BRAC UNIVERSITY

Inspiring Excellence

Emotion Detection & Classification

GROUP 07

SALEHIN RAHMAN KHAN (14101197)

ASHIKUL HAQUE KHAN (14101001)

MAISHA MALIHA (14101013)

FAIYAZ KHALED (14101006)

EHSANUL AMIN KHAN (14101118)

WORK DISTRIBUTION

1. SALEHIN RAHMAN KHAN : **Result Analysis & Environment Setup**
2. MAISHA MALIHA : **Dataset Analysis**
3. FAIYAZ KHALED : **Algorithm Analysis**
4. ASHIKUL HAQUE KHAN : **Image Processing, Code & Library**
5. EHSANUL AMIN KHAN : **Programming, Presentation & Report Writing**

OVERVIEW

- Objective : To Detect and Classify Emotions in Human Faces
- Data Collection : GitHub, Kaggle
- Supervised
- Image Processing
- Neural Network

DATASET ANALYSIS

- We use a set of 28709 pictures of people displaying 7 emotional expression (Angry, Disgusted, Fearful, Happy, Sad, Surprised, Neutral)

- Our dataset contains two columns “Emotion” and “Pixel”. Emotion column contains a numeric code ranging from 0 to 6 and Pixel column contains a string surrounded in quotes for each image.

DATASET ANALYSIS

- In input column there is data consists of 48*48 pixel grayscale images of faces and output column consists of emotions which is represented by numbers from 0 to 6.

- The dataset will be divided into two sets training and testing. Furthermore our dataset has no “nan” or “empty/null” values.

IMAGE PROCESSING

Facial Expressions –

- **Quantitative Dynamics** - determine the amplitude of the expression in terms of intensity levels where the levels correspond to some measures of the extent.

- **Temporal Dynamics** - splitting the expression into three temporal phases like Onset, Apex, Offset.

IMAGE PROCESSING (Cont.)

- Image is converted to grayscale
- Gamma Correction
- Image Enhancement using Gaussian Filter for -
  - **Sharpness**
  - **Low Aliasing**
- After this normalization, the image will be fairly flat limited to noise.

IMAGE PROCESSING (Cont.)

1. Head Pose Identification
2. Face Tracking
3. Face Part Identification
    i. Eye
    ii. Eyebrow
    iii. Mouth

ALGORITHM

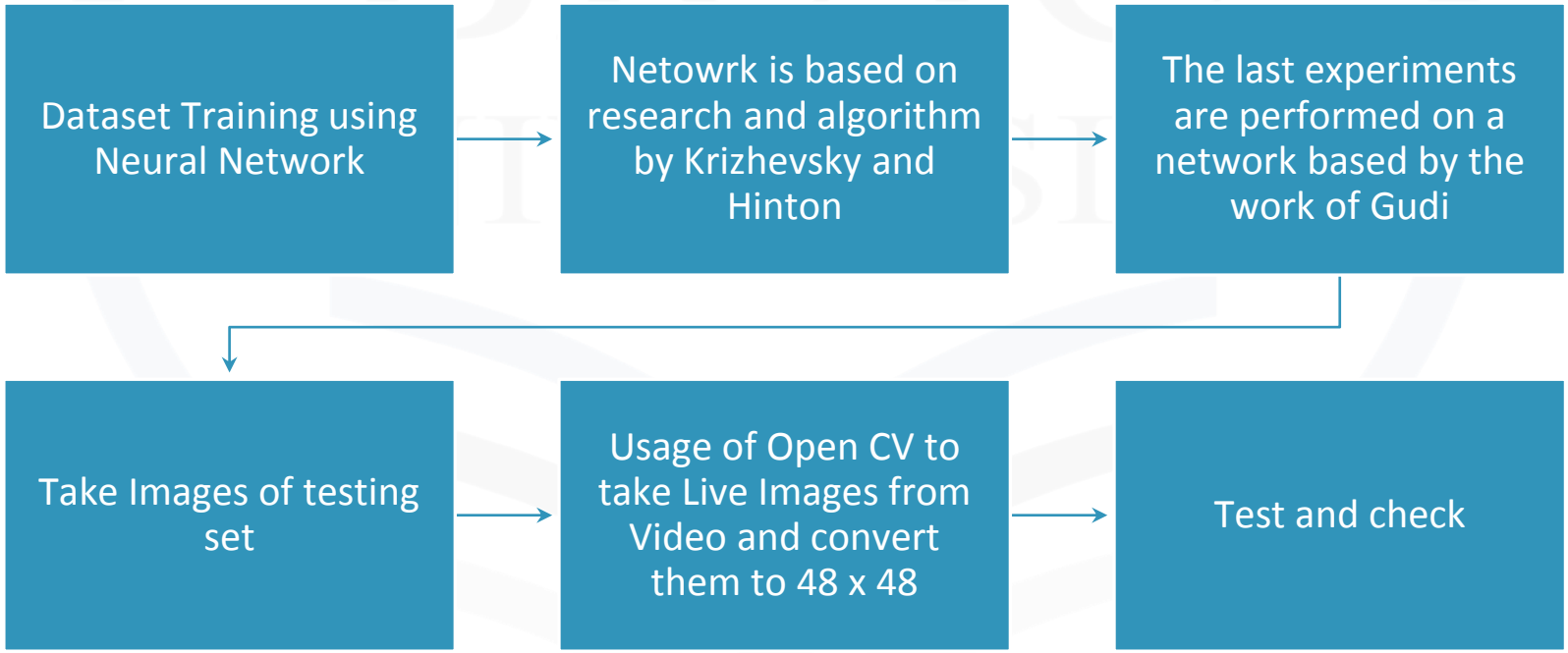


Figure 1

ALGORITHM (Cont.)

