GE-108 Tinkering Lab

Privacy using Computer Vision and Face Detection

Project Report

Submitted by:

Hansin Ahuja:	2018csb1094
Paras Goyal:	2018csb1111
Rohit Tuli:	2018csb1116
Navtejpreet Singh:	2018csb1107
Praful Gupta:	2018csb1112

Introduction

In the Era of Computers, Digital Privacy is of utmost importance. There are 4.1 Billion Internet Users- and the number is increasing at a steady pace.The Internet influenced retail sales to the tune of $2.84 trillion in 2018 and is expected to influence retail sales to the tune of $3.45 trillion in 2019. Social Media websites have users in the tune of Billions. Though there are enough tools to protect against software based cyber crimes, we found that both awareness among masses and specialised tools are very less for Visual threats to Data Privacy. Text on a laptop or mobile screen can be read from a few meters away- even more so for pictures and other graphical media. This creates a dire need for an application that can work in the background and inform the user when another person is looking at it.

We propose a solution which uses Deep Learning Algorithms for Computer Vision to create a software that detects whether an unwanted person is looking at the Computer Screen or not. In such a scenario the software conveys this to the user so that the user can take necessary steps to prevent Data Theft.

Deep learning is a branch of machine learning which is completely based on artificial neural networks, as neural network is going to mimic the human brain so deep learning is also a kind of mimic of human brain. It is currently the most used method for Computer Vision, which helps computers gain a high-level understanding of Images and Videos.

The software uses the webcam / camera on the device to monitor the surroundings. It samples the video input feed to get frame by frame data and then creates 'blobs' from it, eventually generating a confidence score for each structure. If the confidence score exceeds a predefined value and is in the visual range, then it shows a bounding box around the intruder. If it remains in the visual range for more than a few seconds then the user is notified.

Procedure

There are 3 primary aspects to what we've achieved with our project, namely:

1) Deep learning
2) Computer vision and face detection
3) Application of the aforementioned techniques to our project

1) Deep learning

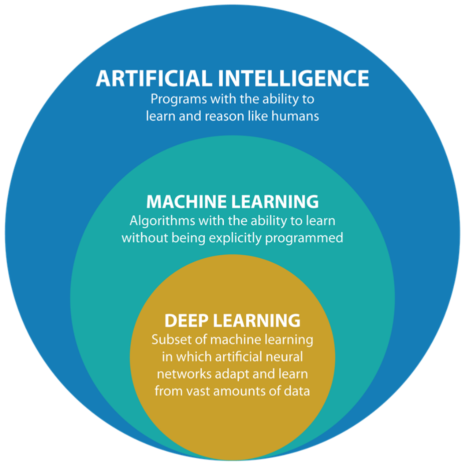
Deep learning is part of a broader family of machine learning algorithms or techniques, which is based on the concept of artificial neural networks. An intuitive idea of how a deep network works is by correlating it with a human brain and its neurons.

Deep neural networks can be thought of as simulating neural networks in the neocortex of our brain, the part of the mammalian brain associated with higher-level cognition, such as sensory perception, generation of motor commands, spatial reasoning and language. The unit component of the brain is a neuron, which receives and transmits electro-chemical information. Putting together enough of these neurons in a structured manner would create a neural network. In computers, these neurons are virtual. They are matrices and functions, which take some input, generate an output, and are then given some sort of feedback as to how correct their output was. Based on this feedback information, the neuron tries to better itself.

An intuitive understanding of this can come by looking at how children learn to identify danger. Children accumulate more and more information as they grow up. As they grow up, they encounter information such as people getting hurt by speeding objects, people actively trying to avoid a burning flame, etc. Accrual of this sort of information over time enables the child to identify actions which might result in an undesirable outcome. In the most abstract sense, this is how neural networks work.

The word ‘deep’ comes from increasing levels of hierarchy and abstraction associated with a neural network. Neural network architects might design smaller networks first to perform rudimentary tasks. Stacking these networks together allows us to perform increasingly complex tasks.

