|  |
| --- |
| A Project Report On  **Online Transaction Security Using Face**  **Recognition**  Submitted in partial fulfilment of the requirement for the 8th semester  **Bachelor of Engineering**  in  Computer Science & Engineering  **VISVESVARAYA TECHNOLOGICAL UNIVERSITY,**  **BELGAUM**    *Submitted by*  **Kamal Shubham Yadav [1DS14CS714]**  **Rohit Ranjan [1DS14CS737]**  **Soumya Bhowmik [1DS14CS746]**  **Soumyajit Kundu [1DS14CS747]**  *Under the guidance of*  Prof. Dhara K.N  **Designation**  Assistant professor, Dept. of CSE, DSCE    **2017-2018**  **Department of Computer Science & Engineering**  **DAYANANDA SAGAR COLLEGE OF ENGINEERING**  **BANGALORE - 560078** |
| **VISVESVARAYA TECHNOLOGICAL**  **UNIVERSITY**  **DAYANANDA SAGAR COLLEGE OF ENGINEERING**  **Shavige Malleshwara Hills, Kumaraswamy Layout, Bangalore - 560078**  **Department of Computer Science & Engineering**    **CERTIFICATE**  This is to certify that the project online Transaction Security Using Face Recognition  is a bonafide work carried out by **Kamal Shubham Yadav [1DS14CS714],**  **Rohit Ranjan [1DS14CS737], Soumya Bhowmik [1DS14CS746]** and **Soumyajit**  **Kundu [1DS14CS747]** in partial fulfilment of 8th semester, Bachelor of Engineering in Computer Science and Engineering under Visvesvaraya Technological University, Belgaum during the year 2017-18.  **Dhara K N Dr. Ramesh Babu Dr. C P S Prakash**  (Internal Guide) Vice Principal & Head of Principal,  Asst.Prof, Department of Department, CSE, DSCE DSCE  CSE, DSCE  Signature:....................... Signature:....................... Signature:..................  Name of the Examiners: Signature with date:  1........................................... ....................................  2........................................... .................................... |
| **Abstract**  Waste sorting is the process of separating waste into different types. The current trend is to efficiently separate the waste in order to appropriately deal with it. The separation must be done as early as possible in order to reduce the contamination of waste by other materials. The need to automate this process is a significant facilitator for waste companies. Recent enforcement of law by the Indian government for the welfare of sanitation workers has raised the need for an automated system in waste management. The existing garbage disposal system in India consists of unclassified waste collected from homes which are then segregated at a station manually. This requires more man power as well.  The objective of this project is to take images of a single piece of garbage and classify it into four classes consisting of plastic, paper, metal and organic. We also create a dataset for training that contains around 400-500 images for each class, which was hand collected.  The proposed system uses Deep learning, YOLO model to achieve the objective. YOLO is a 30 layered neural network, which does object detection and classification. Input to the system is an image of the garbage and the resulting output image will be the type of the waste in the image with the bounding boxes. |
| **Acknowledgement**  We have immense pleasure in successful completion of this project on Waste Segregation using Image based Deep Learning  We would like to take this opportunity to express our gratitude to **Dr. C P S Prakash**, Principal of DSCE, for permitting us to utilize all the necessary facilities of the institution.  We are also very grateful to our respected Vice Principal, HOD of Computer Science & Engineering, DSCE, Bangalore, **Dr. Ramesh Babu D R**, for his support and encouragement.  We are immensely grateful to our respected and learned guide, **Dr. Krishnan R**, Professor CSE, DSCE and our co-guide **Ms. Dhara K N**, Assistant professor, CSE, DSCE for their valuable help and guidance. We are extremely thankful to them for all the encouragement and guidance they have given us during every stage of the project.  We would like to thank our project co-ordinators **Dr. Swetha M D**, Associate Professor, CSE, DSCE and **Ms. Rashmi S R**, Assistant Professor, CSE, DSCE for their guidance and support.  We are also thankful to all the other faculty and staff members of our department for their kind co-operation and help.  Lastly, we would like to express our deep appreciation towards our classmates and our family for providing us with constant moral support and encouragement.  Kamal Shubham Yadav [1DS14CS714]  Rohit Ranjan [1DS14CS737]  Soumya Bhowmik [1DS14CS746]  Soumyajit Kundu [1DS14CS747] |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| 1.1 | Working of a DeepNet . . . . . . . . . . . . . . . . . . . . . . . . . . | 2 |
| 2.1 | Block Diagram of The Proposed Solution . . . . . . . . . . . . . . . . | 5 |
| 2.2 | Phases in the Proposed Solution . . . . . . . . . . . . . . . . . . . . . | 6 |
| 2.3 | Use Case Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 8 |
| 4.1 | System Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 13 |
| 4.2 | System Block Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . | 14 |
| 4.3 | Data Flow Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 15 |
| 4.4 | Sequence Diagram . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 16 |
| 4.5 | DeepNet structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 16 |
| 5.1 | Directory structure . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 18 |
| 6.1 | Training Image: Paper . . . . . . . . . . . . . . . . . . . . . . . . . . | 22 |
| 6.2 | Training Image: Plastic . . . . . . . . . . . . . . . . . . . . . . . . . . | 23 |
| 6.3 | Training Image: Metal . . . . . . . . . . . . . . . . . . . . . . . . . . | 23 |
| 6.4 | Training Image: Organic . . . . . . . . . . . . . . . . . . . . . . . . . | 24 |
| 6.5 | Testing Image: Paper . . . . . . . . . . . . . . . . . . . . . . . . . . . | 24 |
| 6.6 | Testing Image: Plastic . . . . . . . . . . . . . . . . . . . . . . . . . . | 25 |
| 6.7 | Testing Image: Metal . . . . . . . . . . . . . . . . . . . . . . . . . . . | 25 |
| 6.8 | Testing Image: Organic . . . . . . . . . . . . . . . . . . . . . . . . . . | 26 |
| 7.1 | IOU and Recall per Image for 0.00001 learning rate . . . . . . . . . . | 28 |
| 7.2 | IOU and Recall per Image for 0.0001 learning rate . . . . . . . . . . . | 29 |
| 7.3 | IOU and Recall per Image for 100000 iterations . . . . . . . . . . . . | 30 |
| 7.4 | IOU and Recall per Image for 200000 iterations . . . . . . . . . . . . | 31 |
| 7.5 | IOU and Recall per Image for 300000 iterations . . . . . . . . . . . . | 32 |
| 7.6 | IOU and Recall per Image for 600000 iterations . . . . . . . . . . . . | 33 |
| 7.7 | IOU and Recall per Image for 800000 iterations . . . . . . . . . . . . | 34 |
| 7.8 | IOU and Recall per Image for 1000000 iterations . . . . . . . . . . . . | 35 |

**Contents**

[**1 Introduction** **1**](#_Toc49996)

[1.1 Garbage Segregation 1](#_Toc49997)

[1.2 Deep Learning 2](#_Toc49998)

[1.3 Organization of the Project Report 3](#_Toc49999)

[**2 Problem Statement and Proposed Solution** **4**](#_Toc50000)

[2.1 Problem Statement 4](#_Toc50001)

[2.2 Existing Systems 4](#_Toc50002)

[2.3 Proposed Solution 5](#_Toc50003)

[2.4 System Requirements 8](#_Toc50004)

[**3 Literature Survey** **9**](#_Toc50005)

[3.1 Object Detection and Classification 9](#_Toc50006)

[3.2 Waste Segregation using Deep Learning 11](#_Toc50007)

[**4 Architecture and Design** **13**](#_Toc50008)

[4.1 System Overview 13](#_Toc50009)

[4.2 Software Architecture 14](#_Toc50010)

[4.2.1 System Block Diagram 14](#_Toc50011)

[4.2.2 Data flow Diagram 15](#_Toc50012)

[4.2.3 Sequence Diagram 16](#_Toc50013)

[4.3 DeepNet Structure 16](#_Toc50014)

[**5 Implementation** **17**](#_Toc50015)

[5.1 Implementation Platform 17](#_Toc50016)

[5.1.1 Hardware 17](#_Toc50017)

[5.1.2 Software 17](#_Toc50018)

[5.2 Implementation Details 18](#_Toc50019)

[5.2.1 Organization of implementation files 18](#_Toc50020)

[5.2.2 Dataset Collection 18](#_Toc50021)

[5.2.3 Dataset Annotation 19](#_Toc50022)

[5.2.4 Configuration of DeepNet 19](#_Toc50023)

[5.2.5 Weight Extraction 20](#_Toc50024)

[5.2.6 Training the Yolo network 20](#_Toc50025)

[5.2.7 Testing the network 21](#_Toc50026)

[5.2.8 User Interface 21](#_Toc50027)

[**6 Testing** **22**](#_Toc50028)

[6.1 Testing on Training Dataset 22](#_Toc50029)

[6.1.1 Class: PAPER 22](#_Toc50030)

[6.1.2 Class: PLASTIC 23](#_Toc50031)

[6.1.3 Class: METAL 23](#_Toc50032)

[6.1.4 Class: ORGANIC 24](#_Toc50033)

[6.2 Testing on Test Dataset 24](#_Toc50034)

[6.2.1 Class: PAPER 24](#_Toc50035)

[6.2.2 Class: PLASTIC 25](#_Toc50036)

[6.2.3 Class: METAL 25](#_Toc50037)

[6.2.4 Class: ORGANIC 26](#_Toc50038)

[**7 Experiments and Results** **27**](#_Toc50039)

[7.1 Establishing Optimal parameters 27](#_Toc50040)

[7.1.1 Batch Size 27](#_Toc50041)

[7.1.2 Learning Rate 28](#_Toc50042)

[7.1.3 Number of Training Iterations 30](#_Toc50043)

[7.2 Optimal Parameters 36](#_Toc50044)

[**8 Conclusion** **37**](#_Toc50045)

[**References** **38**](#_Toc50046)

[**Appendix :Code** **39**](#_Toc50047)

# Introduction

## Online Transaction Security

Since information of users can leak easily, many authentication approaches are proposed to secure data and systems. Authentication method should be able to “declare anyone that they are who they claim to be”. However, it is not important in authentication process that what identity is authorized for. Moreover, they can be disclosed and stolen by intimates or colleagues. Some systems use an object such as passport or ID card to identify a user. It is practical in use but also is very easy to be stolen or copied. Therefore, biometrics which is unforgettable and more unique becomes a choice for secure process. Biometric technologies can be fingerprint, face, iris, palm print or even hand geometry. Among these technologies, face image seems to be one of the most popular used features due to its usability, collectability and acceptability. For usability and collectability, face technology is non-intrusive technology as test subjects are not required for a direct contact. For acceptability, many studies claim that their correct recognition rates are more than 95%. Other biometrics such as fingerprints yields high matching accuracy rate as well. However, fingerprint can be copied by using plastic sheet and created 3-dimension fake fingerprint. Moreover, fingerprint images that are stored in a database need to be high-quality images for accuracy of recognition process, so this takes a lot of capacity in database in case of applying to a system with a very large number of persons. On the other hand, iris scanning seems to be the most secure and accurate method for biometric technologies. However, iris recognition machines are expensive.

Here we are using Local Binary Pattern (LBP) as our base face recognition algorithm. This algorithm is efficient and has been customized for our specific purpose and more efficient face recognition. It is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

We implement the LBPH using OpenCV. OpenCV was started at Intel in 1999 by **Gary Bradsky** and the first release came out in 2000. Right now, OpenCV supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day.

Currently OpenCV supports a wide variety of programming languages like C++, Python, Java etc and is available on different platforms including Windows, Linux, OS X, Android, iOS etc. Also, interfaces based on CUDA and OpenCL are also under active development for high-speed GPU operations. OpenCV-Python is the Python API of OpenCV. It combines the best qualities of OpenCV C++ API and Python language.

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability.

## Organization of the Project Report

The project report is organized as follows:

In Chapter (2), we discuss about the problem statement and our solution to the problem. The same chapter also deals with the other existing state of arts. The Chapter that follows i.e chapter (3) consists of the details on the literature survey of the papers to the problem statement and the proposed solution. In Chapter (4), we present the System Overview and Proposed system in the form of Data flow diagram and the sequence diagram. The next chapter, chapter (5)gives the requirements and detains about the implementation of the proposed system. Chapter 6 deals with the testing of the system and their results. In Chapter (7), we discuss about the influencing parameters and their effect on the system. The same chapter deals with establishing the optimal parameters for the system. The chapter (8) concludes the paper along with mention of the Future Enhancements. Chapter (9) is details about the references made during the development of the system. The other supporting information and the source code are gathered in the Appendix.

# Problem Statement and Proposed Solution

## Problem Statement

There has been no full proof method to “authenticate” a customer in an online credit card transaction. Authentication is the verification of a credit card owner made during a credit card purchase. In the physical world authentication is achieved through a physical signature which is manually checked at the point of sale. Without effective authentication, following problems may arise like fraudulent transaction, lengthy process of transaction including lack of confidence for customers, higher cost of transactions etc.

## Existing Systems

A simple multi-factor authentication setup involves asking a user for their username and password (something they know) as well as verifying their identity through a second factor such as an SMS message to their phone (something they have). That covers two factors of authentication, but adding in image recognition as well adds an extra layer of security to the login process without making it frustrating or overly complicated for authorized users.

Many banks use image recognition as part of the multi-factor authentication process so their customers can login to their accounts and authorize various financial transactions in a secure environment. On the web, this sort of image recognition authentication is ideal for preventing phishing attacks where another website could mimic the look and feel of your bank. Even if a phishing site looked identical to your bank’s log in page, it would not know what your authentication image looked like, and would not be able to fool users as easily on that page of the process. Also, that sort of phishing attack would render the bank username and password useless. Even with the password, the data thief would not be able to log in to the account without also knowing the authentication image, which they would not be able to steal in a phishing scheme.

Facebook also uses a form of image authentication. The site uses this feature primarily during the first time you log in to an account from a previously unrecognized location or computer. Instead of asking you to answer a security question you previously submitted an answer to, Facebook asks you to name a number of your friends by their profile pictures. This sort of image recognition would deny access to your account unless the hacker was also familiar with the identities of your friends.

Image-based authentication (IBA) can solve many of the common problems with passwords. IBA as a password substitute works primarily by accepting a valid username and then presenting an image set. The user than must identity the correct image or images from the set to be allowed into the account. Another image-based authentication factor could be something like implementing captcha. That can effectively reduce issues of automated brute force attacks since machines are not able to solve the CAPTCHA images reliably. Identifying the person entering a password as human is a strong first step towards security.

## Proposed Solution

In this project, we use standard implementation techniques, face recognition using Local Binary Pattern Histogram algorithm, which has been implemented using OpenCV Python.

