FaceMatch: A System for Dynamic Search and Retrieval of Faces

Dharitri Misra, Michael J. Gill, National Library of Medicine, Bethesda, Maryland, USA

Abstract

Considerable progress has been made in recent years in locating and recognizing faces using advanced machine learning and computer vision techniques and a number of interactive tools are available for general use by individuals to use these technique in an ad hoc manner. However, no known, easily accessible open source framework presently exists to meet organizational needs to search for given faces against their pre-stored image sets, either in data analytics efforts or in time-critical situations, which leverages these techniques.

Consequently, at the US National Library of Medicine (NLM), we have implemented a Web based, publicly accessible system called FaceMatch (FM), which provides customized face matching services to clients through programmatic interfaces to robust face recognition software. In addition, it provides tools for use by the clients to submit requests in interactive or batch environment and visually observe the returned results.

Two key aspects of the FaceMatch system are: (a) it stores the contents of client's images in a repository through their key features instead of pixel values, avoiding potential copyright and other legal problems; (b) it assures near real-time availability of a newly ingested image for subsequent searches, eliminating perceptible delay between ingest and query, which is quite important in time-critical situation such as a natural or manmade disaster.

In addition to providing an HTML/REST-based API for clients to send requests to the FaceMatch server programmatically, the system also provides tools (integrated into a Java application called the FM Workbench), which allow users to submit requests interactively or in batch mode, and to visualize the returned results in real-time.

The FaceMatch system has been built for use with NLM's PEOPLE LOCATOR® Service, replacing an earlier, proprietary visual search system, and is available to be used by others with similar needs.

Motivation

With advances in face detection and recognition algorithms, artificial neural networks, and deep learning techniques, it is comparatively easy to locate faces or landmarks in an image and determine their similarity to other images. However, open source tools to make these techniques programmatically accessible to interested organizations is not common, and to provide them with image search capability without locally storing their images (which may violate privacy policies) is not standard.

There are also situations where it is necessary to build the search set dynamically and allow users to query it with minimal delay. A prime example with such requirements is a natural or man-made disaster, where a missing or found person's photo, submitted anytime, should be made available immediately for searching.

At the US National Library of Medicine, whose PEOPLE LOCATOR® Service (PL) [1] has to deal with such issues, we wanted to build a face matching system for PL in-house, particularly in the absence of available public tools. Furthermore, we wanted it to be flexible and publicly accessible so that it may be useful to other organizations with face matching needs. Below are some examples of how we envision the FM system to be used by others:

a) As an *emergency management tool*: in mass casualty events, where recognition of a person, among certain pre-identified individuals is essential in near real-time expediency. NLM has a demonstration Website (https://TriageTrak.nlm.nih.gov) and app (TriagePic®) for highlighting this situation. Given the frequency of such cases in public or private venues such as schools and businesses, the need for near real time indexing and searching of photos containing faces under a wide variety of conditions would benefit from FM services.

b) As a *curation tool* for persons of interest in historical image collections: Many universities and Government agencies possess photographic collections which could benefit from improved access. By using the FM system, the curators can identify new incoming images against already existing ones. This would reduce curation cost, as well as increase the value of the collection and the historical record of the particular field (Medicine, Science, Law, Engineering, etc.). Historians and the general public may also benefit from similar usage.

c) As a *research tool*: The FaceMatch system may also be used as a research tool for generating ground truth for face related studies and data analytics; for example: the studying effect of facial occlusion in detecting/matching faces in an image set, or facial changes with age.

Approach

The FM system, at the highest level, is built as an HTTP/REST-based Web server. It is written in Java, implemented using open source software/libraries and runs on a Linux platform under Tomcat. This system and its structure are described below.

