Face Detection And Recognition For Distributed Systems

Meng Lin, Ermin Hodzic
School of Computing Science
Simon Fraser University
Surrey/Burnaby, BC, Canada
menglin@sfu.ca, ehodzic@sfu.ca

Abstract

The abstract of your work should state the problem, why it is important, and your proposed solution.

The ability to find faces on images and determine who the faces belong to using computers is a problem researchers have been working on for decades. With the advancements of image recognition algorithms, mostly during the last decade, today we have fast and sufficiently accurate systems. As cloud computing is gaining on popularity, demand for applications written for distributed systems is on the rise. Whil there are solutions for single-machine parallel face detection and recognition, interest in distributed solutions seems low.

I. Introduction

With the increase of computational power and big improvements in field of machine learning, we can say that computers are becoming more and more intelligent. Nowadays we have systems which offer a great level of interaction with users in a way that is natural to humans. Computers are able to perform tasks that are far from things natural to machines but close to humans. For example, since humans are visual beings, distinguishing between objects we see, and recognizing those we have seen before (but now in different conditions) are easy tasks for us. However, for computers, it has been a very hard and unnatural problem which still can not be solved with 100% accuracy. On the other hand, today's software is very advanced and offers excellent results.

Computer face detection and recognition is a topic that has been gaining on popularity over the recent years since face detection tasks are being required more and more. Security cameras combined with

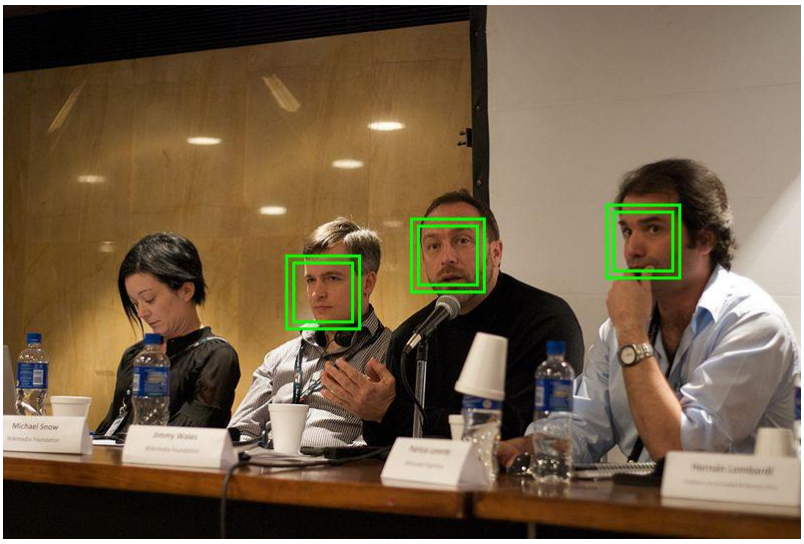

Fig. 1. An example of face detection using OpenCV. Image taken from wikipedia: http://en.wikipedia.org/wiki/File:Face_detection.jpg

real-time face detection and recognition systems is one such example. Also, we know that most of digital cameras and cell phones made in the last decade use some kind of face detection system. Lately, social networks such as Facebook have incorporated face detection and recognition features in their image gallery applications, allowing you to tag people on photos more easily and giving you suggestions on who the people might be. All in all, we are witnesses that demand for face recognition and detection software and algorithms is growing more and more.

Face detection aims to identify and locate faces in images, or videos, usually giving us an outline of the face region. Most of the problems related to face detection lie in variations in scale, orientation, lighting conditions, facial expression etc. Significant progress has been done during the past decade, mostly initiated by the work of Viola and Jones [1] which enabled real-time face detection due to combination of speed and good results.

Face recognition... // TODO

We have designed and implemented parallel face detection and face recognition framework suitable for use on clusters with many nodes. The method is flexible in the sense that any of the well-known methods for face detection can be used at the lowest level to actually find faces. The goal of our method is to partition the image into subimages such that each part can be independently and efficiently analyzed by any algorithm and the results combined.