- Real time Feedback.
- Three convolutional layers and two fully connected layers.
  - -maxpooling layers(reducing image sizes)
  - -dropout layers(reduce the chance of over fitting)
- Hyper Parameters.
  - -number of calculations remains the same.

DIAGRAM

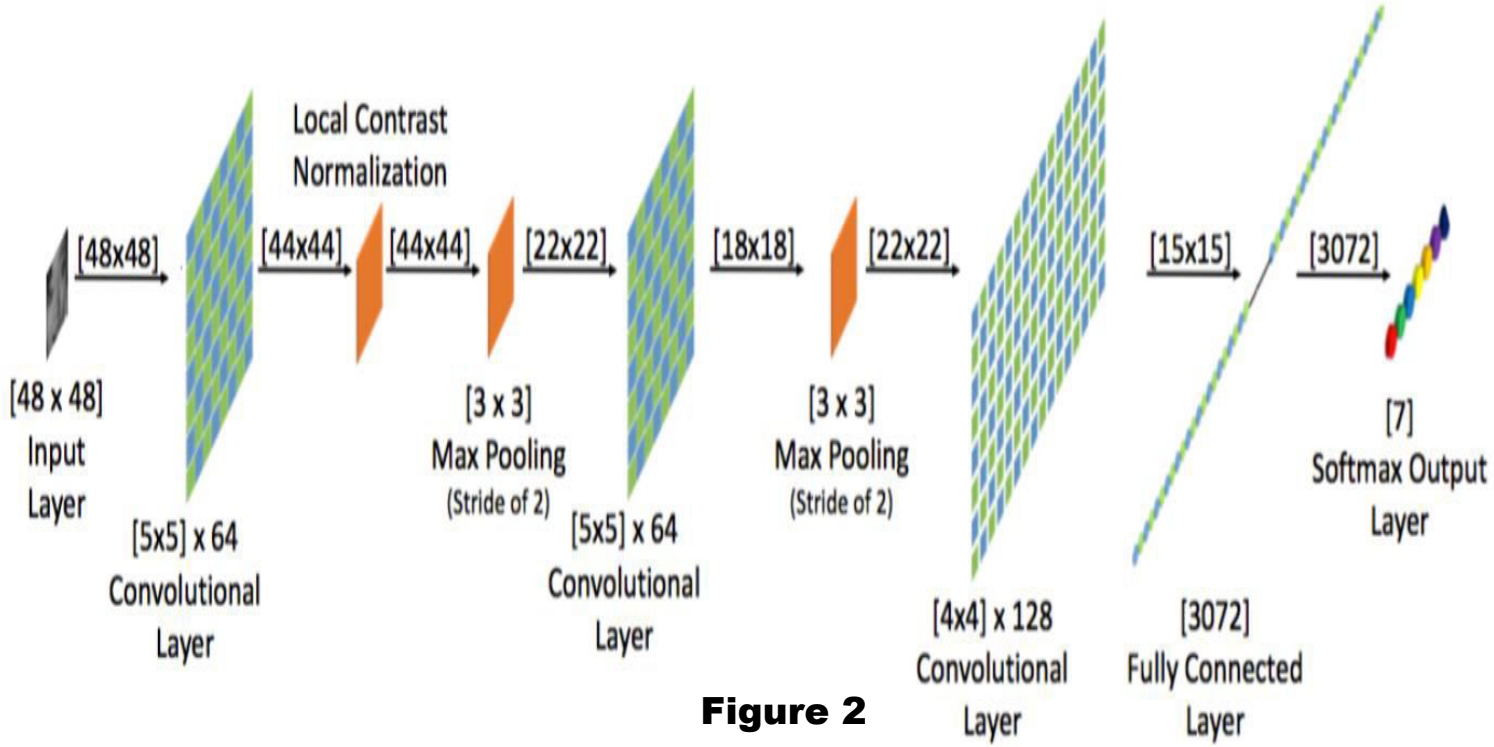


Figure 2

CODE & LIBRARY

- TensorFlow,
- TFLearn,
- OpenCV,
- Python 2.7

TensorFlow WHY & HOW!

TensorFlow is an open-source software library for dataflow programming across a range of tasks. It is a symbolic math library, and also used for machine learning applications such as neural networks

Its Image Processing + learning

What Learning ? Deep learning.

How ? Neural Network

Neural Network

- Artificial neural networks or connectionist systems are computing systems vaguely inspired by the biological neural networks that constitute animal brains.
- RBMs (Restricted Boltzmann Machines)
- DBNs (Deep Belief Networks)
- AlexNET
- GUDI's algorithm

