**Detections by AI**

ESLAM MAGDY REZK EBRAHIM HASSANIN1   
CT002-3-2-AIM-L-4 APU2F2111CS(IS) / APD2F2111CS(IS)  
Asia Pacific TechnologyTechnology Park, Malaysia  
TP062816

GHAARIB KHURSHID 2  
CT002-3-2-AIM-L-4 APD2F2111CS(IS)  
Asia Pacific TechnologyTechnology Park, Malaysia  
TP062554ALI ABDULAMEER ABDALI ABDULLA ALSHEALA3    
CT002-3-2-AIM-L-4 APU2F2111CS(IS) / APD2F2111CS(IS)  
Asia Pacific TechnologyTechnology Park, Malaysia  
TP062836

MOHAMMED ASIF4  
CT002-3-2-AIM-L-4 APU2F2111CS(IS) / APD2F2111CS(IS)  
Asia Pacific TechnologyTechnology Park, Malaysia  
TP062320

MUHAMMAD AKMAL 5  
CT002-3-2-AIM-L-4 APU2F2111CS(IS)/ APD2F2111CS(IS)  
Asia Pacific TechnologyTechnologyPark, Malaysia  
TP062672

*Abstract*

*Face recognition technology is a technique through which a person’s features are determined through a library that contains all the angles of a person’s face in order to identify it with high quality, so it is a biometric technology, and through artificial intelligence, the object identification technology can be used through which all objects within the image are identified, The deep face that determines the state of a person whether he is sad, happy or in a normal situation.*

# Introduction

This project includes developing, modifying parameters, classifying things, and identifying a person’s feelings through a person’s feelings. The developers found a code programmed through artificial intelligence that identifies people, but it has some defects, as it was developed and added some features such as identifying a person’s feelings if he was Sad or happy also to identify all the things that are in the place, where all the libraries were linked to each other in the form of one code, and once the user runs the code, he can identify all of this through the camera of the device available to him, If libraries were placed and linked to each other, such as the CV2 library, the Face Recognition library, Yolo V3, and the Deep Face library that was developed by developers in the Kaggle competition, work was also done to handle the errors within the code, and make it work accurately and in a high method, because when the user turns his face when broadcasting A lot of errors happen, so this was worked on by the developers inside APU University, Also, the Yolo library was used, which is based on the discovery of real-time objects when running the code.

* 1. Literature Review

1. Similar Projects:
2. YOLO object detection

The Detection for Objects project is part of our project that focuses on deep learning using the regression-based Yolo algorithm, which in turn predicts the bounding and classes that are around the object.

Graphical user interface

Description automatically generated with medium confidence

1. Boxes, Confidences and classes inside the code

Boxes, Confidence, and Classes: They are the squares that surround the object, and the trust is the ratio that the YOLO algorithm gives to the object if it is high or low, and the ratio is placed 0.5 for the objects that are not trusted inside the code, as for the classes, this algorithm uses many classifiers, a file called Coco.

Project Structure:

Text

Description automatically generated with medium confidence

1. Project Structure for YOLO object detection

In this project, the Yolo configuration file was used. The file that contains the classes that the algorithm will use, the main file of the Yolo algorithm, which is trained on the classes that were entered, was also used, and there are some pictures and videos that will be worked on.

