American Journal of Systems and Software, 2017, Vol. 5, No. 1, 9-14
Available online at http://pubs.sciepub.com/ajss/5/1/2

DOI:10.12691/ajss-5-1-2

DiFace: A Face-based Video Retrieval System with Distributed Computing

Lan Huang*, Juan Zhou

College of Computer Science, Yangtze University, Jingzhou, Hubei, China
*Corresponding author: lanhuang@yangtzeu.edu.cn

Abstract With the prevalence of video surveillance and the extraordinary number of online video resources, the demand for effective and efficient content-based video analysis tools has shown significant growth in recent years. Human face has always been one of the most important interest points in automatic video analysis. In this paper, we designed a face-based video retrieval system. We analyzed the three key issues in constructing such systems: frame extraction based on face detection, key frame selection based on face tracking and relevant video retrieval using PCA-based face matching. In order to cope with the huge number of videos, we implemented a prototype system on the Hadoop distributed computing framework: DiFace. We populated the system with a baseline dataset consisting of TED talk fragments, provided by the 2014 Chinese national big data contest. Empirical experimental results showed the effectiveness of the system architecture and also the techniques employed.

Keywords: *video retrieval, face-based video retrieval, content-based retrieval, distributed computing, Hadoop*

Cite This Article: Lan Huang, and Juan Zhou, "DiFace: A Face-based Video Retrieval System with Distributed Computing." *American Journal of Systems and Software*, vol. 5, no. 1 (2017): 9-14. doi: 10.12691/ajss-5-1-2.

1. Introduction

With the rapid development of smart city, security cameras and video surveillance has become ubiquitous in recent years [1]. Video sharing is also one of the most popular features on online social networking websites [2]. This results in unprecedented large number of video fragments, and the demand for effective and efficient automatic video content analysis tools has become particularly urgent.

In video content analysis, human face has been a consistent angle of interest [3,4,5,6,7]. Faces identified and extracted from video fragments can be used to develop content-based access and to facilitate intelligent multimedia tools and applications. Although there exists a number of face detectors and trackers [8] that can detect frontal faces with different sizes, locations and background images, recognizing faces of the same person in streaming video fragments remains a challenging task, due to large variation in pose changes, illumination conditions, occlusions and facial expressions. In particular, with large number of videos, conventional standalone methods need to be revised or new methods need to be developed in order to cope with scalability.

In this paper, we proposed a face-based video retrieval system. We analyzed the three key issues in constructing such systems: extraction of frames that containing human faces, selection of key frames that are representative of a given video fragment, and finally retrieval of relevant videos that contain the same person as in the given query video fragment. Furthermore, in order to cope with scalability, efficiency is a particular focus of designing solutions to these issues. Meanwhile, we also employed the off-the-shelf distributed computing framework Hadoop while implementing the proposed system. We name the resulting distributed face-based video retrieval system DiFace.

The rest of this paper is organized as following. Next we review related work on automatic content-based video analysis with a specific focus on the human face angle. Section 3 explains the three key issues in DiFace and our solutions. Sections 4 describes the system design and implementation details. Section 5 presents experimental setup and discusses empirical experimental results. Section 6 concludes the study.

2. Related Work

Face-based video retrieval belongs to the more generic research area of content-based video analysis. Starting from the late 90s, content-based video analysis has developed from low-level image features, temporal and motion features, to high-level semantic objects, sentiments and behaviors [9,10]. Applications of this technique has also expanded from mainly video indexing and searching [9,10], to more generalized scenarios such as video copy detection [11], video quality assessment [12], behavior analysis [13] and video summarization [6]. Human face has been a consistent focus in these applications, for example, in video retrieval [3,4,5], video summarization [6] and in wearable devices [7].

Among these applications, video retrieval is the most relevant area to our study. In this particular field, Pande et

al. [4] combined several face detecting, tracking and grouping techniques to extract face tracks from video segments. The extracted face tracks were then used for video indexing and retrieval. Human faces can be combined with other information to enrich the profile of a given video fragment. In this direction, Satoh et al. [3] extracted and recognized human faces from news videos and associated them with names extracted from transcripts. The resulting system (i.e. the Name-It system) can thereby retrieve video fragments by person's name. Recently, Zhang and Jeong [5] has proposed a face-based image retrieval algorithm to retrieve airport surveillance videos that are relevant to a given person.