This introduction of neural networks was in extremely layman terms. From this point on in this report, it has been assumed that the reader has a good idea of how deep neural networks function.


ARTIFICIAL INTELLIGENCE
Programs with the ability to learn and reason like humans
MACHINE LEARNING
Algorithms with the ability to learn without being explicitly programmed
DEEP LEARNING
Subset of machine learning in which artificial neural networks adapt and learn from vast amounts of data

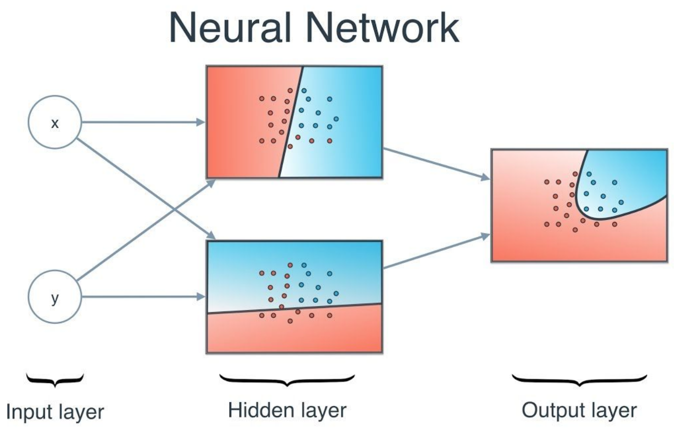


Figure 1 *Figure 2*

Fig 1: Classification of deep learning under the rubric of artificial intelligence and machine learning

Fig 2: A shallow neural network for the purpose of two-dimensional points binary classification

2) Computer Vision and Face Detection

Computer vision is the field of computer science that focuses on replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do. Face Detection is an application of Computer Vision which is used to detect the presence of Human Faces in images and videos.

Face Detection is the first and essential step for face recognition, and it is used to detect faces in the images. It is a part of object detection and can be used in many areas such as security, bio-metrics, law enforcement, entertainment, personal safety, etc.

There are many different ways for face detection- Feature based, Knowledge based, Template Matching and Appearance Based.

We used CaffeModel from the OpenCV Library. We generated blobs,which are the standard array and unified memory interface for the framework from Image of size 300* 300 pixels.

Then size of blob is set and it is normalised. We pass the generated blobs through the Caffe Net, which returns an array of all detections.