1. Deep Nervous Network:

Graphical user interface, text

Description automatically generated

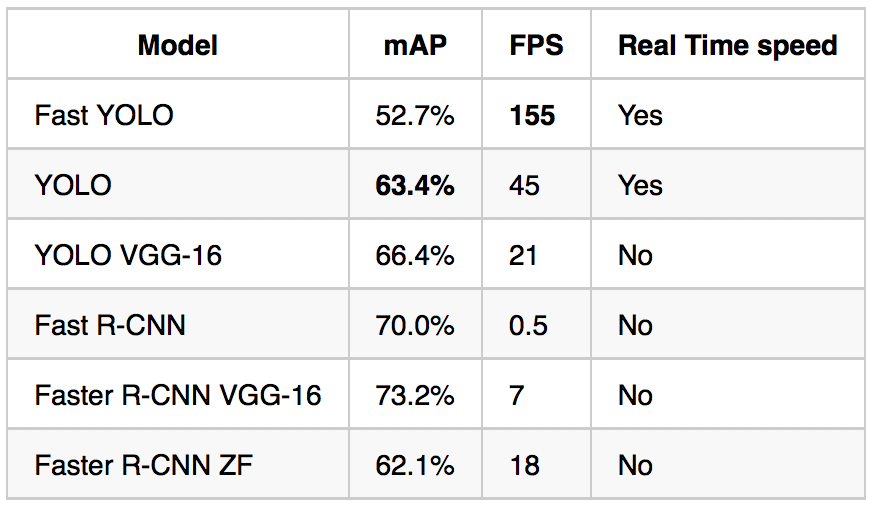
1. Cv2.dnn.readNetFromDarknet function

The Deep Nervous Network module (DNN) is inside the CV Library, which is with CNN, which is trained in facial recognition [1], so for readNetFromDarknet Here, the files in the Darknet format are read, and the file path of the Yolo algorithm is set, which is the Configuration, and the other file that is learned from the network, which is the weight of the Yolo algorithm until it is passed to the net object and used inside the code with the presence of CV [1].

Literature review for YOLO:

(YOLO) is a hugely popular and viral algorithm. (YOLO) is well-known for its object detecting ability. One of the first scientists who gave an introduction to the release of (YOLO) was Redmon et al, and that was in the year 2015 and after that other scientist published other versions of (YOLO).

The YOLO model has the benefit of being compact in size and quick to calculate, as well as having a clear and straightforward structure. The neural network is used to generate YOLO's location and category directly. YOLO's speed is high since it simply needs to upload the image to the network to get the final detection result, allowing it to do video time detection. YOLO detects directly from the global picture, and the detection error rate is low. One of the disadvantages of YOLO is that it needs to increase object detection accuracy. For items that are very close to each other, the YOLO test produces poor results. The handling of big and tiny things, in particular, must be also improved [9].



1. Comparison between yolo methods
2. Face Emotion Recognition:

CNN or Convolutional Neural Networks are advancing and rising in demand. A new CNN architecture named DenseNet are made using denser blocks and does powerful operations where each thick block or layers is a concatenation of the previous feature map. This leads to a more efficient system which deeply analyzes the subject. Ina DenseNet, in each layer, the feature maps of all the previous layers are taken as inputs, and its own feature maps are also used as input for each following layers. Hence the name DenseNet [6].