A number of methods has been developed along the way to tackle face-based video retrieval, including face detection, face image representation, face extraction, face tracking and face matching. For example, for face image representation, mainstream local features include the Harris operators, SIFT descriptors, SURF descriptors and eigen faces [14]. Reviews on these aspects can provide further information [8,15,16].

In recent years, scalability and efficiency has become an increasingly important research focus in video analysis. For example, Yu et al. [17] comprehensively evaluated the trade-offs between efficiency and accuracy brought by different component solutions in a content-based video retrieval system, and obtained a remarkable speed up while retaining reasonable accuracy. Computing clouds and distributed computing techniques have also been employed recently to handle the rapidly-increasing amount of videos on the Internet [18,19]. Yet, how to design an effective and efficient face-based video retrieval system on a distributed computing framework (i.e. Apache Hadoop) still needs exploration.

3. Key Issues and Solutions

As stated previously, there exists three key issues in a face-based video retrieval system: 1) extraction of frames that containing human faces; 2) selection of key frames that are representative of a given video fragment; and 3) retrieval (including ranking) of relevant videos that contain the same person as in the given query video fragment. This section explains our solution to each issue.

3.1. Frame Extraction based on Face Detection

Not every frame of a video fragment is equally important. In our case, in particular, only those that contain a clear (and preferably frontal) face image of the key person of a video are important and thereby worth extracting. Quality of the extracted frames is important, because these frames form the basis for subsequent processing such as key frame selection and profile construction.

We adopted Viola & Jones' algorithm (denoted as the VJ method) for human face detection [20] to detect faces within each frame. By computing the integral image based on the Haar features of the input image, and employing multiple classifiers with AdaBoost, the VJ method can effectively and efficiently detect areas of human faces. Each detected human face area was extracted (e.g. with 150×150 pixels), and collectively they form the candidate images for subsequent key frame selection.

3.2. Key Frame Selection Based on Face Tracking

Sometimes a video can contain more than one person. We define key person as the one that consistently appears in a video segment. Therefore, the extracted human face images are filtered. By computing the pairwise similarity between face images, the key face that consistently occurs is identified. Then matching face images are retained and the others are discarded. Similarity between faces is the decisive factor here. DiFace used the Bhattacharyya distance [21] as the similarity measure of human faces. Given two images and their histograms, the Bhattacharyya distance is calculated as follows:

$$d(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}} \qquad (1)$$

where $H_1(i)$ and $H_2(i)$ are the number of pixels in the ith grey level of the two images. The Bhattacharyya distance is bounded by [0,1], with a smaller value indicating a greater similarity. Figure 1 shows the Bhattacharyya distance values between different face image pairs.

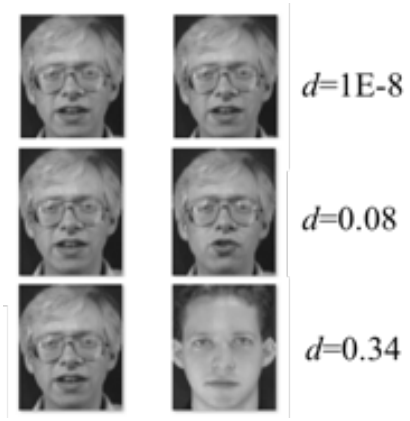

Figure 1. Examples of the Bhattacharyya Distance Between Different Human Faces

Then faces that exceed a pre-defined threshold are retained as the key frames. The selected faces form a face track that runs throughout a video fragment.

3.3. Relevant Video Retrieval based on Principle Component Analysis

After the above step, each query video fragment and each fragment in the database corresponds to a set of face images. Relevance between video fragments is then computed based on the two sets. Due to variations in pose changes, illumination conditions, occlusions, hairstyles, and facial expressions, robust face matching has been a challenging problem. Many face matching methods can be applied here [14]. DiFace uses principle component analysis (PCA) [22] to generate the eigenfaces from a set of face images and then perform face matching based on the eigenfaces.

PCA is a common dimensionality reduction method. By calculating the linear combinations of primitive features that can explain the covariance structure in the input data, PCA transforms the original high-dimensional feature space into a low-dimensional space represented by the linear combinations.

