Emotional Analysis of Students

Submitted in partial fulfilment of the requirements of the degree of

Bachelor of Engineering

by

Amanda Judy Andrade BE-IT 2

Supervisor:

Assistant Professor Vaishali Kavathekar

UNIVERSITY OF MUMBAI

Emotional Analysis of Students

Submitted in partial fulfilment of the requirements of the degree of

Bachelor of Engineering

by

Amanda Judy Andrade BE-IT 2

Supervisor:

Professor Vaishali Kavathekar

Department of Information Technology

Don Bosco Institute of Technology, Mumbai

2020–2021

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

CERTIFICATE

This is to certify that the project entitled "EMOTIONAL ANALYSIS OF STUDENTS" is a bonafide work of

Amanda Judy Andrade	2
Komal Kadam	28
Nilam Kadam	29

submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of **Undergraduate** in **Bachelor of Information Technology**.

Date:

(Prof. Vaishali Kavathekar)

Supervisor

(Prof. Janhavi Baikarekar) **(Dr. Prasanna Nambiar)**

HOD, IT Department **Principal**

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Project Report Approval for B.E.

This project report entitled "EMOTIONAL ANALYSIS OF STUDENTS " by Author Name is approved for the degree of Bachelor of Engineering in Information Technology.

External Examiner: Mr. Chandan Kolvankar

Supervisor: Prof. Vaishali Kavathekar

(**Chairman**)

.

Date:

Place:

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

Andrade

1. Amanda Judy Andrade 02 BE-IT
2. Komal Kadam 28 BE-IT
3. Nilam Kadam 29 BE-IT

Date:

ABSTRACT

Mental health organizations embedded inside instructive frameworks can make a continuum of integrative thought that improves both passionate prosperity and informative accomplishment for youngsters. To strengthen this continuum, and for ideal child improvement, a reconfiguration of preparing and passionate health structures to help execution of verification-based practice might be required. Integrative methods that merge study lobby level and understudy level interventions have a ton of possibilities. Summarizing the proposed procedure by testing it in group and social learning stages, an unpretentious understudy responsibility examination can be used to make shrewd instructing structures more modified.

CONTENTS

LIST OF FIGURES

1 INTRODUCTION

1.1 PROBLEM STATEMENT

According to previous studies and researches conducted on students aged between 8-18 in different countries [1], [2], there existed a common result – Emotional health of the student and the overall performance in Academics, Extra-Curricular Activities are directly correlated to each other. Government bodies allocate funds for education and human welfare in most of the developed and developing countries, in order to improve the economy of the country, this indicates they desire a fruitful educated workforce to act on demand. Addressing the issue of emotional health of students by detecting it at an early stage and acting on the required changes can help in increase of the overall performance. This project helps to aid in this detection by using face recognition and Emotion Recognition of students in a real-life classroom environment.

1.2 SCOPE OF THE PROJECT

The proposed technology will be able to detect and recognize abnormalities in student's behaviour through facial recognition. Generalizing the proposed method by testing it in collaborative and social learning platforms, an unobtrusive student engagement analysis can be used to make intelligent tutoring systems more personalized.

The proposed system will be designed to accomplish the following objectives:

1. Detection of negative emotions of a particular ward during a lecture.
2. Students engagement in a lecture.

1.3 CURRENT SCENARIO

Every year more cases of anxiety and depression are reported in children and teens. 1 in every 8 children has anxiety, according to the Anxiety and Depression Association of America (ADAA) [1]. While there are treatments for anxiety and depression, 80% of children with an anxiety disorder and 60% of children with depression are not receiving treatment. **It can be difficult for teachers to identify anxiety and depression because these disorders often show up differently for different people, but this is why knowing the combinations of behaviours to look for is key**. **A student dealing with one of these disorders can experience negative effects on their attention, interpretation, concentration, memory, social interaction and physical health.**

Individuals will also interpret everyday situations as dangerous or threatening and will often assume the worst case scenario. *When someone is experiencing anxiety or depression the majority of their mental capacity is used to create and process worrisome thoughts.* This can make it extremely difficult to focus on positive thoughts and can be very exhausting for the student, which detracts from their learning abilities.

Not only do these disorders impact memory, which makes it hard for students to recall information, but they can also have negative effects on how students engage in social situations. Often, students with anxiety or depression will avoid interactions with their peers and will perceive neutral situations as threats. This, in turn, makes others uncomfortable and results in the student feeling lonely, outcast and increases anxiety and depression.

Dealing with undiagnosed depression or anxiety can result in students feeling like they are constantly missing out on opportunities and this can lead to substance abuse, conduct problems, further mental health problems and even suicide.

Currently, suicide is the second leading cause of death among college students. Heather Morgan, a crisis line manager for Didi Hirsch Mental Health Services Suicide Prevention Center says, "We receive calls from college students daily. Since people go through different traumatic events in life; calls, texts and chats can range from relationship issues, LGBTQI questions or concerns, financial, and family issues or concerns – to name a few."

Early anxiety disorders typically predict adult anxiety disorders making it crucial to address them early on. **85% of depressed adolescents have a history of having anxiety as a child.** If these issues can be identified early on while the child is in school there is a better chance for treatment and preventative care so that the above consequences can be avoided.

1.4 Need for the Proposed System

Mental health services embedded within school systems can create a continuum of integrative care that improves both mental health and educational attainment for children. To strengthen this continuum, and for optimum child development, a reconfiguration of education and mental health systems to aid implementation of evidence-based practice might be needed. Integrative strategies that combine classroom-level and student-level interventions have much potential. A robust research agenda is needed that focuses on system-level implementation and maintenance of interventions over time. Both ethical and scientific justifications exist for integration of mental health and education: integration democratises access to services and, if coupled with use of evidence-based practices, can promote the healthy development of children.

1.5 Summary of the Results / Task completed

The authors selected FER2013 dataset images as their dataset for training the model. The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is more or less centred and occupies about the same amount of space in each image. The training set consists of 28,709 examples and the public test set consists of 3,589 examples. They developed a web interface to analyse emotion recognition of either stored video or real-time capture with accuracy rate over 67.5% on 50 epochs interval.

2 LITERATURE REVIEW

2.1 SUMMARY OF THE INVESTIGATION IN THE PUBLISHED PAPERS

There were very few papers that had implemented emotion detection using face recognition techniques, only some of the projects achieved the required results expected, but the accuracy rate of prediction on real-life model is much lower than on the training set owing to the following reasons: limited usage of algorithms like SVM, SVR, LBP on either a tableau or an RNN, the projects refused to go beyond the Machine Learning Domain and refused to utilize techniques from Deep Learning, the datasets used were images depicting a certain emotion which fails to provide the need of detecting emotions on real-time basis, and very few projects decided to exploit the benefits that a scaled-variant CNN architecture has to offer.