**LBPH: (Local Binary Pattern Histogram)**

**Local Binary Pattern**(LBP) is a simple yet very efficient texture operator which labels the pixels of an image by thresholding the neighborhood of each pixel and considers the result as a binary number.

LBP is combined with histograms of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

**Parameters**: the LBPH uses 4 parameters:

* Radius:
* Neighbors
* Grid X
* Grid Y

**Radius**: the radius is used to build the circular local binary pattern and represents the radius around the central pixel. It is usually set to 1.

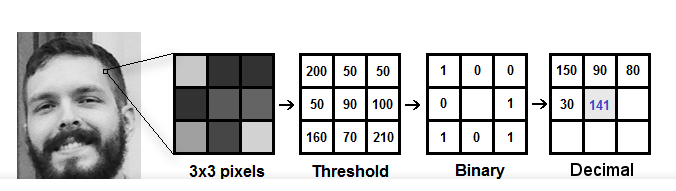
**Neighbors**: the number of sample points to build the circular local binary pattern. Keep in mind: the more sample points you include, the higher the computational cost. It is usually set to 8.

**Grid X**: the number of cells in the horizontal direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

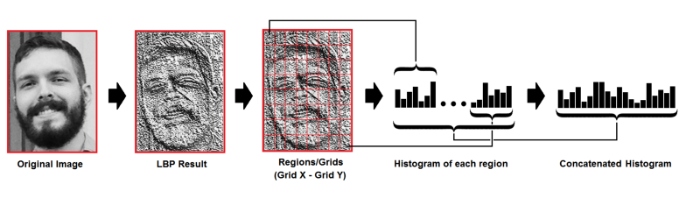
**Grid Y**: the number of cells in the vertical direction. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector. It is usually set to 8.

**Training the Algorithm**: First, we need to train the algorithm. To do so, we need to use a dataset with the facial images of the people we want to recognize. We need to also set an ID (it may be a number or the name of the person) for each image, so the algorithm will use this information to recognize an input image and give you an output. Images of the same person must have the same ID. With the training set already constructed, let’s see the LBPH computational steps.

**Applying the LBP operation**: The first computational step of the LBPH is to create an intermediate image that describes the original image in a better way, by highlighting the facial characteristics. To do so, the algorithm uses a concept of a sliding window, based on the parameters **radius** and **neighbors**.



**Extracting the Histograms**: Now, using the image generated in the last step, we can use the **Grid X** and **Grid Y** parameters to divide the image into multiple grids



We can use various approaches to compare the histograms (calculate the distance between two histograms), for example: **euclidean distance**, **chi-square**, **absolute value**, etc.

**OpenCV Python**

OpenCV-Python is the Python API of OpenCV. It combines the best qualities of OpenCV C++ API and Python language.

Python is a general purpose programming language started by Guido van Rossum, which became very popular in short time mainly because of its simplicity and code readability. It enables the programmer to express his ideas in fewer lines of code without reducing any readability.

Compared to other languages like C/C++, Python is slower. But another important feature of Python is that it can be easily extended with C/C++. This feature helps us to write computationally intensive codes in C/C++ and create a Python wrapper for it so that we can use these wrappers as Python modules. This gives us two advantages: first, our code is as fast as original C/C++ code (since it is the actual C++ code working in background) and second, it is very easy to code in Python. This is how OpenCV-Python works, it is a Python wrapper around original C++ implementation.

And the support of Numpy makes the task more easier. Numpy is a highly optimized library for numerical operations. It gives a MATLAB-style syntax. All the OpenCV array structures are converted to-and-from Numpy arrays. So whatever operations you can do in Numpy, you can combine it with OpenCV, which increases number of weapons in your arsenal. Besides that, several other libraries like Matplotlib which supports Numpy can be used with this. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems.

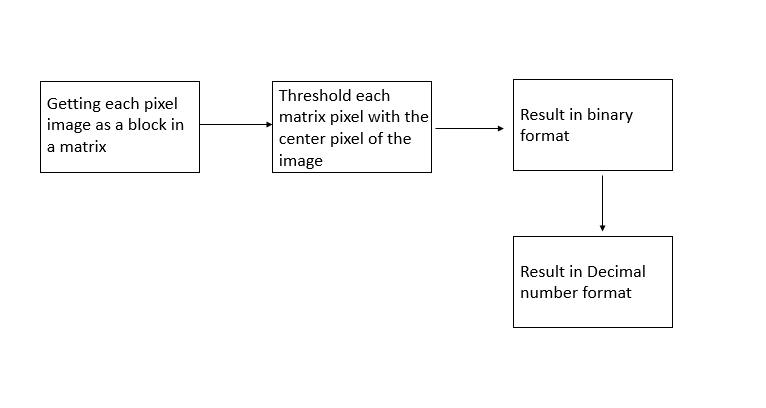


Figure 2.1: Block Diagram of The Proposed Solution

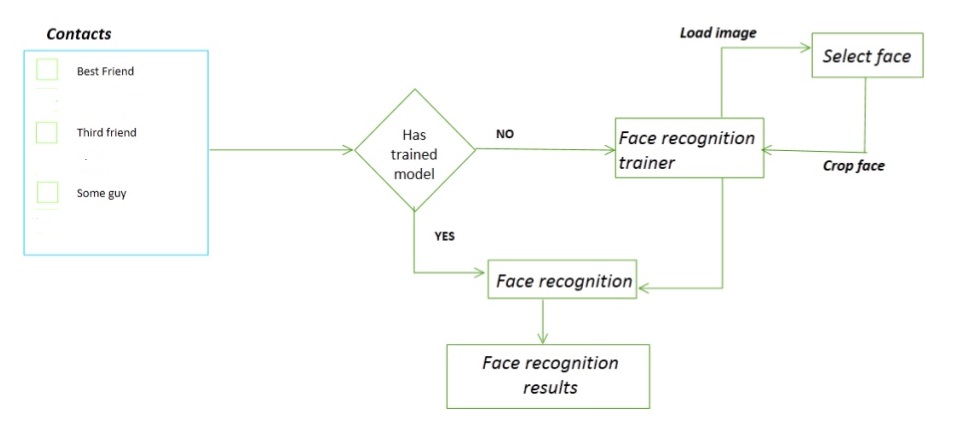


Figure 2.2: Phases in the Proposed Solution

## System Requirements

The system requirements for the proposed system is represented using the Use Case Diagram in Fig.2.3.

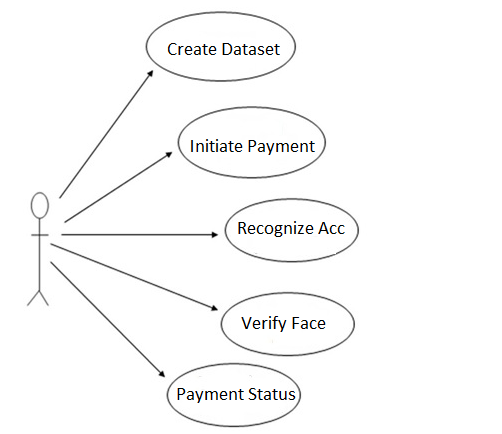


Figure 2.3: Use Case Diagram

The system should be able to:

1. Create datasets for face recognition & save credit card details.
2. Initiate payment process gateway.
3. Enter credit card details, user account is automatically recognised.
4. Take input image from real time and match it with the correct user.
5. Verify user and make payment.
6. Capture user’s image if face id doesn’t match.

# Literature Survey

**3.1 Enhancing User Authentication of Online Credit Card Payment**

**Authors: Gittipat Jetsiktat, Sasipa Panthuwadeethorn**

Description Popularity of online transaction in the half past decade leads to leak of user information. Security of transaction process can be hacked more easily due to advanced technologies. Biometric verification is considered as a key to enhance security problem. In this paper, the new process with face matching verification is proposed to improve security of online payment system. The simulation of the online payment process is also created then performance of the new proposed process is evaluated. The experimental results show that the new proposed process with face matching verification can increase security as well as improve the usability, capability and user satisfaction of online transaction process

**Credit Card Fraud Detection Based on Transaction Behavior**

**Authors: John Richard D. Kho,** **Larry A. Vea**

Description The proliferation of the EMV (EuropayMasterCard-VISA) chip card design in the credit card business mostly resolved the problem posed by the old Magnetic stripe card technology. However, several papers are starting to question the design and implementation of the EMV. This paper is suggesting that a detection model must be available to capture the possible anomalous transactions – a fallback in case the technology will fail. Several classifiers were evaluated during the model creation however only the Random Tree and J48 yielded the highest accuracy value of 94.32% and 93.50% respectively. By thorough analysis of these two (2) classifiers, it shows that the J48 is more fit in understanding the transaction logs data. [3] YOLO9000: Better, Faster, Stronger Authors: Joseph Redmon, Ali Farhadi.

**3.2 Fast and Efﬁcient Implementation of Convolutional Neural Networks**

**Authors: Abhinav Podili, Chi Zhang, Viktor Prasanna**

**Description: :** This paper proposes a deep convolutional neural network architecture code named Inception, for classification and detection in the Image Net Large Scale Visual Recognition Challenge 2014 (ILSVRC14). This paper focus on an efficient deep neural network architecture for computer vision, codenamed Inception, which derives its name from the Network in network paper by Lin et al in conjunction with the famous we need to go deeper internet meme. Network-in-Network approach is applied on the convolutional layers to increase the representational power of neural networks. The method could be viewed as additional 11 convolutional layers followed typically by the rectied linear activation. The most straight forward way of improving the performance of deep neural networks is by increasing their size. This includes both increasing the depth, the number of levels, of the network and its width: the number of units at each level. An Inception network is a network consisting of modules of the above type stacked upon each other, with occasional max-pooling layers with stride 2 to halve the resolution of the grid. For technical reasons (memory efficiency during training), it seemed beneficial to start using Inception modules only at higher layers while keeping the lower layers in traditional convolutional fashion.

#### Convolutional Neural Network Approach for Vision Based Student Recognition System

#### Authors: Nusrat Mubin Ara, Md. Saiful Islam

**Description:** Computers are now too smart to interact with the human in diﬀerent approaches. This interaction will be more acceptable for both human and computer if it is based on recognition process. In this article, author’s concern is to integrate and develop a student recognition system using existing algorithms. Among various face recognition methods, here author use deep learning based face recognition method. This method uses Convolutional Neural Networks (CNN) to generate a low dimensional representation called embeddings. Then those embeddings are used to classify the person’s facial image.By this system diﬀerent types of applications like student attendancesystem, building security etc. can be developed. After building the system, a resultant performance is also showed in this article. Face recognition with deep neural networks or convolutional neural networks is a very popular technique today. This method gives the higher accuracy and precision comparing the previous methods. This method is used in Facebook’s DeepFace [11] and Google’s FaceNet [12]. Also, there is an open source project called openface [13], which provide a python based API for face recognition. The idea to generate embeddings using convolutional neural network was invented in 2005 [12]. There are lots of other research about producing embeddings using convolutional neural network. Such as Visual Geometry Group (VGG) Face Descriptor [14] and Lightened Convolutional Neural Networks (CNNs) [15], and they have their won implemented system. To training a neural network for generating embedding, lots of face data is necessary. Facebook, Google has their own dataset. Openface also releases some trained neural network by training it with some open facial dataset.

#### Facial Expression Recognition via Deep Learning

#### Authors: Abir Fathalla, Gary Thung, Ali Doui

**Description:** Interaction Human-Machine (IHM) has long conﬁned researches to develop techniques based on the use of triplet screen-keyboard-mouse. Today, it is moving towards new paradigms: the user must be able to evolve unimpeded in its natural environment; ﬁngers,hand, face or familiar objects are seen as many devices input/output; the boundary between the physical and electronic worlds is blurring. These new forms of interaction require usually the capture of the observable behavior of the user and his environment. That is why they rely on artiﬁcial perception techniques, including computer vision. IHM is a rapidly evolving discipline. Future generations of human-machine environment will become multimodal integrating new information, from the consideration of the dynamic behavior, speech and/or facial expressions, so as to make the use of machines the most intuitive and natural as possible. The face is the most expressive and communicative part of a human being [1], it represents a major focus in current research concerning the improvement of IHM for establishing a dialogue between the two entities. Facial expression is a visible manifestation of a face from the state of mind (emotion, reﬂection), cognitive activity, physiological (fatigue, pain), personality and psychopathology of a person. The essential of facial expression information

is contained in the deformation of main permanent facial features, characterized by a change visually perceptible. Today, analysis computer assisted of face and its facial expressions is an emerging ﬁeld. Emotion recognition consists in associating an emotion to face image. So the goal is to determine from a face, the internal emotional state of a person. An automatic facial expression recognition system is an important component in human machine interaction. It consists to evaluate the possibility of emotions recognition. However, this is not an easy task. Facial expression recognition usually employs a three-stage training consisting of face Acquisition [2], facial feature extraction [3] and classiﬁer construction [4, 5]. Recently, Many works [6, 7] demonstrated that expression recognition can beneﬁt from collecting two stages together facial feature extraction and classiﬁer construction. Deep learning methods have been successfully applied to extract features and classiﬁcation, in particular Convolutional Neural Networks (CNN) architectures which are biologicallyinspired multi-stage one that learned automatically hierarchies of invariant features [8]. The ConvNets consist of a multistage processing of an input image to extract hierarchical and high-level feature representations. Motivated by this, we present in this paper an effective approach system based on ConvNets for facial expression recognition. We proposed a new architecture which the input of the system is an image; then, we use CNN to predict the facial expression label which should be one these labels[9]: anger,happiness, sadness, disgust, surprise and neutralSystem Overview

## Software Architecture

### System Block Diagram

Figure 4.2: System Block Diagram

The overall block diagram of the proposed system is shown in the Fig.4.2.

-User inputs the image to the system through UI.

-The overall system is represented as a block DeepNet. It consists of: Input processing and DeepNet processing layers.

-Input image is processed and then fed to the DeepNet layers for object detection and classification.

-Output image will be displayed in the UI.