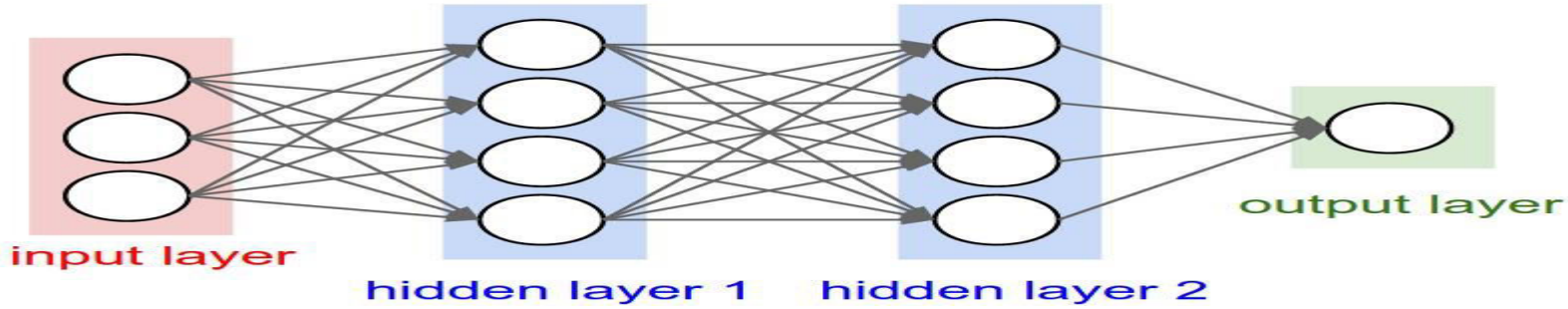


Figure 3

Open CV

- Open Computer Vision
- Used for image processing

Rapid Object Detection using a Boosted Cascade of Simple Features

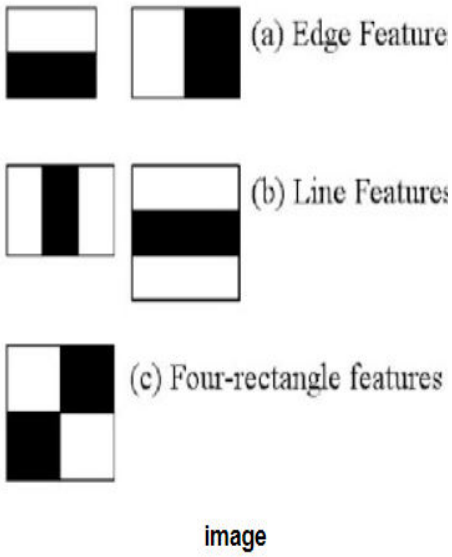