II. Related Work

According to [2], the most famous early face recognition system is the one developed by Kohonen in 1989 who used neural nets to perform face recognition on aligned and normalized images by extracting features known as ”eigenfaces” which capture the most of the important information about the face. However, his system did not perform well in practice due to the need for precise alignment and normalization. Later there appeared improvements to the method ([3]) which increased the efficiency of calculating eigenfaces.

More history of face recognition... // TODO

Probably the most important and the most challenging part of face recognition systems is detecting faces in images. The problem can be very hard problem to solve since faces can be rotated, facial expressions may vary, pose may vary, lighting may differ not only across different images but across faces in the same image. Also, not all faces in the same image are necessarily of the same scale. Thus finding all faces in an image has been a challenging and computationally and algorithmically demanding task.

In 1996, [4] published a neural networks based approach to finding frontal views of faces in grayscale images and reported a success rate between 78.9% and 90.5% of their method.

The paper which probably had the most impact on the area of face detection was the one by Viola and Jones [1] in 2001. It has made face detection practically feasible in real-time applications. It combines ”integral image” representation for fast features evaluation, AdaBoost learner for constructing a classifier and a method for combining the classifiers into cascade to quickly identify regions of interest, thus dramatically increasing the speed of the process. The results were comparable to the state of the art of the time and the method was much faster than competition. However, while the haar features used in Viola and Jones method are very simple and effective for frontal face detection, they don’t perform as well when face is allowed to be in arbitrary pose.

In [5], they did a survey of recent advances in face recognition in 2010, describing various methods, and concluding that a lot of work still needs to be done to achieve good results in unconstrained settings. They reported about 50-70% detection rate of the state of the art face detectors with 0.5-3% false positive rate. In the same year, [6] have done comparative testing of face detection systems and report VeriLook to be the best in terms of performance, followed by the popular and free-for-use OpenCV.

While there has been work done on parallelizing face detection algorithms, such as Intel’s AIM View [7] which is based on OpenCV and utilizes multiple cores on modern processors, moving face detection systems to clusters has not received much attention.

More on state of art in face recognition... // TODO

For image processing on Hadoop Mapreduce. In [8], they proposed an open-source Hadoop Image Processing Interface (HIPI). Their goal was to provide a general framework to computer vision research on big data. Usually, identical operations are performed on the whole input set, which makes Hadoop ideal for big data vision research. In their work, they packed images into Image Bundles, because hadoop file system is better working for small number of big files rather than big number of small files[8]. HIPI hides the details on image encoding and tranfering, so users can focus on the image itself. Instead of JPEG or PNG, images are distributed as float images with extra header information. As a result, users can filter out unwanted images without decoding, and perform pixel-wise operations like croping more easily.

III. Research Ideas

With the increase in popularity of cloud computing and writing applications for distributed systems, and since face detection and recognition systems built for clusters have not received much attention, our idea was to design and implement such a system. The goal was to think of an algorithm that would distribute work in such a way that would enable easy parallelization of the job being done, thus making it easier to be done on a cluster. Much care has been put into making it work with MapReduce programming model.

In the case of face detection, one thing this project does not deal with is algorithm for deciding whether there are (and where they are) faces in a given image and we mainly focused on making the job easier by distributing the work. For the job of actually deciding whether there is a face in a given region, we employed the free-to-use solution OpenCV with its underlying Viola-Jones algorithm. Of course, any other tool can be used instead of OpenCV.