### Data flow Diagram

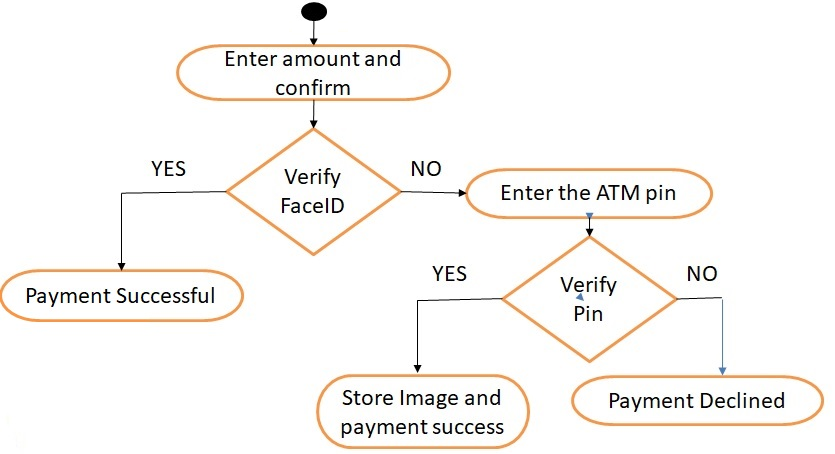


Figure 4.3: Data Flow Diagram

A data flow diagram (DFD) is a graphical representation of the flow of data through an information system. A DFD gives the preliminary overview of the system without going into great detail. Fig.4.2.1 represents the DFD of our proposed system. The flow of the system is as follows:

1. User enters amount and confirms.
2. Face verification is done by comparing input image with datasets.
3. If face is successfully verified, payment is successful.
4. If face verification fails, pin code is asked.
5. If pin code is verified, face image is captured for verification and payment is successful.
6. If not verified, payment is declined.

### Sequence Diagram

Figure 4.4: Sequence Diagram

A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. Fig.4.2.2 represents the sequence diagram that shows the interaction between the User, interface, server and the Trained DeepNet. It gives the sequence of actions from the moment of user entering the image till the resulting image displayed to the user.

## DeepNet Structure

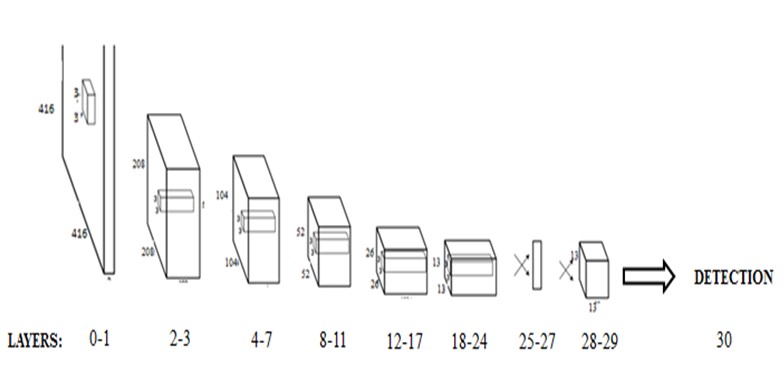


Figure 4.5: DeepNet structure

The DeepNet structure is represented in Fig.4.2.3. It has 30 layers, combination of convolution and max pool layers.

Each block in the diagram consists of a number of convolution layers followed by one maxpool layer.

# Implementation

## Implementation Platform

### Hardware

• **Processor:** Intel Duo Core

#### • RAM: 8 GB

• **GPU:** NVIDIA Tesla K80

The above requirements were satisfied by the use of **gcloud** where in the training could be done on an instance that matches the necessary requirements as mentioned above.

### Software

* **Operating System:** Ubuntu (14) (64bit), Windows 10
* **Programming Languages:** C, Python, JavaScript, CSS, php5, html5
* **Deep Learning Framework:** Darknet
* **Server:** Apache2

## Implementation Details

### Organization of implementation files

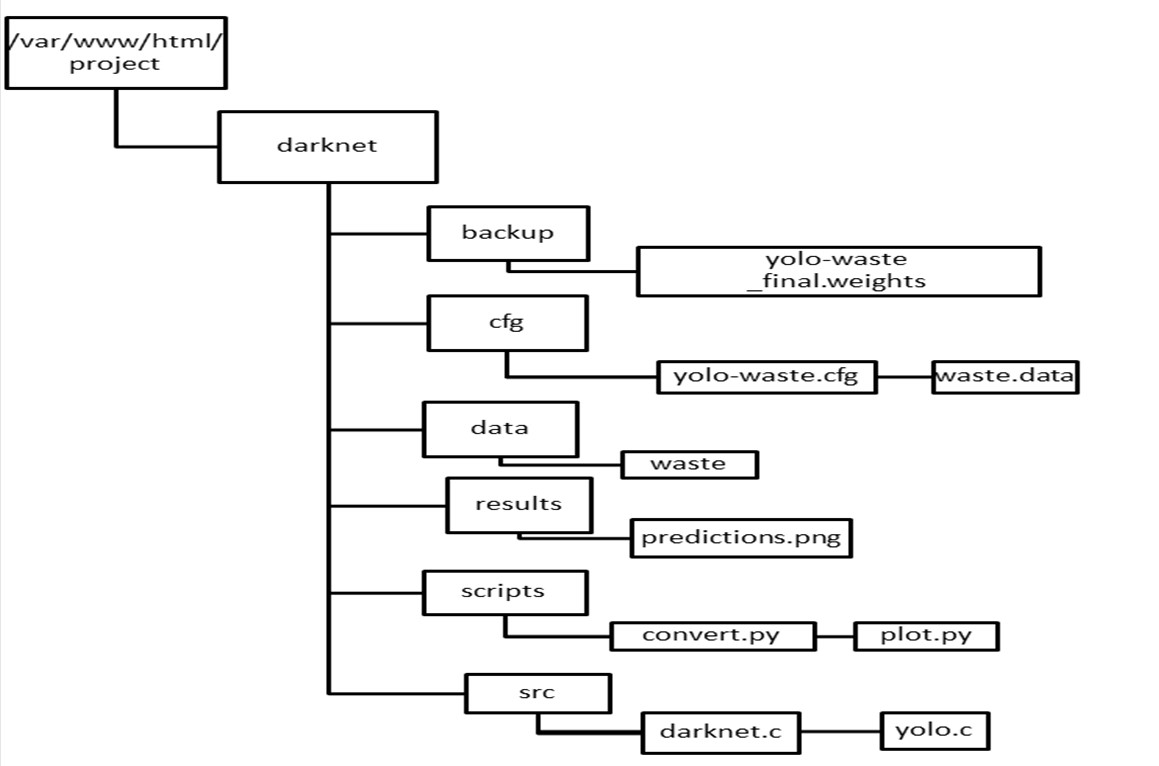


Figure 5.1: Directory structure

The above diagram shows the organization of the implementation files. The detailed explanation of the figure is given in further sections.

### Dataset Collection

The dataset collection process is one of the most important steps of the system. The user while creating his details of the account along with the credit card details, creates a dataset of his face. The camera continuously captures around 40 images in a span of few seconds. This is stored in the database which is linked with the user’s new id. The images taken are first converted to greyscale, then contour mapping is done on them. This enhances the ability of the applied algorithm to recognize facial patterns more efficiently. This enables the face recognition algorithm to quickly recognise a face all in a fraction of a second. The Haarcascade Classifier does the main task of creating the dataset. This classifier has an exceptional ability to pick up images from real time footage using the webcam.

### Training the Datasets

Once the datasets are created, the datasets have to be trained using their respective IDs which has to correspond to the users card details. All this is stored in a separate file named **trainer.yml**

### User Interface

Setting up **LAMP** on gcloud for launching web interface for the project can be done by installing apache2 and php5 and transferring the UI files into the instance.

The code files are then saved in the /var/www/html folder so as to be accessible by the server.

# Testing

Testing is carried out with 20% of the images from the dataset. The accuracy of the bounding boxes in the results and the prediction percentage depends on the learning rate and number of iterations for training. The following cases describe the results on testing data for different weights obtained from different learning rates and iterations.

## Testing on Training Dataset

Images from the training dataset were tested on the network and the results were as follows:

### Class: PAPER

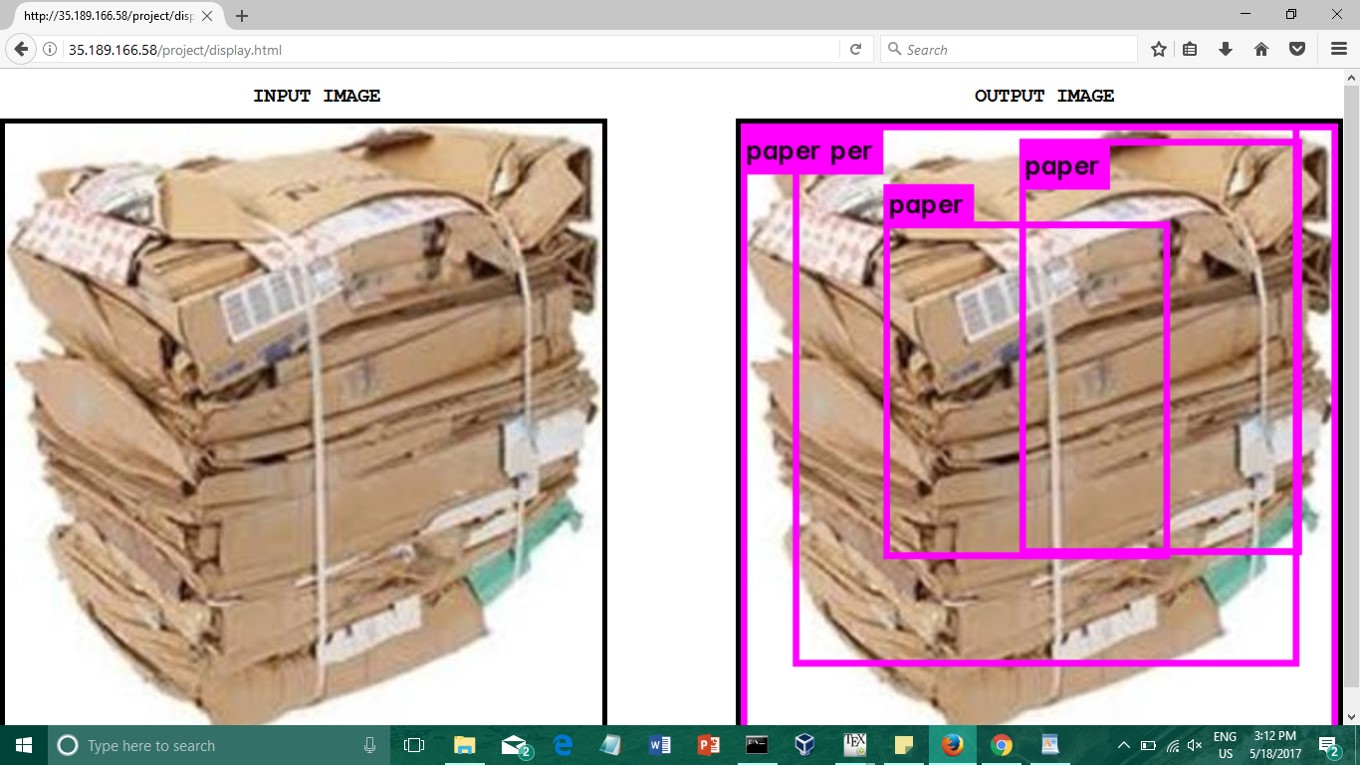


Figure 6.1: Training Image: Paper

### Class: PLASTIC

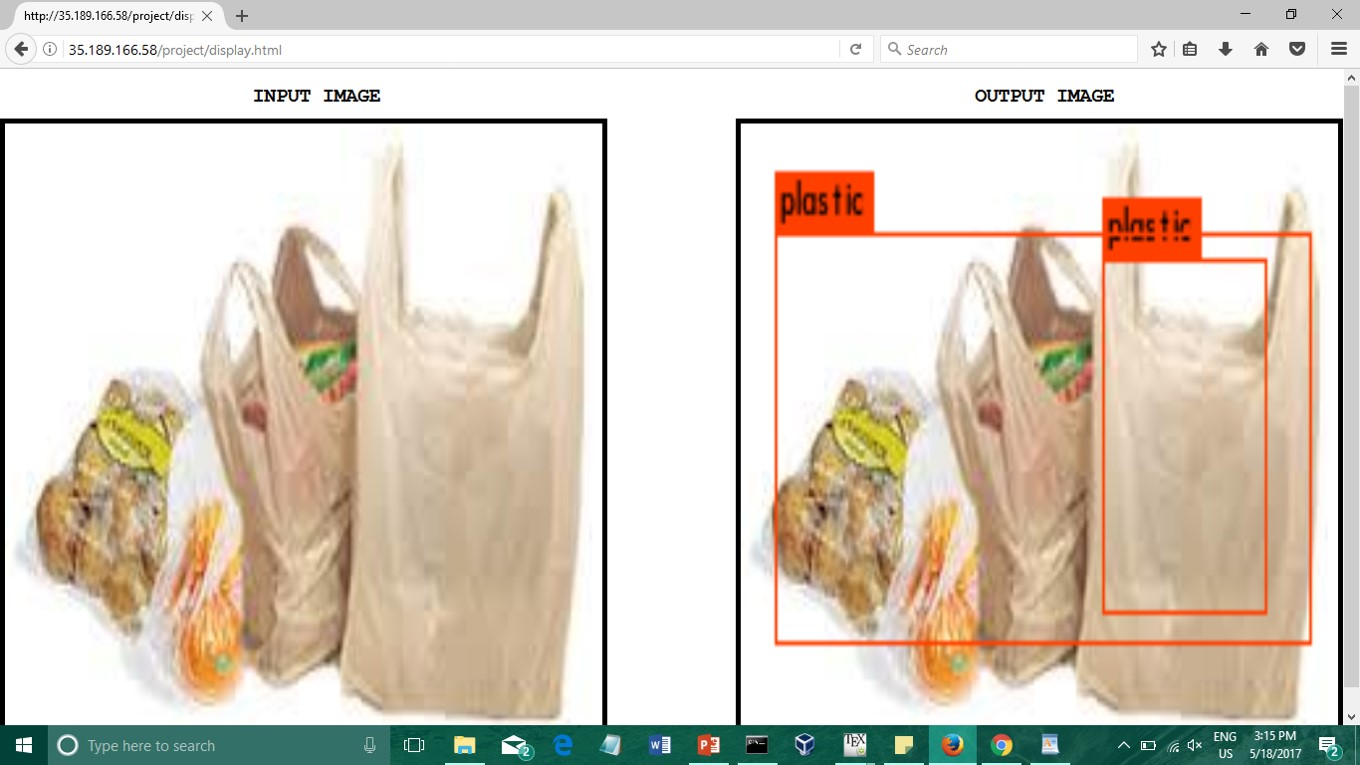


Figure 6.2: Training Image: Plastic

### Class: METAL

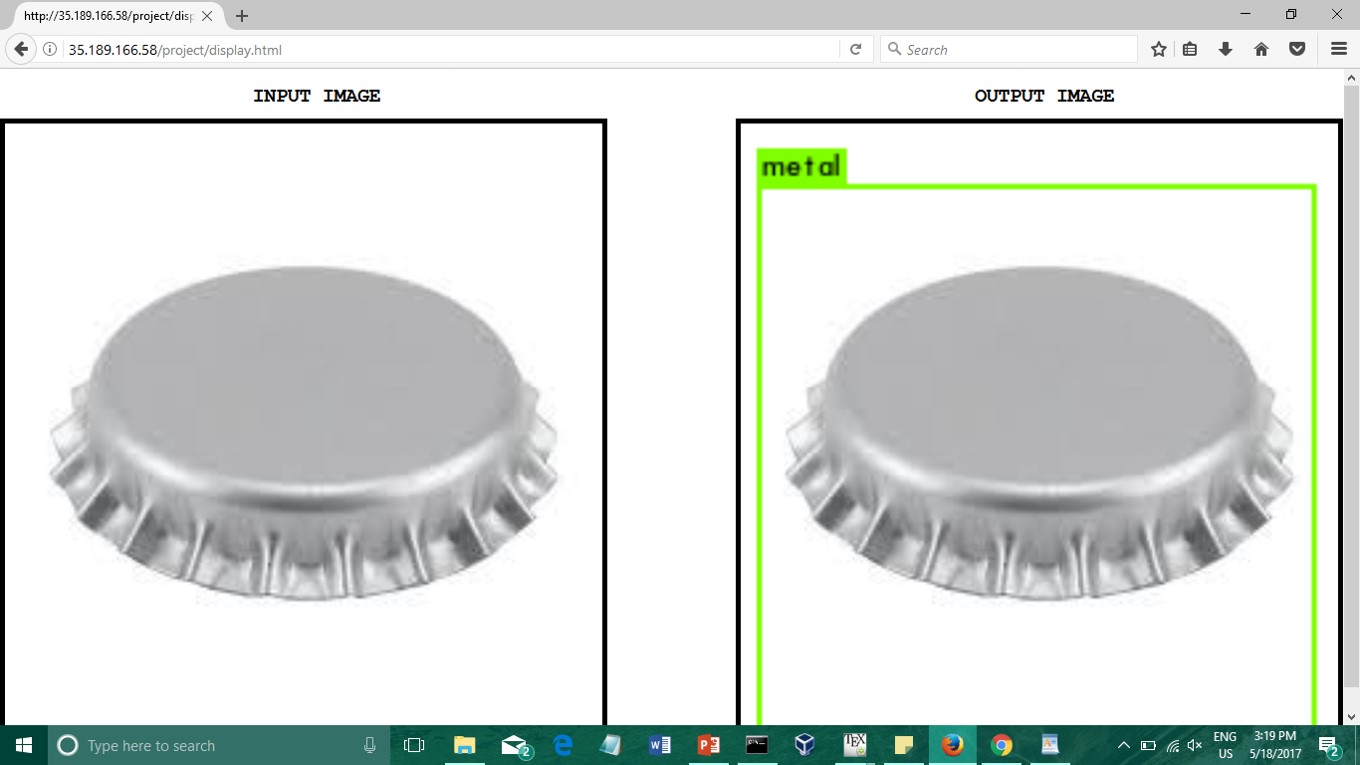


Figure 6.3: Training Image: Metal

### Class: ORGANIC

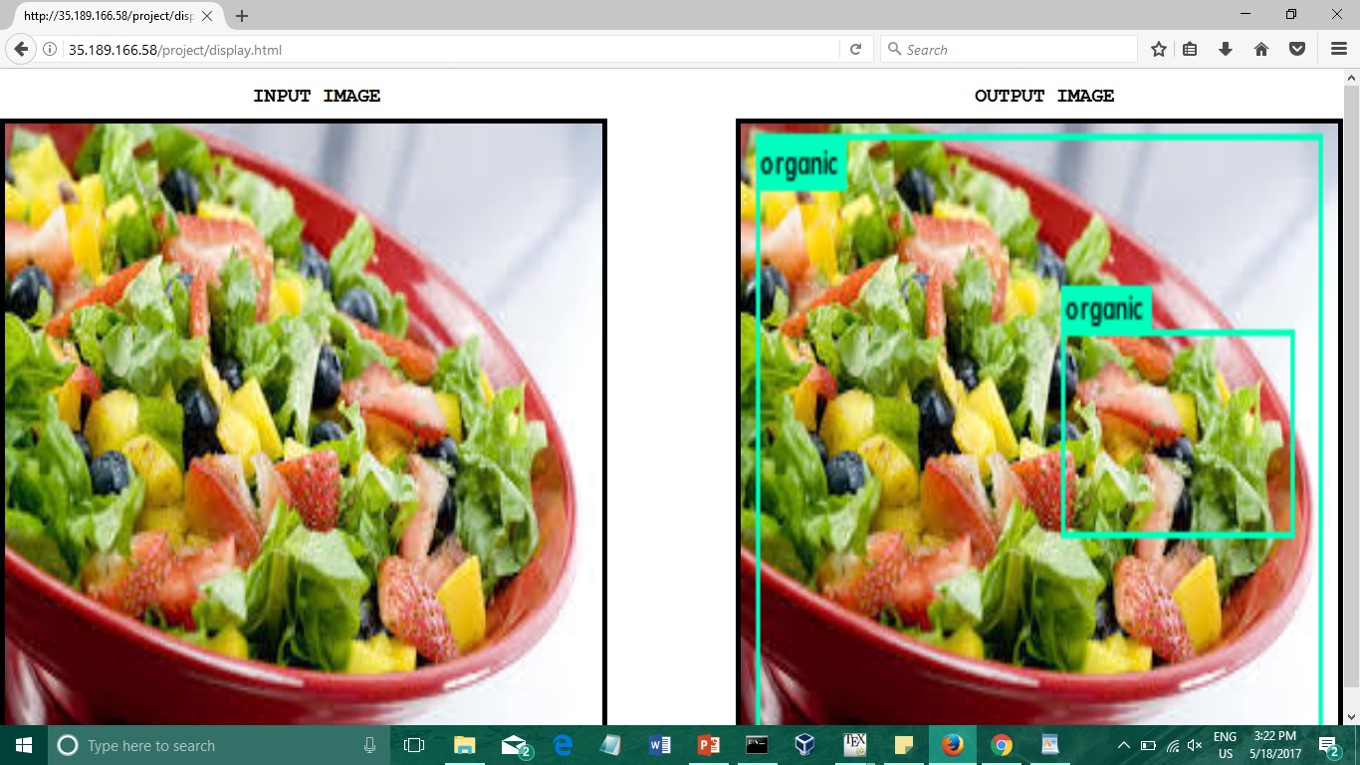


Figure 6.4: Training Image: Organic

## Testing on Test Dataset

Images from the test dataset, when tested on the network with pre-trained gave the following results:

### Class: PAPER

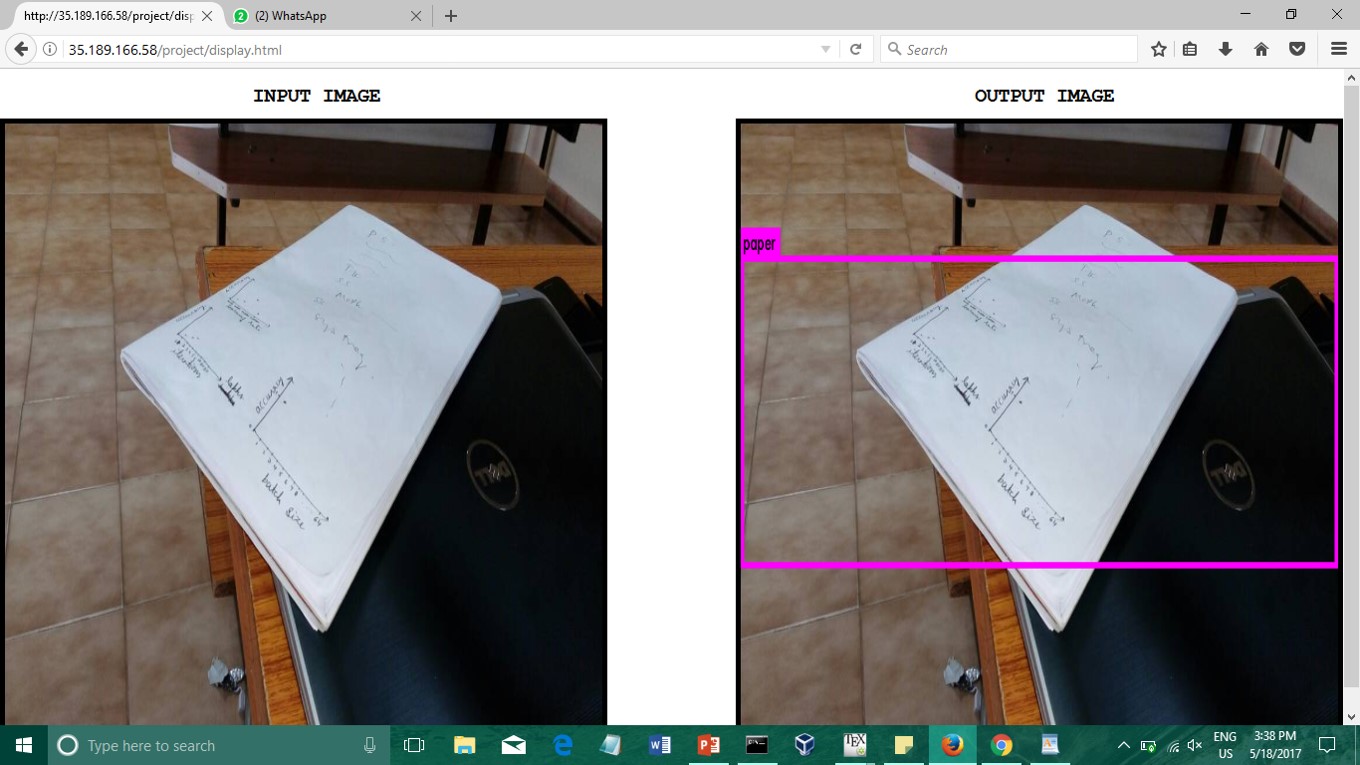


Figure 6.5: Testing Image: Paper

### Class: PLASTIC

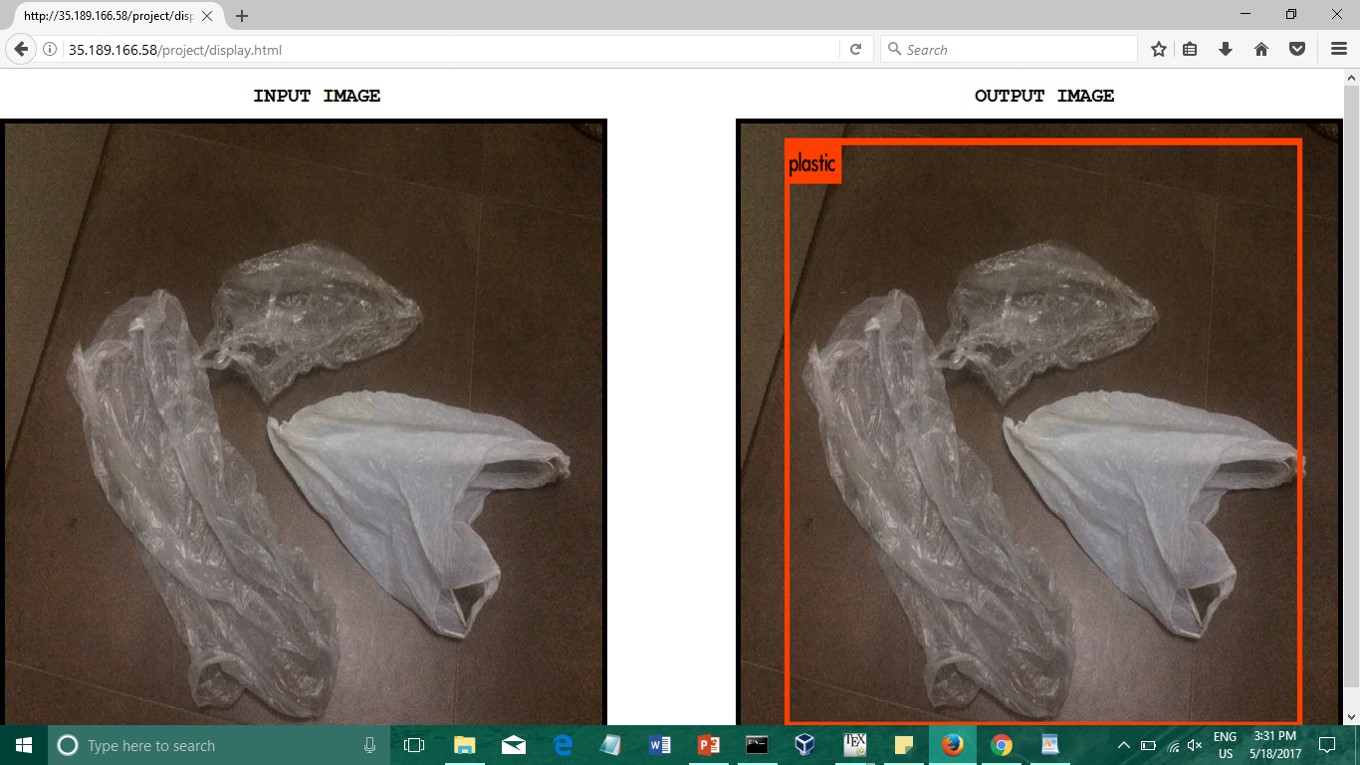


Figure 6.6: Testing Image: Plastic

### Class: METAL

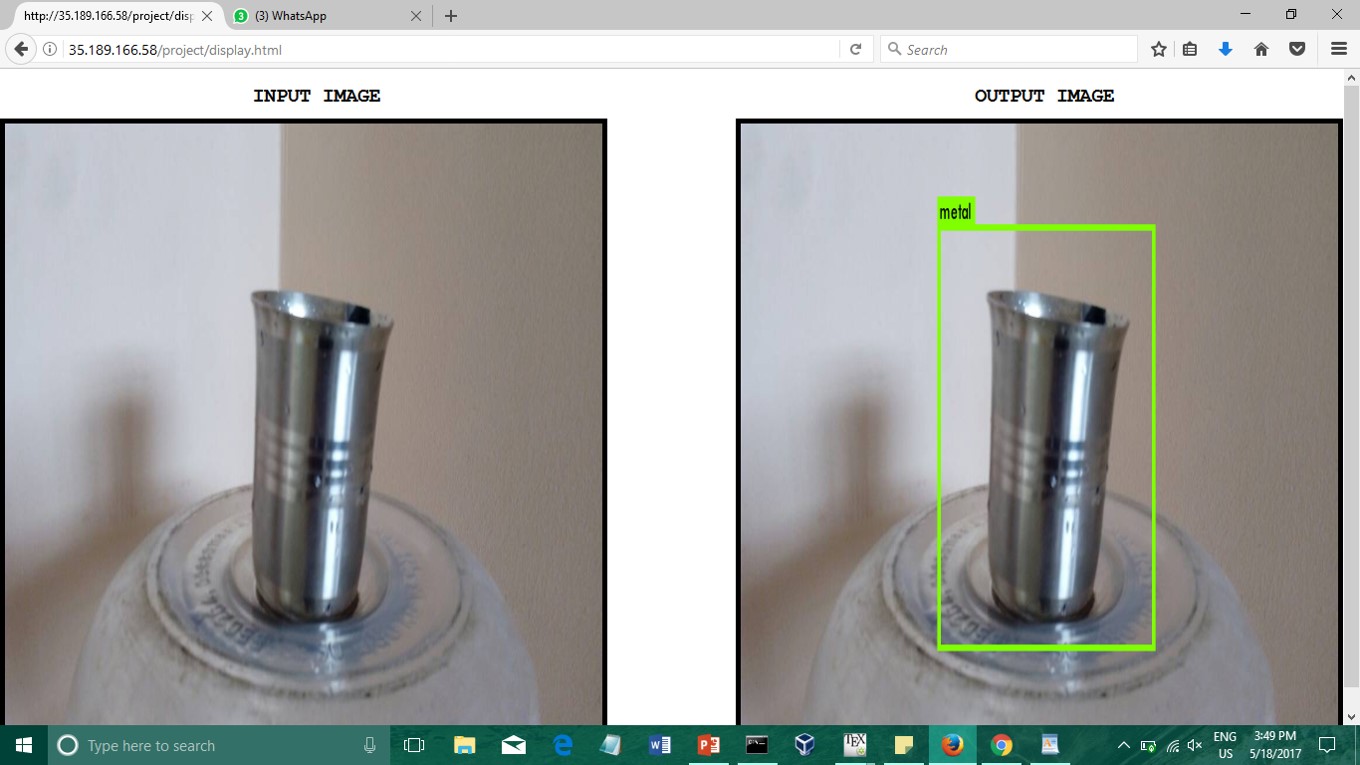


Figure 6.7: Testing Image: Metal

### Class: ORGANIC

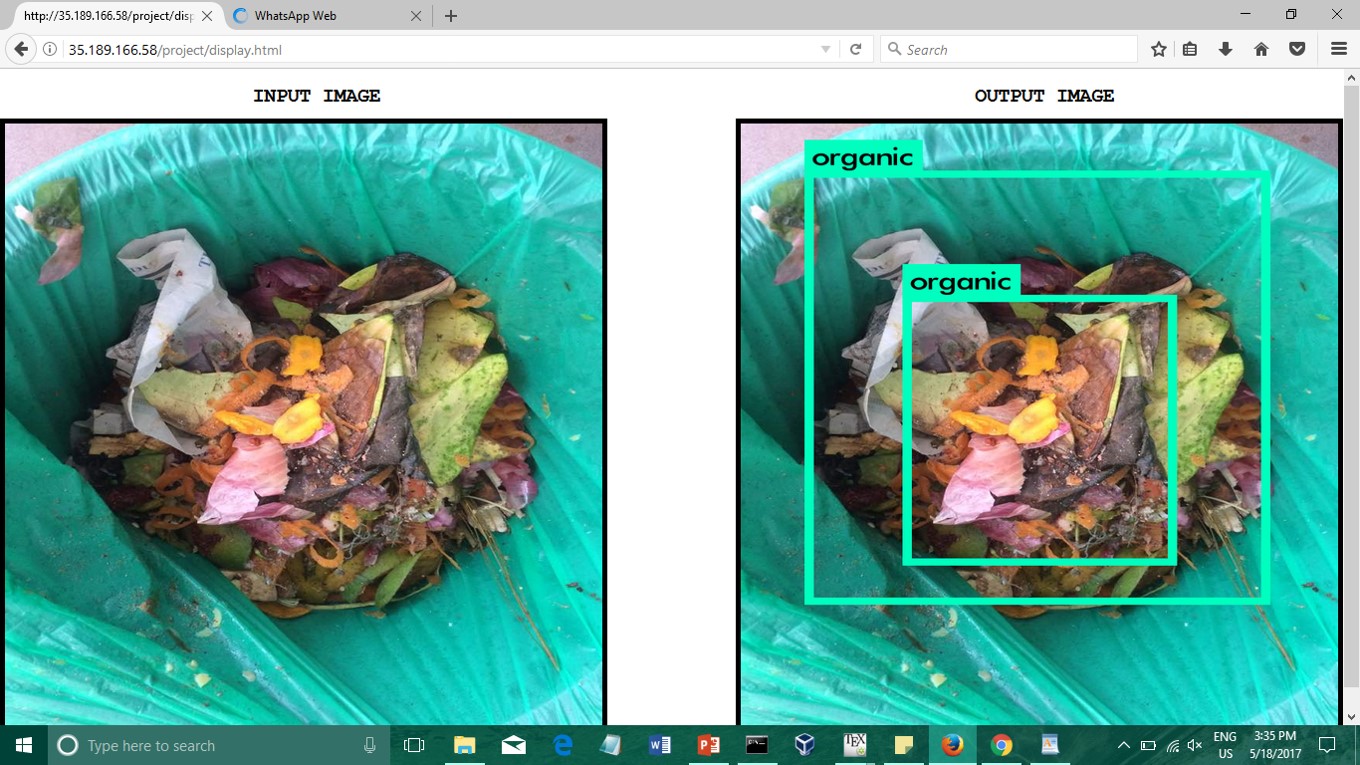


Figure 6.8: Testing Image: Organic

# Experiments and Results

The prediction percentage and the accuracy of the bounding boxes in the results depends on the 1) Batch size

1. Learning rate
2. Number of training iterations

**Batch size** is the number of images that are trained per batch in one iteration of training.

**Learning Rate** is the training parameter that controls the size of weight and bias changes during learning.

**Number of Iterations** is the number of training iterations after which the network is optimally trained.

**IOU:** Intersection over Union is an evaluation metric used to measure the accuracy of an object detector on a particular dataset. In the numerator we compute the area of overlap between the predicted bounding box and the ground-truth bounding box. The denominator is the area of union, or more simply, the area encompassed by both the predicted bounding box and the ground-truth bounding box. Dividing the area of overlap by the area of union yields our final score the Intersection over Union.

## Establishing Optimal parameters

This section deals with experimenting with various training parameters are carried out to decide on an optimal set of parameters.

The effects of the various changes and the issues are discussed in Section.7.2

### Batch Size

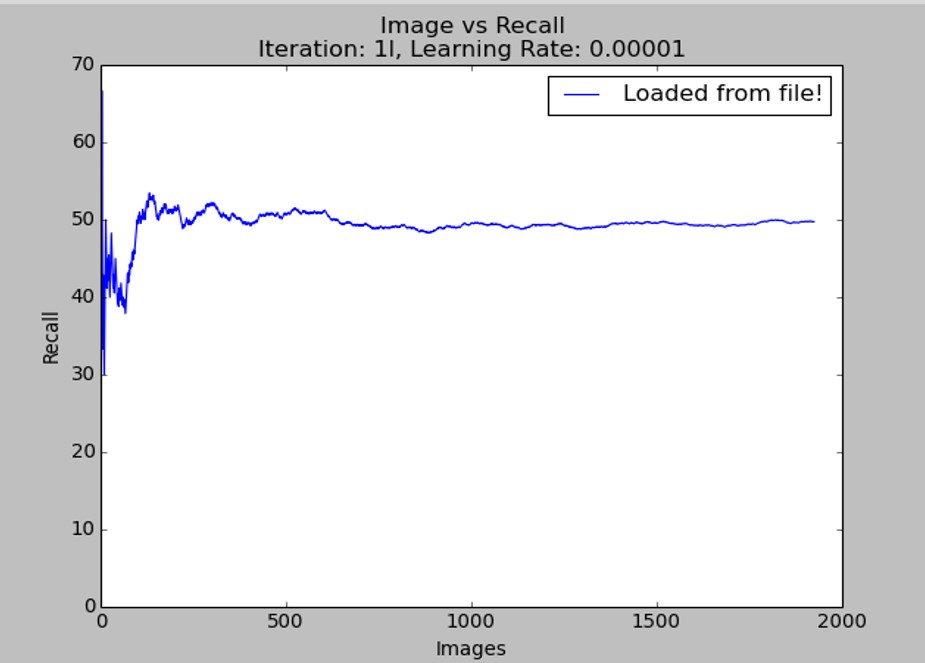
* Batch sizes were set to 64, 8 and 1 and experiments were carried out.
* Batch sizes of 64 and 8 took a long time to train and did not produce bounding boxes for object detection.
* Batch size 1 proved optimal with a fast training speed and also efficient bounding box predictions when tested

### Learning Rate

1. Learning Rate (0.00001)



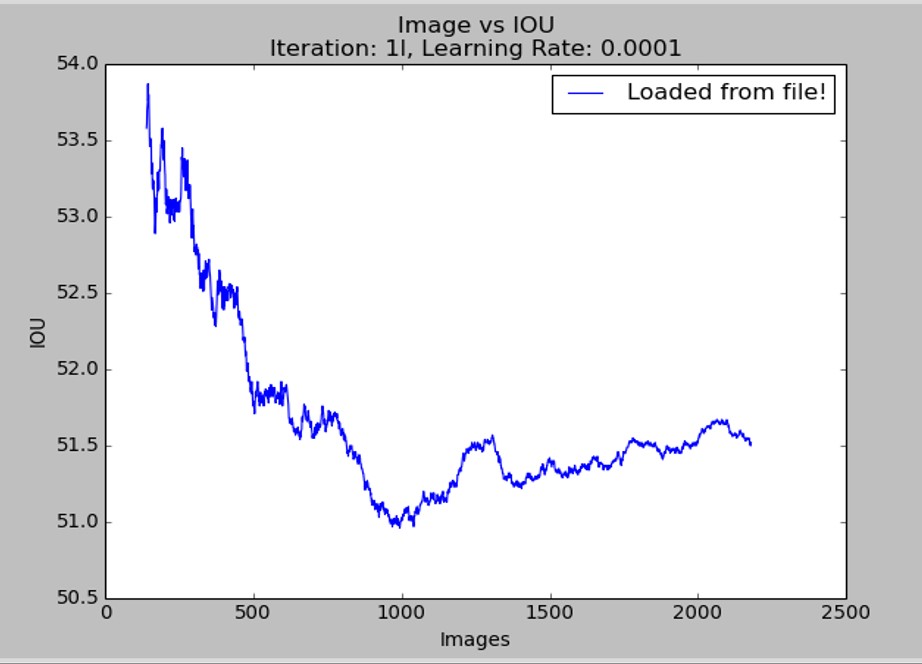
**IOU per Image**



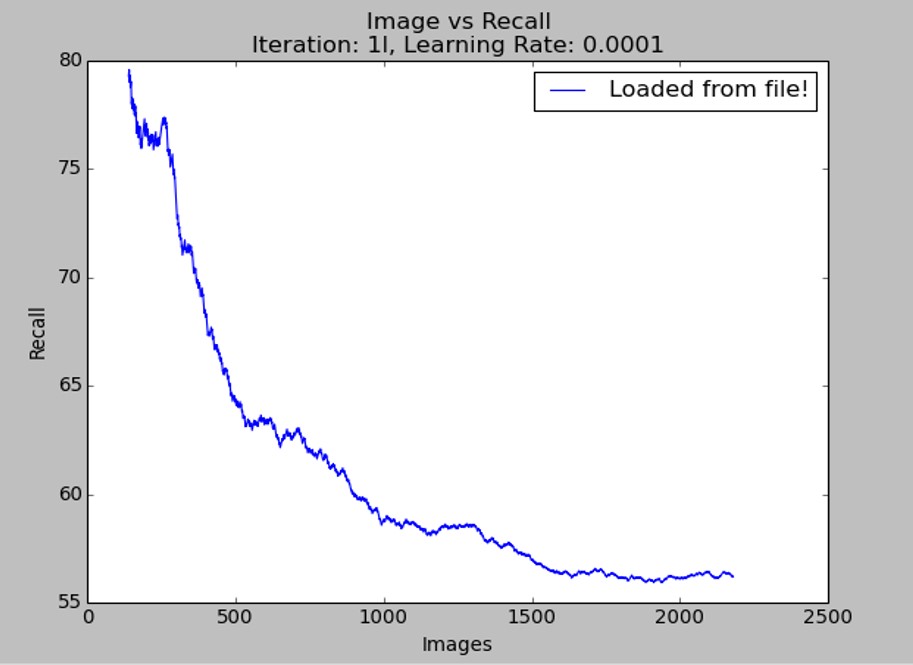
**Recall per Image**

Figure 7.1: IOU and Recall per Image for 0.00001 learning rate

1. Learning Rate (0.0001)



**IOU per Image**



**Recall per Image**

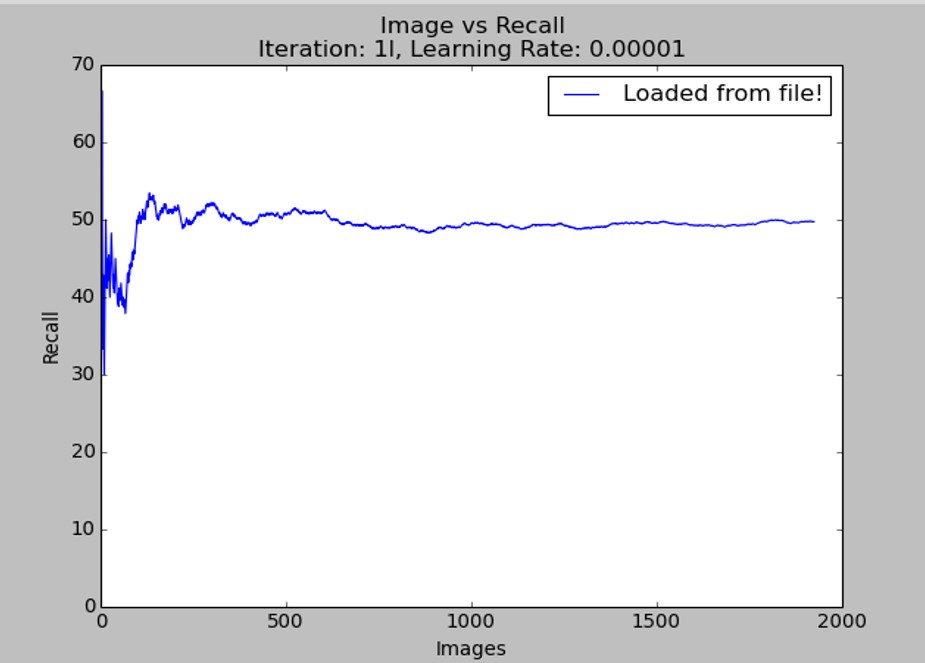
Figure 7.2: IOU and Recall per Image for 0.0001 learning rate