Image 1

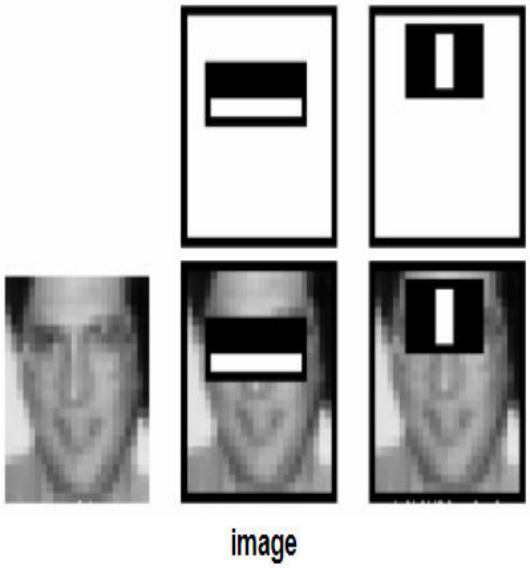


Image 2

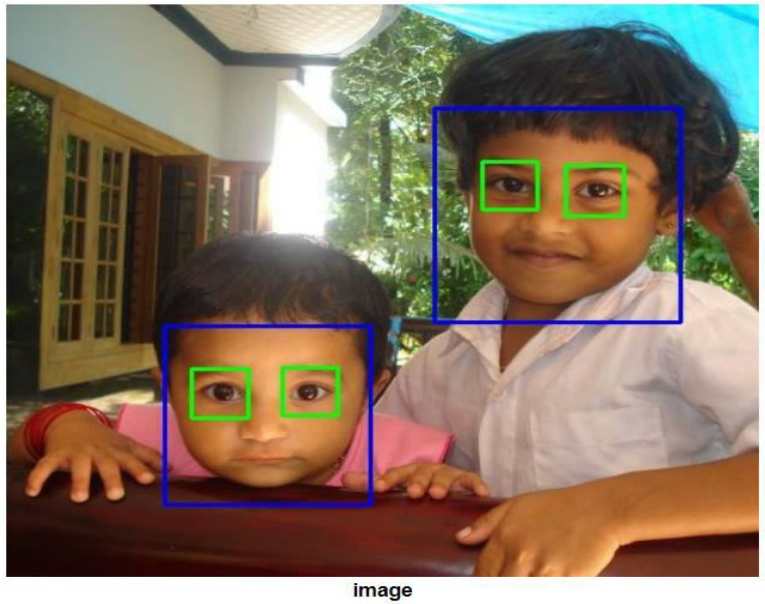
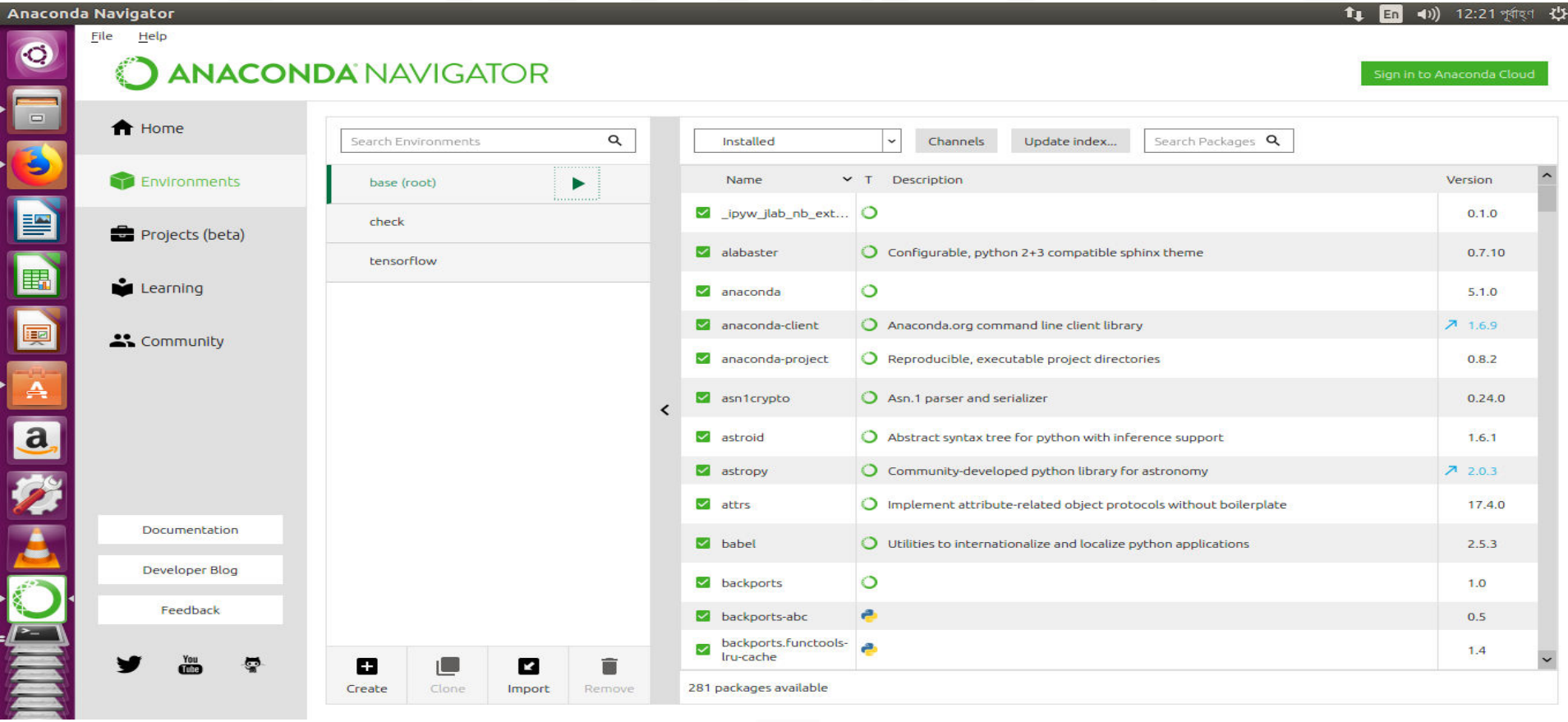
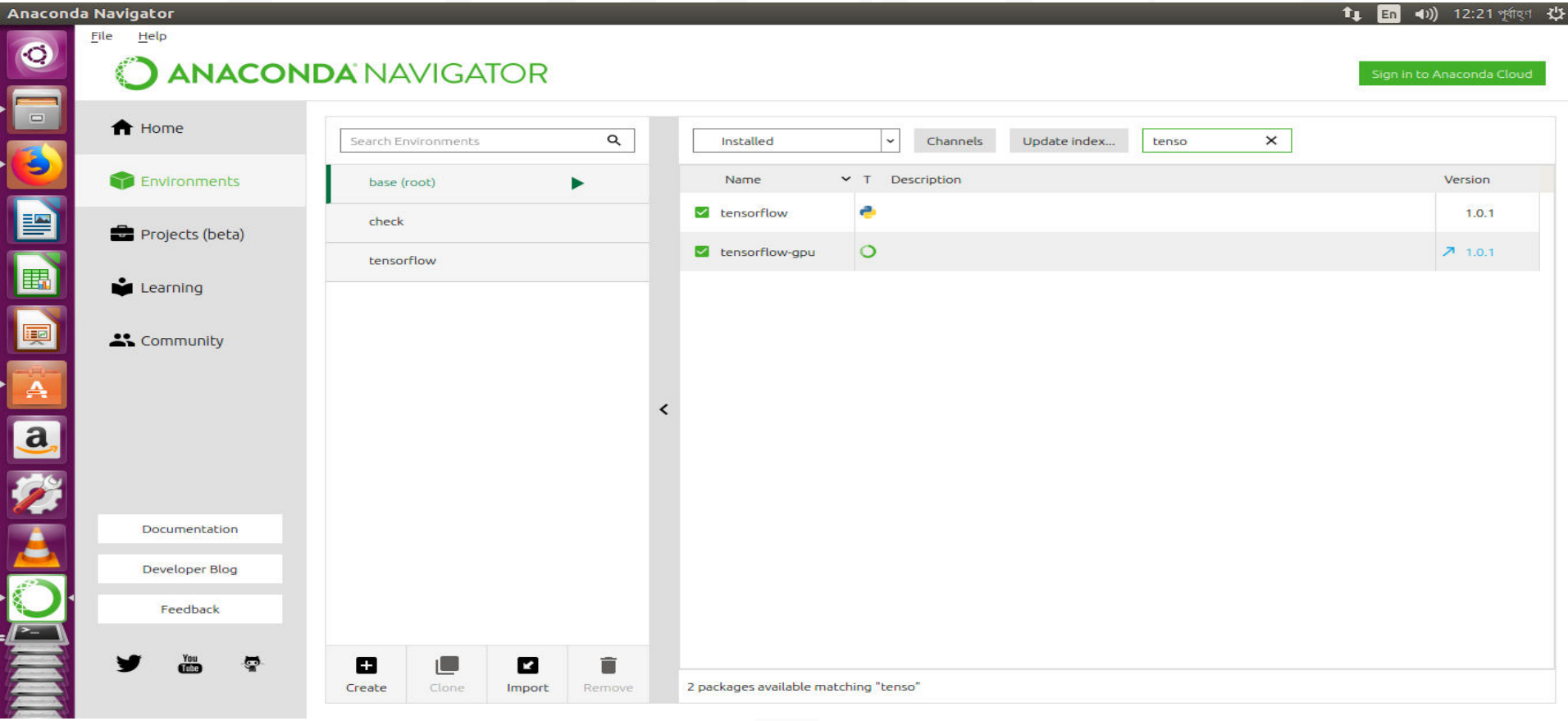