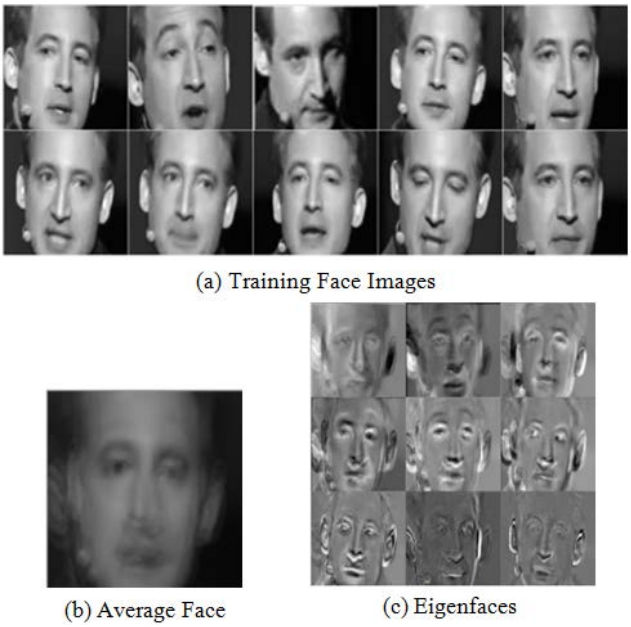

Figure 2. Eigenfaces Computed by PCA

When applied to face matching, PCA usually works in three steps. Firstly, it computes an "average face" from a set of training face images, and then subtracts the average face from each training image. Secondly, the eigenfaces are calculated from the covariance matrix between each training image and the average face image. Finally, given a new face image, it compares the input against the training images, and outputs images that result in a similarity (e.g., based on the Euclidean distance) value exceeding a pre-specified threshold. Figure 2 illustrates the working process of PCA with examples from our experimental dataset: (a) ten training images of the same person, (b) the resulting average face, and (c) the nine eigenfaces generated from the ten training images.

When applied to face-based video retrieval, PCA also involves three steps. Firstly, for each video fragment t in the database, constructs its eigenface representation $\{e_1, e_2, \ldots, e_k\}$ based on the key frames $\{f_1, f_2, \ldots, f_t\}$ selected in the above section. Secondly, for each query video fragment q, extracts its key frames $\{f_1, f_2, \ldots, f_q\}$ and projects each frame onto the representation space of $\{e_1, e_2, \ldots, e_k\}$. Finally, computes the average distance between $\{f_1, f_2, \ldots, f_q\}$ and $\{f_1, f_2, \ldots, f_t\}$. If the resulting distance is below a pre-defined threshold, then the two video fragments are considered as containing the same key person.

3.4. Distributed Computing Framework

As the amount of video resources increase rapidly, scalability and high efficiency has become an important requirement for many automatic video analysis tools. The Apache Hadoop framework [23] is the most popular open-source distributed computing environment. Because Hadoop is scalable, generalized and extensible, it has been used for parallelized video processing in several previous studies [18,19].

The most relevant major components of Hadoop are the distributed fault-tolerant file system HDFS and the distributed computing programming paradigm MapReduce. All video fragments are stored on HDFS, which provides distributed and highly reliable file storage. Following the MapReduce programming paradigm, programmers only need to convert their standalone programs into the Mapper and Reducer classes, so as to parallelize their programs. All the complicated parallel computing functions such as resource allocation and task monitoring and management, are all performed by the Hadoop framework, thus significantly facilitates the transformation into distributed computing.

4. System Design and Implementation

4.1. Processing Pipeline

Given a static image or a video fragment, face-based video retrieval finds videos that have the same faces as those in the input video. An automatic face-based video retrieval system usually consists of three components: representative frame extraction (i.e., images containing clear faces with a reasonable size), video profile construction (i.e., extracting, selecting and organizing face images that can represent the person appearing in a video fragment), and face matching (i.e., relevance calculation) and ranking. Accordingly, DiFace's processing pipeline consists of four major steps, as shown in Figure 3.