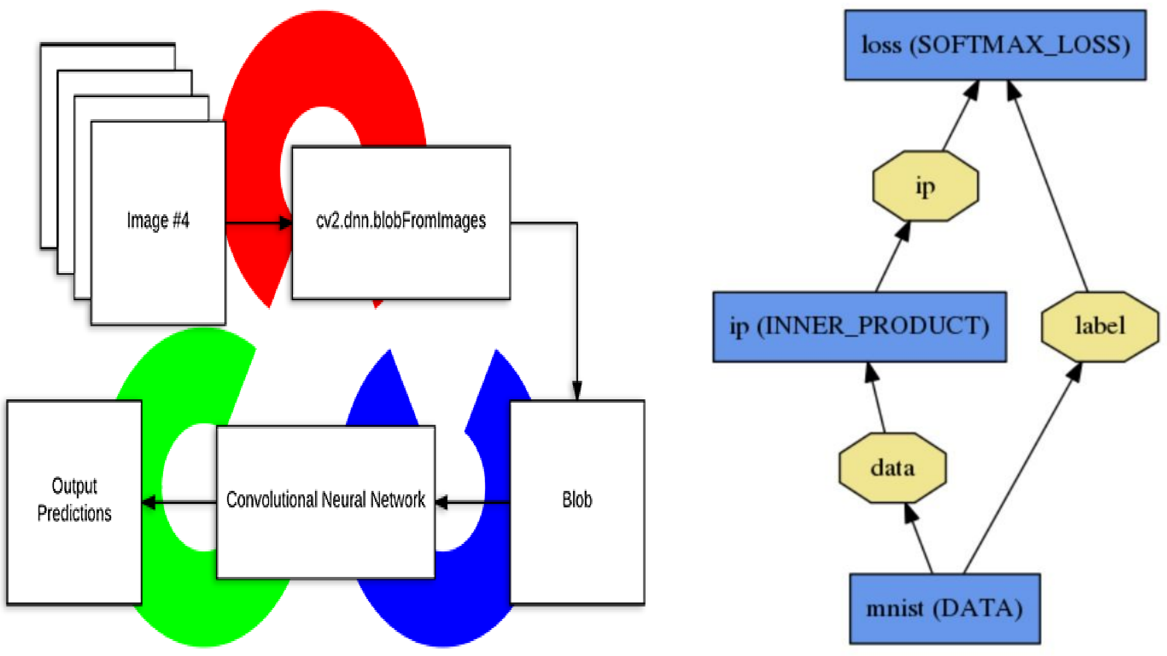


Fig 3: Data and control flow of the program.(Left), Neural Network of the model used (Right)

We loop over the detections and check the confidence value with a predefined minimum confidence set by us. If the confidence value is higher, it means that the face has successfully been detected and we create a bounding box around the face.

The aforementioned model has been trained on MNIST dataset. The MNIST database contains 60,000 training images and 10,000 testing images taken from American Census Bureau employees and American high school students

3) Application of the above in our project

Given that we've understood how the aforementioned model works, we now modify it to find an application in privacy. Oftentimes, we wouldn't want multiple reasons looking at our laptop screens. That may be for the purpose of maintaining privacy while carrying out online banking transactions, online shopping or simply because we don't like someone peeking into our screen. We first describe the modifications to our model, and then highlight some problems that might be associated with its applications.

Starting off from the point where our algorithm has learnt how to introduce bounding boxes around the faces that it detects (which happens every 2 nanoseconds), our task now is to count the number of faces currently on-screen. The code can be modified in any number of ways, and this has been detailed in the future section titled “Possible improvements.” As it exists right now, the code has been modified to return to the desktop home screen if it detects.

We maintain a simple count variable, which is incremented by one each time the algorithm iterates over a bounding box. This count variable is evaluated every 2 nanoseconds, our frequency of image capture. If this count is greater than one, we return to the homescreen. This task is done using the Python library ‘keyboard.’ This library which is used to get full control and functionality of the keyboard. It’s a small library which can hook global events, register hotkeys, simulate key presses and more.

- It helps to enter keys, record the keyboard activities and block the keys until a specified key is entered and simulate the keys.
- It captures all keys, even onscreen keyboard events are also captured.
- Keyboard module supports complex hotkeys.
- Using this module we can listen and send keyboard events.

We use this module to gain control of the keyboard, utilize conditional branching to navigate to whichever screen we want. In particular, if count > 1, we use the command `keyboard.press_and_release('win+d')` and toggle to the homescreen, else remain put. However, this is the base case. The conditional branching also checks my current state of the screen. If the current state is the homescreen, and the algorithm detects more than one face in the next iteration, we don’t toggle. Several of these cases can be similarly enumerated. We took note of all possibilities and addressed them in our code.

Current limitations and problems we faced

The biggest limitation of our application is the field of view of the camera. Our laptop cameras have a narrow field of view, and can't capture faces from a skewed angle. This limitation is further amplified in a mobile phone. This problem can possibly be fixed by installing a wide angle camera to the operating device.

Another issue that we faced was the heavy RAM occupancy of our application. Using the camera to capture images at a quick rate and feeding the capture to the neural network at a high frequency is extremely RAM intensive. This can possibly be fixed by decreasing the frequency at which we sample images from the device camera.

Future Work

In the future we intend to convert our code into a full application for both desktop and mobile users with the added functionality to allow users to tune the parameters (like the number of people allowed to watch the screen) based on the kind of work the user is doing. We would also add a face recognition module so that the user can create exceptions for peers.

References

- https://opencv.org/
- https://caffe.berkeleyvision.org/
- https://pypi.org/project/keyboard/
- https://towardsdatascience.com/face-detection-for-beginners-e58e8f21aad9