Image 3

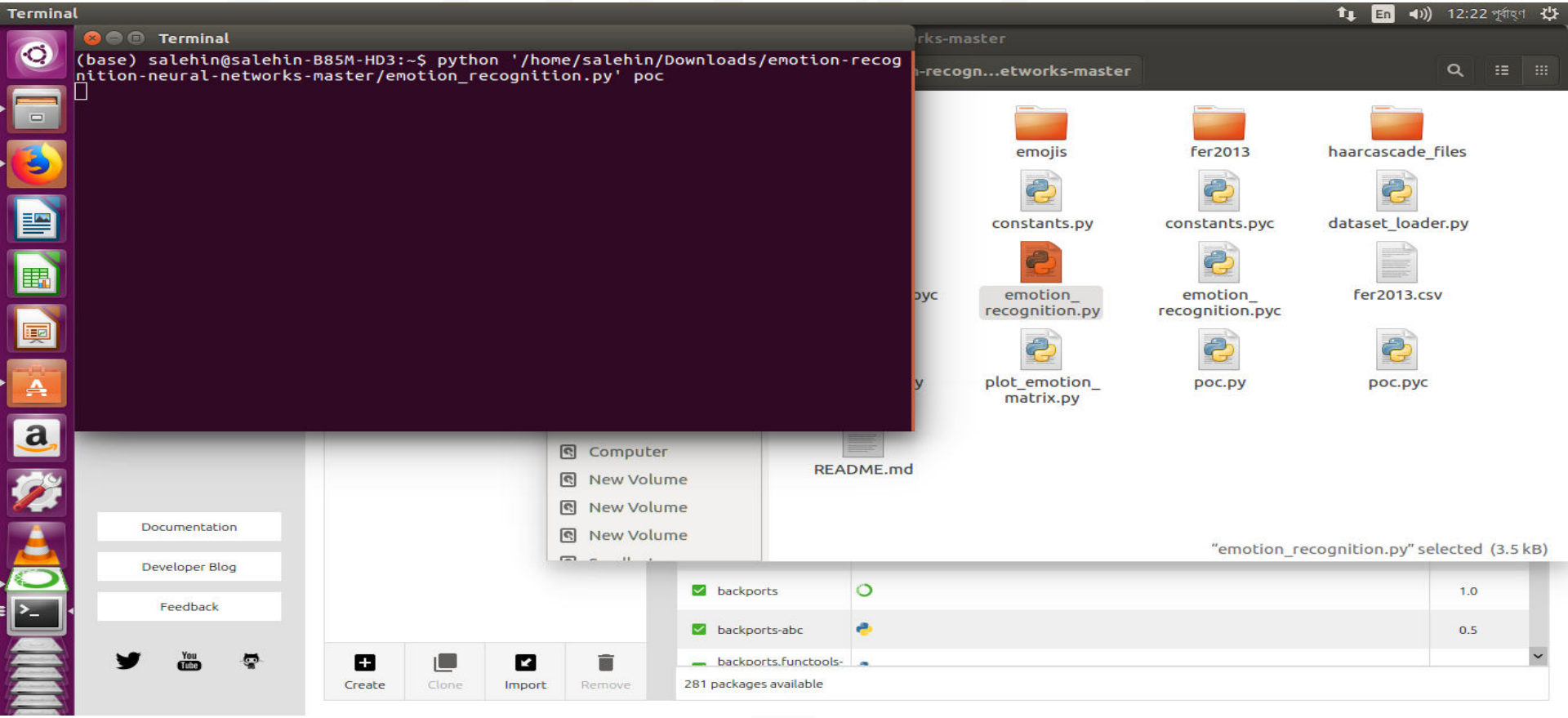
CODES


Anaconda Navigator
File Help
ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Search Environments
base (root)
check
tensorflow
Create
Clone
Import
Remove
Installed
Channels
Update index...
Search Packages
Name
T
Description
Version
_ipyw_jlab_nb_ext... 0.1.0
alabaster Configurable, python 2+3 compatible sphinx theme 0.7.10
anaconda 5.1.0
anaconda-client Anaconda.org command line client library 1.6.9
anaconda-project Reproducible, executable project directories 0.8.2
asn1crypto Asn.1 parser and serializer 0.24.0
astroid Abstract syntax tree for python with inference support 1.6.1
astropy Community-developed python library for astronomy 2.0.3
attrs Implement attribute-related object protocols without boilerplate 17.4.0
babel Utilities to internationalize and localize python applications 2.5.3
backports 1.0
backports-abc 0.5
backports.functools-lru-cache 1.4
281 packages available