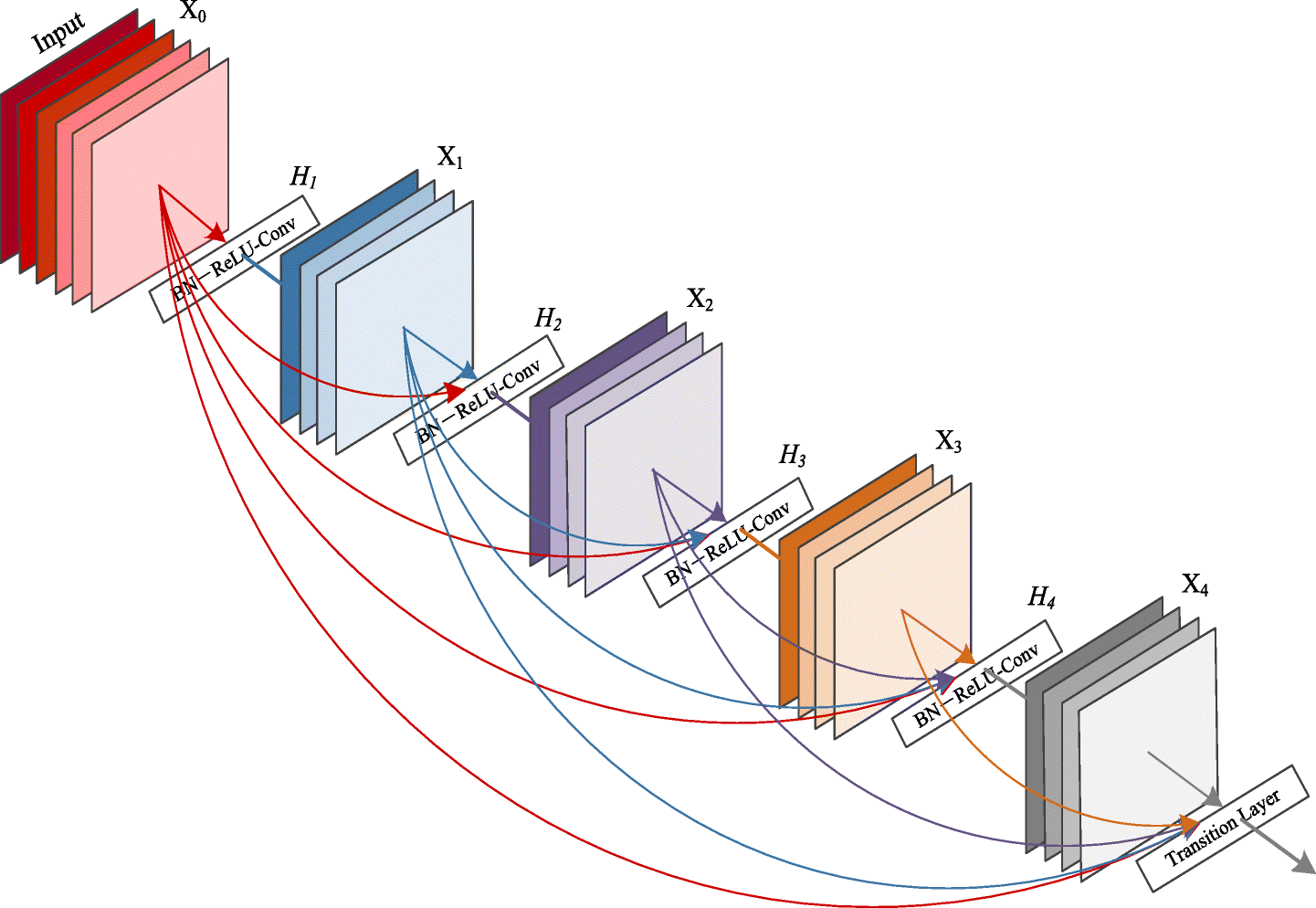


Figure 5 - DenseNet

A total layer count of 56 with a growth rate of 12 gave rise to amazing recognition capabilities. At the input node, 3 different features were present. The RGB image, the first half of the image and the bottom half of the image.

The DenseNet model was put to test using the EmotiW 2015 benchmark test & emotions were assigned to undergo the test: angry, disgust, happy, sad, surprise, fear and neutral. The dataset of images included people from all ages, genders and with different head poses.

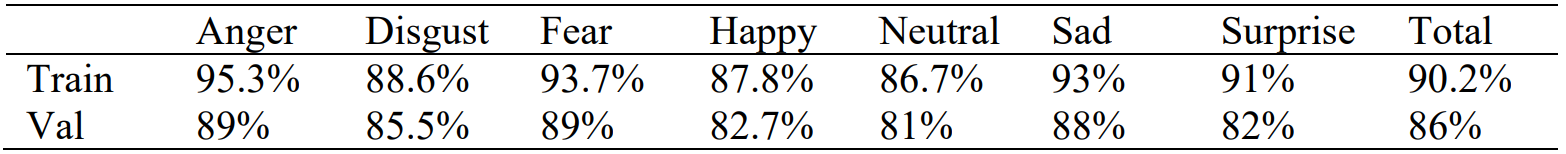


Figure 5 - Results

The result produced by DenseNet was also compared to other architectures. It showed that DenseNet can give an 18% improvement when it comes to recognizing emotions [5].

1. Deep face:

Nonverbal communication is frequently expressed through facial expressions. Facial expression recognition is a popular topic these days. The most prevalent face emotions are happy, sad, angry, neutral, fear, disgust, and surprise, each with its own meaning. As a result, the DeepFace algorithm is used to implement it digitally. Deepface is a python framework developed by Facebook AI Research for face recognition and facial attribute analysis (age, gender, emotion, and race) (FAIR). There are four steps in modern face recognition:

* Detect
* Align
* Represent
* Classify

The purpose of Alignment is to construct a frontal face from an input image that may contain faces in a variety of positions and angles. The frontal face was extracted using 3D frontalization of faces based on fiducial (facial feature points). The following are the steps involved in the alignment process:

* Using six fiducial points, locate the face in an input image. Two eyes, the tip of the nose, and three places on the lips are the six fiducial points. Face recognition in images relies on these characteristic points. Create a 2D-face picture cut from the source image using 6 fiducial points.

Graphical user interface, website

Description automatically generated

Figure 6 2D-Cropped face

* A 7 fiducial point map is applied to the 2D-aligned cropped image with the necessary Delauney Triangulation. The goal of this step is to align the rotations that are out of plane. A 3D model is also built using a generic 2D to 3D model generator, and 67 fiducial points are manually plotted.

A picture containing text

Description automatically generated

Figure 3D shape generated from the align 2D-crop image (pawangfg, Deep Face Recognition, 2021)

* Using the provided relation, construct a relationship between 2D and 3D.

