VIRGINIA TECH

FlickrIDR

A web-based multimodal search interface based on the SuperIDR

Client: Uma Murthy
Class: CS4624
University: Virginia Tech
City: Blacksburg, VA

Kaslin Fields, James Kaplan, Martin Overstrom

5/8/2013

Table of Contents

1.0 Executive Summary

In many educational fields, it is important to be able to recognize by sight, such as identifying a fish's species in wildlife sciences. To develop these types of skills, it is important to be able to look at databases of images pertaining to the topic of study and to be able to examine relationships of images. For example, if one is doing an assignment on identifying a fish's species by its picture, it may be useful to be able to search for images similar to the one given, taking note of certain unique aspects. The FlickrIDR allows a user to specify a Flickr Image Group pertaining to their field of interest and to search through that group for images similar to a given image and/or text. The goal of this project is to develop a web-based interface and system to enable multi-modal (image and annotation) search on Flickr image collections. The project will build upon the SuperIDR prototype and will provide SuperIDR's functionality, focusing on search, in a web-based environment.

The project was separated into phases. The first phase consisted of doing research regarding the Flickr API and Flickr groups. In this phase, the Flickr API was used to gather information about Flickr groups including total number of images, number of images with annotations, and average number of annotations per image. The second phase involved creating a website as well as enabling searching of the Flickr SuperIDR image group. This was the most time and effort-intensive portion, as the website had to be designed and implemented as well as the ability to search a Flickr image group. In the third phase, the Flickr image group search capability was to be generalized. Whereas the website in phase 2 could search only the SuperIDR image group, the goal of phase 3 was to be able to indicate any Flickr image group for indexing so that any indexed group would be used in searches. As stretch goals, the project was to be released as open source, and with any extra time, research would be done on improving search algorithms.

The team was able to complete effectively complete the 3 primary phases, with some work still left to be done. Phase 1 was completely finished on-schedule after only a few weeks. Phase 2 was completed, though later than anticipated, and also the UI designed and developed in this stage was inadequate for the functionality to be provided in phase 3. The web development proved to be quite difficult due to the need to use javascript to implement subimage selection capabilities, which are still not fully functional. Phase 3 further required of the UI that the user be able to specify a Flickr image group for indexing. This change to the UI was not implemented. The functionality to accomplish this task exists in the code base, however it will need to be integrated with the UI to allow the user to dynamically select groups for indexing. The general functionality of indexing a Flickr image group beyond the Super IDR image group is complete. Although the ability for the user to specify that image group is not complete.

The client's goal was sufficiently fulfilled; however there is still a significant amount of work left to be completed in the future. For example, due to unexpected difficulties in learning new technologies during phase 2, the schedule had to be altered, leaving no time to create a sufficient automated test suite. The subimage selection in the user interface requires further work, as does image selection via url. The UI and backend functionality need to be updated to enable the user to specify Flickr groups for indexing, though the backend code does have the capability to index Flickr groups. It may also be interesting to consider updating indexed Flickr groups and recognizing groups which have already been indexed to save time, as the indexing process is very time and bandwidth intensive. This project serves

as a proof of concept and base for further development in creating a tool which allows users the kind of multi-modal search capabilities available in the SuperIDR in an easy-to-use and generic web interface.

2.0 User's Manual

2.1 Overview

Flickr IDR is a web-based tool that allows a user to perform search queries on groups of Flickr-hosted images. It supports search by image similarity and by text keywords. In addition to the web component, Flickr IDR has a command-line-driven indexing program, which performs the initial indexing on Flickr image groups that is necessary to enable search of that group using the web application.

2.2 System Requirements and Setup

2.2.1 Necessary installed software

Flickr IDR requires the Microsoft .NET framework version 3.5 or later to run. The web application requires a web server capable of running ASP.NET applications (usually Microsoft Internet Information Services web server)

2.2.2 External Libraries

Flickr IDR utilizes CBISC, Lucene, and Flickr NET. These are distributed in the bin directory of the application.