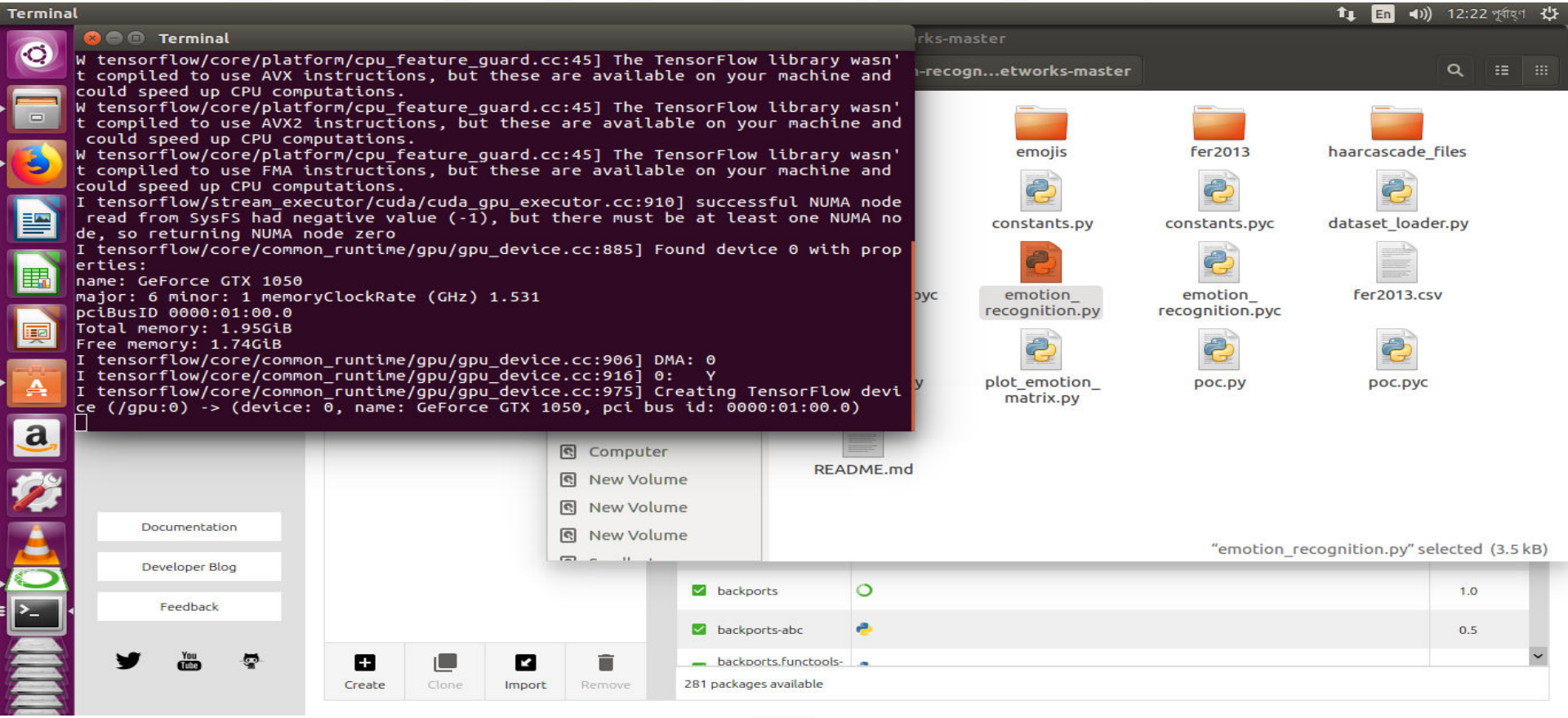
CODES (Cont.)


Anaconda Navigator
File
Help
ANACONDA NAVIGATOR
Sign in to Anaconda Cloud
Home
Environments
Projects (beta)
Learning
Community
Documentation
Developer Blog
Feedback
Search Environments
base (root)
check
tensorflow
Create
Clone
Import
Remove
Installed
Channels
Update index...
tenso
Name
T
Description
Version
tensorflow
1.0.1
tensorflow-gpu
1.0.1
2 packages available matching "tenso"

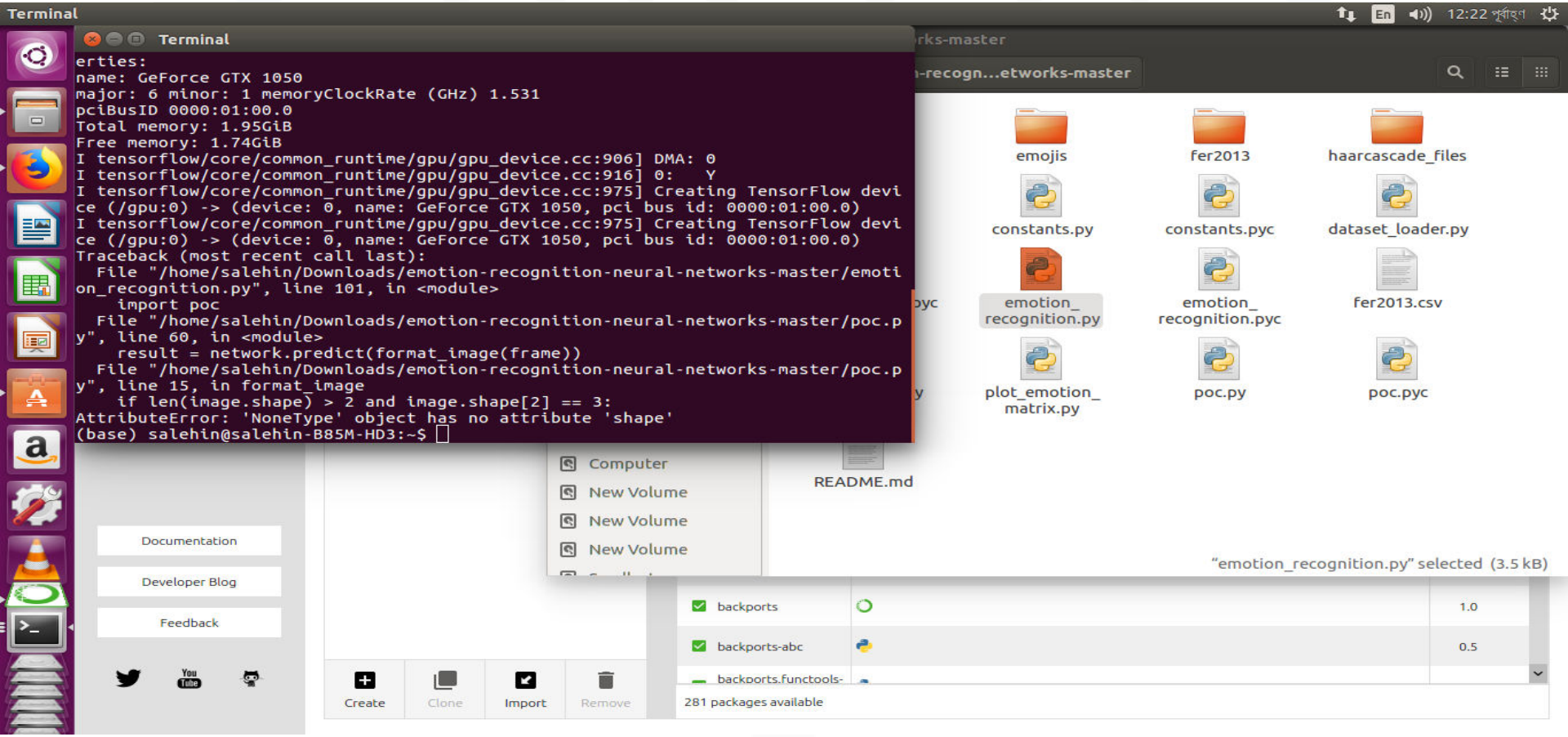
CODES (Cont.)


