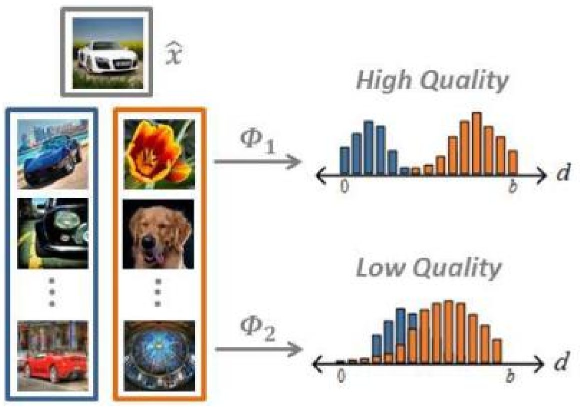


2.3 Online Hashing with Mutual Information

2.3.1.Introduction

Hashing is a widely used approach for practical nearest neighbor search in many computer vision applications. It has been observed that adaptive hashing methods that learn to hash from data generally outperform data-independent hashing methods such as Locality Sensitive Hashing . In this paper, we focus on a relatively new family of adaptive hashing methods, namely *online* adaptive hashing methods . These techniques employ online learning in the presence of streaming data, and are appealing due to their low computational complexity and their ability to adapt to changes in the data distribution.

Despite recent progress, a key challenge has not been addressed in online hashing, which motivates this work the computed binary representations, or the "hash table", may become outdated after a change in the hash mapping. To reflect the updates in the hash mapping, the hash table may need to be recomputed frequently, causing inefficiencies in the system such as successive disk I/O, especially when dealing with large datasets.



Merits:

- Highly efficient .
- Very fast and high precision.
- Low storage cost .

.

Demerits :

- Require a significant degree of effort in large-scale applications.
- Expensive training time .
- Insufficient precision rate.

Challenges :

1. Mutual information based update criterion, the Trigger Update module (TU).

2.TU can be used to filter updates to the hash mapping with negative or small improvement.

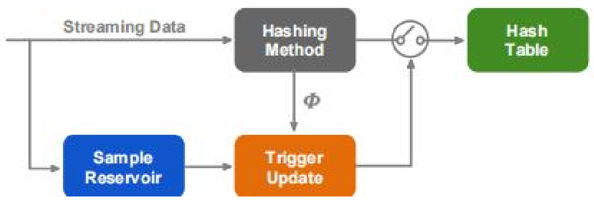
.

.

2.3.2 Implementation of Online Hashing with Mutual Information :

1. The goal of hashing is to learn a hash mapping
$\Phi : X \rightarrow Hb$ such that a desired neighborhood structure is preserved.
2. We consider an online learning setup where Φ is continuously updated from input streaming data, and at time t, the current mapping Φt is learned from $\{x1, . . . , xt\}$.
3. We follow the standard setup of learning Φ from pairs of instances with similar/dissimilar labels
4.These labels, along with the neighborhood structure, can be derived from a metric, *e.g*. two instances are labeled similar (*i.e*. neighbors of each other) if their Euclidean distance in X is below a threshold

Output:



CHAPTER 3

PROPOSED SYSTEM

3.1 Objective of Proposed Model

a. Different from OSupH(Online Supervised Hashing) and HCOH(Hadamard Codebook based Online Hashing) in which the target codes are generated heuristically and the hash functions are learned collectively according to the generated codes, in our method, the hash functions are learned independently. Hence, the effective target codes can be constructed by selecting the effective bits online to reduce the correlation between hash functions and improve the robustness of hash functions, while the target codes of OSupH and HCOH are generated heuristically only in advance and cannot be learned online. To our knowledge, the existing online hashing methods based on target codes do not support constructing target codes online according to the data.

b. A metric is proposed to select the effective binary bits to construct the target codes. The metric consists of two selection criteria. One selection criterion is designed to select the bits that can make the hash functions resistant to the bit flipping error, and the other one is designed to select the bits that can result in the low correlation between the hash functions. c. We compare our method with other online hashing methods in both a static environment and a dynamic environment,while in prior work online hashing methods are compared in either the static environment or the dynamic environment. Extensive experiments show that our method achieves better search accuracy than other online hashing methods.

3.2. Algorithms Used for Proposed Model

1. RSA (Rivest, Shamir and Adleman)

2. ONLINE HASHING ALGORITHM

3. BIT SELECTION ALGORITHM

1. RSA (Rivest, Shamir and Adleman)

RSA algorithm is a public key encryption technique and is considered as the most secure way of encryption. It was invented by Rivest, Shamir and Adleman in year 1978 and hence name RSA algorithm.

A hashing algorithm is a cryptographic hash function. It is a mathematical algorithm that maps data of arbitrary size to a hash of a fixed size.

Working of RSA Algorithm

1. Consider 2 huge prime numbers from image pixels
2. p, q such that $p \neq q$.
3. Calculate $n= p \times q$, $\phi(n) = (p-1) (q-1)$
4. Select e such that $\gcd(e,\phi(n))\equiv 1$.
5. Calculate d using Extended Euclidean algorithm (
6. $d \equiv e-1 \bmod (())$
7. Public key = {e, n}
8. Private key = {d, n}
9. Encryption .

ONLINE HASHING ALGORITHM

Online hashing is a technique for generating compact binary codes for images.
These codes can be used to efficiently search large image databases.
Online hashing has several benefits, including fast query times and low storage requirements.

BIT SELECTION ALGORITHM

Bit selection is the process of choosing the optimal number of bits for a hash code
The optimal number of bits depends on the size of the image database and the desired retrieval performance
Different bit selection methods can have a significant impact on image retrieval performance

SHA ALGORITHM

SHA stands for secure hashing algorithm. SHA is a modified version of MD5 and used for hashing data and certificates. A hashing algorithm shortens the input data into a smaller form that cannot be understood by using bitwise operations, modular additions, and compression functions.

Bit Selection Strategies

● Random Selection: A simple and effective strategy that selects bits randomly from the hash code Easy to implement and computationally efficient May not be optimal for complex datasets.

● **Mutual Information-based Selection:**

A strategy that selects bits based on their mutual information with the class labels Can improve retrieval performance for structured datasets May require additional supervision and computational resources.

Hashing Techniques

● **Supervised Hashing**: A technique that learns hash functions from labeled data

○ Requires labeled data for training

○ Can achieve high retrieval accuracy with sufficient training data

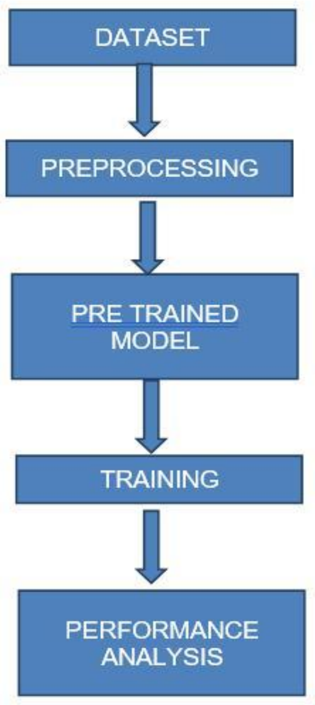
● **Unsupervised Hashing**: A technique that learns hash functions from unlabeled data

○ Does not require labeled data for training

○ May not achieve high retrieval accuracy without additional supervision

3.3. Designing

Working:



Use cases :

Image Server

In this module, the Image server has to login by using valid user name and password. After login successful he can do some operations such as View all images with its Description and ranking,View all images ranking,List all users and authorize,List end user transaction,View all similar images with similar hash sign,Find the distance between images,Find all view non similar images based on different hash sign.