2.2.3 Setup

Extract the files in \FlickrIDR to a folder configured as an application on your web server.
Edit web.config and change the value attribute of "appRootFolder" to the path to the "flickrdata" folder on your computer
If you will be using the import tool, also perform the same change to the App.config file located in FlickrIDRImport/FlickrIDRImport.
If you will be using the import tool, update the "flickrApiKey" and "flickrApiSecret" to the relevant values from your Flickr account.

2.3 Use of Indexing Tool

Build and run the FlickrIDRImport application, or run the precompiled FlickrIDRImport.exe
Required arguments to FlickrIDRImport in order:
[USER or GROUP] - to indicate whether you want to index photos from a particular Flickr user or Flickr image group.
ID – the Flickr ID of the user or group you want to index
NAME- the name that will be displayed to describe the image group in the web interface. Enclose in quotes for multiple words
LIMIT (optional) – optionally specify a maximum number of images to index

2.4 Use of web application

Select an image group from the dropdown (freshwater fish and mineral specimens are currently available)

Optionally specify an image to use as a search query by selecting a file from your local machine, or enter a URL of a web-hosted image
Optionally use the crop tool by clicking and dragging to select the desired portion of the image.
Optionally enter text keywords to the box
Select the weighting of image vs. text search results, if you've entered queries for both.
Click the 'Search' button
View search results and click on the thumbnail images to visit the image's Flickr page

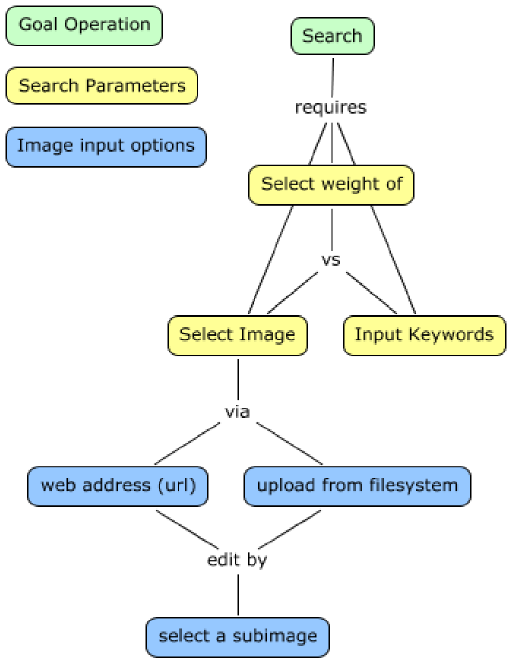

Figure 6.1: Use Flow Diagram

3.0 Developer's Manual

3.1 FlickrIDR Package

The FlickrIDR package contains files to provide the search and webpage functionalities. This includes the html and javascript documents as well as C# code to implement searching using Lucene and CBISC according to user input. The following is an overview of each of the folders and files in the package.

3.1.1 data

Local SuperIDR image database

3.1.2 images

Static images contained on webpages

3.1.3 js

3.1.3.1 jquery.color.cs

3.1.3.2 jquery.crop.cs

image cropping functions

3.1.3.3 jquery.min.cs

jquery include

3.1.4 styles

css to support page styling

3.1.4.1 jquery.Jcrop.css

necessary for Jcrop library

3.1.4.2 style.css

style and formatting for webpage

3.1.5 CBISCInst.cs

Singleton which holds reference to CBISC instance

3.1.6 Default.aspx

Allows user input to search functions. Contains Javascript interface supporting interactive image selection and cropping

3.1.6.1 Default.aspx.cs

Gathers user input from search page and makes it available for retrieval by the results page

3.1.7 LuceneManager.cs

3.1.7.1 FieldKeys

Holds constants containing Lucene field keys

3.1.7.2 LuceneManager

Acts as a singleton for a Lucene instance
Supports photo and annotation data import
Supports photo data retrieval
Supports retrieval of text search results

3.1.8 results.aspx

Page for display of search results

3.1.8.1 results.aspx.cs

Retrieves user input from search page and performs search

3.1.9 Search.cs

Determines if image search, text search, or both are needed
Performs searches with CBISC and Lucene, saving image ids in a map structure with their corresponding scores, combining results from image and text searches
Retrieves photo data from Lucene, converts to data table, and sorts by score
Returns data table of results

3.1.10 Web.config

Various configuration settings for the web page.
Includes setting address of CBISC and Lucene information in file system

3.2 FlickrIDRImport Package

This package contains the functionality required for indexing Flickr groups to be used in searches.