System Description

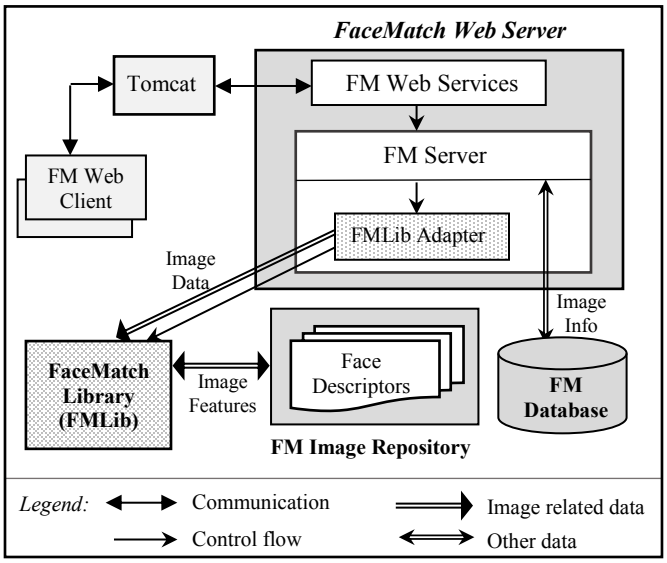

Figure 1- FaceMatch system architecture

The high level FM system architecture is presented in Figure 1. The FM Web server comprises two main software components: the *FM Web Services*, which is the system front end that interfaces with Tomcat for all client communications, and passes the service requests to the second component, a back end unit called the *FM Server*. The FM Server handles the execution of all system functions, including face matching, by invoking the appropriate computational service modules.

For simplicity, the diagram depicts only the components directly involved with face matching. The three main elements related to this operation and shown in the figure are described below.

(a) FaceMatch Library (FMLib)

This C++ software library, previously developed in-house, provides the face matching capability based on machine learning and artificial neural networks (ANN) [2]. FMLib performs image indexing by finding the key features of one or more detected faces in the image. During image ingest this index data, called an image descriptor, is stored in a *face-specific* binary file with user provided image tags. This granularity allows matching a query face against individual faces, rather than images, in the collection.

FMLib is connected to the FM server through the customized FMLib Adapter module, which includes a JNI (Java Native Interface) layer, to bridge the gap between the Java and C++ software interface. The JNI layer in not shown in the diagram for simplicity. This modularity allows us to provide better face detection and matching capability in the future by plugging in a more advanced library with its custom adapter.

(b) FM Image Repository

This data repository resides on a distributed file system and is partitioned into a set of storage regions corresponding to each client and their image collections. Each partition holds a set of image descriptors containing the facial image features generated by FMLib as described above. The data and file structures are independent of the submitted image file format; and therefore not concerned with image format obsolescence. The FM repository uses a hierarchical structure: connecting a client, to an image, to a facial zone. Files may be deleted from the repository by the client at individual collection, image, or face level using the appropriate Web service.

(c) FM Database

This MySQL database contains all relevant information related to a client and their ingested images. A client may group its images into one or more collections, called *image extents*, establishing individual search spaces for an incoming query. An image extent may be created, deleted, activated/deactivated and assigned its own search preferences through the associated Web service parameters.

The module *FM Web Client* in Figure 1 represents a client application, which sends REST-based service requests to the FM system and retrieves the returned results following the protocols in FM server's *web.xml* file. It is expected to be developed by the users of FM Web server. However, to alleviate the need to write this module by each user, FM also provides a stand-alone Java Web client application, called the ***FM Workbench***, described later in this paper.

FM Web Services

The FaceMatch system provides the following four groups of services to a client for face matching operations:

- Detection of faces in an image
- Creation and maintenance of an image collection
- Ingest of faces to an image collection
- Matching of faces in a query image against previously ingested images in a collection

All service requests are made using a client-specific software key, assigned by the FM administrator in *offline* communication, and the outputs are returned by the Server as JSON strings. A high level description of these services follows:

Face Detection Service: This service allows a client to analyze the quality of facial images, and does not depend on any image collection. It is useful for ascertaining the detectability of faces, particularly of unconstrained ones, before ingesting the corresponding image. The FM Server returns the coordinates of all detected faces in the input image, optionally with landmarks, in the service result but does not store them internally.