The authors, A. Sharma and V. Mansotra; [3] conducted their research based on facial emotion recognition of students in a classroom arrangement and have proposed a deep learning approach to analyze emotions with improved emotion classification results and offer optimized feedback to the instructor. A deep learning-based convolution neural network algorithm was used in this paper to train FER2013 facial emotion images database and they used transfer learning technique to pre-train the VGG16 architecture-based model with Cohn-Kanade (CK+) facial image database, with its own weights and basis. A trained model captured the live steaming of students by using a high-resolution digital video camera that faces towards the students, capturing their live emotions through facial expressions, and classifying the emotions like sad, happy, neutral, angry, disgust, surprise, and fear, it offered the authors insight into the class group emotion that is reflective of the mood among the students in the classroom. This experimental approach can be used for video conferences, online classes etc. They presented their research methodologies and their achieved results on student emotions in a classroom atmosphere and had proposed an improved CNN model based on transfer learning that could suggestively improve the emotions classification accuracy. Krithika L.B & Lakshmi Priya GG, [4]proposed a system that would detect emotions based on Guassian distance between the eyes and eyebrows. It eliminated the need of any device usage requiring physical contact to the subject under study. The existing system helped to identify emotions and classify learner involvement and interest in the topic were plotted as feedback to the instructor to improve the learning experience. This serves as a stepping stone to our proposed system. Experiments, (J. Guo et al; [5]) indicated that pairs of compound emotion (e.g., surprisingly-happy vs happily-surprised) were more difficult to be recognized if compared with the seven basic emotions. The recognition of compound emotions on the iCV-MEFED dataset demonstrated to be very challenging, leaving a large room for improvement. Top winners' methods from FG 2017 workshop have been analyzed and compared. How to incorporate prior information of dominant and complementary categories into compound facial emotion recognition is one question we want to address in future work. The authors (T. S. Ashwin and R. M. R. Guddeti) proposed a convolutional neural network [6] architecture for unobtrusive students' engagement analysis using non-verbal cues. The proposed architecture was trained and tested on faces, hand gestures and body postures in the wild of more than 350 students present in a classroom environment, with each test image containing multiple students in a single image frame. The data annotation was performed using the gold standard study, and the annotators reliably agree with Cohen's $\varkappa = 0.43$. They obtained 71% accuracy for the students' engagement level classification. Further, a pre-test/post-test analysis was performed, and it was observed that there is a positive correlation between the students' engagement and their test performance. This existing research is considered for our proposed system. The authors (Fennell, P.G., Zuo, Z. & Lerman, K) [7] described a statistical approach to modelling behavioural data called the structured sum-of-squares decomposition

(S3D). The algorithm, which was inspired by decision trees, selects important features that collectively explain the variation of the outcome, quantifies correlations between the features, and bins the subspace of important features into smaller, more homogeneous blocks that correspond to similarly-behaving subgroups within the population. They proved that S3D creates parsimonious models that can predict outcomes in the held-out data at levels comparable to state-of-the-art approaches, but in addition, produces interpretable models that provide insights into behaviours. This is important for informing strategies aimed at changing behaviour, designing social systems, but also for explaining predictions, a critical step towards minimizing algorithmic bias. The authors Y. Tang, Q. Mao, H. Jia, H. Song and Y. Zhan; proposed an emotion-embedded visual attention model (EVAM) [8]to learn emotion context information for predicting affective dimension values from video sequences. First, deep CNN was used to generate a high-level representation of the raw face images. Second, a visual attention model based on the gated recurrent unit (GRU) was employed to learn the context information of the feature sequences from facial features. Third, the k-means algorithm was adapted to embed previous emotion into attention model to produce more robust time-series predictions, which emphasize the influence of previous emotion on current effective prediction. In this paper, all experiments were carried out on database AVEC 2016 and AVEC 2017. The experimental results validate the efficiency of the proposed method, and competitive results were obtained. We consider this project as an important guiding stone for our proposed system. In this paper (Q. Mao, Q. Zhu, Q. Rao, H. Jia and S. Luo), a novel three-stage method [9] was proposed to learn hierarchical emotion context information (feature-and label-level contexts) for predicting affective dimension values from video sequences. In the first stage, a feed-forward neural network was used to generate a high-level representation of the raw input features. Then, in the second stage, the bidirectional long short-term memory (BLSTM) layers learn the context information of the feature sequences from the high-level representation and get the initial recognition results of the input. Finally, in the third stage, a BLSTM neural network was used to learn the context information from emotion label sequences by an unsupervised way, which was used to correct the initial recognition results and get the final results. The authors explored the influence of different sequence lengths by sampling from the original sequences. The experiment performed on the video data of AVEC 2015 demonstrated the effectiveness of the proposed method. Their framework highlights that incorporating both feature/label level dependencies and context information is a promising research direction for predicting the continuous dimensional emotion. This research is a stepping stone to our proposed system as it tells us which dataset can help the us in getting a better yield.

2.2 COMPARISON BETWEEN THE TOOLS / METHODS / ALGORITHMS

2.2.1 MODEL COMPARISONS:

In the previous projects mainly Deep Neural Networks were used. Deep Neural Networks [10] that have multiple layers and activation functions (non-linearities as ReLU, elu, tanh, sigmoid etc.) Second of all nonlinearities and multiple layers introduce a nonconvex and usually rather complex error space which means that we have many local minimums that the training of the deep neural network can converge to. This means that a lot of hyperparameters have to be tuned in order to get to a place in the error space where the error is small enough so that the model will be useful. A lot of hyper parameters which could start from 10 and reach up to 40 or 50 are dealt with bayesian optimization using Gaussian processes to optimize them which still does not guarantee good performance.

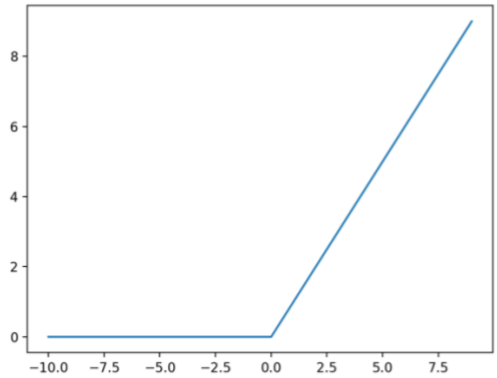


Figure 1: ReLU function graph

Their training is very slow and adding the tuning of the hyperparameters into that makes it even slower where in comparison the linear model would be much faster to be trained. This introduces a serious cost-benefit trade-off. A trained linear model has weights which are interpretable and give useful information to the data scientist onto how various features play a role for the task at hand. Though **RNNs** operate over sequences of vectors: sequences in the input, the output, or in the most general case both in comparison with **CNN** which not only have constrained **Application Programming Interface (API)** but also fixed amount of computational steps. This is why **CNN** is kind of more powerful now than **RNN**. This is mostly because **RNN** has gradient vanishing and exploding problems (over 3 layers, the performance may drop) whereas **CNN** can be stacked into a very deep model, for which it's been proven quite effective.