First, let us take a look at the way most of current face detection algorithms work. First of all, a face size is assumed. Then whole image is scanned with the fixed scan window size and for each position a classifier is used to decide whether current window location represents a face or not. Window is then scaled to a different size and the process is repeated. In the end, we may end up with many rectangles which overlap a lot. Since they most likely represent the same face, such rectangles are averaged to a single rectangle. Scaling factor is usually set to value between 5% to 10% for a good ratio of quality and running speed. Increasing scale factor reduces number of times we scan the whole image and increases running time but then we might skip a scale that would identify a face and thus miss it. Decreasing it will improve sensitivity but also increase running time quite a lot. First way to introduce parallelization

would obviously be to distribute different window sizes across multiple workers. Next idea is to split image into subimages so as to reduce space that needs to be scanned at each pass.

For face recognition part, we use Eigen Face Recognizer in OpenCV. We split the training image set to different nodes to enjoy data parallelism. For example, we have 800 labeled face images and 4 nodes, then we can deliver 200 images to each node. Every node will train an independent recognizer based on its training set. When a new face image come, we deliver it to every node, and every node will return a prediction result with confidence. We reduce these feedback to report the most confident one.

Inspired by HIPI[8], We want to implement an image bundle data structure. We pack float images into the bundle and the whole bundle can be exported as a text file, which is the default file format for Hadoop.

IV. Problem Statement

Facial recognition is a problem of determining and identifying faces in images using computers. People have worked on it for decades and there has been much of improvement, mostly in last decade with the advance of digital image and video recording devices. It is a popular area of research in computer vision and has applications in surveillance systems, marketing, video recording, social networks etc.

Obvious application in security systems is real-time location and identification of people being recorded on surveillance cameras. One possible application in marketing would be to have saved preferences for faces detected by the system and to show adds that the person is the most interested in. In digital video recording, such as taking photos with your phone or camera, detecting faces helps automatically adjust focus, lighting and zoom level to better capture people on the image taken. Social networks have been deploying facial recognition algorithms to find people on images. Those are just some of the examples of usage of facial recognition algorithms.

When it comes to face recognition algorithms, we usually divide them in two groups. One is group of algorithms for face detection, and the other is group of algorithms for face recognition.

Face detection is a problem of determining locations and size of faces in input images. Output is usually a set of rectangles placed on the original image such that each rectangle contains a face. // More to fill in

Face recognition is a problem of identifying a given face with faces in a database. User will input a face image and the system should return the users identity. Some system even return a list of users that look similar to the input one. We don't provide similar face list in our system.// More to fill in

Fig. 2. An illustration of image partitioning given the window size (maximum face size). Mesh with white lines is the cut based on window size. Green, yellow, red, purple and blue boxes represent examples of different scan areas. Scan areas overlap to make sure we don't miss any face on the boundary. Gray-shaded region represents parts of the image that cannot contain a face alone, because they are smaller than assumed face size.

V. Proposed Solution

Our project's work is separated into two parts. First part deals only with the problem of detecting faces in images.

For face recognition, we spread training images to different nodes to improve speed. In terms of mapreduce, each mapper will make its own prediction and emit (prediction, confidence) pairs as new value, see fig.3. The emit key is set to "KEY" so that every prediction comes to the same reducer. Each reducer will combine these pairs, in the way that only the most confident one remains. We also introduce a Combiner to locally reduce mapper results.

VI. Evaluation

In this section, you demonstrate that your solution is cool and it outperforms previous works.