### Number of Training Iterations

1. Iterations: 100000



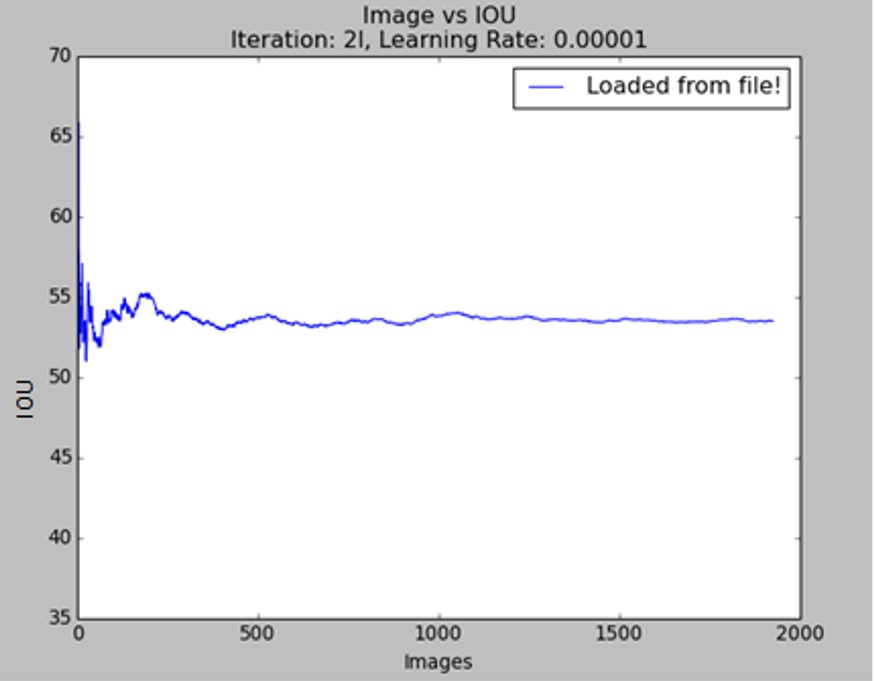
**IOU per Image**



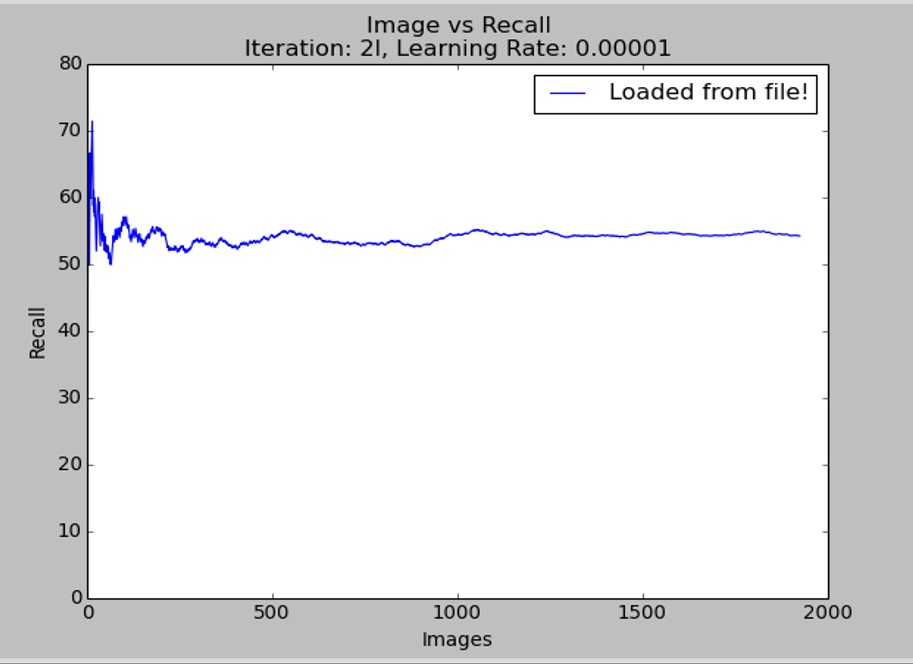
**Recall per Image**

Figure 7.3: IOU and Recall per Image for 100000 iterations

1. Iterations: 200000



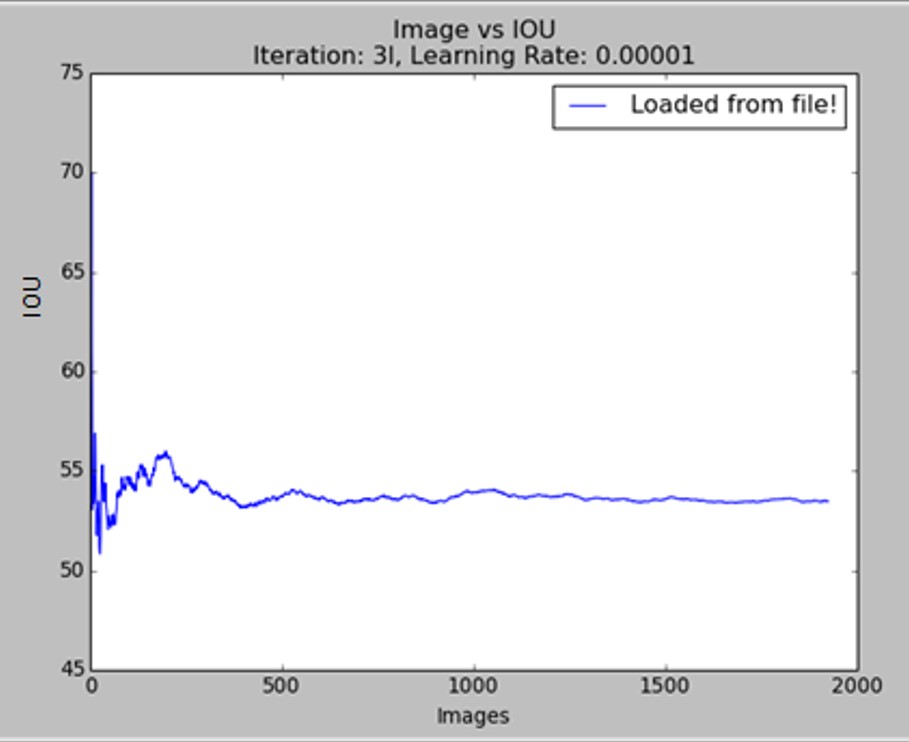
**IOU per Image**



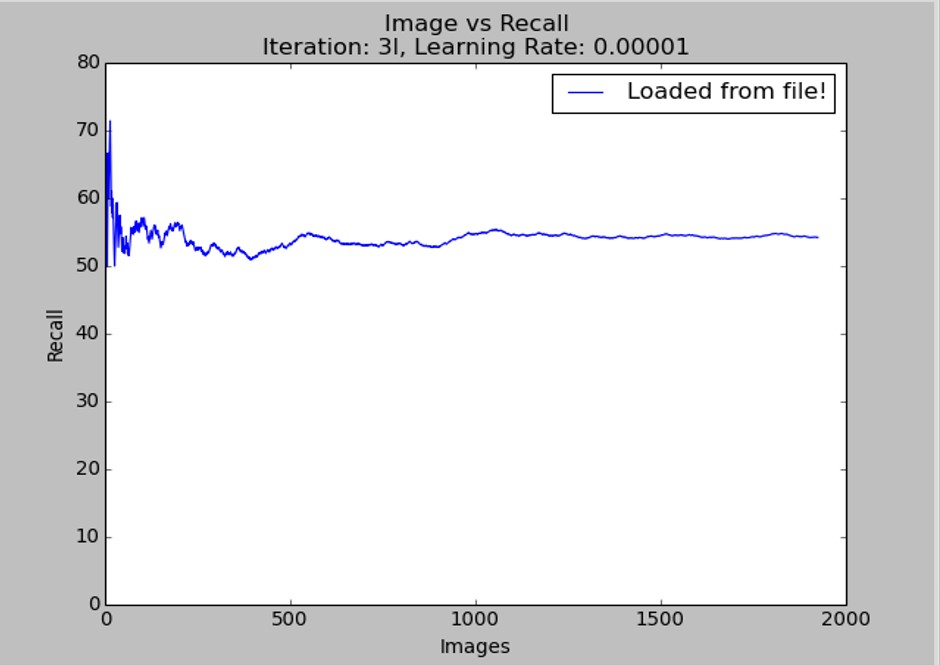
**Recall per Image**

Figure 7.4: IOU and Recall per Image for 200000 iterations

1. Iterations: 300000



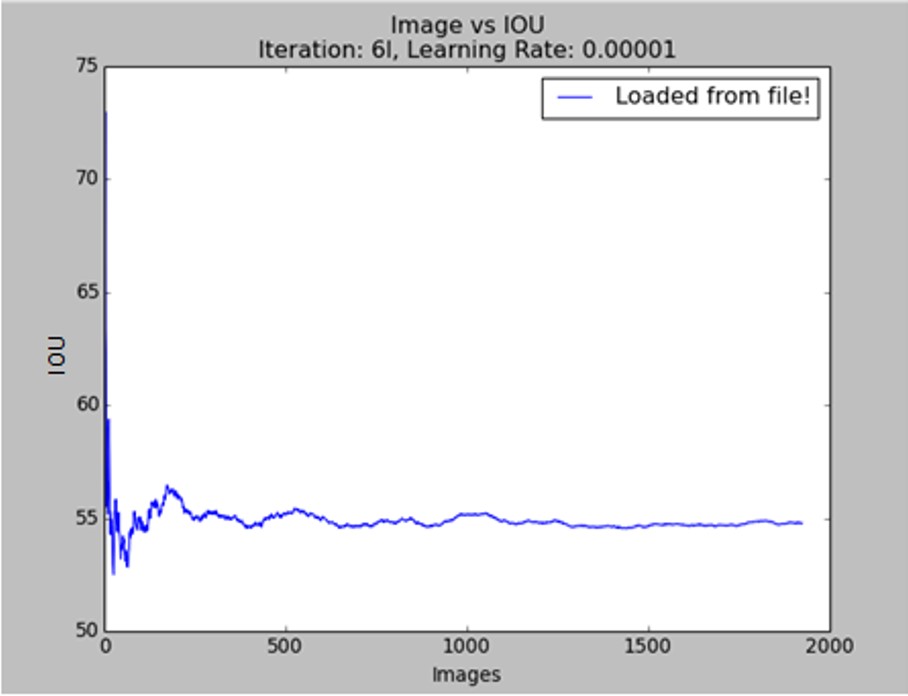
**IOU per Image**



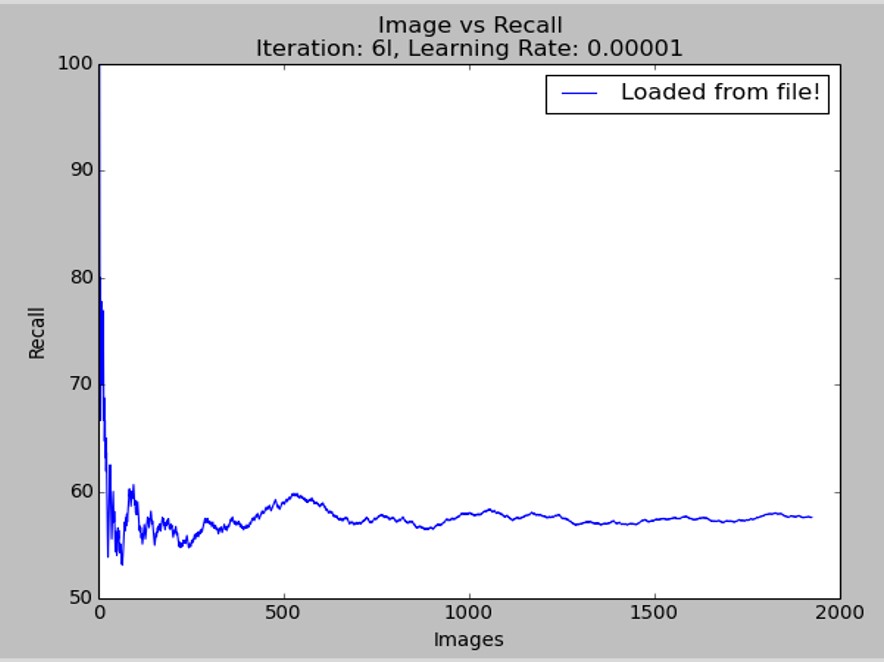
**Recall per Image**

Figure 7.5: IOU and Recall per Image for 300000 iterations

1. Iterations: 600000



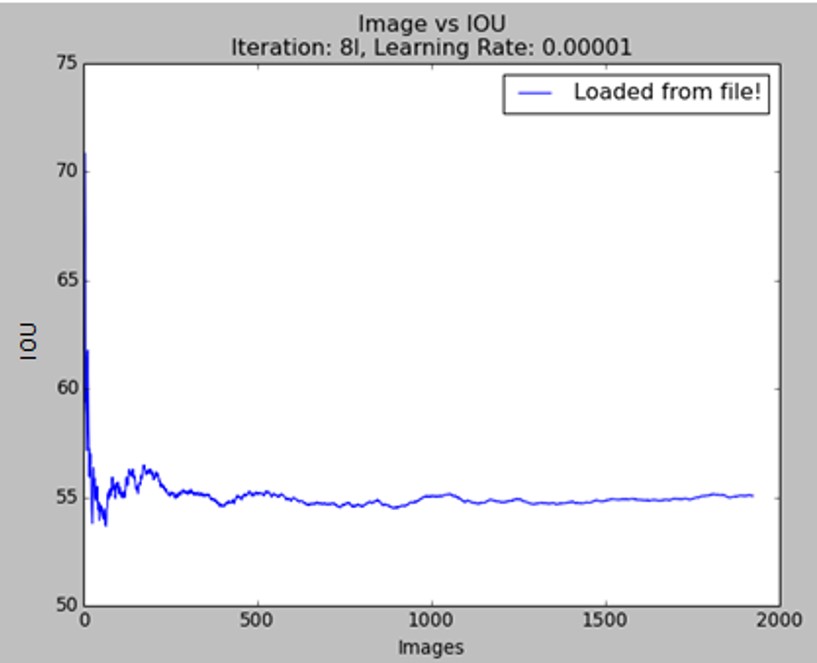
**IOU per Image**



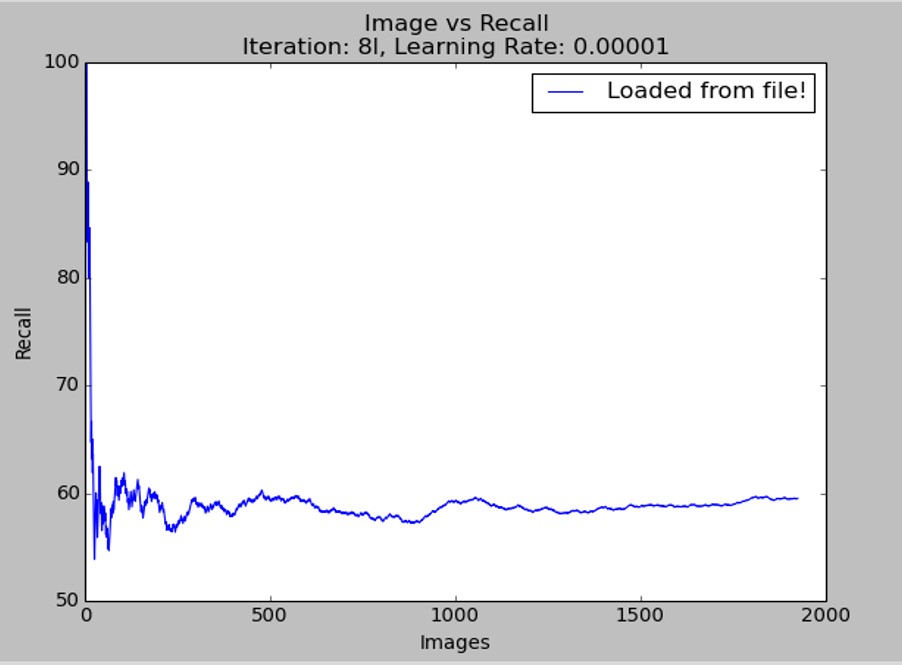
**Recall per Image**

Figure 7.6: IOU and Recall per Image for 600000 iterations

1. Iterations: 800000



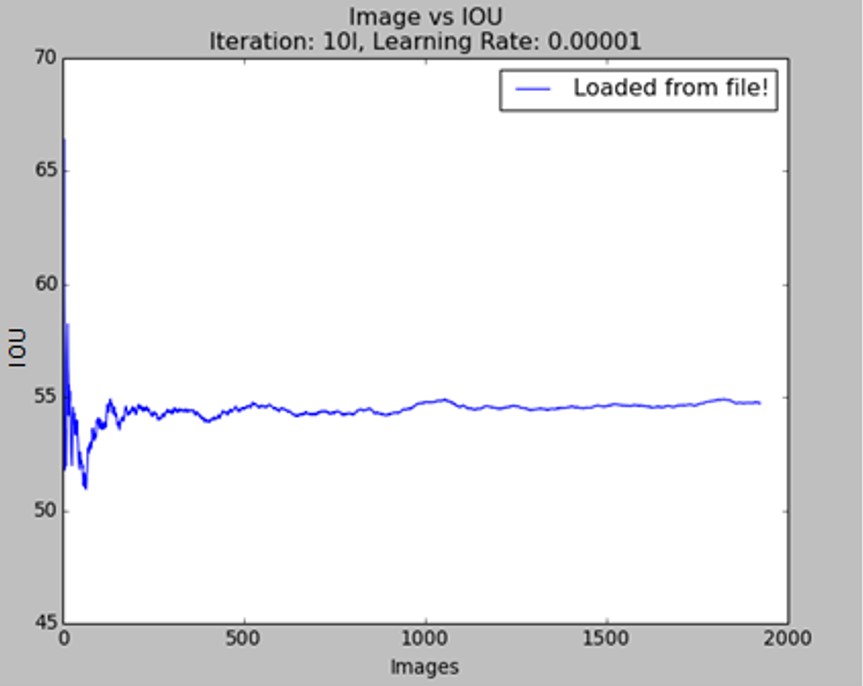
**IOU per Image**



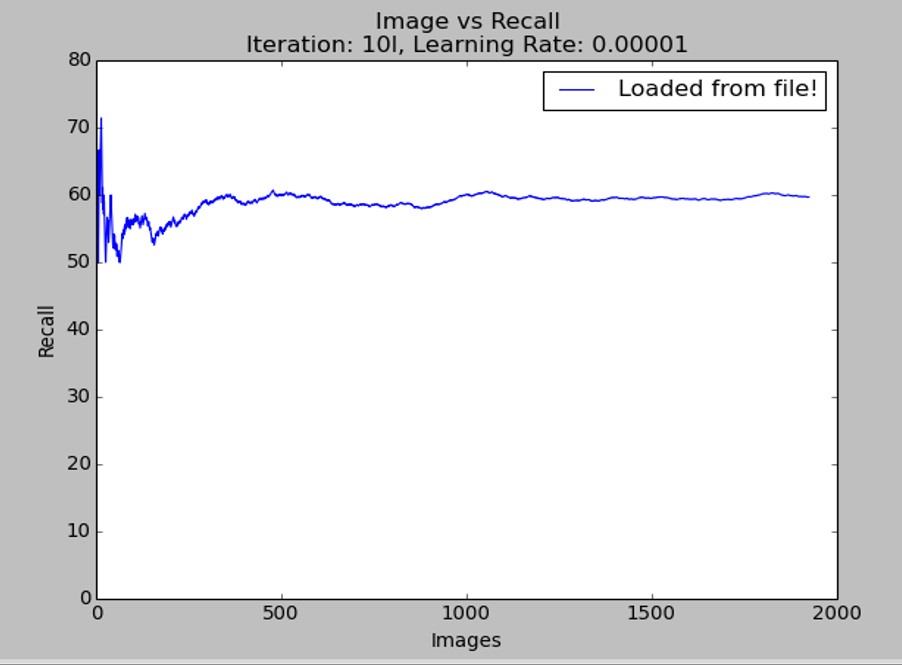
**Recall per Image**

Figure 7.7: IOU and Recall per Image for 800000 iterations

1. Iterations: 1000000



**IOU per Image**



**Recall per Image**

Figure 7.8: IOU and Recall per Image for 1000000 iterations

## Optimal Parameters

After conducting the above experiments with respect to varying training parameters such as Batch Size, Learning Rate and number of training iterations, the optimal values were found to be as follows:

#### • Batch Size: 1

When each iteration trains a single image, it was found that the featues of the image were learnt better.

#### • Learning Rate: 0.00001

With Learning Rate below 0.00001, it was found that the network was unable to detect objects at all. Learning Rate of 0.0001 detected objects, but with lesser accuracy in terms of classification. Hence, the optimal learning rate for good detection and classification is found to be 0.00001

#### • Number of training iterations: 800000

For training below 800000 iterations, the network was found to predict fewer bounding boxes with less accuracy. For training for iterations from 900000 to 1000000, the network was found to be over fit - i.e, performing well only on the training images. Hence, the optimal number of training iterations was concluded to be as 800000.

# Conclusion

This system can convert the less security into more security transaction. This system create a more reliable communication means user and bank system. Also it generate face at the end of transaction which is verify by various algorithm.

* + - 1. payment system for online transaction is based on face recognition provides authenticate user data privacy and prevents misuse of data in world.
      2. . This technique identify theft and prevent customer data which improve security.
      3. This system can convert the less security into more security transaction. This system create a more reliable communication means user and bank system.
      4. This is useful to reduce or totally remove fake transaction, i.e to read OPT then it required face of user which is provide for transaction. Then the transaction is done.
      5. . It will also go a long way in ensuring that innocent people are not wrongly arrested based on previous crimes

**Future Work**

The proposed system gives a method for online transaction security using facial recognition. The system can be developed to automate the process. Some future enhancements for the proposed system can be:

1. Improving the system for banking transactions.
2. To add some more feature to make the transaction secure.
3. 3. More efficient Object Detection Neural network.

4. To implement the system in a banking and online transaction section web is mot more convenient so develop the app for this application.

The proposed system can be evolved to meet several other operations which are not included in this project. Expanding the system will result in more efficient and hassle free operations.

## 

## References

1. Teichman, Alexander William. "METHOD AND SYSTEM FOR VISUAL AUTHENTICATION." U.S. Patent No. 20,170,300,685. 19 Oct. 2017.
2. Zhao, W., Chellappa, R., Phillips, P.J. and Rosenfeld, A., 2003. Face recognition: A literature survey. *ACM computing surveys (CSUR)*, *35*(4), pp.399-458.
3. Savvides, Marios, BVK Vijaya Kumar, and Pradeep K. Khosla. "Cancelable biometric filters for face recognition." *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*. Vol. 3. IEEE, 2004.
4. McKenna, Stephen J., and Shaogang Gong. "Non-intrusive person authentication for access control by visual tracking and face recognition." *International Conference on Audio-and Video-Based Biometric Person Authentication*. Springer, Berlin, Heidelberg, 1997.
5. R.S. Choras, “Facial feature detection for face authentication,” in the Proceeding of IEEE Conference on Cybernetics and Intelligent Systems., 2013, pp.112-116
6. P. Nagesh and B. Li, “A compressive sensing approach for expressioninvariant face recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 1518–1525