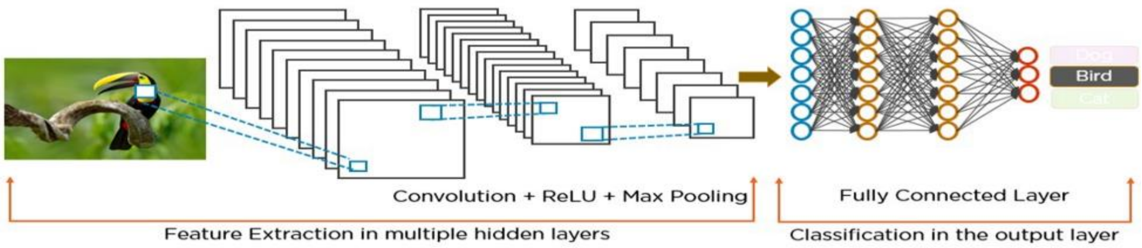


Figure 2:Convolutional Neural Network (CNN)

But **CNNs** are not also flawless. A typical **CNN** can tell the type of an object but can't specify their location. This is because **CNN** can regress one object at a time thus when multiple objects remain in the same visual field then the **CNN** bounding box regression cannot work well due to interference. As for example, **CNN** can detect the bird shown in the model below but if there are two birds of different species within the same visual field it can't detect that.

While an **R-CNN** (R standing for regional, for object detection) can force the **CNN** to focus on a single region at a time improvising dominance of a specific object in a given region. Before feeding into **CNN** for classification and bounding box regression, the regions in the **R-CNN** are resized into equal size following detection by selective search algorithm. Therefore, it helps to specify a preferred object.

2.2.1.1 ARCHITECTURE MODELS

Sequence to Sequence Learning and Attention

Seq2Seq models are particularly good at translation, where the sequence of words from one language is transformed into a sequence of different words in another language. A popular choice for this type of model is Long-Short-Term-Memory (LSTM)-based models. With sequence-dependent data, the LSTM modules can give meaning to the sequence while remembering (or forgetting) the parts it finds important (or unimportant). Sentences, for example, are sequence-dependent since the order of the words is crucial for understanding the sentence. LSTM are a natural choice for this type of data. Seq2Seq models consist of an Encoder and a Decoder. The Encoder takes the input sequence and maps it into a higher dimensional space (n-dimensional vector). That abstract vector is fed into the Decoder which turns it into an output sequence. The output sequence can be in another language, symbols, a copy of the input, etc. The attention-mechanism looks at an input sequence and decides at each step which other parts of the sequence are important. An attention-mechanism works similarly for a given sequence. In other words, for each input that the LSTM (Encoder) reads, the attention-mechanism takes into account several other inputs at the same time and decides which ones are important by attributing different weights to those inputs. The Decoder will then take as input the encoded sentence and the weights provided by the attention-mechanism.

2.2.1.2 IMPLEMENTATION FRAMEWORKS

Theano

Theano [11] is completely Python based library that allows user to define, optimize and evaluate mathematical expressions evolving multi-dimensional arrays efficiently. It can be deployed on single GPU and allows transparent use of GPU to perform data intensive computations. It integrates with Numpy allowing efficient symbolic differentiation. It has been powering large scale computationally intensive investigations quiet efficiently.

Advantages of Theano:

- Open-source deep-libraries such as Keras, Lasagne and Blocks have been built on top of Theano.
- Computational graph is nice abstraction
- Raw Theano is somewhat low-level
- It has some high level wrappers such Keras, Lasagne which increases it usability

Drawbacks of Theano:

- It can be troublesome on AWS
- It can be deployed on single GPU
- Large models can demand long compile times
- Error messages doesn't help much in debugging
- Much fatter than Torch

TensorFlow

TensorFlow [10] is offering users a great amount of documentation for installation and learning materials aimed at helping beginners understand the theoretical aspects of neural networks and help in setting it up. TensorFlow also has the ability to do partial sub graph computation, which is not offered in other frameworks. It is worth noting that one of the Theano frameworks, Keras, supports TensorFlow.

Existing for over a year in the industry, TensorFlow has become one of the most widely adopted open source library for performing fast gradient based machine learning on GPUs, and has a flexible architecture allowing users to deploy computation to one or more CPUs and GPUs in a desktop or mobile.

Advantages of TensorFlow:

- It supports reinforcement learning and other algorithms

- Offers computational graph abstraction
- Has a faster compile time than Theano
- Offers TensorBoard for visualisation
- Offers data and model parallelism
- It can be deployed on multiple CPUs and GPUs

Drawbacks of TensorFlow:

- It doesn't support matrix operations, making copying these large matrices a costly affair
- It runs dramatically slower than other frameworks
- It doesn't have pertained models
- Computational graph can be slow
- It is not commercially supported
- Drops out to Python to load each new training batch
- Not very toolable
- Dynamic typing is error-prone on large software projects

But TensorFlow is comparatively easier to use as it provides a lot of Monitoring and Debugging Tools. Theano takes the Lead in Usability and Speed, but TensorFlow is better suited for Deployment. Paperwork or Documentation for Theano is more than TensorFlow and TensorFlow being a new Language people don't have many resources, to begin with. Open-source deep-libraries such as Keras, Lasagne and Blocks have been built on top of Theano.

2.2.1.3DATASET USED:

FER2013 Database

The data consists of 48x48 pixel grayscale images of faces [12]. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

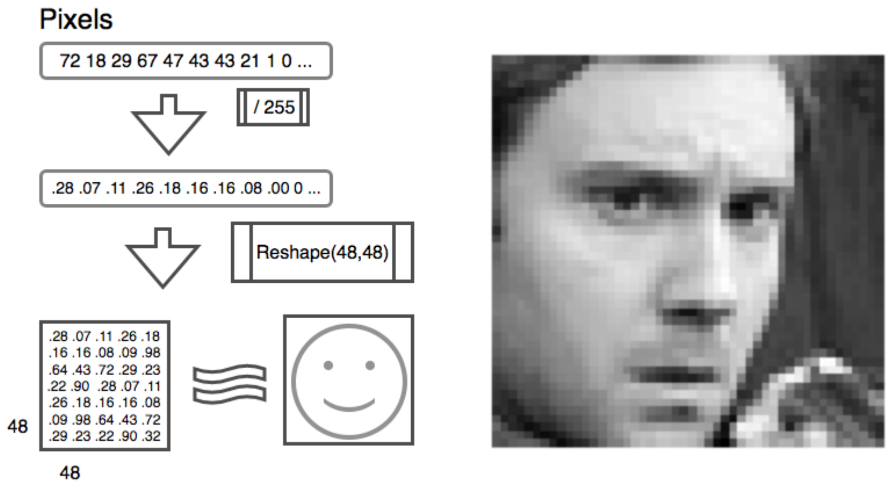


Figure 3: Pixels to Face

The contents of this string a space-separated pixel values in row major order. test.csv contains only the "pixels" column and your task is to predict the emotion column.

The training set consists of 28,709 examples. The public test set used for the leaderboard consists of 3,589 examples. The final test set, which was used to determine the winner of the competition, consists of another 3,589 examples.

This dataset was prepared by Pierre-Luc Carrier and Aaron Courville, as part of an ongoing research project. They have graciously provided the workshop organizers with a preliminary version of their dataset to use for this contest.

3 Analysis and Design

3.1 Methodology / Procedure adopted

The research began by reviewing existing research papers and defining our problem statement. Once this was achieved could we have come to a stand of deciding a final method of implementation. The authors were suggested to try implementing the research on a single person to detect whether the algorithm works in the right manner. They have successfully implemented the proposed methodology in the ReLU 4-layered 2D CNN Architecture with the algorithm's learning rate ($\boldsymbol{\alpha}$) to 0.0005, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and epsilon $\varepsilon = e - 5$ with the accuracy of the model at 66.7% and lower loss rate of 89% over 50 epochs, thus surpassing previous models.