Search History

This is controlled by admin; the admin can view the search history details. If he clicks on search history button, it will show the list of searched user details with their tags such as user name, user searched for image name, time and date.

Rank of images

In user's module, the admin can view the list of ranking images. If admin click on list of ranking images, then the server will give response with their tags image and rank of image.

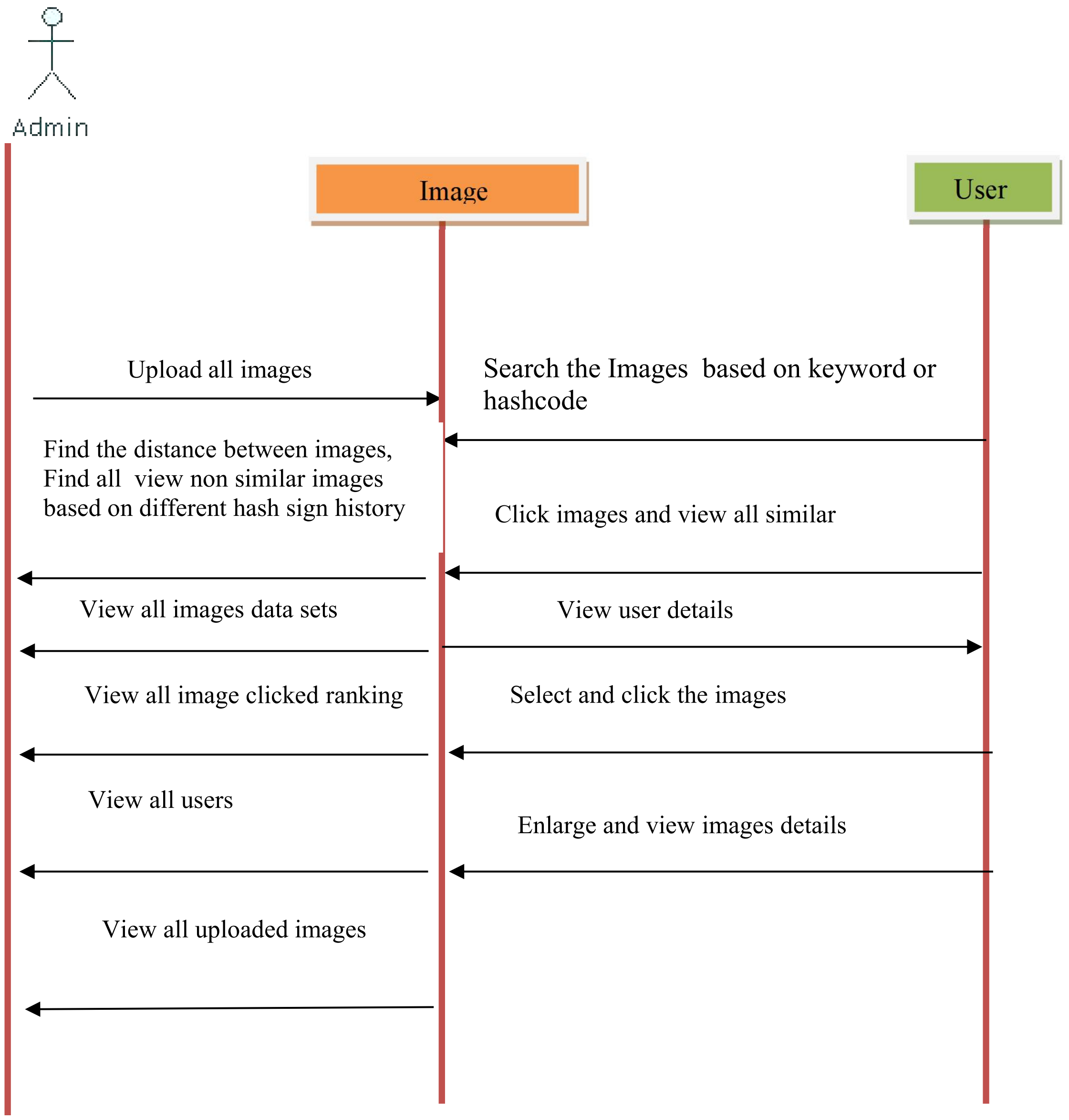
Upload Images

In this module, the admin can upload n number of images. Admin want to upload new image then he has enter some fields like image name, image color, image description, image type, image usage, browse the image file and upload. After uploading successfully he will get a response from the server. Initially new uploaded image rank is zero. After viewing that image rank will re-rank.

User

In this module, there are n numbers of users are present. User should register before doing some operations. And register user details are stored in user module. After registration successful he has to login by using authorized user name and password. Login successful he will do some operations like view my details, search images, request secrete key and logout. The user click on my details link then the server will give response to the user with all details such as user name, phone no, address, e mail ID and location. An end user can search the images based on **Images based on keyword or hash code** and gets the details like image name, image color, image usage and image type. And server will give response to the user, then that image rank will be increased.

3.3.1.UML DIAGRAM


Admin
Image
User
Upload all images
Search the Images based on keyword or hashcode
Find the distance between images, Find all view non similar images based on different hash sign history
Click images and view all similar
View all images data sets
View user details
View all image clicked ranking
Select and click the images
View all users
Enlarge and view images details
View all uploaded images

IMPLEMENTATION

USER :