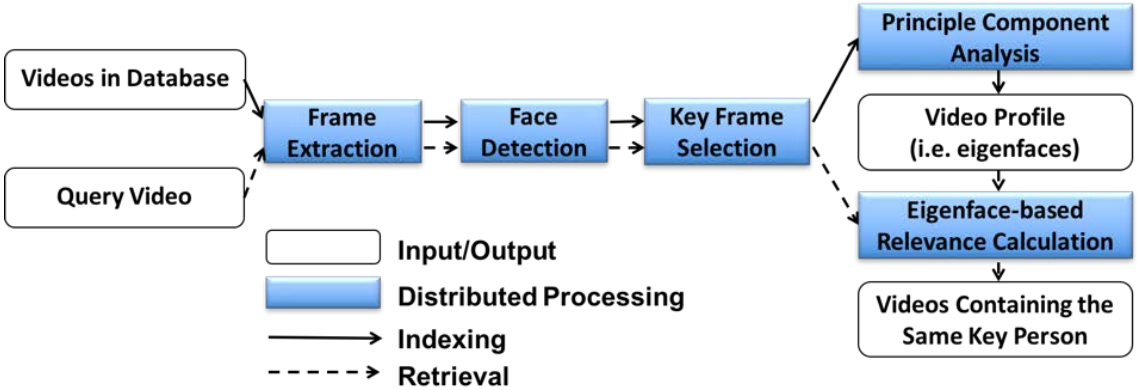

Figure 3. DiFace System's Processing Pipeline

For video fragments in the database, the indexing phase consists of four steps: frame extraction, face detection, key frame selection and video profile construction (based on PCA). For a query video fragment, it also goes through four steps, with the same first three steps as in the indexing phase and the last step being profile-based video matching and ranking.

4.2. System Implementation

Figure 4 – Figure 6 shows the process of the three key issues as discussed in the previous section. Figure 4 and Figure 5 shows the key frame extraction process. Starting from the first frame of a video fragment, DiFace first generates its grayscale image. Face detector is then applied to the grayscale image. Since the frequency of videos in our experimental dataset is 15 frames/second, if the current frame contains a human face, it is likely that the next frame also contains a face. Two step size is then designed for sliding the frame extractor: 0.07 second if the current frame contains a face and 1 second otherwise. The frame extraction process iterates until ten human face images are collected. Then the images are filtered based on their pairwise Bhattacharyya distances. Only images with the most number of matches (i.e. images with a Bhattacharyya distance below the threshold) are retained.

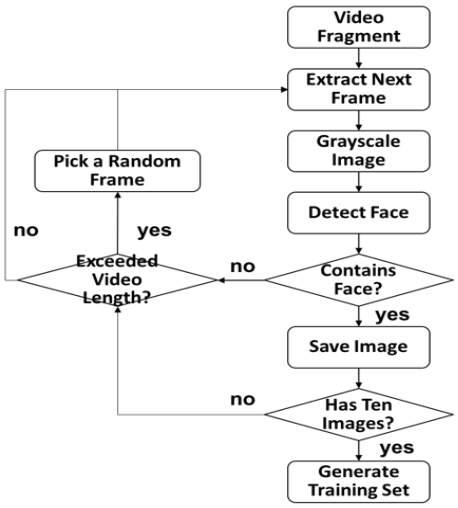

Figure 4. Frame Extraction Process

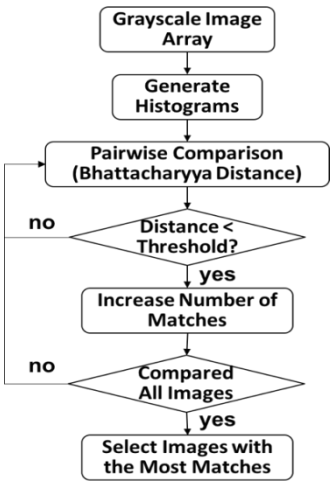

Figure 5. Key Frame Selection Process

During the retrieval process, face images are first detected and extracted. Then for each video fragment in the backend database and its eigenface representation, the extracted faces are projected onto the same space. Euclidean distances are calculated between the query video's images and those of the indexed videos. Relevance is simply the reverse of the distance value: a smaller distance indicating greater relevance.