3.1.1 Algorithm Used:

3.1.1.1 Viola-Jones Face Detection Algorithm:

The job of each stage is to determine whether a given sub-window is definitely not a face or may be a face. A given sub-window is immediately discarded as not a face if it fails in any of the stages.

```
Algorithm: Viola-Jones Face Detection Algorithm
Input: original test image
Output: image with face indicators as rectangles
for i ← 1 to num of scales in pyramid of images do
    Downsample image to create image_i
    Compute integral image, image_ii
    for j ← 1 to num of shift steps of sub-window do
        for k ← 1 to num of stages in cascade classifier do
            for l ← 1 to num of filters of stage k do
                Filter detection sub-window
                Accumulate filter outputs
            end for
            if accumulation fails per-stage threshold then
                Reject sub-window as face
                Break this k for loop
            end if
        end for
        if sub-window passed all per-stage checks then
            Accept this sub-window as a face
        end if
    end for
end for
```

Figure 4: Viola-Jones Algorithm (Generalized)

A simple framework for cascade training is given below:

- f = the maximum acceptable false positive rate per layer.
- d = the minimum acceptable detection rate per layer.
- Ftarget = target overall false positive rate.
- P = set of positive examples.
- N = set of negative examples.

```
F(0) = 1.0; D(0) = 1.0; i = 0


while F(i) > Ftarget
   increase i
   n(i) = 0; F(i)= F(i-1)
```

while F(i) > f × F(i-1)

increase n(i)

use P and N to train a classifier with n(I) features using AdaBoost

Evaluate current cascaded classifier on validation set to determine F(i) and D(i)

decrease threshold for the ith classifier (i.e. how many weak classifiers need to accept for strong classifier to accept)

until the current cascaded classifier has a detection rate of at least d × D(i-1) (this also affects F(i))

N = ∅

if F(i) > Ftarget then

evaluate the current cascaded detector on the set of non-face images

and put any false detections into the set N.

The cascade architecture has interesting implications for the performance of the individual classifiers. Because the activation of each classifier depends entirely on the behaviour of its predecessor, the false positive rate for an entire cascade is: $F = \prod_{i=1}^{K} f_i$. Similarly, the detection rate is: $D = \prod_{i=1}^{K} d_i$.

3.1.1.2 ADAM OPTIMIZER

Adam optimization [13] is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments.

According to Kingma et al., 2014 [14], the method is "*computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/parameters*".

Arguments:

- **learning_rate**: A Tensor, floating point value, or a schedule that is a tf.keras.optimizers.schedules.LearningRateSchedule, or a callable that takes no arguments and returns the actual value to use, The learning rate. Defaults to 0.001.
- **beta_1**: A float value or a constant float tensor, or a callable that takes no arguments and returns the actual value to use. The exponential decay rate for the 1st moment estimates. Defaults to 0.9.
- **beta_2**: A float value or a constant float tensor, or a callable that takes no arguments and returns the actual value to use, The exponential decay rate for the 2nd moment estimates. Defaults to 0.999.
- **epsilon**: A small constant for numerical stability. This epsilon is "epsilon hat" in the Kingma and Ba paper (in the formula just before Section 2.1), not the epsilon in Algorithm 1 of the paper. Defaults to 1e-7.
- **amsgrad**: Boolean. Whether to apply AMSGrad variant of this algorithm from the paper "On the Convergence of Adam and beyond". Defaults to False.
- **name**: Optional name for the operations created when applying gradients. Defaults to "Adam".
- ****kwargs**: Keyword arguments. Allowed to be one of "clipnorm" or "clipvalue". "clipnorm"(float) clips gradients by norm; "clipvalue" (float) clips gradients by value.

3.1.2 SOFTWARE REQUIREMENT SPECIFICATIONS (SRS)

Introduction

Purpose

The project is intended for detection of emotions amongst students using face detection.

Intended Audience
Students.

Intended Use
Analysing and detecting student's emotions in order to diagnose student's mental health or to help teachers to get constructive healthy feedback based on their lectures.

Scope
The proposed system will be designed to accomplish the following objectives:

1. Detection of negative emotions of a particular ward during a lecture.
2. Students engagement in a lecture.

Definitions and Acronyms
NA

Overall Description
User Needs

- Better overall improvement in students' performance.
- Better student-teacher interaction will be achieved.

Assumptions and Dependencies
Students will be paying attention to the lecture by looking at the teacher's direction thereby allowing the CCTV camera to detect and analyse emotions.

System Features and Requirements
Functional Requirements

- **Processor:** 2.6GHz 6-core Intel Core i7, Turbo Boost up to 4.3GHz, with 9MB shared L3 cache.
- **Storage:** 512 SSD.
- **RAM:** 16GB of 2400MHz DDR4 onboard memory.
- **Graphics:** 2.2GHz Radeon Pro 555X with 4GB of GDDR5 memory and automatic graphics switching, Intel UHD Graphics 630.
- **Camera:** 720p FaceTime HD camera.

External Interface Requirements

    - 8th Gen Intel Core i7-4700MQ
    - 16GB RAM
    - 512GB SSD
- Python 3.8.3
- TensorFlow 2.4.1
- Keras 2.4.3

System Features

- Able to detect emotions in a 48 × 48 Haar Cascades boxes.
- Analysis's emotion based on pixel coloration.

Non-functional Requirements

- **Accuracy:** 66.7% over 50 epochs and identify 100% of the emotions.
- **Security:** NA
- **Performance:** Detects faces with 95% accuracy.
- **Quality:** NA.

3.2 PROPOSED SYSTEM

3.2.1 HARDWARE / SOFTWARE REQUIREMENTS

Hardware Requirements

- 8th Gen Intel Core i7-4700MQ
- 16GB RAM
- 512GB SSD

Software requirements

- Mac OS 10.15.6
- TensorFlow 2.4.1
- Keras 2.4.3

System requirements

- Python 3.5–3.8
  - Python 3.8 support requires TensorFlow 2.2 or later.
- pip 19.0 or later (requires manylinux2010 support)
- Ubuntu 16.04 or later (64-bit)
- macOS 10.12.6 (Sierra) or later (64-bit) *(no GPU support)*
- Windows 7 or later (64-bit)
  - Microsoft Visual C++ Redistributable for Visual Studio 2015, 2017 and 2019
- Raspbian 9.0 or later
- GPU support requires a CUDA®-enabled card *(Ubuntu and Windows)*

Python Packages:

- OS
- Random
- Scipy
- sklearn
- argparse
- opencv-python
- itertools
- numpy
- pandas
- seaborn
- matplotlib
- flask