7. S. Liao, A. K. Jain, and S. Z. Li, “Partial face recognition: Alignmentfree approach,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 5, pp. 1193–1205, 2013.
8. Q. Yin, X. Tang, and J. Sun, “An associate-predict model for face recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, June 2011, pp. 497–504.
9. M. V. Prakash, P. A. Infant and S. J. Shobana, “Eliminating Vulnerable Attacks Using One Time Password and PassText - Analytical Study of Blended Schema,” Proceedings of the International Conference on VLSI, Communications and Instrumentation, pp. 35-41, Nov. 2010.

## Appendix :Code

**Configuration File (.cfg)**

[ net ] batch=64 subdivisions=8 height=416 width=416 channels=3

momentum=0.9 decay=0.0005 angle=0 saturation = 1.5 exposure = 1.5 hue=.1

learning rate =0.0001 max batches = 45000 policy=steps steps =100,25000,35000 scales =10 ,.1 ,.1

[ convolutional ] batch normalize=1

f i l t e r s =32 size=3 stride=1

pad=1

activation=leaky

[ maxpool ] size=2 stride=2

[ convolutional ] batch normalize=1

f i l t e r s =64 size=3 stride=1

pad=1

activation=leaky

[ maxpool ] size=2 stride=2

[ convolutional ] batch normalize=1 f i l t e r s =128 size=3 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =64 size=1 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =128 size=3 stride=1 pad=1 activation=leaky

[ maxpool ] size=2 stride=2

[ convolutional ] batch normalize=1 f i l t e r s =256 size=3 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =128 size=1 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =256 size=3 stride=1 pad=1 activation=leaky

[ maxpool ] size=2 stride=2

[ convolutional ] batch normalize=1 f i l t e r s =512 size=3 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =256 size=1 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =512 size=3 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =256 size=1 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =512 size=3 stride=1 pad=1 activation=leaky

[ maxpool ] size=2 stride=2

[ convolutional ] batch normalize=1 f i l t e r s =1024 size=3 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =512 size=1 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =1024 size=3 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =512 size=1 stride=1 pad=1 activation=leaky

[ convolutional ] batch normalize=1 f i l t e r s =1024 size=3 stride=1 pad=1 activation=leaky

#######

[ convolutional ] batch normalize=1 size=3 stride=1 pad=1 f i l t e r s =1024 activation=leaky

[ convolutional ] batch normalize=1 size=3 stride=1 pad=1 f i l t e r s =1024 activation=leaky

[ route ] layers=−9

[ reorg ] stride=2 [ route ] layers=−1,−3

[ convolutional ] batch normalize=1

size=3 stride=1

pad=1

f i l t e r s =1024

activation=leaky

[ convolutional ] size=1 stride=1

pad=1

f i l t e r s =45

activation=linear

[ region ] anchors = 1.08 ,1.19 , 3.42 ,4.41 , 6.63 ,11.38 , 9.42 ,5.11 , 16.62 ,10.52 bias match=1

classes=4

coords=4

num=5

softmax=1 j i t t e r =.2

rescore=1

object scale=5 noobject scale=1 class scale=1 coord scale=1

absolute=1 thresh = .6 random=0

**convert.py**

import os from os import walk , getcwd from PIL import Image classes = [” paper ” ,” plastic ” ,” metal ” ,” organic ”]

def convert ( size , box ): dw = 1./ size [0] dh = 1./ size [1] x = (box [0] + box [1])/2.0 y = (box [2] + box [3])/2.0

w = box [1] − box [0] h = box [3] − box [2] x = x∗dw w = w∗dw y = y∗dh h = h∗dh

return (x ,y ,w,h );

””” Configure Paths””” mypath = ”/home/medhu/BBox−Label−Tool/Labels /002/”

outpath = ”/home/medhu/darknet/ scripts / labels /paper/”

cls = ”paper” i f cls not in classes : exit (0)

cls id = classes . index ( cls )

wd = getcwd ()

l i s t f i l e = open(’%s/%s l i s t . txt ’%(wd, cls ) , ’w’)

””” Get input text f i l e l i s t ””” txt name list = [ ] for ( dirpath , dirnames , filenames ) in walk(mypath ):

txt name list . extend ( filenames )

break

print ( txt name list )

””” Process ””” for txt name in txt name list :

# t x t f i l e = open(” Labels/ stop sign /001. txt ” , ”r ”)

””” Open input text f i l e s ””” txtpath = mypath + txt name print (” Input :” + txt path ) t x tf i l e = open( txt path , ”r ”) lines = t x t f i l e . read (). split ( ’\ r\n ’)

use ”\r\n” instead of ”\n”

””” Open output text f i l e s ””” txt outpath = outpath + txt name print (”Output :” + txt outpath ) txt outfile = open( txt outpath , ”w”)

””” Convert the data to YOLO format ””” ct = 0 for line in lines :

#print ( ’ lenth of line is : ’)

#print ( len ( line ))

#print ( ’\n ’) i f ( len ( line ) *>*= 2): ct = ct + 1

print ( line + ”\n”)

elem = line . split ( ’\n ’) elems = elem [ 1 ] . split ( ’ ’) print ( elems )

xmin = elems [0] xmax = elems [2] ymin = elems [1] ymax = elems [3] #

img path = str (’%s/ labels/%s/%s . jpeg ’%(wd, cls , os . path . splitext ( txt name ) [ 0 ] ) )

#t = magic . from file ( img path )

#wh= re . search ( ’(\d+) x (\d+)’, t ). groups () im=Image . open( img path )

w= int (im. size [0]) h= int (im. size [1])