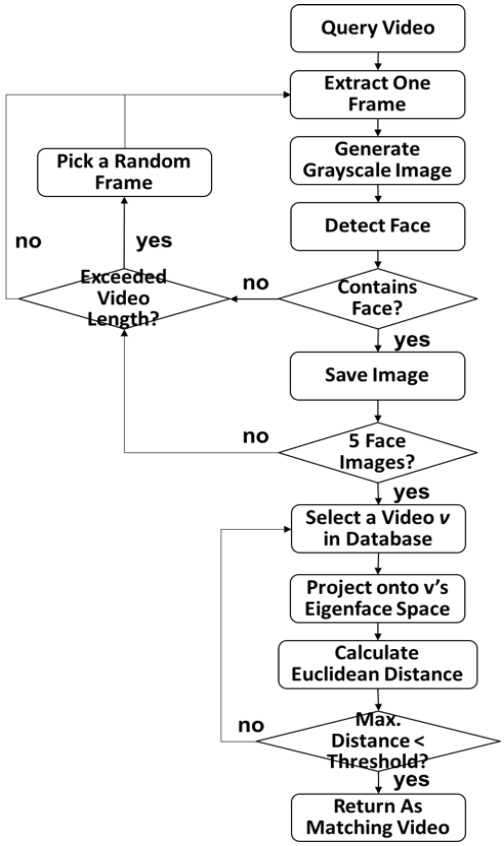

Figure 6. Face-based Video Matching Process

During the retrieval process, face images are first detected and extracted. Then for each video in the backend database and its eigenface representation, the extracted faces are projected onto the same space. Euclidean distances are calculated between the query video's images and those of the indexed videos. Relevance is simply the reverse of the distance value: a smaller distance indicating greater relevance.

DiFace utilizes a number of video processing tools, including Fuse-DFS [24] and JavaCV [25]. Fuse-DFS allows Hadoop's HDFS to be mounted as a standard file system onto the common Linux systems. Then files on the HDFS can be accessed directly from local file systems, and programming libraries such as many image processing libraries can be accessed and used in MapReduce programs. JavaCV provides the Java language encapsulation of the popular OpenCV vision processing libraries and common video processing tools like FFmpeg. Using these tools, all processes are performed in a distributed fashion.

5. Experimental Results and Discussion

To evaluate the effectiveness of the DiFace system, we used video fragments collected from the Technology

Entertainment Design (TED) talks. This experimental dataset, created and provided in the face-based video retrieval task of the second big data contest held by the China Computer Federation, consists of 1,197 short video fragments from 73 TED talks. Every fragment contains a key speaker. Figure 7 shows the distribution of the number of fragments per talk. In order to prevent long talks dominating the evaluation, we randomly selected ten talks from the middle of the distribution, i.e., within the [10,20] range, resulting in 160 fragments. The length of a single fragment varies from 0.6 to 11.2 minutes, with an average of 4.3 minutes.

The search task was then for any given query video fragment, find out the other video fragments that belong to the same talk as the query fragment. System effectiveness was measured by standard performance indicators in information retrieval: *Precision*, *Recall* and the *F-measure*, which are calculated as follows:

$$Precision = \frac{\#relevant\ video\ fragments\ returned}{\#video\ fragments\ returned}, \quad (2)$$

$$Recall = \frac{\#relevant\ video\ fragments\ returned}{\#relevant\ video\ fragments\ in\ database}. \quad (3)$$

F-measure averages out Precision and Recall:

$$F-measure = \frac{2 \times Recall \times Precision}{Recall + Precision}. \quad (4)$$

Results were the average of ten independent runs of a 10-fold cross-validation. In each run, the 160 video fragments were divided into ten different groups: nine for training and one for testing.

In the PCA-based video relevance calculating process, there are three ways to compute the relevance (i.e. distance) between the query video and existing video: taking the average (AVG), the maximum (MAX) and the minimum (MIN) distance between their associated face image sets. Figure 8 compares these methods in terms of their impact on system's retrieval performances. In comparison, the baseline represents a brute-force method by retrieving fragments in the dominating talk: the talk that has the largest number of fragments in the dataset. In other words, no content-based analysis was involved in the baseline approach. The X-axis shows the performance when calculated with different number of returned results.

As shown in Figure 8, the proposed face-based video retrieval system and the techniques it employed were effective in general. Besides, taking the maximum distance between face images as the distance between videos was the most effective.