3.2.2 DESIGN DETAILS

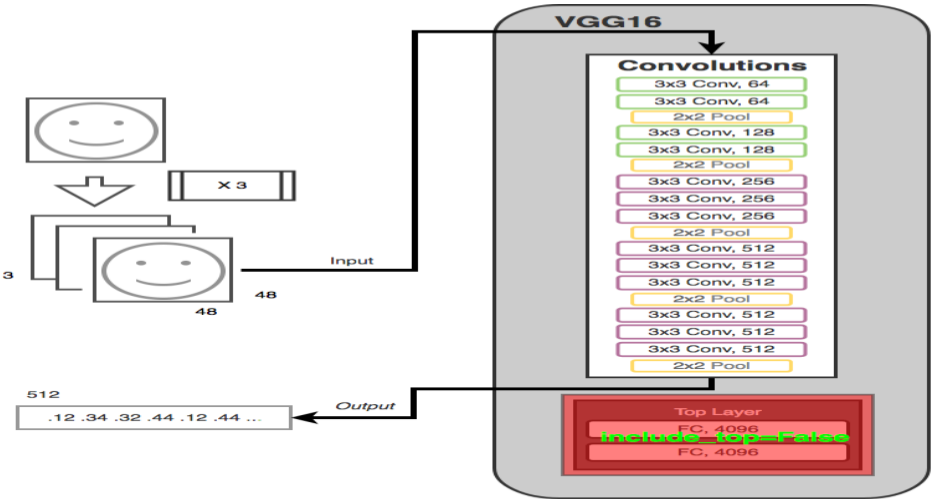


Figure 5: VGG-16 Architecture

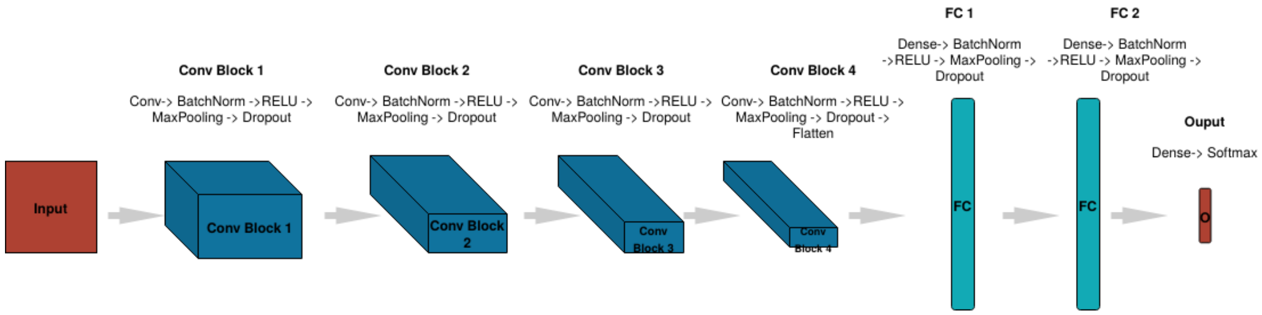


Figure 6:Model Implementation

4 IMPLEMENTATION

4.1 IMPLEMENTATION PLAN

Implementation plan was mainly focused on the Improvisation of accuracy and creating a UI for depicting the Outputs.

4.2. CODING STANDARD

4.2.1 MAIN TRAINING DATASET FILE:

```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import utils
import os
%matplotlib inline

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Dense, Input, Dropout,Flatten, Conv2D
from tensorflow.keras.layers import BatchNormalization, Activation, MaxPooling2D
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau
from tensorflow.keras.utils import plot_model

from IPython.display import SVG, Image
from livelossplot import PlotLossesKeras
import tensorflow as tf
```

```
print("Tensorflow version:", tf.__version__)

utils.datasets.fer.plot_example_images(plt).show()

for expression in os.listdir("train/"):
    print(str(len(os.listdir("train/" + expression))) + " " + expression + " images")

img_size=48
batch_size=64

datagen_train=ImageDataGenerator(horizontal_flip=True)
train_generator=datagen_train.flow_from_directory("train/", target_size=(img_size,img_size), color_mode='grayscale',batch_size=batch_size, class_mode='categorical', shuffle=True)

datagen_test=ImageDataGenerator(horizontal_flip=True)
test_generator=datagen_test.flow_from_directory("test/", target_size=(img_size,img_size), color_mode='grayscale',batch_size=batch_size, class_mode='categorical', shuffle=True)

model=Sequential()

# 1st Convolution Layer
model.add(Conv2D(64,(3,3),padding='same',input_shape=(48,48,1)))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

# 2nd Convolution Layer
model.add(Conv2D(128,(5,5),padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

# 3rd Convolution Layer
model.add(Conv2D(512,(3,3),padding='same'))
model.add(BatchNormalization())
```

```
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

# 4th Convolution Layer
model.add(Conv2D(512,(3,3),padding='same'))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())

# Fully Connected Neural Networks
model.add(Dense(512))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))

model.add(Dense(512))
model.add(BatchNormalization())
model.add(Activation('relu'))
model.add(Dropout(0.5))

model.add(Dense(7,activation='softmax'))
opt=Adam(lr=0.0005,beta_1=0.9,beta_2=0.999,epsilon=1e-5)
model.compile(optimizer=opt, loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()

epochs=50
steps_per_epoch=train_generator.n//train_generator.batch_size
validation_steps=test_generator.n//test_generator.batch_size

checkpoint=ModelCheckpoint("/Users/amanda/Projects/BE-
Project/fer_weights.h5",monitor='val_accuracy',save_weights_only=True,mode='max',verbose=1)
reduce_lr=ReduceLROnPlateau(monitor='val_loss',factor=0.1,patience=2,min_lr=0.00001,mode='auto')
callbacks=[PlotLossesKeras(),checkpoint,reduce_lr]
```

```
history=model.fit(x=train_generator,steps_per_epoch=steps_per_epoch,epochs=epochs,validation_data=test
_generator,callbacks=callbacks)

model_json=model.to_json()
with open("/Users/amanda/Projects/BE-Project/fer_model.json","w") as json_file:
    json_file.write(model_json)
print("Saved Model to JSON File")
```

Total Compilation Time

Model(Improved with 50 epochs): 5 hours.

Model with no set parameters: 8 hours (initial)

4.3 Actual Emotion Recognition

4.3.1 Model Deployment:

```
from tensorflow.keras.models import model_from_json
from tensorflow.python.keras.backend import set_session
import numpy as np

import tensorflow as tf

config = tf.compat.v1.ConfigProto()
config.gpu_options.per_process_gpu_memory_fraction = 0.15
session = tf.compat.v1.Session(config=config)
set_session(session)


class FacialExpressionModel(object):

    EMOTIONS_LIST = ["Angry", "Disgust",
                     "Fear", "Happy",
                     "Neutral", "Sad",
                     "Surprise"]

    def __init__(self, model_json_file, model_weights_file):

        with open(model_json_file, "r") as json_file:
            loaded_model_json = json_file.read()
            self.loaded_model = model_from_json(loaded_model_json)

        self.loaded_model.load_weights(model_weights_file)
```

```
    def predict_emotion(self, img):
        global session
        set_session(session)
        self.preds = self.loaded_model.predict(img)
        return FacialExpressionModel.EMOTIONS_LIST[np.argmax(self.preds)]
```