A picture containing text

Description automatically generated

• The loss function is converted to an ordinary least squares function via Cholesky decomposition.

A map of a city

Description automatically generated with medium confidence

Figure 7 ficudial point mapping on 2D-3D affine face.

* Before achieving frontalization, add the residual component to the 3D warp's x-y coordinates to reduce 3D-warp corruption. Finally, piece-wise affine is used to the Delauney triangulation produced from the 67-fiducial points to accomplish frontalization.

A picture containing text, person, person, primate

Description automatically generated

Figure 8 final frontalization

**Representation and Classification Architecture:**

DeepFace has been trained for multi-class face recognition, which means it can categorize photographs of a variety of people based on their identities. It takes a 3D aligned 152\*152 RGB picture for input. Following that, the image is processed with a Convolution layer with 32 filters and a size of 11\*11\*3, followed by a 3\*3 max-pooling layer with a stride of 2. A 16-filter convolution layer with a dimension of 9\*9\*16 is then applied. Low-level features from image edges and textures are extracted with these layers. The following three layers are locally connected layers, which are a type of fully connected layer in each feature map with various types of filters. This aids in the improvement of the model because different parts of the face have different discrimination abilities.

A picture containing text, screenshot

Description automatically generated

Figure 9 DeepFace Full architecture

The model's final two tiers are made up of layers that are totally interconnected. These layers aid in the formation of a link between two facial areas that are otherwise distinct. Two examples include the location and shape of the eyes, as well as the position and shape of the mouth. The output of the second last fully connected layer is used as a face representation, whereas the output of the last layer is the SoftMax layer K classes for face categorization. There are around 120 million parameters in this network, with the final fully linked layers accounting for the great bulk (95 percent). This network's feature map/vector formed during training is exceedingly sparse, which is an intriguing characteristic. At the highest levels, for example, 75% of the values are zero. This could be due to the fact that every convolutional network makes use of the ReLU activation function, which is effectively maxed out (0, x). This network also employs Drop-out Regularization, which contributes to its sparsity. Dropout, on the other hand, is only used on the first layer that is totally linked. In the final stages of this network, we also normalize the feature to be between 0 and 1. This also reduces the overall impact of lighting fluctuations [6].

1. Methodology & Approach
2. Approach:

Deep learning

Machine Learning

1. Methodology:

Hybrid Methods for face recognition (ALI):

In the recent studies related to the Facial recognition systems it could be seen that many of the systems are combining different face recognition approaches in an either serial or parallel to produce a better outcome than the individual methods. In this section some hybrid face recognition approaches from several papers were reviewed to present a different scope from the conventional face recognition systems.

According to [7] the conventional facial recognition systems usually apply one feature domain and one classifier where the neural network is used for classification. Hence, when it applied to the real-world applications the face appearance could get perceived differently due to different degree of illumination, expression, and posture. The figure shows how a simple face recognition system is proceeded.

Graphical user interface, text

Description automatically generated

1. Standard phases for facial recognition system

The proposed facial recognition system from [5] in figure 8 utilizes a hybrid approach where it is presented in two modes training and classification. In training mode, a dataset of face images is created through normalization and two face feature extraction methods which are Principal Component Analysis (PCA) and Independent Component Analysis (ICA). Therefore, the extracted features are trained in parallel using Back-propagation neural networks (BPNN) ,which is a multi-layer feed-forward neural network with a backpropagation algorithm. Further, the trained PCA BPNN and ICA BPNN are introduced with new images in classification mode to produce new inputs through the neural network. These inputs are presented to a combiner which it is function is for producing a new output from fusing both feature extractors scores. Overall, Classification is accomplished by reflecting an input face image into a subspace known as a face space defined by "Eigenfaces," and then categorizing the incoming face image by comparing it with the stored image of known persons.

Diagram

Description automatically generated

1. Hybrid Multi-Feature Face recognition system

Through the comparison of the results from different implementations using PCA ,ICA or both together it indicates that a hybrid approach produces better performance than any simple PCA and ICA Implementations.

Haar Cascade Classifier (Ali):

Facial detection is one of the methods to produce some unique facial features such as facial landmarks. Therefore, these extracted features can be used for many facial recognition purposes. Hence, the Haar cascade algorithm by [12] is used primarily for face detection in the proposed system. The Haar cascade algorithm was implemented for developing the facial detection system of the mini quadcopter made by DJI company. The algorithm provides the quadcopter with the ability to follow the face object and determine the required position. The Haar cascade approach utilized in this study is implemented through Python and OpenCV programing languages and using the installed camera in the quadcopter.