Image collection Services: This set of services allows a client to partition their images into collections *(extents)* according to image property, duration, search criteria and the like. Images may be ingested only to a previously created collection. An entire collection may be deleted when it is no longer needed, or deactivated temporarily, disallowing their search.

Ingest Services: These services allow the client to ingest (add) new images to an existing collection, delete a previously ingested image, or simply delete a specific face from an ingested image.

During ingest, the client may specify one or more rectangular regions in an image as *pre-designated* faces. Otherwise, FM first detects all the faces in the image, and then ingests the ones larger than a pre-determined size. Faces smaller than that size (nominally: 36x36 pixels) are not used for feature-based image matching.

To limit the number of relevant faces which should be searched against a query image, FM allows a client to associate a set of searchable *metadata*, such as a person's age group and gender, in the ingest request, and saves it in the database with the image URL. If none specified, it uses a default value for each metadata type.

Query Service: This service allows the user to search for face(s) in a given image against those in a specific collection. A query request may include the input image metadata, so as to search only the subset of images with similar metadata. As in ingest, the user may specify the coordinates of a face in the query image that should be matched, which is especially useful if the image contains multiple faces.

The match results for a query face are returned in ranked order, determined by their similarity distances to the given face. To limit the size of the return result set, the client may specify the maximum search distance and match set size in the query request,

The input/output format corresponding to these services and other details are provided in an FM Interface Control Documents.

System Operations

As mentioned earlier, FaceMatch system operates as a standard Web server under Tomcat, and provides the services specified in its *web.xml* file. The FM server uses a configuration file at system startup time for setting various operational parameters.

Next, it reads all relevant information for active image extents from the FM database as Java-based *Data Objects* into an in-memory cache, and builds *inverted index trees* based on image metadata for these image extents. To facilitate synchronization between ingest and query operations, the FM Server dynamically updates the pertinent branches of a metadata tree following each new ingest. When a query image is submitted, parallel searches are conducted through FMLib, using all available CPUs, for the relevant

metadata branches compliant with the query metadata - reducing overall search time.

Face detection and matching techniques

FMLib's face localization and search techniques are based upon research work conducted in-house using a large number of image collections, and comparing the results against that obtained from other tools such as the commercially available FaceSDK [*https://www.luxand.com/facesdk/*]. FMLib uses the OpenCV computer vision library [3] for computing initial face regions using Viola-Jones Haar feature-based cascade classifiers [4]. It improves this detection accuracy by looking for skin patches in both detected and rejected regions using a skin color map model. This model, referenced in here as skinANN, was developed at NLM using artificial neural networks and more than 13 million skin pixel samples [5]. Note that detection of fully profiled faces are not yet very successful with existing models.

For best effects, FaceMatch system uses several different feature extractors, namely: SHFT (Scale Invariant Feature Transform), SURF (Speeded up Robust Features), ORB (Oriented FAST and Rotated BRIEF), and LBPH (Local Binary Pattern Histograms). It generates image descriptors corresponding to each of these extractors during ingest and stores them in the image repository. A weighted average of these features, named DIST indexing scheme, is later used for similarity matching against a query image [2]. During search, features from the query image were extracted using the same scheme as during ingest, and matched against the stored descriptors.

Since image indexing is performed at the time of ingest, it is made available for search immediately afterwards by loading this index data to the appropriate search space prior to processing the next query. FM uses a *linear search* algorithm to compute match distances using the stored feature sets and returns the results as a ranked list. The list size is determined by user chosen limits in the query request.

FaceMatch system allows the same user to submit different service requests in parallel without affecting the results, with the only restriction being an image cannot be submitted for query until its ingest is complete. There is no restriction in serving multiple clients simultaneously, except for its possible impact on shared system resources.

Performance considerations

An important goal of the FaceMatch system is to return the results of a service request to users without perceptible delay. Consequently, it uses a number of optimization schemes including the use of a GPU and parallel search of relevant image sets, described earlier. Image metadata use narrows the search set further, but if no metadata is specified with a query image, FM searches all images within an extent. Similarly, it adds all images ingested with no metadata to each search set – resulting in slower response.