Terminal
(base) salehin@salehin-B85M-HD3:~$ python '/home/salehin/Downloads/emotion-recognition-neural-networks-master/emotion_recognition.py' poc
emojis
fer2013
haarcascade_files
constants.py
constants.pyc
dataset_loader.py
emotion_recognition.py
emotion_recognition.pyc
fer2013.csv
plot_emotion_matrix.py
poc.py
poc.pyc
README.md
Computer
New Volume
New Volume
New Volume
Documentation
Developer Blog
Feedback
"emotion_recognition.py" selected (3.5 kB)
backports
backports-abc
1.0
0.5
Create
Clone
Import
Remove
281 packages available

CODES (Cont.)


Terminal
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX2 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use FMA instructions, but these are available on your machine and could speed up CPU computations.
I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:910] successful NUMA node read from SysFS had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
I tensorflow/core/common_runtime/gpu/gpu_device.cc:885] Found device 0 with properties:
name: GeForce GTX 1050
major: 6 minor: 1 memoryClockRate (GHz) 1.531
pciBusID 0000:01:00.0
Total memory: 1.95GiB
Free memory: 1.74GiB
I tensorflow/core/common_runtime/gpu/gpu_device.cc:906] DMA: 0
I tensorflow/core/common_runtime/gpu/gpu_device.cc:916] 0: Y
I tensorflow/core/common_runtime/gpu/gpu_device.cc:975] Creating TensorFlow device (/gpu:0) -> (device: 0, name: GeForce GTX 1050, pci bus id: 0000:01:00.0)
emojis
fer2013
haarcascade_files
constants.py
constants.pyc
dataset_loader.py
emotion_recognition.py
emotion_recognition.pyc
fer2013.csv
plot_emotion_matrix.py
poc.py
poc.pyc
README.md
Computer
New Volume
New Volume
New Volume
Documentation
Developer Blog
Feedback
Create
Clone
Import
Remove
backports
backports-abc
281 packages available
"emotion_recognition.py" selected (3.5 kB)

CODES (Cont.)


Terminal
Terminal
erties:
name: GeForce GTX 1050
major: 6 minor: 1 memoryClockRate (GHz) 1.531
pciBusID 0000:01:00.0
Total memory: 1.95GiB
Free memory: 1.74GiB
I tensorflow/core/common_runtime/gpu/gpu_device.cc:906] DMA: 0
I tensorflow/core/common_runtime/gpu/gpu_device.cc:916] 0: Y
I tensorflow/core/common_runtime/gpu/gpu_device.cc:975] Creating TensorFlow device (/gpu:0) -> (device: 0, name: GeForce GTX 1050, pci bus id: 0000:01:00.0)
I tensorflow/core/common_runtime/gpu/gpu_device.cc:975] Creating TensorFlow device (/gpu:0) -> (device: 0, name: GeForce GTX 1050, pci bus id: 0000:01:00.0)
Traceback (most recent call last):
File "/home/salehin/Downloads/emotion-recognition-neural-networks-master/emotion_recognition.py", line 101, in <module>
import poc
File "/home/salehin/Downloads/emotion-recognition-neural-networks-master/poc.py", line 60, in <module>
result = network.predict(format_image(frame))
File "/home/salehin/Downloads/emotion-recognition-neural-networks-master/poc.py", line 15, in format_image
if len(image.shape) > 2 and image.shape[2] == 3:
AttributeError: 'NoneType' object has no attribute 'shape'
(base) salehin@salehin-B85M-HD3:~$
emojis
fer2013
haarcascade_files
constants.py
constants.pyc
dataset_loader.py
emotion_recognition.py
emotion_recognition.pyc
fer2013.csv
plot_emotion_matrix.py
poc.py
poc.pyc
README.md
Computer
New Volume
New Volume
New Volume
Documentation
Developer Blog
Feedback
Create
Clone
Import
Remove
backports
backports-abc
281 packages available
"emotion_recognition.py" selected (3.5 kB)

CODES (Cont.)

emotion_recognition.py (~/Downloads/emotion-recognition-neural-networks-master) - gedit