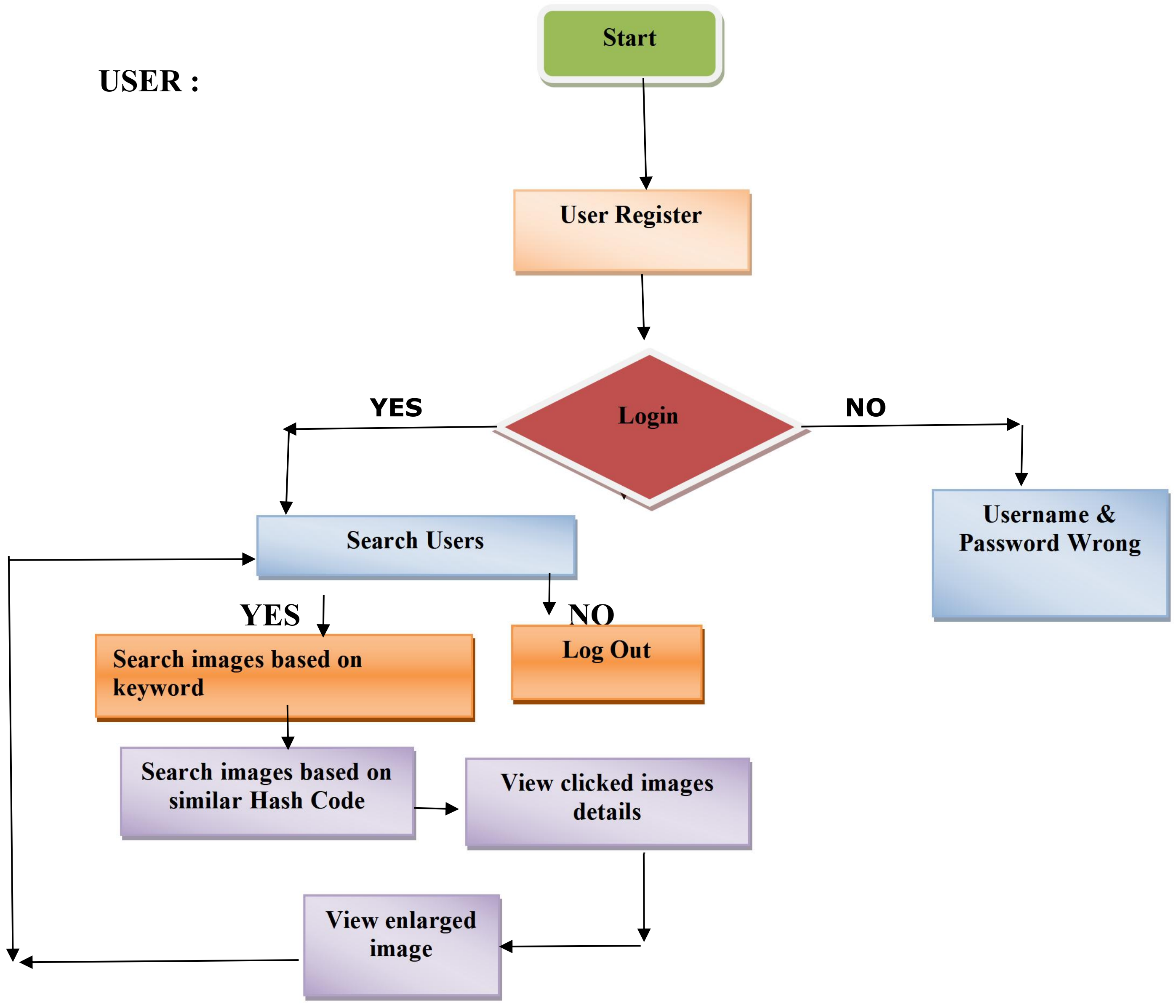
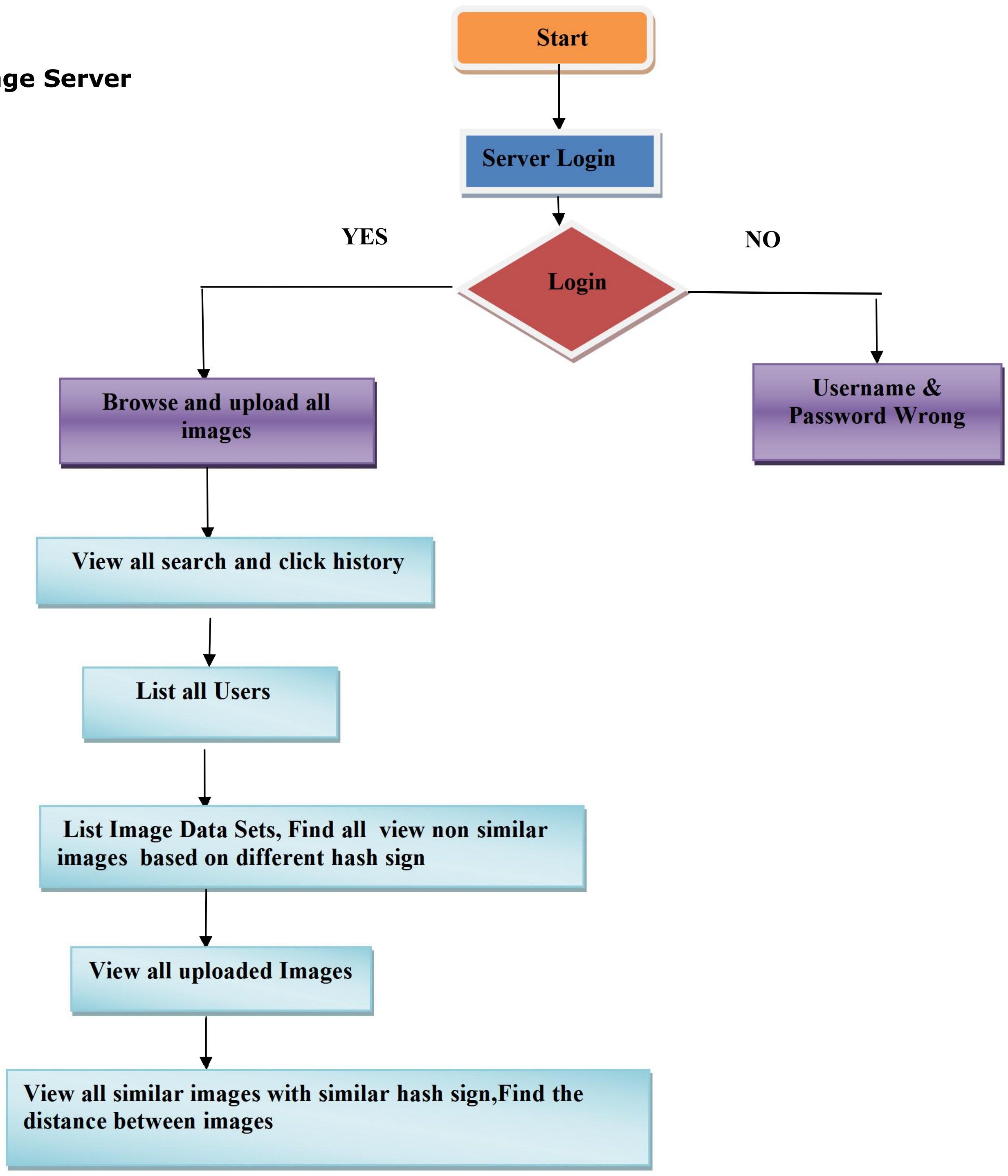

Start
User Register
Login
YES
NO
Username & Password Wrong
Search Users
YES
NO
Search images based on keyword
Log Out
Search images based on similar Hash Code
View clicked images details
View enlarged image

Image Server


Start
Server Login
YES
NO
Login
Browse and upload all images
Username & Password Wrong
View all search and click history
List all Users
List Image Data Sets, Find all view non similar images based on different hash sign
View all uploaded Images
View all similar images with similar hash sign,Find the distance between images

CODE :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Image Adding</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
<style type="text/css">
<!--
.style1 {color: #7FB600}
.style2 {color: #DD5411}
.style3 {font-size: 15px}
-->
</style>
</head>
<body>
<center>
  <div id="wh_bg">
    <div id="bd_bg">
      <!--top panel starts-->
      <div id="main1">
<div id="top"> <span class="what"> <img src="images/what1.gif" width="0.1" height="0.1"
alt="" class="wh_img" /> </span> <img src="images/logo1.jpg" width="284" height="60"
alt="" class="logo" />
          <div class="menu">
<a href="index.html">HOME</a> <img src="images/line.gif" width="3" height="13" alt=""
class="flt" style="margin-top:20px;" />
<a href="admin_Login.jsp" style="color:#7FB600;">IMAGE SERVER </a> <img
src="images/line.gif" width="3" height="13" alt="" class="flt" style="margin-top:20px;" />
<a href="user_Login.jsp">USER</a> <img src="images/line.gif" width="3" height="13"
alt="" class="flt" style="margin-top:20px;" />
<a href="project_Details.html"></a> </div>
        </div>
      </div>
 <div id="main2">
```