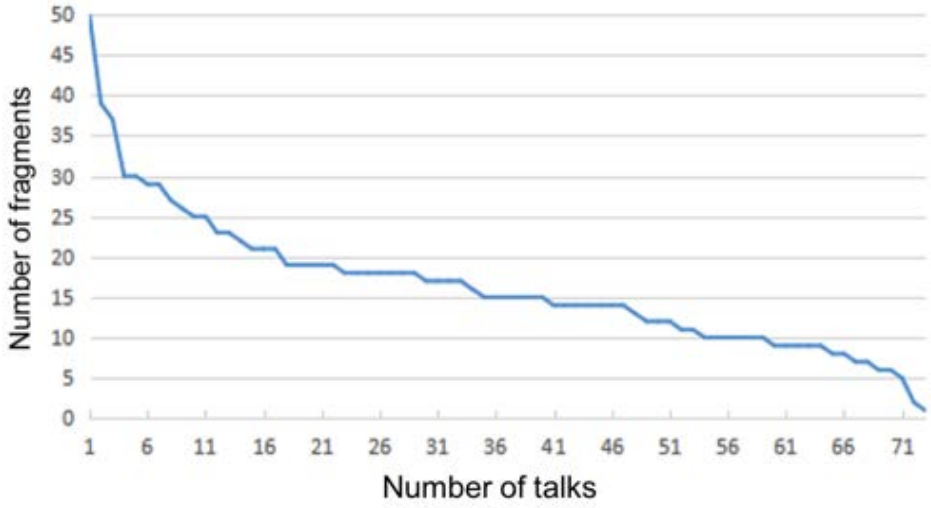

Figure 7. Number of Video Fragments Per Talk

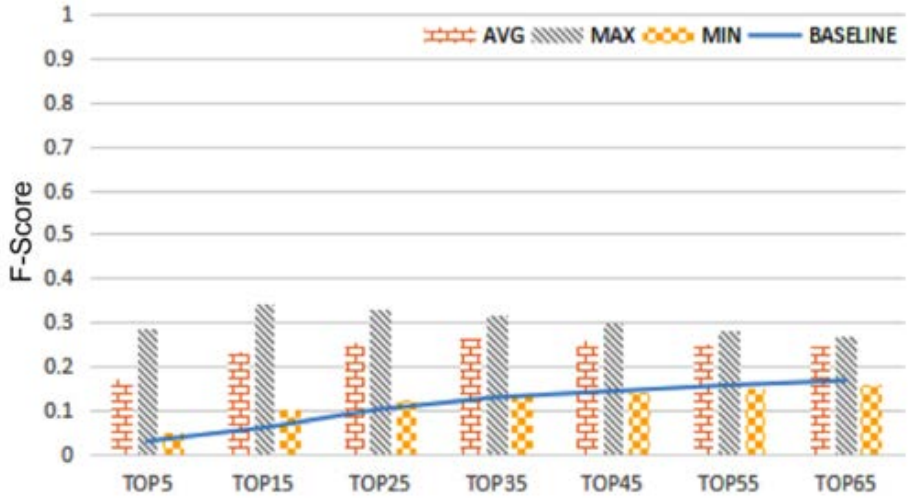

Figure 8. DiFace's Retrieval Performances

6. Conclusions

Facing the pressing demand for effective and efficient content-based video analysis tools, we designed and implemented a distributed face-based video retrieval prototype system DiFace. Mainstream frameworks for distributed visual processing was adopted, including Hadoop, Fuse-DFS and OpenCV, so as to ensure system scalability. Experimental results showed the effectiveness of the proposed system. In future work, we plan to apply the DiFace system to real-world video resources, such as surveillance videos collected from public cameras and within industrial plants. These more complicated scenarios might require advanced video mining techniques, such as motion detection and tracking.

Acknowledgements

This study was funded by Yangtze University (grant no. JY2014032 and 2015cqn52).

References

[1] Dey, S., Chakraborty, A., Naskar, S., Misra, P.. Smart city surveillance: Leveraging benefits of cloud data stores. In: IEEE 37th Conference on Local Computer Networks Workshops, 2012, IEEE, pp. 868-876.

[2] Boyd, D. M., Ellison, N. B.. Social Network Sites: Definition, History, and Scholarship. Journal of Computer-Mediated Communication, 2007, 13(1), pp. 210-230.

[3] Satoh, S., Nakamura, Y., Kanade, T.. Name-It: Naming and Detecting Faces in News Videos. Journal of IEEE MultiMedia, 1999, 6(1), pp. 22-35.