The Haar cascade approach according to [8] (Priambodo, Arifin, Nasuha, & Winursito, 2021)can be exerted by training images through a cascade function in four stages as it is shown in Figure 9 . The first stage in the haar cascade algorithm is responsible for collecting the haar features. The haar feature defined by [8] as “ the calculation of the difference between the sum results of the pixel intensities in a certain location of the detection window”. To prevent the emergence of undesirable problems while extracting a Haar feature. Integral images are produced in the second stage to provide a reduction to the number of the operations required for the feature extraction. The integral image is overall built by accumulating the values of the top and left pixels to produce the pixel values for an image.

Diagram

Description automatically generated

1. Stages of Haar Cascade Classifier

Due to the lower accuracy of the selected Haar feature when detecting the objects, the Ada boost algorithm is implemented by choosing the optimal Haar feature to improve the object detection. The improvement from the Ada boost is constructed by combing a variety of simple classifiers into strong classifiers, which results in increasing the accuracy of the Haar features. The fourth stage ends when the classifications are merged in a cascade structure. Hence, the duration of the detection process is decreased due to the swift object detection.

Haar cascade classifier method is considered to be one of the simple and lightweight facial detection approaches. Although the Haar cascade algorithm could provide some facial detection results in a limited computer system but when compared to other object detection algorithms the performance of the haar cascade method is still lacking.

Convolutional Neural Network:

Face recognition are the specialties of CNNs. No need to configure parameters before using CNN algorithms. CNN algorithms are viable due to model training and online information. For data representation, CNN algorithms employ mathematics. Similar findings are found using weight sharing. This method allows for quick data categorization. This is a huge challenge. Examples include hardware processing power and dataset parameter count. CNN had a 15.3% mistake rate from 1998 until 2012, when ZF-net began. Google and VGGNet cut mistakes. Its 2015 error rate is 3.6%, whereas the human eye's is 5.1%.

A conventional CNN has input, convolutional, active, pooling, fully connected, and output layers. Some CNN models include extra layers for specific tasks. Figure 1 shows a CNN's basic design.

Diagram

Description automatically generated with low confidence

1. CNN Structure

This multi-layered architecture uses forward pass and error backpropagation calculations to achieve target competence. A collection of photographic data and labels for each aspect of the structure is needed to train it as a model After training, final weights for testing will be set. Consider the following for a better grasp of levels.

The first one is the input layer, which initializes the image data and centers it. To speed up convergence, this layer normalizes the scale of all input data to a range of 0 to 1. Normalization lightens the data, reducing duplication. PCA is used to reduce the number of accessible dimensions while retaining the most important ones.

Second is the convolutional layer, which is the most critical in a CNN. These neurons are designed to untangle preceding layer attributes. Many local and mutual interactions use a CONV kernel filter that slides on the original image input. It multiplies and adds each pixel of the local associated data it contains before adding the convolutional result. That's because the CONV kernel may use convolution to retrieve an image's properties. A CONV kernel's shared weights are used to filter distinct parts of an image. It is possible to identify and categorize neutral cells using shared weights. It takes kernel size, depth, stride, zero padding, and filter amount.

A vanishing gradient problem caused by underfitting is addressed by the active layer. Insufficient prior convolutional layer produces nonlinearity. After that, an active layer function like Sigmoid, Tanh, ReLu, ELU, Leaky ELU, or Max out may be used to solve underfitting. Although the Sigmoid and Tanh functions are still commonly used for their simplicity and efficiency, the ReLu function has the quickest convergence.

With the pooling layer, the convolutional layer's discoveries may be reduced in size. Weakening component maps and strengthening selected extractions are achieved by combining the outputs of one layer's neurons into a single neuron in another. Spatial Pyramid Pooling is a common pooling layer found between two convolutional layers (SPP). The width of a pooling layer is its stride. There's max and regular pooling. Max pooling utilizes the most severe rewards from the previous layer's cell groups. Regular pooling is done for normal rewards. Broader than stride pooling happens. The input layer's aberrant state features may be recovered and obtained using convolutional layers and a final pooling layer.

Among the CNN layers, the completely linked layer that feeds data to the output layer. Using the preceding layer's neuron and linking them simplifies and speeds up data processing. In all cases, a yield layer follows it.