4.3.2 Camera File:

```
import cv2
from model import FacialExpressionModel
import numpy as np

facec = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
model = FacialExpressionModel("fer_model.json","fer_weights.h5")
font = cv2.FONT_HERSHEY_SIMPLEX

class VideoCamera(object):
    def __init__(self):
        self.video = cv2.VideoCapture("/Users/amanda/Projects/BE-Project/video/presidential_debate.mp4")

    def __del__(self):
        self.video.release()

    def get_frame(self):
        _, fr = self.video.read()
        gray_fr = cv2.cvtColor(fr, cv2.COLOR_BGR2GRAY)
        faces = facec.detectMultiScale(gray_fr, 1.32, 5)

        for (x, y, w, h) in faces:
            fc = gray_fr[y:y+h, x:x+w]

            roi = cv2.resize(fc, (48, 48))
            pred = model.predict_emotion(roi[np.newaxis, :, :, np.newaxis])

            cv2.putText(fr, pred, (x, y), font, 1, (255, 255, 0), 2)
            cv2.rectangle(fr,(x,y),(x+w,y+h),(255,0,0),2)

        _, jpeg = cv2.imencode('.jpg', fr)
        return jpeg.tobytes()
```

4.3.3. Main file:

```
from flask import Flask, render_template, Response
```

```
from camera import VideoCamera


app = Flask(__name__)

@app.route('/')
def index():
    return render_template('index.html')

def gen(camera):
    while True:
        frame = camera.get_frame()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n\r\n')

@app.route('/video_feed')
def video_feed():
    return Response(gen(VideoCamera()),
                    mimetype='multipart/x-mixed-replace; boundary=frame')

if __name__ == '__main__':
    app.run(host='0.0.0.0',debug=True)
```

4.3.4 Real-Time Detection:

```
import os
import cv2
import numpy as np
from keras.models import model_from_json
from keras.preprocessing import image

#load model
model = model_from_json(open("/Users/amanda/Projects/BE-Project/fer_model.json", "r").read())
#load weights
model.load_weights('/Users/amanda/Projects/BE-Project/fer_weights.h5')


face_haar_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')


cap=cv2.VideoCapture(0)

while True:
    ret,test_img=cap.read()# captures frame and returns boolean value and captured image
```

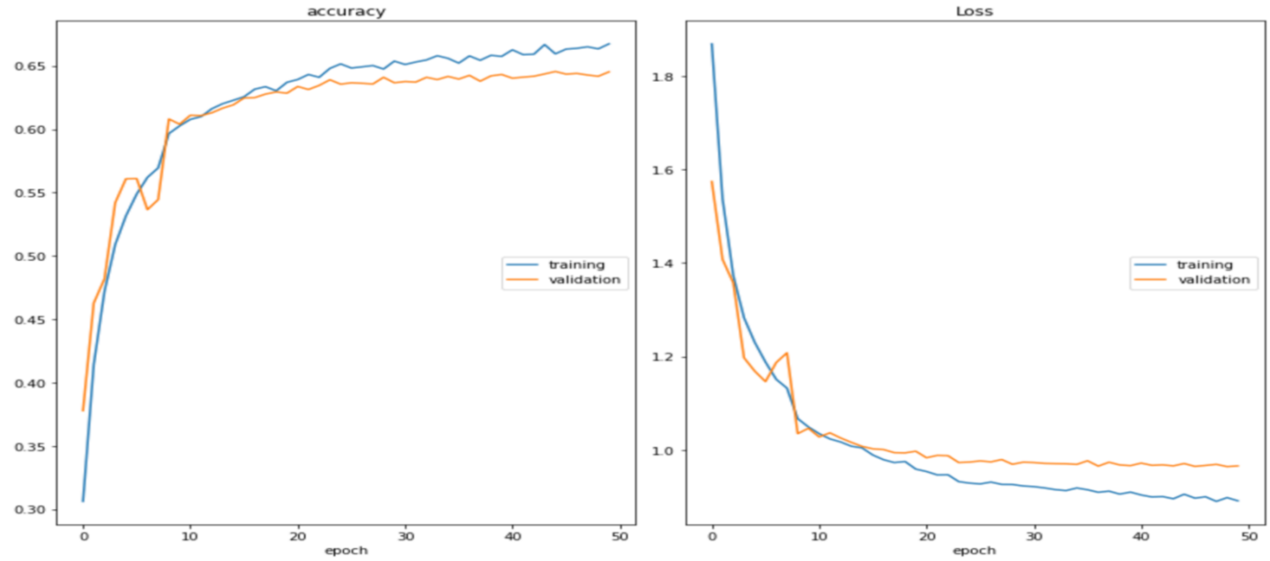
```
    if not ret:
        continue
    gray_img= cv2.cvtColor(test_img, cv2.COLOR_BGR2GRAY)
    faces_detected = face_haar_cascade.detectMultiScale(gray_img, 1.3, 5)

    for (x,y,w,h) in faces_detected:
        cv2.rectangle(test_img,(x,y),(x+w,y+h),(255,0,0),thickness=7)
        roi_gray=gray_img[y:y+w,x:x+h]#cropping region of interest i.e. face area from  image
        roi_gray=cv2.resize(roi_gray,(48,48))
        img_pixels = image.img_to_array(roi_gray)
        img_pixels = np.expand_dims(img_pixels, axis = 0)
        img_pixels /= 255

        predictions = model.predict(img_pixels)

        #find max indexed array
        max_index = np.argmax(predictions[0])

        emotions = ('angry', 'disgust', 'fear', 'happy', 'sad', 'surprise', 'neutral')
        predicted_emotion = emotions[max_index]

        cv2.putText(test_img, predicted_emotion, (int(x), int(y)), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255), 2)

    resized_img = cv2.resize(test_img, (1000, 700))
    cv2.imshow('Emotion Detection  wait until q key is pressed for Quit',resized_img)



    if cv2.waitKey(10) == ord('q'):#wait until 'q' key is pressed
        break

cap.release()
cv2.destroyAllWindows
```

4.4 Testing

4.4.1 Test Cases

```
File - mainExecute
59  449/449 [==============================] - 175s 391ms/step - loss: 0.
    8919 - accuracy: 0.6582 - val_loss: 1.1902 - val_accuracy: 0.5729
60  Epoch 28/30
61  449/449 [==============================] - 188s 418ms/step - loss: 0.
    8829 - accuracy: 0.6665 - val_loss: 1.1987 - val_accuracy: 0.5756
62  Epoch 29/30
63  449/449 [==============================] - 155s 346ms/step - loss: 0.
    8782 - accuracy: 0.6677 - val_loss: 1.1935 - val_accuracy: 0.5807
64  Epoch 30/30
65  449/449 [==============================] - 141s 315ms/step - loss: 0.
    8545 - accuracy: 0.6732 - val_loss: 1.1977 - val_accuracy: 0.5798
66
67  Process finished with exit code 0
```

Figure 7: This model was a 3 layered CNN with no set parameters




```
accuracy
        training             (min:    0.306, max:    0.667, cur:    0.667)
        validation           (min:    0.378, max:    0.646, cur:    0.645)
Loss
        training             (min:    0.890, max:    1.869, cur:    0.891)
        validation           (min:    0.964, max:    1.574, cur:    0.966)
```

Figure 8: 4-layered CNN Model with set parameters give the following results over training at 50 epochs

4.4.2 ACTUAL TEST SUBJECTS:

Real-Time Detection:

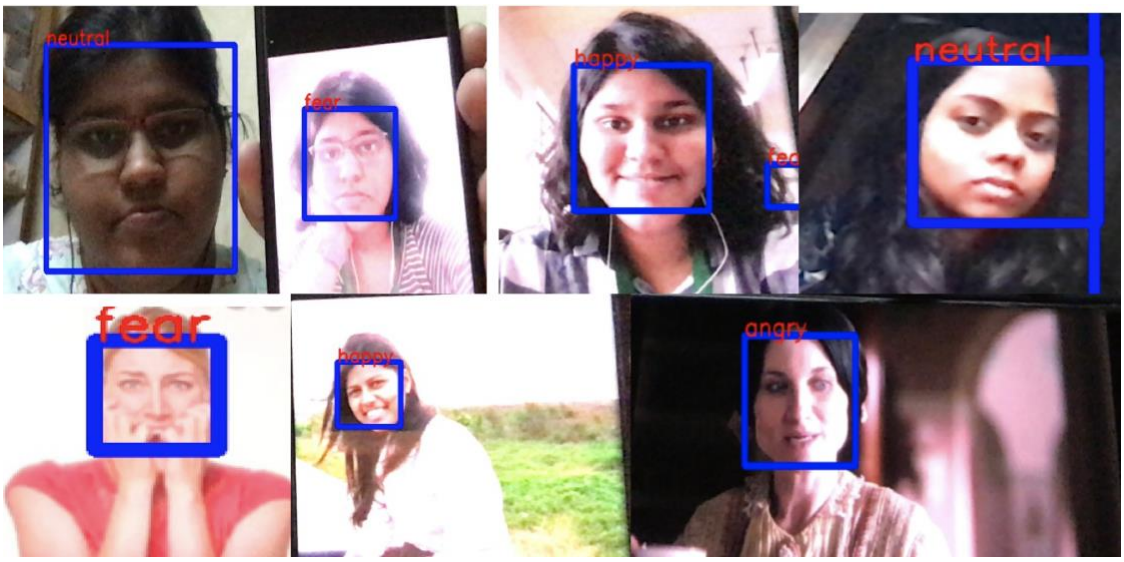


Figure 9: Actual Test Subjects. This includes a running TV Video show (Victoria). In this case the emotions neutral, fear, happy are being identified with 66.7% accuracy. Most often emotions like sad and angry were rarely detected. Disgust was identified as fear.

Detection done on Stored video:

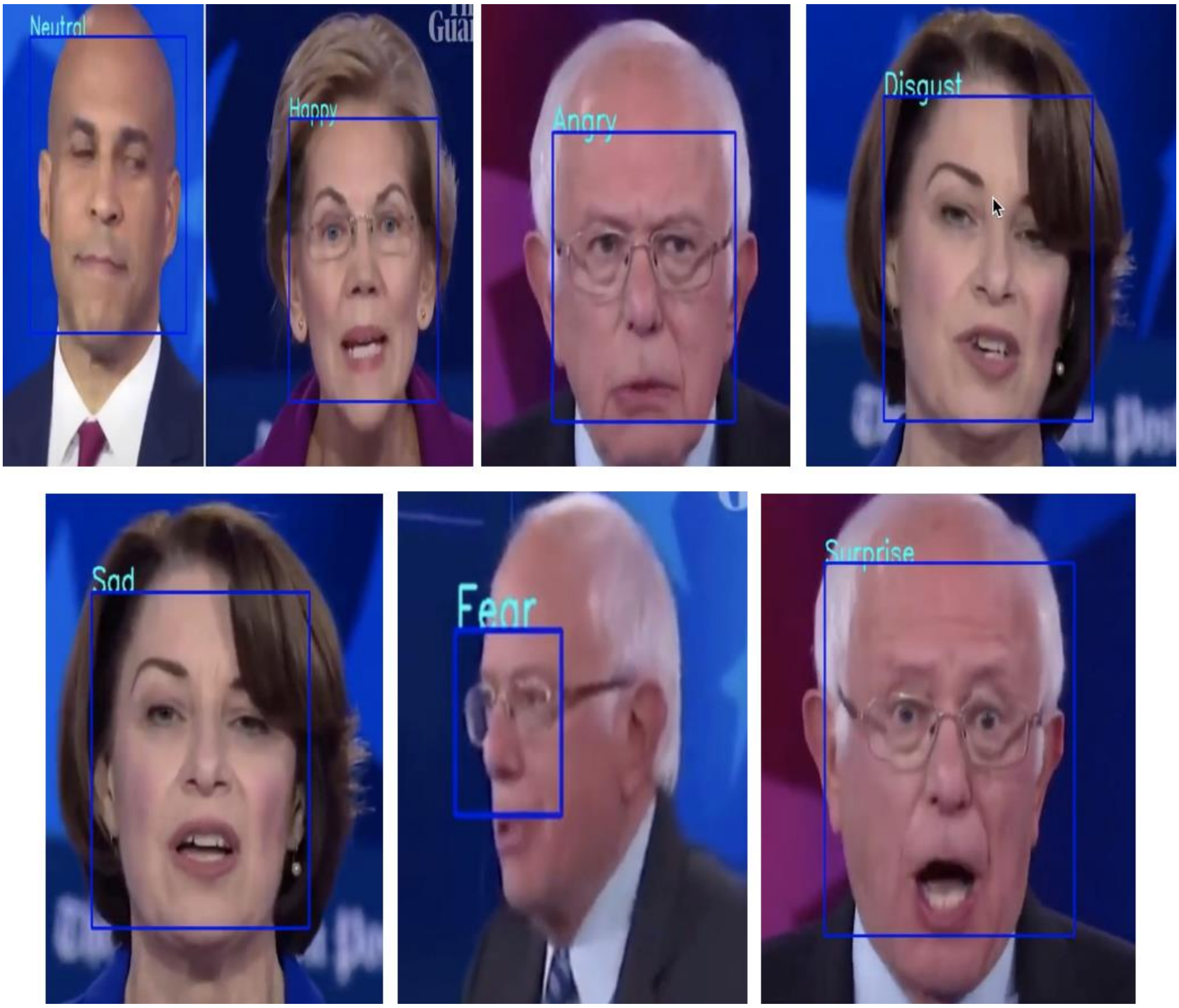


Figure 10: Emotion Detection over Stored Video. Presidential Debate 2020(https://youtu.be/BIm_KpBshtA)

5 Results and Discussion

5.1 Results

The project when implemented in real-life subjects gives approximately 67% accuracy.

5.2 Discussion

Instead of using Viola-Jones Face Detection algorithm for face detection, future groups can use Amazon Rekognition. This module can only work provide students give rights to you as an admin to use their images for facial expression detection.