```
    <div id="wel_bg">  </div>
   </div>
   <!--top panel ends-->
   <!--content panel starts-->
   <div id="main3">
    <div id="con">
     <div id="content">
      <div id="left"> <h2 class="style1">Sidebar Menu</h2>
       <div class="flt" style="width:260px;padding:15px 0px 0px 9px;">

     <li><a href="admin_Main.jsp"style="color:#DD5411;">Admin Main</a> </li></br>
     <li><a href="admin_Login.jsp"style="color:#DD5411;">Log Out</a> </li></br>
        <p>&nbsp;</p>

       </div>

    <h2 class="style1">Main </h2>
      </div>
      <div id="right"> <h2 class="style2" >Upload Image Details . <span
class="style1"></span></h2>
       <div class="flt" style="width:590px;padding:15px 0px 0px 7px;">
        <p></p>
        <form action="admin_Add_Images1.jsp" method="post" enctype="multipart/form-
data">
        <table width="506" border="0"  cellpadding="0" cellspacing="0"  style="border-
collapse: collapse; display:inline; margin:10px 10px 10px 10px; font-family:Verdana, Arial,
Helvetica, sans-serif; font-size:14px;">

    <%@ page import ="java.util.*" %>
        <%@ include file="connect.jsp" %>
    <%
```

```
 try{
     ArrayList a1=new ArrayList();
     String str = " select * from titles ";
     Statement st = connection.createStatement();
     ResultSet rs = st.executeQuery(str);
     while(rs.next())
     {
     a1.add(rs.getString(2));
     }
     %>

     <tr>
        <td  width="188" valign="middle" height="45" style="color: #2c83b0;"><div
align="left" class="style3 style7 style1" style="margin-left:20px;"><strong> Select Title
</strong></div></td>
        <td  width="318" valign="middle" height="45" style="color:#000000;"><div
align="left" style="margin-left:20px;">



     <select id="s1" name="title">
      <option>--Select--</option>
     <% for(int i=0;i<a1.size();i++)
        {
        %>
       <option><%= a1.get(i)%></option>

       <%}%>
     </select>
   </div>



                  <div align="right"> New <a

href="admin_Add_Titles.jsp">Titles</a>?</div></td>

       </td>
      </tr>


     <tr>
```

```
      <td  width="188" valign="middle" height="45" style="color: #2c83b0;"><div
align="left" class="style3 style7 style1" style="margin-left:20px;"><strong> Name
</strong></div></td>
      <td  width="318" valign="middle" height="45" style="color:#000000;"><div
align="left" style="margin-left:20px;"><input type="text" name="name"></div></td>
     </tr>
     <tr>
      <td  width="188" valign="middle" height="45" style="color: #2c83b0;"><div
align="left" class="style3 style7 style1" style="margin-left:20px;"><strong>  Color
</strong></div></td>
      <td  width="318" valign="middle" height="45" style="color:#000000;"><div
align="left" style="margin-left:20px;"><input type="text" name="color"></div></td>
     </tr>
     <tr>
       <td  width="188" valign="middle" height="45" style="color: #2c83b0;"><div
align="left" class="style3 style7 style1" style="margin-left:20px;"><strong> Description <br
/>
         </strong></div></td>
      <td  width="318" valign="middle" height="45"><div align="left" style="margin-
left:20px;">
       <textarea name="desc"></textarea>
      </div></td>
     </tr>
     <tr>
      <td  width="188" align="left" valign="middle" height="45" style="color:
#2c83b0;"><div align="left" class="style3 style7 style1" style="margin-left:20px;"><strong>
Uses </strong></div></td>
      <td  width="318" align="left" valign="middle" height="45"><div align="left"
style="margin-left:20px;"><input type="text" name="uses"></div></td>

         <span class="style8 style1"><strong>
          </tr>
         </strong></span>
    <tr>
```

```
          <td   width="188" align="left" valign="middle" height="74" style="color:
#2c83b0;"><div align="left" class="style3 style8 style1" style="margin-
left:20px;"><strong>Select Image </strong></div></td>
          <td  width="318" align="left" valign="middle" height="74" style="color: #2c83b0;">
          <div align="left" style="margin-left:20px;"><input type="file"
name="pic"></div></td>
        </tr>
        <div > <tr><td height="45" colspan="2" id="learn_more"
align="center"  style="color:#FFFFFF;"><input type="submit" value="Submit"
style="width:100px; height:25px; background-color:#000000;
color:#FFFFFF;"/>&nbsp;&nbsp;<input type="reset" name="Reset" style="width:100px;
height:25px; background-color:#000000; color:#FFFFFF;"/></td></tr></div>
      </table>
       </form>
       <%



         connection.close();
        }

        catch(Exception e)
        {
          out.println(e.getMessage());
        }
%>
            </p>
             <p>&nbsp;</p>
             <p align="right" class="style3"><a href="admin_Main.jsp">Back</a></p>
             <p align="right" class="style3">&nbsp;</p>
           </div>
           <br/>
       <h2 class="style2" >&nbsp;</h2>
          </div>
        </div>
```

```
 </div>
      </div>
      <!--content panel ends-->
      <!--footer panel starts-->
      <div id="main4">

      </div>
      <!--footer panel ends-->
    </div>
  </div>
</center>
</body>
</html>
```

```
<title>User Profile</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
<style type="text/css">
<!--
.style1 {color: #7FB600}
.style2 {color: #DD5411}
.style3 {font-size: 15px}
.style4 {font-size: 14px}
.style5 {color: #000000}
-->
</style>
</head>
<body>
<center>
  <div id="wh_bg">
    <div id="bd_bg">
      <!--top panel starts-->
      <div id="main1">
<div id="top"> <span class="what"> <img src="images/what1.gif" width="0.1" height="0.1"
alt="" class="wh_img" /> </span> <img src="images/logo1.jpg" width="284" height="60"
alt="" class="logo" />
          <div class="menu">
<a href="index.html">HOME</a> <img src="images/line.gif" width="3" height="13" alt=""
class="flt" style="margin-top:20px;" />
<a href="admin_Login.jsp">IMAGE SERVER </a> <img src="images/line.gif" width="3"
height="13" alt="" class="flt" style="margin-top:20px;" />
<a href="user_Login.jsp" style="color:#7FB600;">USER</a> <img src="images/line.gif"
width="3" height="13" alt="" class="flt" style="margin-top:20px;" /></div>
```

```
    </div>
   </div>
   <div id="main2">
    <div id="wel_bg">  </div>
   </div>
   <!--top panel ends-->
   <!--content panel starts-->
   <div id="main3">
    <div id="content">
     <div id="content">
      <div id="left"> <h2 class="style1">Sidebar Menu</h2>
       <div class="flt" style="width:260px;padding:15px 0px 0px 9px;">