[4] Pande, N., Jain, M., Kapil, D., Guha, P.. The Video Face Book. In: 18th International Conference on Advances in Multimedia Modeling, 2012, IEEE, pp. 495-506.

[5] Zhang, N., Jeong, H-Y.. A Retrieval Algorithm for Specific Face Images in Airport Surveillance Multimedia Videos on Cloud Computing Platform, 2017, 76(16), pp. 17129-17143.

[6] Lee, Y-S., Hsu, C-Y., Lin, P-C., Chen, C-Y., Wang, J-C.. Video summarization based on face recognition and speaker verification. In 10th Conference on Industrial Electronics and Applications, 2015, IEEE, pp. 621-625.

[7] Mandal, B.. Face recognition: Perspectives from the real world. In 14th International Conference on Control, Automation, Robotics and Vision, 2016, IEEE.

[8] Belaroussi, R., Milgram, M.. A comparative study on face detection and tracking algorithms. Expert Systems with Applications, 2012, 39(8), pp. 7158-7164.

[9] Zhang, H. J. Wu, J., Zhong, D., Smoliar, S. W.. An integrated system for content-based video retrieval and browsing. Pattern Recognition, 1997, 30(4), pp. 643-658.

[10] Hu, W., Xie, N., Li, L., Zeng, X., Maybank, S.. A survey on visual content-based video indexing and retrieval. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), 2011, 41(6), pp. 797-819.

[11] Lian, S., Nikolaidis, N., Sencar, H. T.. Content-Based Video Copy Detection—A Survey. Intelligent Multimedia Analysis for Security Applications, 2010, pp. 253-273.

[12] Chikkerur, S., Sundaram, V., Reisslein, M., Karam, L. J.. Objective Video Quality Assessment Methods: A Classification, Review, and Performance Comparison. IEEE Transactions on Broadcasting, 2011, 57(2), pp. 165-182.

[13] Popoola, O. P., Wang, K.. Video-Based Abnormal Human Behavior Recognition—A Review. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), 2012, 42(6), pp. 865-878.

[14] Zhou, J., Huang, L. A Comparative Study of Local Features in Face-based Video Retrieval. Journal of Computing Science and Engineering, 2017, 11(1), pp. 24-31.

[15] Yang, H., Shao, L., Zheng, F., Wang, L., Song, Z.. Recent advances and trends in visual tracking: A review. Neurocomputing, 2011, 74(18), pp. 3823-3831.

[16] Zafeiriou, S., Zhang, C., Zhang, Z.. A survey on face detection in the wild: Past, present and future. Computer Vision and Image Understanding, 2015, 138, pp. 1-24.

[17] Yu, S-I., Jiang, L., Xu, Z., Yang, Y., Gauptmann, A. G.. Content-Based Video Search over 1 Million Videos with 1 Core in 1 Second. In: 5th ACM International Conference on Multimedia Retrieval, 2015, ACM, pp. 419-426.

[18] Ryu, C., Lee, D., Jang, M., Kim, C., Seo, E.. Extensible video processing framework in Apache Hadoop. In: 5th IEEE International Conference on Cloud Computing Technology and Science, vol. 2, 2013, IEEE, pp. 305-310.

[19] Tan, H., Chen, L.. An approach for fast and parallel video processing on Apache Hadoop clusters. In: 2014 IEEE International Conference on Multimedia and Expo, 2014, IEEE, pp. 1-6.

[20] Viola, P., Jones M. Rapid object detection using a boosted cascade of simple features. In: 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2011, IEEE, pp. 511-518.

[21] Bhattacharyya, A.. On a measure of divergence between two statistical populations defined by their probability distributions. Bulletin of the Calcutta Mathematical Society, 1943, 35, pp. 99-109.

[22] Turk, M., Pentland, A.. Eigenfaces for recognition. Journal of Cognitive Neuroscience, 1999, 3(1), 71-86.

[23] White, T. Hadoop: The Definitive Guide. O'Reilly Media, Inc. 2015.

[24] MountableHDFS. [Online]. Available: https://wiki.apache.org/hadoop/MountableHDFS. [Accessed: August 17, 2017].

[25] JavaCV. [Online]. Available: https://github.com/bytedeco/javacv. [Accessed: August 17, 2017].