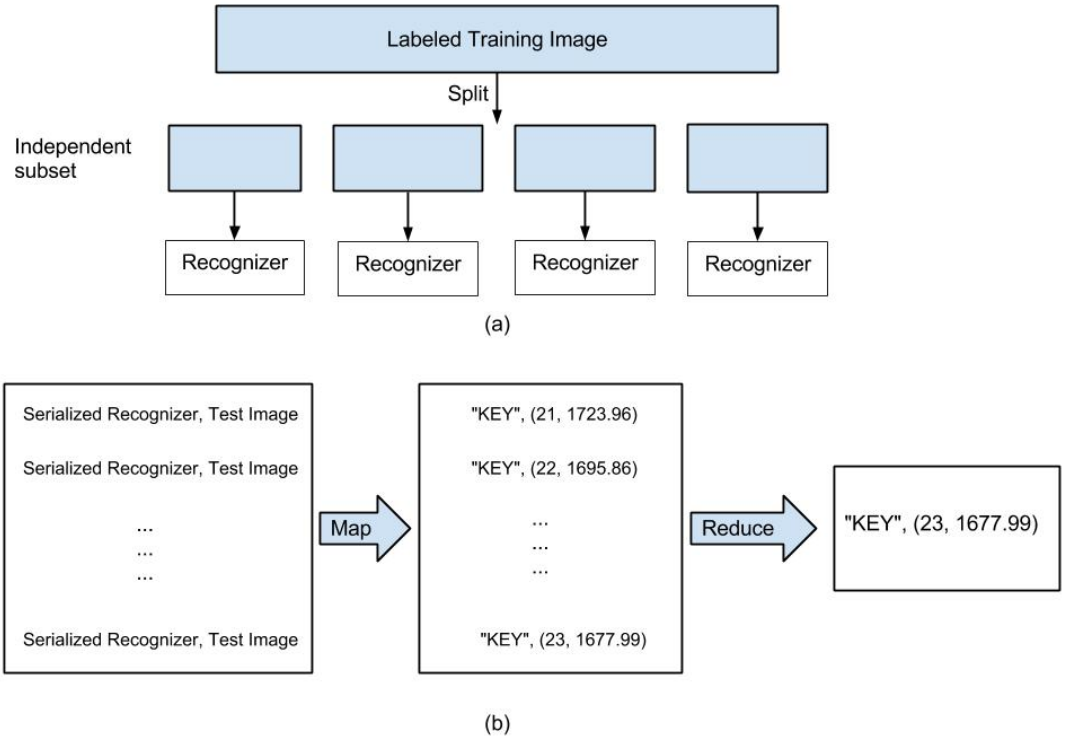

Fig. 3. Distribute face recognition. (a) The training set is split so that every computing node has its own training data. Independent recognizers are trained in the trainning stage. (b) When a test image comes, we generate the input file for hadoop(the leftmost rectangle). It contains serialized recognizer, which contains all the required data to make a prediction, and the test image itself. Every line of the input file is distributed to a different mapper as input, and the output will be the prediction result, in (predict, confidence) pair. Reducers and Combiners will take these output to find the most confident result.

VII. Conclusions and Future Work

What are the lessons that we should learn from this paper? What are the possible extensions of this work?

References

[1] P. Viola and M. Jones, "Robust real-time object detection," *Second International Workshop On Statistical And Computational Theories Of Vision Modeling, Learning, Computing, And Sampling*, July 2001.

[2] A. Pentland and T. Choudhury, "Personalizing smart environments: Face recognition for human interaction," The Media Laboratory, Massachusetts Institute of Technology, Tech. Rep., January 2000.

[3] M. Kirby and L. Sirovich, "Application of the karhunen-loeve procedure for the characterization of human faces," *IEEE Pattern Analysis and Machine Intelligence*, vol. 12, no. 1, pp. 103–108, 1990.

[4] S. B. Henry A. Rowley and T. Kanade, "Neural network-based face detection," *Computer Vision and Pattern Recognition*, 1996.

[5] C. Zhang and Z. Zhang, "A survey of recent advances in face detection," Microsoft Research, Tech. Rep. MSR-TR-2010-66, June 2010.

[6] N. Degtyarev and O. Seredin, "Comparative testing of face detection algorithms," 2010.

[7] A. Ranjan and S. Malik, "Parallelizing a face detection and tracking system for multi-core processors," *2012 Ninth Conference on Computer and Robot Vision*, 2012.

[8] C. Sweeney, L. Liu, S. Arietta, and J. Lawrence, "Hipi: A hadoop image processing interface for image-based mapreduce tasks," *Chris. University of Virginia*, 2011.