3.2.1 App.config

Various configuration settings for the application (including CBISC and Lucene address)

3.2.2 FlickrImporter.cs

Supports indexing of Flickr user accounts or photo groups
Utilizes Flickr NET API wrapper
Downloads images and feeds to CBISC
Loads text data and feeds to Lucene

3.2.3 Program.cs

Handles command-line input and initiates indexing

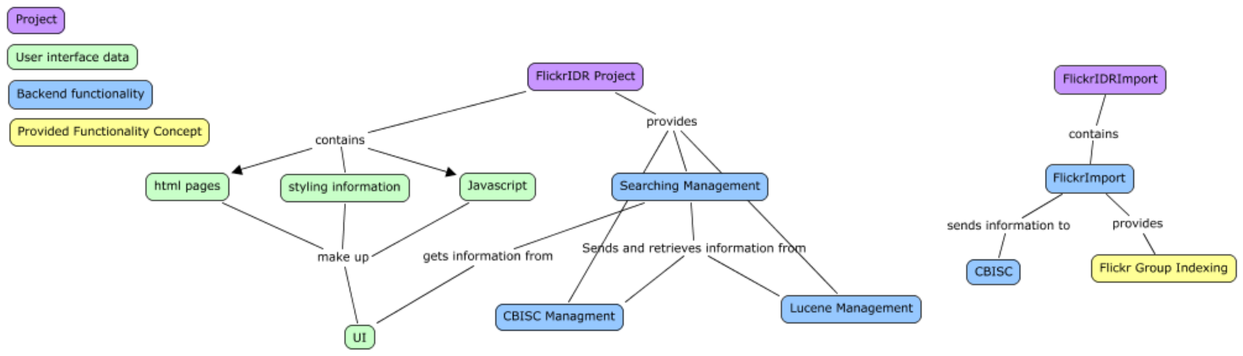

Figure 7.1: Project Diagrams

4.0 Tester's Manual

4.1 Normal Use Case

4.1.1 Use Case

The user selects an image to be used as a query as either an image file or a url. The user opens the FlickrIDR web page and selects their image via their preferred method (url or file upload). After the image is selected, a pop-up asks the user to select a subimage. The user selects the portion of the image they are interested in matching. In the "Keywords" textbox, the user enters search terms in the form of key words separated by commas. Upon confirming that his/her input is correct, the user clicks the search button. After examining the results of the search, the user selects an appropriate image which takes them to that image on Flickr.

4.2 Test Prep

4.2.1 About tests

Click the search button at the end of each test case. Numbers indicate steps. *Italicized* lines indicate a full test case. This test plan is fairly comprehensive.

4.2.2 Output

Each test case should produce image results on the results page unless otherwise stated.

4.2.3 Preparation

Select two images to be used in testing from http://www.flickr.com/photos/superidr/. Also select a subimage from each image to be used where testing calls for selection of a subimage. Use keywords such as "fin, gills".

4.2.4 Total

There are a total of 30 tests described in section 3.0.

4.3 Test Cases

4.3.1 Flickr Group Indexing

Indexing groups will allow the FlickrIDR to include the images from the group when searching for results. Choose one or more images (as the case requires) from a Flickr group which matches the requirements stated below. After indexing that Flickr group, you may use that image in the test cases in sections 4.3.2 – 4.3.7.