                                    <li><a    href="user_Main.jsp"style="color:#DD5411;">User
Main</a> </li></br>
                              <li><a href="user_Login.jsp"style="color:#DD5411;">Log Out</a>
</li></br>
        <p>&nbsp;</p>

       </div>

                            <h2 class="style1">Main </h2>
      </div>
      <div    id="right">    <h2    class="style2"    >User    :<span    class="style1">
<%=(String)application.getAttribute("user")%></span> Profile</h2>
       <div class="flt" style="width:590px;padding:15px 0px 0px 7px;">
        <p>

        </p>
        <p></p>
        <p><table    width="547"    border="1.5"    align="center"        cellpadding="0"
cellspacing="0"  >



<%@ include file="connect.jsp" %>
<%@ page import="org.bouncycastle.util.encoders.Base64"%>

                              <%

                                    String
user=(String )application.getAttribute("user");

                                    String s1,s2,s3,s4,s5;
                                    int i=0;
                                    try
                                    {
                                          String  query="select  *  from  user  where
username='"+user+"'";
                                          Statement
st=connection.createStatement();
                                          ResultSet rs=st.executeQuery(query);
                                          if ( rs.next() )
                                          {
```

```
 i=rs.getInt(1);
s1=rs.getString(4);
s2=rs.getString(5);
s3=rs.getString(6);
s5=rs.getString(7);
s4=rs.getString(10);
%>
<tr>
<td width="230" rowspan="6" >
<div class="style7" style="margin:10px 13px 10px 13px;" ><a class="#" id="img1"
href="#" >
<input  name="image" type="image" src="user_Pic.jsp?id=<%=i%>" style="width:200px;
height:200px;" />
                                                </a></div>
        </td>
        </tr>

        <tr>
  <td  width="145" valign="middle" height="40" style="color: #2c83b0;"><div align="left"
class="style14 style7 style15 style20 style8 style9 style2 style3" style="margin-
left:20px;"><strong>E-Mail</strong></div></td>
                                              <td   width="164" valign="middle" height="40"
style="color:#000000;"><div align="left" class="style23 style9 style10 style4 style5"
style="margin-left:20px;"><%out.println(s1);%></div></td>
                                          </tr>
                                          <tr>
                                              <td   width="145" valign="middle" height="40"
style="color: #2c83b0;"><div align="left" class="style14 style7 style15 style20 style8 style9
style2 style3" style="margin-left:20px;"><strong>Mobile</strong></div></td>
```

```
<td width="164" valign="middle" height="40"><div align="left" class="style23 style9 style10 style4 style5" style="margin-left:20px;"><%out.println(s2);%></div></td>
                                        </tr>
                                        <tr>
<td width="145" align="left" valign="middle" height="40" style="color: #2c83b0;"><div align="left" class="style14 style7 style15 style20 style8 style9 style2 style3" style="margin-left:20px;"><strong>Address</strong></div> </td>
<td width="164" align="left" valign="middle" height="40"><div align="left" class="style23 style9 style10 style4 style5" style="margin-left:20px;"><%out.println(s3);%></div></td>
                                        </tr>
                                        <tr>
<td width="145" align="left" valign="middle" height="40" style="color: #2c83b0;"><div align="left" class="style14 style7 style15 style20 style8 style9 style2 style3" style="margin-left:20px;"><strong>Date of Birth</strong></div> </td>
<td width="164" align="left" valign="middle" height="40"><div align="left" class="style23 style9 style10 style4 style5" style="margin-left:20px;"><%out.println(s5);%></div></td>
                                        </tr>
                                        <tr>
                                            <td width="145" align="left" valign="middle" height="51" style="color: #2c83b0;"><div align="left" class="style14 style7 style15 style20 style8 style9 style2 style3" style="margin-left:20px;"><strong>Status</strong></div ></td>
                                                <td width="164" align="left" valign="middle" height="51" style="color: #2c83b0;"><div align="left">
                                          <div align="left" class="style23 style9 style10 style4 style5" style="margin-left:20px;"><%out.println(s4);%></div></td>
                                        </tr>
```

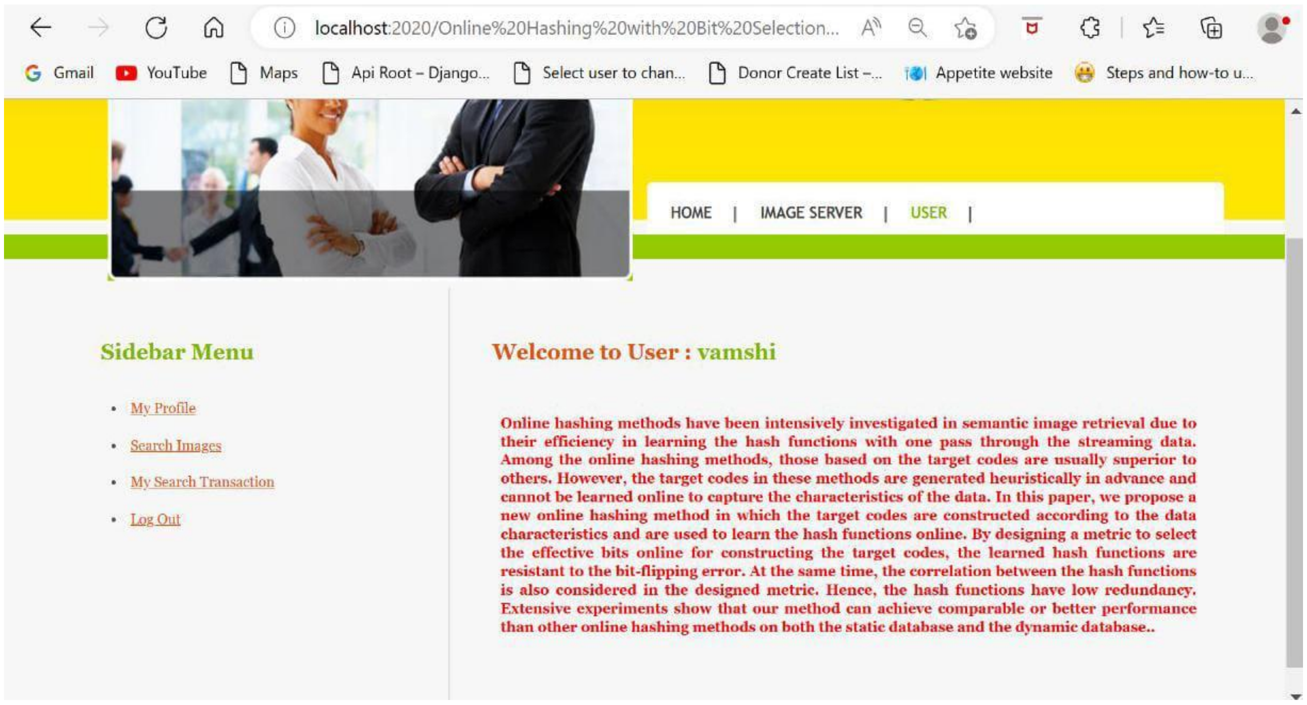
```
<%
        }
connection.close();
}
catch(Exception e)
                {
out.println(e.getMessage());
                }
%>
</table></p>
          <p>&nbsp;</p>
          <p align="right" class="style3"><a href="user_Main.jsp">Back</a></p>
          <p>&nbsp;</p>