#w = int (xmax) − int (xmin)

#h = int (ymax) − int (ymin)

# print (xmin) print (w, h)

b = ( float (xmin) , float (xmax) , float (ymin) , float (ymax)) bb = convert ((w,h) , b)

print (bb)

txt outfile . write ( str ( cls id ) + ” ” + ” ”

. join ([ str (a) for a in bb ]) + ’\n ’)

””” Save those images with bb into l i s t ””” i f ( ct != 0):

l i s t f i l e . write(’%s/ labels/%s/%s . jpeg\n’%(wd, cls ,

os . path . splitext ( txt name ) [ 0 ] ) ) l i s t f i l e . close ()

**yolo.c file**

#include ”network .h”

#include ” detection layer .h”

#include ” cost layer .h”

#include ” utils .h”

#include ”parser .h”

#include ”box .h”

#include ”demo.h” char ∗voc names [ ] = {”Paper” ,” Plastic ” ,”Metal” ,” Organic ”};

void trainyolo ( char ∗ cfgfile , char ∗ weightfile )

{

char ∗train images = ”/data/voc/ train . txt ”; char ∗backup directory = ”/home/pjreddie/backup /”; srand ( time (0)); char ∗base = basecfg ( c f g f i l e );

printf(”%s\n” , base ); float avg loss = −1;

network net = parse network cfg ( c f g f i l e );

i f ( weightfile ){ load weights(&net , weightfile );

}

printf (” Learning Rate : %g , Momentum: %g , Decay :

%g\n” , net . learning rate , net .momentum, net . decay ); int imgs = net . batch∗net . subdivisions ; int i = ∗net . seen/imgs ; data train , buffer ;

layer l = net . layers [ net .n − 1];

int side = l . side ; int classes = l . classes ; float j i t t e r = l . j i t t e r ;

l i s t ∗ plist = get paths ( train images );

// int N = plist −*>*size ; char ∗∗paths = ( char ∗∗) list to array ( plist );

load args args = {0}; args .w = net .w; args .h = net .h; args . paths = paths ; args .n = imgs ; args .m = plist −*>*size ; args . classes = classes ; args . j i t t e r = j i t t e r ; args . num boxes = side ; args .d = &buffer ; args . type = REGION DATA; args . angle = net . angle ; args . exposure = net . exposure ; args . saturation = net . saturation ; args . hue = net . hue ;

pthread t load thread = load data in thread ( args ); clock t time ;

//while ( i ∗imgs *<* N∗120){ while ( get current batch ( net ) *<* net . max batches ){ i += 1;

time=clock (); pthread join ( load thread , 0); train = buffer ; load thread = load data in thread ( args ); printf (”Loaded : %l f seconds\n” , sec ( clock()−time ));

time=clock ();

float loss = train network (net , train ); i f ( avg loss *<* 0) avg loss = loss ;

avg loss = avg loss ∗.9 + loss ∗.1;

printf(”%d: %f , %f avg , %f rate , %l f seconds ,

%d images\n” , i , loss , avg loss , get current rate ( net ) , sec ( clock()−time ) , i ∗imgs );

i f ( i%1000==0 | | ( i *<* 1000 && i%100 == 0)){ char buff [256];

sprintf ( buff , ”%s/%s %d. weights ” , backup directory , base , i ); save weights (net , buff );

} free data ( train );

} char buff [256];

sprintf ( buff , ”%s/%s final . weights ” , backup directory , base ); save weights (net , buff );

}

void print yolo detections (FILE ∗∗fps , char ∗id ,

box ∗boxes , float ∗∗probs , int total ,

int classes , int w, int h)

{

int i , j ;

for ( i = 0; i *<* total ; ++i ){ float xmin = boxes [ i ] . x − boxes [ i ] .w/2.; float xmax = boxes [ i ] . x + boxes [ i ] .w/2.; float ymin = boxes [ i ] . y − boxes [ i ] . h/2.; float ymax = boxes [ i ] . y + boxes [ i ] . h/2.;

i f (xmin *<* 0) xmin = 0; i f (ymin *<* 0) ymin = 0; i f (xmax *>* w) xmax = w; i f (ymax *>* h) ymax = h;

for ( j = 0; j *<* classes ; ++j ){ i f ( probs [ i ] [ j ]) fprintf ( fps [ j ] , ”%s %f %f %f %f %f \n”

, id , probs [ i ] [ j ] , xmin , ymin , xmax, ymax);

}

}

}

void test yolo ( char ∗ cfgfile , char ∗ weightfile , char ∗filename , float thresh )

{

image ∗∗alphabet = load alphabet (); network net = parse network cfg ( c f g f i l e );

i f ( weightfile ){ load weights(&net , weightfile );

}

detection layer l = net . layers [ net .n−1]; set batchnetwork(&net , 1); srand (2222222); clock t time ;

char buff [256]; char ∗input = buff ; int j ; float nms=.4;

box ∗boxes = calloc ( l . side ∗ l . side ∗ l .n, sizeof (box ));

float ∗∗probs = calloc ( l . side ∗ l . side ∗ l .n, sizeof ( float ∗)); for ( j = 0; j *<* l . side ∗ l . side ∗ l .n; ++j )

probs [ j ] = calloc ( l . classes , sizeof ( float ∗));

while (1){ i f ( filename ){ strncpy ( input , filename , 256);

} else { printf (”Enter Image Path : ”); fflush ( stdout );

input = fgets ( input , 256 , stdin );

i f (! input ) return ;

strtok ( input , ”\n”);

}

image im = load image color ( input ,0 ,0); image sized = resize image (im , net .w, net .h );

float ∗X = sized . data ;

time=clock (); network predict (net , X); printf(”%s : Predicted in %f seconds .\n” , input , sec ( clock()−time )); get detection boxes ( l , 1 , 1 , thresh , probs , boxes , 0);

i f (nms) do nms sort (boxes , probs , l . side ∗ l . side ∗ l .n, l . classes , nms);

// draw detections (im , l . side ∗ l . side ∗ l .n, thresh , boxes , probs , voc names , alphabet , 20);

draw detections (im , l . side ∗ l . side ∗ l .n, thresh , boxes ,

probs , voc names , alphabet , 20);

save image (im , ” predictions ”); show image(im , ” predictions ”);

free image (im ); free image ( sized );

#ifdef OPENCV

cvWaitKey (0); cvDestroyAllWindows ();

#endif i f ( filename ) break ;

}

}

**UI:**

**index.html**

*<*html*>*

*<*head*>*

*<*title *>*project front page*<*/title *>*

*<*script language=”javascript ” type=”text/ javascript ” src=”asd . js”*>* function testJS () { var b = document . getElementById ( ’name ’ ) . value , url = ’ http :// path to your html files / next . html?name=’ + encodeURIComponent(b );

document . location . href = url ;

}

*<*/script *>*

*<*/head*>*

*<*body style=”background−color : white;”*>*

*<*h1 align=”center”*><*font color=”orange” size=”10”*>*

GARBAGE SEGREGATION *<*/font*><*/h1*><*br/*>*

*<*br/*>*

*<*img align=l e f t src=”sampleinput . png” alt=”sample input” style=”width :380px ; height :390px;” /*>*

*<*img align=right src=”sampleoutput . png” alt=”sample output” style=”width :350px ; height :390px;” /*>*

*<*br/*><*br/*><*br/*><*br/*><*br/*>*

*<*h1 style=”text−align : center;”*>*

*<*form action=’bup. html’*>*

*<*input type=’submit ’ name=’upload btn ’ value=’START!’*> <*/form*>*

*<*br/*><*/h1*>*

*<*br/*><*br/*><*br/*>*

*<*/body*>*

*<*/html*>*

**display.html**

*<*html*>*

*<*head*>*

*<*pre*><*h2*>* INPUT IMAGE OUTPUT IMAGE*<*/h2*><*/pre*>*

*<*body*>*

*<*img src=”darknet/new.JPEG” border=”5” alt=”uploaded image” style=”width : 600px ; height :620px ; position : absolute ; top : 50px ;

l e f t : 0px;”*>*

*<*img src=”darknet/ predictions . png” border=”5” alt=”output image” style=”width : 600px ; height :620px ;

position : absolute ; top : 50px ;

right : 0px;”*>*

*<*/body*>*

*<*/html*>* **process.html**

*<*!DOCTYPE html*>*

*<*html*>*

*<*head*>*

*<*script src=”https :// ajax . googleapis .com/ajax/ libs /jquery/

3.2.1/ jquery . min. js”*><*/script *> <*script *>*

$(document ). ready ( function (){

$(document ). ajaxStart ( function (){

$(”#layers ”). html ();

$(”#div1 ”). fadeIn ();

$(”#div2 ”). fadeIn ();

$(”#div45 ”). fadeIn ();

$(”#div34 ”). fadeIn (12000);

$(”#div3 ”). fadeIn (12000);

$(”#div4 ”). fadeIn (12000);

$(”#div35 ”). fadeIn (12000);

$(”#div5 ”). fadeIn (18000);

$(”#div6 ”). fadeIn (18000);

$(”#div7 ”). fadeIn (18000);

$(”#div8 ”). fadeIn (18000);

$(”#div36 ”). fadeIn (18000); $(”#div9 ”). fadeIn (24000);

$(”#div10 ”). fadeIn (24000);

$(”#div11 ”). fadeIn (24000);

$(”#div12 ”). fadeIn (24000);

$(”#div37 ”). fadeIn (24000);

$(”#div13 ”). fadeIn (30000);

$(”#div14 ”). fadeIn (30000);

$(”#div15 ”). fadeIn (30000);

$(”#div16 ”). fadeIn (30000);

$(”#div17 ”). fadeIn (30000);

$(”#div18 ”). fadeIn (30000);

$(”#div38 ”). fadeIn (30000);

$(”#div19 ”). fadeIn (36000);

$(”#div20 ”). fadeIn (36000);

$(”#div21 ”). fadeIn (36000);

$(”#div22 ”). fadeIn (36000);

$(”#div23 ”). fadeIn (36000);

$(”#div24 ”). fadeIn (36000);

$(”#div25 ”). fadeIn (36000);

$(”#div39 ”). fadeIn (36000);

$(”#div26 ”). fadeIn (42000);

$(”#div27 ”). fadeIn (42000);

$(”#div28 ”). fadeIn (42000);

$(”#div29 ”). fadeIn (42000);

$(”#div40 ”). fadeIn (42000);

$(”#div30 ”). fadeIn (48000);

$(”#div31 ”). fadeIn (48000);

$(”#div41 ”). fadeIn (48000);

$(”#div32 ”). fadeIn (54000);

$(”#div42 ”). fadeIn (54000);

});

$(document ). ajaxComplete ( function (){ $(”#layers ”). html(”*<*img src=\”layers . png\” width=\”800px\” height=\”300px\”*>*”); });

$(” button ”). click ( function (){

$(”#txt ”). load (” upload .php”);

});

});

*<*/script *>*

*<*/head*>*

*<*body*>*

*<*div id=”layers ” style=”position : absolute ; top :

170px ; l e f t : 290px;”*><*/div*>*

*<*div id=”txt”*><*h2*><*/h2*><*/div*>*

*<*br /*><*br /*>*

Upload an image to test :

*<*form action=’upload .php ’ method=’POST’ enctype=’multipart/ form−data ’ id=”form1”*>*

*<*input type=’ file ’ name=’userFile ’*><*br*>*

Enter detection threshold : *<*input type=’text ’ name=”cmd”*><*br*>*

*<*/form*>*

*<*button type=”submit” form=”form1” value=”Submit”*>*SUBMIT*<*/button*>*

*<*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*><*br*>*

*<*div id=”div1” style=”width :900px ; height :1px ; display : none ; background−color : white ; color : Black”*>*

*<*pre*>*layer f i l t e r s size input output*<*/pre*><*/div*><*br*>*

*<*div id=”div2” style=”width :900px ; height :1px ; display : none ; background−color : white ; color :Red”*><*pre*>*

0 conv 32 3 x 3 / 1 416 x 416 x 3 −*>* 416 x 416 x

32

*<*pre*>*30 detection *<*/pre*><*/div*><*br*><*br*><*br*><*br*>*

*<*div id=”div34” style=”width :85px ; height :180px ; display : none ; position : absolute ; top : 190px ; l e f t : 360px ;

background−color : red ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div35” style=”width :80px ; height :160px ; display : none ; position : absolute ; top : 210px ; l e f t : 450px ;

background−color : green ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div36” style=”width :60px ; height :110px ; display : none ; position : absolute ; top : 260px ; l e f t : 540px ;

background−color : blue ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div37” style=”width :65px ; height :85px ; display : none ; position : absolute ; top : 290px ; l e f t : 610px ;

background−color : green ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div38” style=”width :60px ; height :55px ; display : none ; position : absolute ; top : 325px ; l e f t : 685px ;

background−color : red ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div39” style=”width :60px ; height :55px ; display : none ; position : absolute ; top : 325px ; l e f t : 755px ;

background−color : blue ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div40” style=”width :15px ; height :55px ; display : none ; position : absolute ; top : 325px ; l e f t : 845px ;

background−color : green ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div41” style=”width :40px ; height :60px ; display : none ; position : absolute ; top : 325px ; l e f t : 890px ;

background−color : red ; opacity : 0.5”*><*/div*><*br*>*

*<*div id=”div42” style=”width :105px ; height :60px ; display : none ; position : absolute ; top : 325px ; l e f t : 1000px ; background−color : blue ; opacity : 0.5”*><*/div*>*

*<*/body*>*

*<*/html*>* **upload.php**

*<*?php

//to upload the image . . .

$info = pathinfo ($ FILES [ ’ userFile ’ ] [ ’ name ’ ] ) ;

$ext = $info [ ’ extension ’ ] ; // get the extension of the f i l e

$newname = ”new .”. $ext ; $target = ’ darknet / ’.$newname; move uploaded file ( $ FILES [ ’ userFile ’ ] [ ’ tmp name ’] , $target ); // getting threshold value . . .

$thresh=$ POST[ ’cmd ’ ] ; //echo running details . . .

$running=”./darknet detector test cfg/waste . data cfg/ yolo−waste . cfg yolo−waste 400000 . weights . . /images/new.JPEG −thresh ”. $thresh ; echo ”Running command ”. $running .” ...”.” *<* br /*>*”; //run the detector command . . . $pleaseput = shell exec ( ’ sh detect . sh ’ ) ; // display the prediction percentage . . .

echo ”*<*pre*>*$pleaseput *<*/pre *>*”; //button to view output image . . . echo ”Detection successful !”; echo ”*<*form action=\”display . html\”*>*”; echo ”*<*input type=\”submit\” value=\”view image\”*>*”; echo ”*<*/form*>*”;

?*>*