4.3.1.1 ***Index a small Flickr group, ie ~300-1000 images***

4.3.1.2 ***Index a large Flickr group, ie ~10,000 images***

4.3.2 Combinations

Both Flickr groups should be indexed for searching.

4.3.2.1 ***Index a large, then a small Flickr group***

4.3.2.2 ***Index a small, then a large Flickr group***

4.3.3 Doubles

Both Flickr groups should be indexed for searching

4.3.3.1 ***Index a large, then a different large Flickr group***

4.3.3.2 ***Index a small, then a different small Flickr group***

4.3.4 Repeats

The system should recognize that the group has already been indexed and only add images that were not in the group when it was last indexed.

4.3.4.1 ***Index a large Flickr group, then index the same Flickr group again***

4.3.4.2 ***Index a small Flickr group, then index the same Flickr group again***

4.3.5 From Flickr

4.3.5.1 **With Subimage**

4.3.5.1.1 Enter the url of the image

4.3.5.1.2 Select a subimage

4.3.5.1.3 ***With Keywords***

4.3.5.1.3.1 Enter keywords

4.3.5.1.4 ***Without Keywords***

4.3.5.1.4.1 Do not enter keywords

4.3.5.2 **Without Subimage**

4.3.5.2.1 Enter the url of the image

4.3.5.2.2 Do not select a subimage

4.3.5.2.3 ***With Keywords***

4.3.5.2.3.1 Enter keywords

4.3.5.2.4 ***Without Keywords***

4.3.5.2.4.1 Do not enter keywords

4.3.6 From Upload

4.3.6.1 **With Subimage**

4.3.6.1.1 Select the image file after clicking the “Browse” button

4.3.6.1.2 Select a subimage

4.3.6.1.3 ***With Keywords***

4.3.6.1.3.1 Enter keywords

4.3.6.1.4 ***Without Keywords***

4.3.6.1.4.1 Do not enter keywords

4.3.6.2 **Without Subimage**

4.3.6.2.1 Select the image file

4.3.6.2.2 Do not select a subimage

4.3.6.2.3 ***With Keywords***

4.3.6.2.3.1 Enter keywords

4.3.6.2.4 ***Without Keywords***

4.3.6.2.4.1 Do not enter keywords

4.3.7 Without Image, With Keywords

4.3.7.1 Do not upload an image

4.3.7.2 Do not select a subimage

4.3.8 No Input

The system should return an error.

4.3.9 Multiple File Submissions

Expected result for this section is that the most recently selected image should persist unless stated otherwise.

4.3.10 Repeated Upload

4.3.10.1 ***With Subimages***

4.3.10.1.1 Select an image for upload

4.3.10.1.2 Select a subimage

4.3.10.1.3 Select a different image for upload

4.3.10.1.4 Select a subimage

4.3.10.2 ***Without Subimages***

4.3.10.2.1 Select an image for upload

4.3.10.2.2 Select a different image for upload

4.3.10.3 **Subimage Combination**

4.3.10.3.1 ***Subimage First***

4.3.10.3.1.1 Select an image for upload

4.3.10.3.1.2 Select a subimage

4.3.10.3.1.3 Select a different image for upload

4.3.10.3.2 ***Subimage Last***

4.3.10.3.2.1 Select an image for upload

4.3.10.3.2.2 Select a different image for upload

4.3.10.3.2.3 Select a subimage

4.3.10.4 **Repeated URL**

Same steps as 4.3.6.1, but use a URL for every image instead of uploading.

4.3.10.5 **Repeated Combination**

Follow the following steps, and then repeat using a URL where an upload is called for and an upload where a url is called for.