         </div>
         <img    src="images/cn_bar.gif"    width="605"    height="1"    alt=""    class="flt"
style="margin-top:15px;" />
                         <br/>
                         <h2 class="style2" >Image </h2>
        </div>
       </div>
      </div>
    </div>
    <!--content panel ends-->
    <!--footer panel starts-->
    <div id="main4">
<html>
```

CHAPTER-4

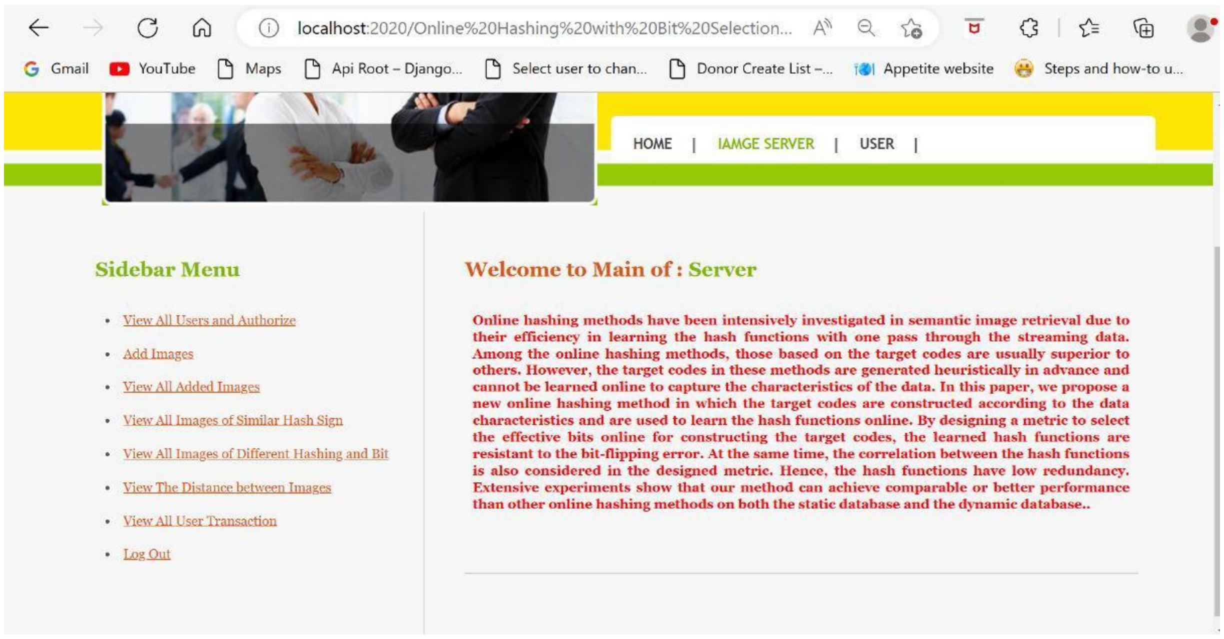
RESULTS & DISCUSSION

Results



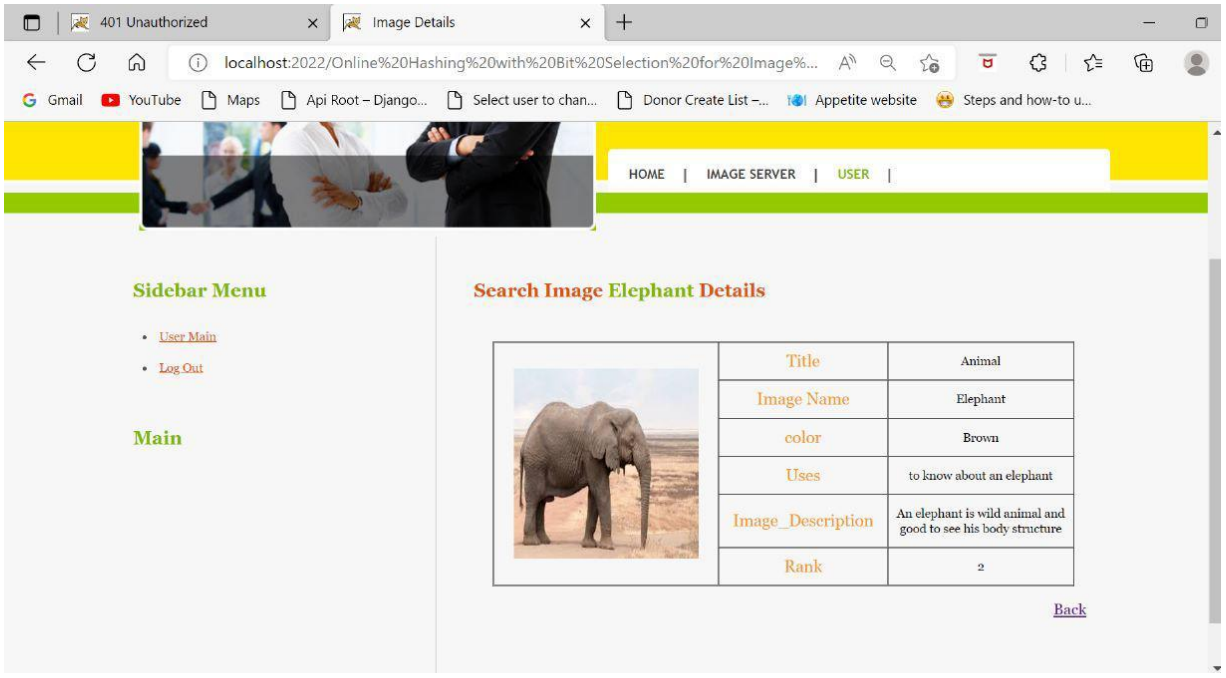
Welcome to User : vamshi

Online hashing methods have been intensively investigated in semantic image retrieval due to their efficiency in learning the hash functions with one pass through the streaming data. Among the online hashing methods, those based on the target codes are usually superior to others. However, the target codes in these methods are generated heuristically in advance and cannot be learned online to capture the characteristics of the data. In this paper, we propose a new online hashing method in which the target codes are constructed according to the data characteristics and are used to learn the hash functions online. By designing a metric to select the effective bits online for constructing the target codes, the learned hash functions are resistant to the bit-flipping error. At the same time, the correlation between the hash functions is also considered in the designed metric. Hence, the hash functions have low redundancy. Extensive experiments show that our method can achieve comparable or better performance than other online hashing methods on both the static database and the dynamic database..


localhost:2020/Online%20Hashing%20with%20Bit%20Selection...
Gmail
YouTube
Maps
Api Root – Django...
Select user to chan...
Donor Create List –...
Appetite website
Steps and how-to u...
HOME | IAMGE SERVER | USER |
Sidebar Menu
View All Users and Authorize
Add Images
View All Added Images
View All Images of Similar Hash Sign
View All Images of Different Hashing and Bit
View The Distance between Images
View All User Transaction
Log Out

Welcome to Main of : Server

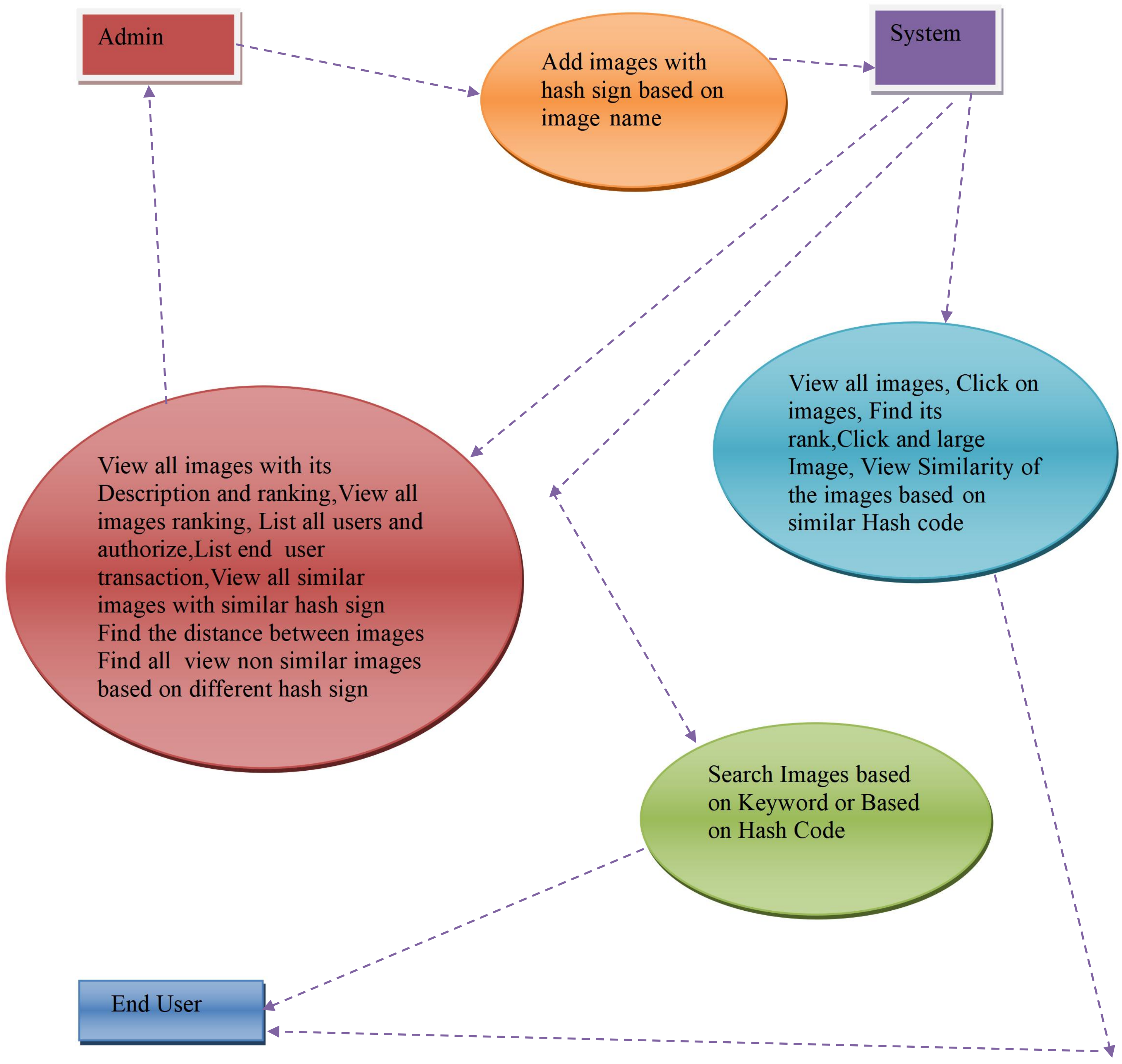
Online hashing methods have been intensively investigated in semantic image retrieval due to their efficiency in learning the hash functions with one pass through the streaming data. Among the online hashing methods, those based on the target codes are usually superior to others. However, the target codes in these methods are generated heuristically in advance and cannot be learned online to capture the characteristics of the data. In this paper, we propose a new online hashing method in which the target codes are constructed according to the data characteristics and are used to learn the hash functions online. By designing a metric to select the effective bits online for constructing the target codes, the learned hash functions are resistant to the bit-flipping error. At the same time, the correlation between the hash functions is also considered in the designed metric. Hence, the hash functions have low redundancy. Extensive experiments show that our method can achieve comparable or better performance than other online hashing methods on both the static database and the dynamic database..