Time expended by the FM Service to complete major steps (such as image download, face detection, similarity search, and the entire operation) is returned to the user in each service result.

To identify bottlenecks and improve system performance through post-analysis, the FM system administrator has the option to record relevant operation timelines in the FM database.

Speed vs. accuracy

The time taken by the FaceMatch library to locate faces in an image depends upon the image quality, size, facial orientation (tilted, profile, etc.) and other factors. For example, to get accurate results for tilted faces, the image may be rotated counter-clockwise at 30 degree intervals until a face is found. The user has the option to choose between speed and accuracy on a *per collection* basis.

In general, various factors determine the total service time for a client request, even under optimal circumstances, as follows:

- Face Detection: dimension of the input image, number and size of faces in the image, orientation of the face (frontal vs. profile), use of skin color map (skinANN),
- Face Ingest: Face detection time plus the type and number of indexing schemes used for computing the weighted average, and
- Face Matching: Face detection time plus size of the search set and the number of parallel search streams.

Results

A number of benchmarks, evaluating both the accuracy and response times related to face detection and matching operations, were conducted by FM using a number of well-known image datasets. The image sets shown in this section are the following:

HEPL: This large dataset consists of images from the Y2010 Haiti Earthquake, and is obtained from the PL system. These are mostly low quality, color images, which were curated at NLM by the FM team for these studies [6].

Caltech [*http://www.vision.caltech.edu/archive.html*]: It is an openly accessible datasets of 450 facial images of about 27 persons, collected from the Web.

ColorFERET [*https://www.nist.gov/itl/iad/image-group/color-feret-database*]: It is another openly accessible dataset, from the National Institute of Standards and Technology (NIST), with more than 10,000 images. It consists of good quality single-face images in studio-type settings, in different orientations of each subject. About 2800 of selected faces from this set were used for FMLib accuracy benchmarks.

Indian Faces Database [*http://vis-www.cs.umass.edu/~vidit/IndianFaceDatabase/*]: This non-public dataset comprises 676 studio-setting images of 61 individuals in varying facial poses.

The ground truth for faces and skin colors were generated for 4000 HEPL images by the FM team for these tests, the other three image sets were released with their own ground truth data.

Accuracy

Accuracy of face detection and recognition operations by FMLib, compared against ground truth, is discussed in detail in papers [4, 5]. Those results were also compared against the commercial face matching engine FaceSDK (FSDK). For convenience, some of those results are reproduced here in Tables 1-a and 1-b.

Table 1-a. Face Detection Scores on Different Data Sets

Data	Method	Recall	Precision	F-Score
HEPL-500	VJ	.76	.87	.81
	VJ+SC	.81	.84	**.82**
	FSDK	.73	.87	.79
HEPL-4000	VJ	.45	.81	.58
	VJ+SC	.51	.81	**.63**
	FSDK	.47	.86	.60
Caltech	VJ	.95	.88	.91
	VJ+SC	.98	.97	**.98**
	FSDK	.96	.94	.95

Table 1-a shows the face detection accuracy of FM against HEPL (two subsets) and Caltech images. The first two rows for each set show the recall, precision and F-Scores with the OpenCV

provided Viola-Jones (VJ) models alone, and then VJ combined with FM's ANN-based skin color map, skinANN, denoted as SC. The third row indicates the results using FaceSDK. As evident, VJ with skin color map yields the best accuracy in each case, reaching up to 98% for good quality images.

Table 1-b. FaceMatch (FM) hit rate accuracy in top-N queries

	Caltech		ColorFERET		Indian Faces DB	
Top-N queries	FSDK	FMLib	FSDK	FMLib	FSDK	FMLib
1	.98	.98	.74	.98	.69	.79
3	.99	.98	.75	.98	.73	.85
5	.99	.99	.75	.99	.76	.87
10	.99	.99	.76	.99	.79	.90
20	.99	**1.0**	.76	**1.0**	.88	**.92**

Table 1-b shows FMLib face matching accuracy in terms of the probability of locating and retrieving an image within a containing set. It varies from 0.79 to 0.92 as the worst to best case scenario, 0.79 being the probability that the first image in the returned set is the correct one. It also illustrates that the FMLib results are consistently better than that of FaceSDK.