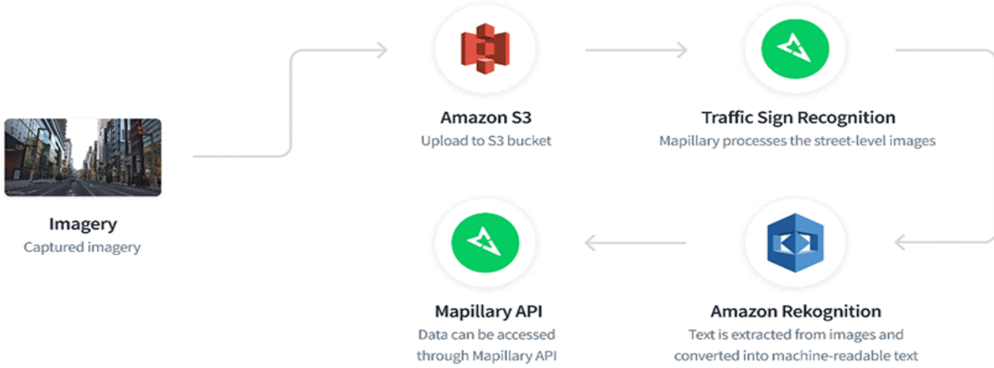


Figure 11:Mapillary uses Amazon Rekognition to work towards building parking solutions for US cities

Amazon Rekognition [15] makes it easy to add image and video analysis to your applications using proven, highly scalable, deep learning technology that requires no machine learning expertise to use. With Amazon Rekognition, you can identify objects, people, text, scenes, and activities in images and videos, as well as detect any inappropriate content. Amazon Rekognition also provides highly accurate facial analysis and facial search capabilities that you can use to detect, analyze, and compare faces for a wide variety of user verification, people counting, and public safety use cases.

With Amazon Rekognition Custom Labels, you can identify the objects and scenes in images that are specific to your business needs. For example, you can build a model to classify specific machine parts on your assembly line or to detect unhealthy plants. Amazon Rekognition Custom Labels takes care of the heavy lifting of model development for you, so no machine learning experience is required. You simply need to supply images of objects or scenes you want to identify, and the service handles the rest.

6 Conclusion and Future Work

6.1 Conclusion

The project can be implemented in real-life scenarios.

6.2 Future Work

Rekognition [16] provides a number of computer vision capabilities, which can be divided into two categories: Algorithms that are pre-trained on data collected by Amazon or its partners, and algorithms that a user can train on a custom dataset.

As of July 2019, Rekognition provides the following computer vision capabilities.

The use of pre-trained model for detecting emotions using cameras via Mobile App Development. This can be implemented through thorough use of Firebase and TensorFlow Lite.

6.2.1 Pre-trained algorithms

- *Celebrity recognition* in images
- *Facial attribute detection* in images, including gender, age range, emotions (e.g. happy, calm, disgusted), whether the face has a beard or moustache, whether the face has eyeglasses or sunglasses, whether the eyes are open, whether the mouth is open, whether the person is smiling, and the location of several markers such as the pupils and jaw line.
- *People Pathing* enables tracking of people through a video. An advertised use-case of this capability is to track sports players for post-game analysis.
- *Text detection and classification* in images
- *Unsafe visual content detection*

6.2.2 Algorithms that a user can train on a custom dataset

- *SearchFaces* enables users to import a database of images with pre-labelled faces, to train a machine learning model on this database, and to expose the model as a cloud service with an API. Then, the user can post new images to the API and receive information about the faces in the image. The API can be used to expose a number of capabilities, including identifying faces of known people, comparing faces, and finding similar faces in a database.
- *Face-based user verification*

Appendix-I

Installation Procedure - Development Software

To use the project on your system the user has to ensure that the system the project is intended to deploy on meets the following constraints:

<u>Hardware Constraints:</u>

- 8th Gen Intel Core i7-4700MQ
- 16GB RAM
- 512GB SSD

Python Version: 3.5 to 3.8

TensorFlow: 2.4.1
Keras: 2.4.3

BIBLIOGRAPHY

[1] [Online]. Available: https://ibcces.org/blog/2019/05/01/impact-anxiety-depression-student-progress/.

[2] H. K. S. S. F. T. Fazel M, “Mental Health Interventions in Schools in High-Income Countries,” *Lancet Psychiatry,* vol. 1, no. 5, pp. 377-387, 2014.

[3] V. M. Archana Sharma, “Deep Learning based Student Emotion Recognition from Facial Expressions in Classrooms',” *International Journal of Engineering and Advanced Technology (IJEAT),* vol. 8, no. 6, p. ISSN: 2249 – 8958, August 2019.

[4] L. P. G. Krithika L.B, “'Student Emotion Recognition System (SERS) for e-learning Improvement Based on Learner Concentration Metric',” no. https://doi.org/1.

[5] J. G. e. al., “"Dominant and Complementary Emotion Recognition From Still Images of Faces,",” *IEEE Access,* vol. 6, no. DOI: 10.1109/ACCESS.2018.2831927, pp. 26391-26403, 2018.

[6] T. S. A. a. R. M. R. Guddeti, “"Unobtrusive Behavioral Analysis of Students in Classroom Environment Using Non-Verbal Cues,",” *IEEE Access,* vol. 7, no. doi: 10.1109/ACCESS.2019.2947519, pp. 150693-150709, 2019.

[7] P. Z. Z. &. L. K. Fennell, “Predicting and explaining behavioral data with structured feature space decomposition.,” *EPJ Data Sci,* vol. 8, no. https://doi.org/10.1140/epjds/s13688-019-0201-0, 2019.

[8] Q. M. H. J. H. S. a. Y. Z. Y. Tang, “"An Emotion-Embedded Visual Attention Model for Dimensional Emotion Context Learning,",” *IEEE Access,* vol. 7, no. doi: 10.1109/ACCESS.2019.2911714., pp. 72457-72468, 2019.

[9] Q. Z. Q. R. H. J. a. S. L. Q. Mao, “"Learning Hierarchical Emotion Context for Continuous Dimensional Emotion Recognition From Video Sequences,",” *IEEE Access,* vol. 7, no. doi: 10.1109/ACCESS.2019.2916211, pp. 62894-62903, 2019.

[10] [Online]. Available: https://datascience.stackexchange.com/questions/21627/what-are-the-disadvantages-of-using-deep-neural-networks-compared-to-a-linear-mo.

[11] a. i. mag. [Online]. Available: https://analyticsindiamag.com/tensorflow-vs-theano-researchers-prefer-artificial-intelligence-framework/.

[12] [Online]. Available: [30]. https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data.

[13] K. A. Python. [Online]. Available: https://keras.io/api/optimizers/adam/.

[14] J. B. Diederik P. Kingma, “Adam: A Method for Stochastic Optimization,” 22 December 2014. [Online]. Available: https://arxiv.org/abs/1412.6980. [Accessed 30 January 2017].

[15] “Amazon Rekognition,” [Online]. Available: https://aws.amazon.com/rekognition/?blog-cards.sort-by=item.additionalFields.createdDate&blog-cards.sort-order=desc.

[16] Wikipaedia, "Amazon Rekognition Wikipaedia," [Online]. Available: https://en.wikipedia.org/wiki/Amazon_Rekognition.

[17] U. o. S. California, Distress Analysis Interview Corpus (DAIC), 2014. [Online]. Available: https://dcapswoz.ict.usc.edu.

[18] J. G. B. S. F. R. D. L. e. a. Michel Valstar, "AVEC 2016 – Depression, Mood, and Emotion Recognition Workshop and Challenge.," 2016.

[19] B. S. M. V. J. G. R. C. e. a. Fabien Ringeval, "AVEC 2017 - Real-life Depression, and Affect Recognition Workshop and Challenge. 7th International Workshop on Audio/Visual Emotion Challenge, AVEC'17, co-locate".

Black Book.pdf

5/21/2021 5:29:56 AM

Plagiarism Free!

Unique Content

100%