4.3.10.5.1 ***With Subimages***

4.3.10.5.1.1 Select an image for upload

4.3.10.5.1.2 Select a subimage

4.3.10.5.1.3 Select a different image via url

4.3.10.5.1.4 Select a subimage

4.3.10.5.2 ***Without Subimages***

4.3.10.5.2.1 Select an image for upload

4.3.10.5.2.2 Select a different image via url

4.3.10.5.3 **Subimage Combination**

4.3.10.5.3.1 ***Subimage First***

4.3.10.5.3.1.1 Select an image for upload

4.3.10.5.3.1.2 Select a subimage

4.3.10.5.3.1.3 Select a different image via url

4.3.10.5.3.2 ***Subimage Last***

4.3.10.5.3.2.1 Select an image for upload

4.3.10.5.3.2.2 Select a different image via url

4.3.10.5.3.2.3 Select a subimage

4.3.11 Browsers

Perform tests in multiple browsers.

5.0 Lessons Learned

This project provided a great learning opportunity. Our team was able to work together on a client-driven project and we accomplished great progress in a short time frame. Important lessons which were learned through this project include communication, distributing work and time-management, as well as how to deal with issues in development.

In a project such as this, communication between all parties was key. By keeping in touch with our client, Ms. Uma Murthy, we were able to receive feedback from her, develop our timeline, make the connections we needed, and to generally better understand our project goals. By keeping solid lines of contact between team members, we were able to set up meetings and get status updates easily. And as we made more connections, such as with Mr. Sunshin Lee, who helped us acquire server resources for our project, it was important to also keep a line of communication with them. For example, we originally asked Mr. Lee for any kind of server, and he was able to give us access to a Linux server. Later on in the project, we decided to pursue ASP.net instead of using PHP, which meant that we required a Windows server instead. We were able to contact Mr. Lee and he provided us with access to a Windows server.

Work distribution and time management are key to any team project. At the beginning of the project, the three teammates had several meetings to better understand each other's skill sets, which led to a more efficient load distribution and role assignment which utilized everyone's best skills. The roles were assigned such that Martin Overstrom, who had the most experience with this type of programming, was given the role of backend functionality developer. Kaslin Fields, who was primarily experienced in testing with some experience in program management, was assigned the roles of program manager and testing engineer. James Kaplan primarily had skills with development and showed a talent for learning, and was so assigned the role of JavaScript developer, as no one in the team had any real experience with this skill. Assigning one person to be the primary manager of the team was useful, as it promoted connection and uniform spread of work and time information throughout the team.

The development portion of this project was, as most likely any other project would be, fraught with difficulties which needed to be overcome. Firstly, it was originally planned that the primary functionality of the project should be done in PHP. However, due to a lack of experience in this field, and existence of legacy code in c-sharp, the decision was made that, due to time concerns, the project's primary functionality should be coded using ASP.net. This meant that a Windows server was required to run the program, which was handled as discussed above. There were also problems due to sparse documentation for CBISC, however the ability to use the c-sharp code from the SuperIDR project gave the team examples to supplement existing documentation. Furthermore, much of the development time at the beginning of the project needed to be spent simply understanding previous work and what needed to be done, while this is not unusual, it put strain on the deliverable timeline. This strain was further exacerbated by the difficulty of implementing the initial search functionality, which relied in great part on CBISC, making process in this area very slow.

The project's original timeline did have to be altered by a couple weeks to account for primarily coding and learning difficulties. However, strong communication and teamwork allowed the team to complete the primary goals for the project in the time allotted.

6.0 Acknowledgements

We would like to express our deepest gratitude to Mrs. Uma Murthy who took time out of her very busy schedule to work with us to make this project a success. Thank you to our professor, Dr. Edward Fox, who introduced us to this project through his class and provided feedback to help improve the final product. Our thanks also go to Ricardo Torres and his students for their work on the CBISC project which is used for image searching functionality in this project. Thank you to Mr. Sunshin Lee who worked to get us access to university server resources.

7.0 References

The app garden. (n.d.). Retrieved from http://www.flickr.com/services/api/

Murthy, U. (2011). *Digital libraries with superimposed information: Supporting scholarly tasks that involve fine grain information*. (Doctoral dissertation), Available from etds@vt.edu. (etd-04142011-175752)Retrieved from http://scholar.lib.vt.edu/theses/available/etd-04142011-175752/