Service Response Time

The response time of FM services depends upon both the FM software architecture as well as the hardware platform and devices being used. Here we present the response times for the ColorFERET images, as well as that from real-life image sets of persons affected by natural disasters, namely: the 2013 flood in Uttarakhand, India, the 2013 typhoon Haiyan in Philippines, and the 2010 earthquake in Christchurch, New Zealand. These images were retrieved using their publicly available URLs from the PL system [6]. The quality of these images is mostly similar to that of HEPL.

The FM timing benchmarks were conducted under a lower grade Linux/Red Hat development system with 12 GB memory, eight CPUs, and a single GTX GeForce 750 GPU. It used MySQL V5.1 database, and OpenCV V2.4.13 and CUDA V8.0 libraries.

Face detection time

Faces were detected both with and without the ANN-based skin color maps to check the time variance against image characteristics. This mapping, which corrects for both false positive and false negative results yielding better accuracy, is performed in a time consuming second pass. This is reflected in the corresponding columns of Table 2-a, where VJ refers to Viola-Jones and SC to skin color map or skinANN.

Table 2-a. Face detection times with and without skin color map

Data	Image set size	Type	# of images w/faces	Ave msec	Min msec	Max msec
Uttara-khand	1000	VJ+SC	534	216	139	1,018
		VJ	422	58	23	139
HEPL	1000	VJ+SC	597	791	82	1,715
		VJ	265	94	79	137
Color-FERET	1000	VJ+SC	678	682	108	1,705
		VJ	633	121	97	215

In the first two image sets no faces were found in more than 40% of the images, even with skin maps, due the poor quality of the images that failed the minimum criteria (size, sharpness, lighting, background, etc.) for face detection. The third set, ColorFERET, contains many profile or near-profile faces which current models do not locate properly. The large variance between minimum (Min) and maximum (Max) times are often due to the extraction of single vs. multiple faces in an image as shown in Figure-2, or due to rotation of images to detect tilted faces, if any.

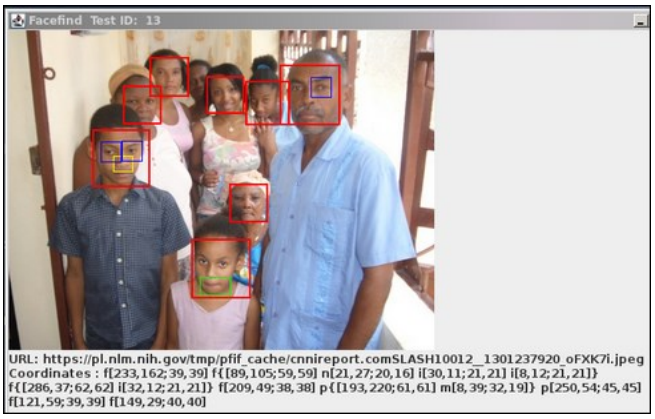

Figure 2. Example of multiple faces detected by FM in an HEPL image

Face query time

Table 2-b shows the query results, which includes face detection, for images in different sets under *linear search* with weighted DIST indexing strategy mentioned earlier.

As seen, for small search sets, face detection time is significant compared to the match time, but decreases as a function of set size. As presented, for the ColorFERET set with more than 7,500 images, the average query time in a development machine is less than four seconds.

It should be mentioned that unlike face detection time, query time does not depend upon the size of faces in the image set, as the same number of features are extracted from each face region during ingest irrespective of its size.

Table 2-b. Face query time for different image collections

Data	# of ingested images	Ave Face Detect (msec)	Ave Query (msec)	Min Query (msec)	Max Query (msec)
Christ-church	252	66.4	273	195	359
Typhoon Haiyan	5822	62.9	1,528	449	4,162
Color-FERET	7535	130.7	2,380	96	3,970

The results in Tables 2-a and 2-b do not include the Web service overhead, which is between 15 to 20 milliseconds for such operations.