Search Image Elephant Details

	Title	Animal
	Image Name	Elephant
	color	Brown
	Uses	to know about an elephant
	Image_Description	An elephant is wild animal and good to see his body structure
	Rank	2

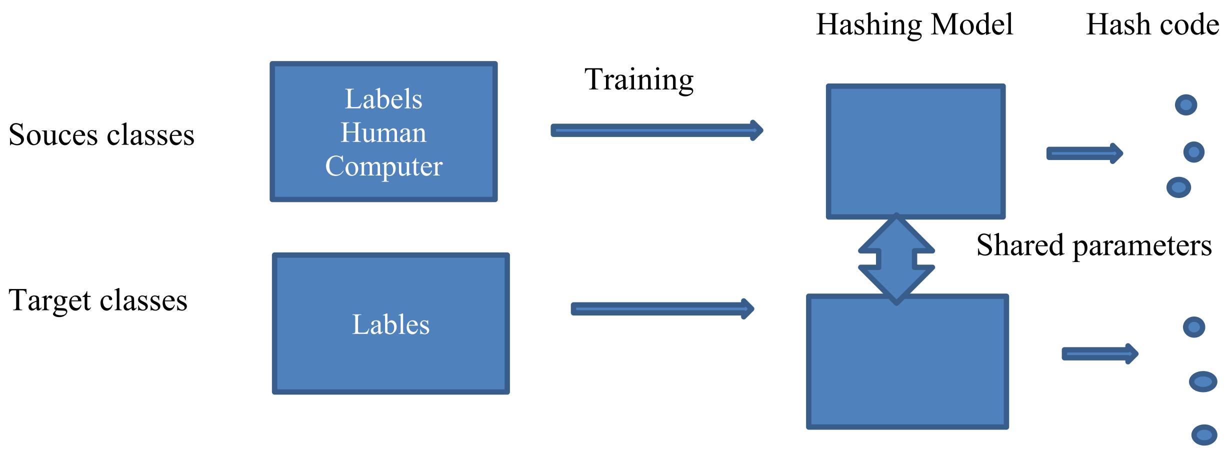
4.1. Data Collection and Performance metrics


Admin
Add images with hash sign based on image name
System
View all images with its Description and ranking,View all images ranking, List all users and authorize,List end user transaction,View all similar images with similar hash sign Find the distance between images Find all view non similar images based on different hash sign
View all images, Click on images, Find its rank,Click and large Image, View Similarity of the images based on similar Hash code
Search Images based on Keyword or Based on Hash Code
End User

- Precision: The fraction of retrieved images that are relevant to the query.
- Recall: The fraction of relevant images that are retrieved by the system.
- F1 Score: The harmonic mean of precision and recall.
- These metrics are used to evaluate the effectiveness of online hashing with bit selection for image retrieval.

- various performance metrics that can be used to evaluate predictions for classification problems.

1. Confusion Matrix
2. Accuracy
3. Precision
4. Recall



1.CONFUSION MATRIX

Total samples tested= 165

Predicted Values

Actual Values

TP=100	FP=5
FN=100	TN=50

2. **ACCURACY** = $\frac{TP+TN}{T.S}$

$= \frac{100+50}{165}$

$= 0.91$

3. **PRECISION** =

$\frac{TP}{TP+FP}$

$= \frac{100}{100+5}$

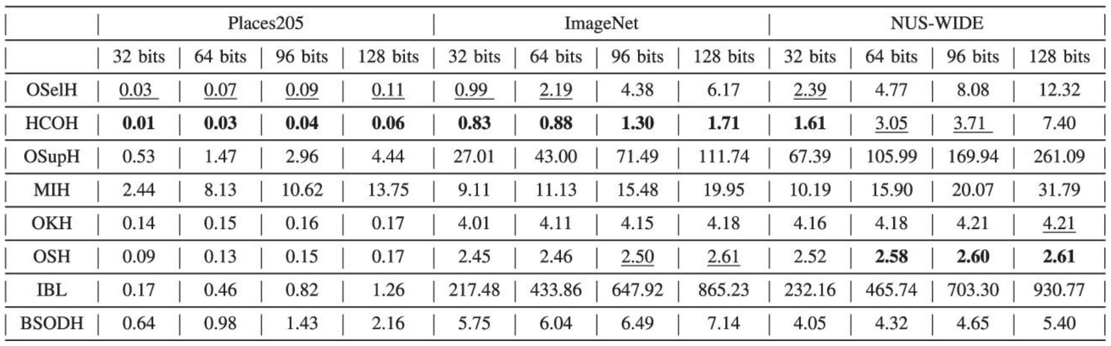
$= 0.95$

6. **RECALL** =

$\frac{TP}{TP+FP}$

$= \frac{100}{100+10}$

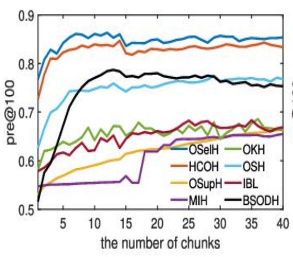
$= 0.909$



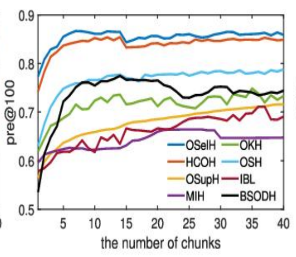
	Places205				ImageNet				NUS-WIDE			
	32 bits	64 bits	96 bits	128 bits	32 bits	64 bits	96 bits	128 bits	32 bits	64 bits	96 bits	128 bits
OSelH	0.03	0.07	0.09	0.11	0.99	2.19	4.38	6.17	2.39	4.77	8.08	12.32
HCOH	**0.01**	**0.03**	**0.04**	**0.06**	**0.83**	**0.88**	**1.30**	**1.71**	**1.61**	3.05	3.71	7.40
OSupH	0.53	1.47	2.96	4.44	27.01	43.00	71.49	111.74	67.39	105.99	169.94	261.09
MIH	2.44	8.13	10.62	13.75	9.11	11.13	15.48	19.95	10.19	15.90	20.07	31.79
OKH	0.14	0.15	0.16	0.17	4.01	4.11	4.15	4.18	4.16	4.18	4.21	4.21
OSH	0.09	0.13	0.15	0.17	2.45	2.46	2.50	2.61	2.52	**2.58**	**2.60**	**2.61**
IBL	0.17	0.46	0.82	1.26	217.48	433.86	647.92	865.23	232.16	465.74	703.30	930.77
BSODH	0.64	0.98	1.43	2.16	5.75	6.04	6.49	7.14	4.05	4.32	4.65	5.40

4.2

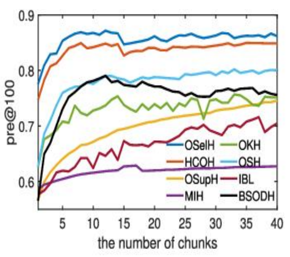