Then there are additional layers needed for particular CNN models to get the appropriate results. Dropout and regression layers are included. It updates neural cell knot weights with a probability to combat overfitting (which is decided by the stochastic policy). Classification of characteristics by the regression layer utilizing logistic regression, Bayesian linear regression, and Gaussian Processes (GPR). In a regression layer, all possible object types are parabolized [6].

CNN for Face Recognition:

As we know face recognition is given a database of K images recognize the person. One of the difficulties with face recognition is that you must be able to recognize a person from a single photograph or sample of that person's face. Furthermore, deep learning algorithms have typically performed poorly when just one training example is available. So, one technique would be to enter the person's picture, feed it to a ConvNet, and have it output a label, y, using a SoftMax unit with N outputs or maybe N+1 outputs corresponding to each of the N people in the database, or none of the above. However, this does not function very well.

Text, letter

Description automatically generated

1. Similarity function

The function d's duty is to take two faces and tell you how similar or distinct they are. A Siamese network is a nice method to achieve this: we end up with a feature vector of 128 numbers after a series of convolutional, pooling, and fully connected layers. We train the neural network such that the encoding it computes results in a function d that tells us when two images are of the same person.

Gradient descent on the triplet loss function is one way for learning neural network parameters that give a good encoding for face photos. We must compare two pictures in order to use the triplet loss. Terminology for triplet loss: anchor image, positive image (same as anchor image), and negative image. So, we want the distance of the squared norm between the anchor encoding and the negative encoding to be less than or equal to the distance of the squared norm between the anchor encoding and the negative encoding. To ensure that the neural network does not just output 0 for every encoding, we will change the aim so that it is much smaller than zero rather than less than or equal to zero. This is also known as a margin. So, if we have a training set of 5,000 photographs of 500 distinct people, we pick 5,000 (multiple pictures of the same person) pictures, choose triplets, and then train the learning algorithm on the triples cost function using gradient descent [4].

CNN for Open Cv2:

A digital image, as we all know, is a 2-D matrix of different-valued pixels. All photos are made up of pixels, which are the basic building blocks of visuals. Pixels are placed in a grid to form images. An image with 640 columns (width) and 480 rows (height) has 640 columns (width) and 480 rows (height) (the height). An image of this size has 640 \* 480 = 307200 pixels.

There is a basic step for image process which is Importing image using Image acquisition tools, Image Pre-Processing or Analyzing and manipulating images, and output in which either can alter an image or make some analysis out of it. For all image pre-processing tasks, we will utilize the OpenCV library. OpenCV reads data from a single place in memory. We will utilize the HDF5 format for reading and writing picture data just for this purpose. We will go through all of the necessary tools and libraries in order to have a better understanding.

OpenCV is a free and open-source software library for computer vision and machine learning. OpenCV was created to offer a standard foundation for computer vision applications and to speed up the incorporation of machine perception into commercial products. We will utilize the OpenCV library to resize the photos and generate feature vectors from them, which will be accomplished by transforming the image data to NumPy arrays. For training the model, we will utilize CNN (Convolutional Neural Network), one of the Deep Neural Net extensions [5].

1. YOLO Network structure
2. Conclusion/Recommendation

YOLO is a cutting-edge object identification and face recognition algorithm which outperforms various approaches and has quickly become the industry standard for recognizing objects in computer vision. Techniques such as sliding window object identification, R CNN, Fast R CNN, and Faster R CNN were previously used. Nonetheless, due to its speed and precision, YOLO has become an industry standard for object detection since its inception in 2015.

# Material and Methods

This research is conducted through a simple computer with a processor of i7 and 8GB of ram. The facial recognition system also requires a functioning webcam connected to the operating computer to perceive the video inputs and store the captured frames as a binary data by utilizing the pickle library. The facial recognition system requires an operating system installed with python and OpenCV programming languages to run the recognition.py and embedding.py python files.

The most importing methods in the facial recognition system are implemented through YOLO V3 algorithm, which is a state-of-the-art, real time object detection architecture. The algorithm is pretrained by the COCO dataset and it provide a quick and accurate object recognition. Furthermore, the OpenCV library played a major role in providing the system with the required algorithms for the machine perception and computer vision.