FM Workbench: the Pre-built FM Web Client

This stand-alone Java Web client application may be obtained from the FM system as a Java jar file, and configured to run at a client's facility. It can be used to request FM services individually or in batch mode. Some of its usages are as follows:

- Send a service request to the FM server and display returned results visually in real-time.

- Manually delineate face regions in an image prior to submitting for ingest or query
- Batch a set of requests, to help in ingest or research/test activities, and display results in table form; then click on any table row to display further details.
- Save the return data in JSON format on disk; then load and display through the Workbench later, or provide as input to other applications, for further analysis.

Figure 3-a shows, as an example, the face detection results displayed by the FM Workbench for a user submitted image batch from the Haiyan dataset. Figure 3-b shows the top nine matches, in ranked order with similarity distances between 0 and 0.5, for a query image (the top leftmost panel) in the ColorFERET collection.

Figure 3-a. Batch-based face finding result, with display of selected rows

Figure 3b – A query image (upper leftmost) and its first nine matched images ranked according to their distances as shown by the FM Workbench

Conclusion

The FaceMatch system was tested for face detection and matching services for a variety of image types, with satisfactory results, for NLM's PEOPLE LOCATOR® and other simulated clients.

Presently, we are working to increase the accuracy of face detection and search results by developing a state-of-the-art FaceMatch library, using recent research work in computer vision and machine learning. Thus, we are exploring Deep Learning and convolutional neural network (CNN)-based object detection and similarity search techniques, such as Faster R-CNN [7], especially as the accuracy and efficiency of related software tools have improved to the extent of making their use in operational systems both practical and desirable.

Based upon our work, we believe that in addition to providing PL the much needed service, the FaceMatch system would meet our related goal, namely: be useful to other organizations as a productive service. The FM Workbench, which makes interaction with the server simpler, may be fine-tuned to meet each client's specific needs, helping in this effort.

Acknowledgement

This research was supported by the Intramural Research Program of the National Institutes of Health (NIH), National Library of Medicine, and Lister Hill National Center for Biomedical Communications (LHNCBC). We thank Dr. Sameer Antani, NLM Staff Scientist, for his helpful suggestions in preparing this manuscript.

References

[1] Find Missing Persons During A Disaster | People Locator and ReUnite (https://lpf.nlm.nih.gov/PeopleLocator-ReUnite)

[2] E. Borovikov and S. Vajda, "FaceMatch: real-world face image retrieval," Recent Trends in in Image Processing and Pattern Recognition (2016) (https://lhncbc.nlm.nih.gov/system/files/pub9450.pdf)

[3] Open Source Computer Vision Library (https://docs.opencv.org/2.4/modules/refman.html)

[4] P. Viola and M. Jones, "Robust real-time face detection", *Int. Journal of Computer Vision*, vol. 57, pp. 137–154, (2004)

[5] E. Borovikov, S. Vajda, et.al., Face matching for post-disaster family reunification (https://lhncbc.nlm.nih.gov/system/files/pub8893.pdf)

[6] OHSR Response to Request for Review of Research Activity Involving Human Subjects (https://wiki.nlm.nih.gov/confluence/pages/viewpage.action?pageId=74619832&preview=/74619832/105547932/GillM_NLM_5080_CY2010.pdf)

[7] Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks (http://ieeexplore.ieee.org/document/7485869/?reload=true)

Author Biography

Dharitri Misra is a Staff Scientist at the US National Library of Medicine. Her work involves machine learning, image recognition, and natural language processing, including development of frameworks and tools to support such work. Her current work focuses on the application of Deep Learning to face recognition. She received her M.S. and Ph.D. degrees in Physics from the University of Maryland.

Michael Gill is a senior electronics engineer with the National Library of Medicine. His work includes the design, development and operation of the NLM PEOPLE LOCATOR system which is a service to speed family reunification after mass casualty incidents. He has a BSEE from the University of Maryland.