Map-Curve



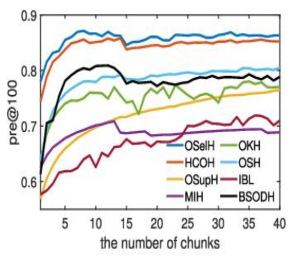
(a) 32 bits



(b) 64 bits



(c) 96 bits



(d) 128 bits

CHAPTER-5

CONCLUSION

CHAPTER-5

CONCLUSION

Different from other online hashing methods that generate the target codes heuristically in advance and learn the hash functions collectively according to the target codes, our proposed online supervised hashing method learns the hash
functions independently. As the hash functions are learned independently, the target codes can be constructed online by selecting the effective bits, and the hash functions can be learned according to the constructed target codes. A metric is designed to select the hash functions that have high robustness to resist the bit-flipping error and have low redundancy.

The experiments show that the proposed method can achieve comparable or better performance compared with other online hashing methods on both the static database and the dynamic database.

FUTURE ENHANCEMENT

Online hashing with bit selection faces several challenges, including reduced accuracy compared to other methods and the need for careful selection of performance metrics.Future directions for research in this area include the development of new online hashing algorithms and bit selection techniques and the exploration of new applications for online hashing with bit selection.

REFERENCES

REFERENCES

[1] A. Oliva and A. Torralba, “Modeling the shape of the scene: A holistic representation of the spatial envelope,” *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.
[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton,“Imagenet classifification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.
[3] C. Silpa-Anan and R. Hartley, “Optimised kd-trees for fast image descriptor matching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.
[4] J. Tang, Z. Li, and X. Zhu, “Supervised deep hashing for scalable face image retrieval,” *Pattern Recognition*, vol. 75, pp. 25 – 32, 2018.
[5] C. Deng, Z. Chen, X. Liu, X. Gao, and D. Tao, “Triplet-based deep hashing network for cross-modal retrieval,” *IEEE Trans. on Image Processing*, vol. 27, no. 8, pp. 3893–3903, Aug 2018.
[6] Q. Jiang, X. Cui, and W. Li, “Deep discrete supervised hashing,” *IEEE Trans. on Image Processing*, vol. 27, no. 12, pp. 5996–6009, Dec 2018.
[7] Y. Cao, B. Liu, M. Long, and J. Wang, “Hashgan: Deep learning to hash with pair conditional wasserstein gan,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2018, pp. 1287–1296.
[8] E. Yang, C. Deng, C. Li, W. Liu, J. Li, and D. Tao, “Shared predictive cross-modal deep quantization,” *IEEE Trans. on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5292–5303, Nov 2018.
[9] F. Cakir, S. A. Bargal, and S. Sclaroff, “Online supervised hashing,” *Computer Vision and Image Understanding*, vol. 156, pp. 162–173, 2017.

GitHub Link :

https://vamshikandukala.github.io/Hashing-bit-/