1. Yolo
2. OpenCv
3. Deep Face
4. Face Recognition
   1. Algorithm Implementation
5. Purpose:

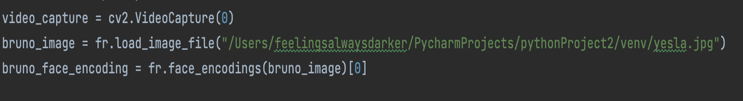
The purpose of the code was to recognize faces, and after that it was modified and many libraries and methods were added to it, and the content of the code itself was modified so that it could distinguish faces with great accuracy. Also, a library containing the captured images was added, and the focus was also on identifying things at the same time and knowing the status of Emotional person using algorithms such as face identification, YOLO and deep face.

1. Parameters:
2. Import pickle:

Before:

Before that, the user could only add one picture and could not add more than one picture, and here there was a problem in identifying more than one person, but after that, after adding a database of pictures, the situation became easier by adding more than one picture of the same for different people

The Pickle library has been added so that the user can identify himself by taking more than one photo and it is placed inside the database of the code. The file containing the user’s photos is called and linked to the facial recognition library.



The library can add more than one image, which makes the camera able to identify more than one person and not just one person.

After:

Text

Description automatically generated

1. Connected deep face library to the data:

Before:

Before that, the code was not able to identify the person’s emotional state, and after that, codes were added that recognized the person’s emotional state, but the code was unable to identify the person’s emotional state when turning his face to the left, so the code was mediated by Enforce detection = false.

Text

Description automatically generated

After:

The Deep Faces library has been added through which a person's feelings are identified, whether he is happy or sad, and through that a nested lobe was made to link the code to each other and connect the data to each other so that they work at the same moment on the same user.

Text

Description automatically generated

1. Errors inside the code of deep face:

Before:



Here, the code was not working when moving his face to the left, the user's face, ending the camera, and the code had to be engineered and the parameter was modified and added enforce detection = False

After:

A picture containing text

Description automatically generated

There were defects inside this code that when the code was run and the user turned his face away from the camera, an error occurred inside the code and the camera would be closed automatically, and here there were only two ways to solve this problem, the first one, using try and expects to handle the error then the output inside expects will be not detected, or use enforce detection and it should be false

1. Cannot detect unknown people:

The other problem was that the code recognized only the people who were registered in the database, while the unknown people were when they appeared in the camera, an error occurred because the compiler could not understand who these people were, because there was a defect between the user name and the name of the customer, and the link between the two codes leads To a problem, therefore, this defect was modified by removing the user name permanently and putting the name of the customer and linking them in the two codes.

OLD CODE:

**cv2.putText(frame, ref\_dictt[name], (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1)**

NEW CODE:

cv2.putText(frame, name, (leftScreen + 6, bottomScreen - 6), font, 1.0, (255, 255, 255), 1)

1. Convert from transflow to YOLO:

There were a lot of problems inside the code that contains the Transflow library in order to identify things and then the change was made from Mobile net SSD to Yolo weight and configuration, by using dnn to read paths.

net = cv2.dnn.readNet('/Users/feelingsalwaysdarker/PycharmProjects/pythonProject2/venv/yolov3.weights','/Users/feelingsalwaysdarker/PycharmProjects/pythonProject2/venv/yolov3.cfg')

1. Change in the confidences:

Before:

Text

Description automatically generated

The code was not able to literally trust inside the images, so the code was partially changed and a code was added that works to trust inside the images and through the size and shape of the image and its relationship to a classes that can be given a percentage.

After:

The percentage has been modified to ensure that the box contains an object. The confidence has been modified and placed between 0.4 and 0.9.

Text

Description automatically generated

1. Convert from image detection to live video by using haar cascade:

Before:

The code was previously working on image analysis only, but the parameter was modified in the following example and working on the Har algorithm, calling it from GitHub and inserting it directly into the code.

Text

Description automatically generated

After:

The HAR algorithm was linked to the code and also transformed it from just a code that works on images to a code that runs on the live broadcast, and the algorithm was called from GitHub.

Graphical user interface, text

Description automatically generated

1. Result and Discussion

The result here is that the Yolo algorithm was linked to the algorithm for identifying a person’s emotions and the facial recognition algorithm after the code was only able to identify the face and could not also identify unknown people, also the code was able to add only one image and could not identify more than one person, the work was done on With that, adding a library library to work on storing images for more than one person, the confidence rate was also developed because it was previously based on expectations, but by using repetition it gives results through the spaces I have in the image and its relationship to classes.

* 1. Discussion on Implementation

In the running stage, it will appear in the operation window inside the Compiler, knowing the faces of people, identifying things and their accuracy compared to their sizes and quality, or it is also possible to identify unknown people, it can also identify more than one person inside the code when it works. The code works efficiently and without errors, it can also give the special emotional state of a person.

* 1. Results

A picture containing text, person, indoor

Description automatically generated

A picture containing person, person

Description automatically generated

1. Conclusions

In the summary of the topic, three algorithms are used at the same time, which are the Yolo algorithm to identify things, the face identification algorithm and the user emotion detection algorithm, and this is done once the code is executed and run by the user.

# Acknowlgements

* Learn how the Yolo algorithm works and identify objects.
* Knowing how the face identification algorithm works and how to add a library to all people's faces.
* Knowing how the deep face algorithm works and how to detect a user's face.
* Knowing how to link all the algorithms to each other, developing their own parameters, modifying errors within the code and raising its efficiency.

# References

1. OpenCv, (n.d*.) readNetFromDarknet ()* Retrieved from OpenCV: <https://docs.opencv.org/3.4/d6/d0f/group__dnn.html#gafde362956af949cce087f3f25c6aff0d>
2. Dhami, D (2018, November 26). *Face Recognition/Special Applications of CNN*. Retrieved February 28, 2022, from Medium: <https://medium.com/@dhartidhami/face-recognition-special-applications-of-cnn-51b928a3cd40>
3. Mehta, C. (2020, October 11). *Image Processing using OpenCV, CNN, and Keras backed by Tensor Flow*. Retrieved February 28, 2022, from Medium: <https://medium.com/analytics-vidhya/image-processing-using-opencv-cnn-and-keras-backed-by-tensor-flow-c9adf22bb271>
4. Approach. Proceedings of ICSCN 2007: International Conference on Signal Processing Communications and Networking. 420 - 425. 10.1109/ICSCN.2007.350774
5. Zhang, Zhiqin. (2018). Deep Face Emotion Recognition. Journal of Physics: Conference Series. 1087. 062036. 10.1088/1742-6596/1087/6/062036.
6. Amaarora. (2020, August 2). *DenseNet Architecture Explained with PyTorch Implementation from TorchVision*. Retrieved from Amaarora: https://amaarora.github.io/2020/08/02/densenets.html

Amaarora. (2020, August 2). DenseNet Architecture Explained with PyTorch Implementation from TorchVision. Retrieved from Amaarora: https://amaarora.github.io/2020/08/02/densenets.html

Kodinariya, T. M. (2014). Hybrid Approach to Face Recognition System using Principle component and Independent component with score based fusion process. *arXiv*. doi:arxiv.org/abs/1401.0395v1

pawangfg. (2021, June 11). *Deep face recognition*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/deep-face-recognition/

pawangfg. (2021, June 11). *Deep Face Recognition*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/deep-face-recognition/

Priambodo, A. S., Arifin, F., Nasuha, A., & Winursito, A. (2021). Face tracking for flying robot quadcopter based on haar cascade classifier and pid controller. *Journal of Physics: Conference Series, 2111*, 012046.

1. Kodinariya, T. M. (2014). Hybrid Approach to Face Recognition System using Principal component and independent component with score based fusion process. arXiv. https://doi.org/arxiv.org/abs/1401.0395v1
2. Priambodo, A. S., Arifin, F., Nasuha, A., & Winursito, A. (2021). Face tracking for flying robot quadcopter based on haar cascade classifier and pid controller. Journal of Physics: Conference Series, 2111, 012046.

Amaarora. (2020, August 2). *DenseNet Architecture Explained with PyTorch Implementation from TorchVision*. Retrieved from Amaarora: https://amaarora.github.io/2020/08/02/densenets.html

Kodinariya, T. M. (2014). Hybrid Approach to Face Recognition System using Principle component and Independent component with score based fusion process. *arXiv*. doi:arxiv.org/abs/1401.0395v1

pawangfg. (2021, June 11). *Deep face recognition*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/deep-face-recognition/

pawangfg. (2021, June 11). *Deep Face Recognition*. Retrieved from GeeksforGeeks: https://www.geeksforgeeks.org/deep-face-recognition/

Priambodo, A. S., Arifin, F., Nasuha, A., & Winursito, A. (2021). Face tracking for flying robot quadcopter based on haar cascade classifier and pid controller. *Journal of Physics: Conference Series, 2111*, 012046.

1. Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, Bo Ma,

A Review of Yolo Algorithm Developments, Procedia Computer Science, Volume 199, 2022, Pages 1066-1073, ISSN 1877-0509,https://doi.org/10.1016/j.procs.2022.01.135