Open Save

```
from __future__ import division, absolute_import
import re
import numpy as np
from dataset_loader import DatasetLoader
import tflearn
from tflearn.layers.core import input_data, dropout, fully_connected, flatten
from tflearn.layers.conv import conv_2d, max_pool_2d, avg_pool_2d
from tflearn.layers.merge_ops import merge
from tflearn.layers.normalization import local_response_normalization
from tflearn.layers.estimator import regression
from constants import *
from os.path import isfile, join
import random
import sys

class EmotionRecognition:

  def __init__(self):
    self.dataset = DatasetLoader()

  def build_network(self):
    # Smaller 'AlexNet'
    # https://github.com/tflearn/tflearn/blob/master/examples/images/alexnet.py
    print('[+] Building CNN')
    self.network = input_data(shape = [None, SIZE_FACE, SIZE_FACE, 1])
    self.network = conv_2d(self.network, 64, 5, activation = 'relu')
    #self.network = local_response_normalization(self.network)
    self.network = max_pool_2d(self.network, 3, strides = 2)
    self.network = conv_2d(self.network, 64, 5, activation = 'relu')
    self.network = max_pool_2d(self.network, 3, strides = 2)
    self.network = conv_2d(self.network, 128, 4, activation = 'relu')
    self.network = dropout(self.network, 0.3)
    self.network = fully_connected(self.network, 3072, activation = 'relu')
    self.network = fully_connected(self.network, len(EMOTIONS), activation = 'softmax')
    self.network = regression(self.network,
      optimizer = 'momentum',
      loss = 'categorical_crossentropy')
    self.model = tflearn.DNN(
      self.network,
      checkpoint_path = SAVE_DIRECTORY + '/emotion_recognition',
```

Python Tab Width: 8 Ln 23, Col 63 INS

CODES (Cont.)

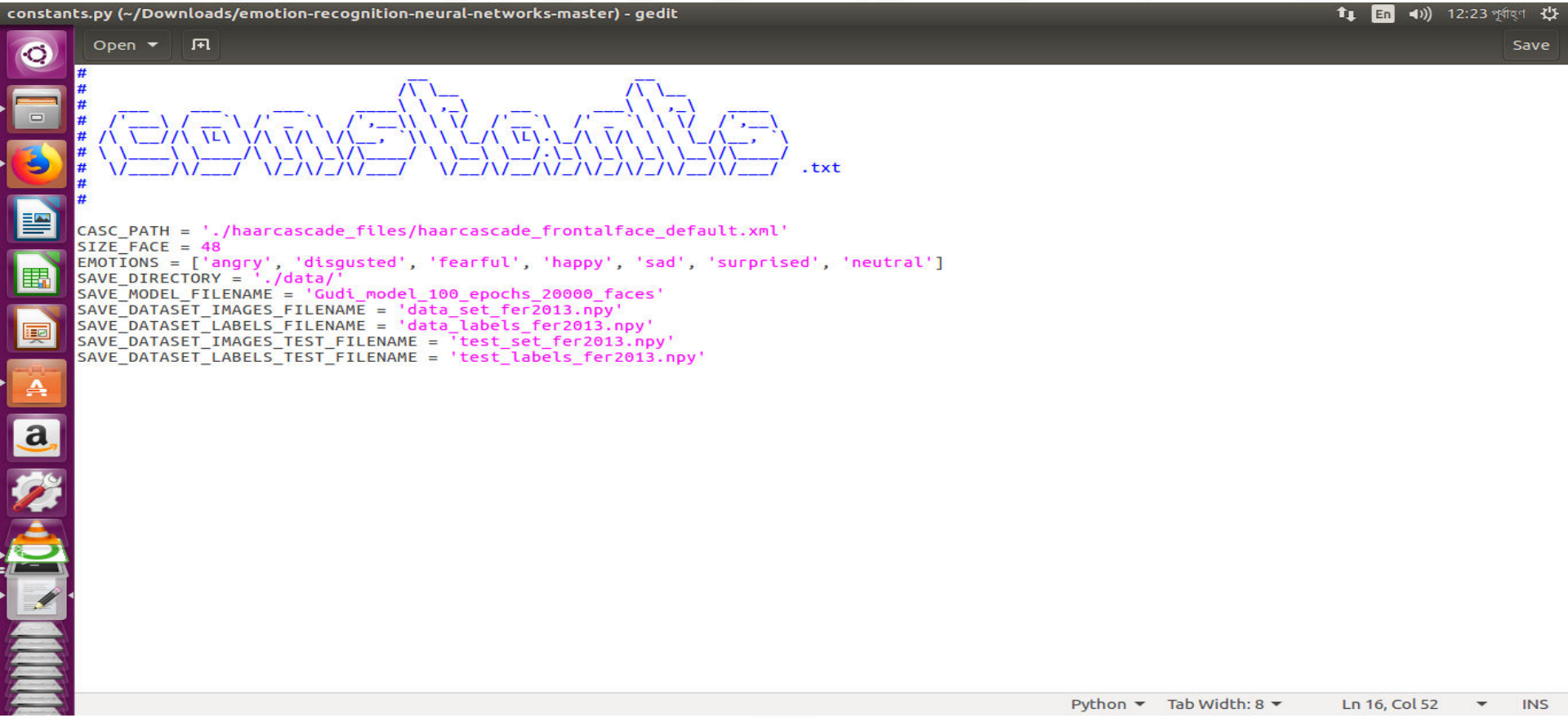
dataset_loader.py (~/Downloads/emotion-recognition-neural-networks-master) - gedit

Open Save

```
from os.path import join
import numpy as np
from constants import *
import cv2

class DatasetLoader(object):

  def __init__(self):
    pass

  def load_from_save(self):
    self._images      = np.load(join(SAVE_DIRECTORY, SAVE_DATASET_IMAGES_FILENAME))
    self._labels      = np.load(join(SAVE_DIRECTORY, SAVE_DATASET_LABELS_FILENAME))
    self._images_test = np.load(join(SAVE_DIRECTORY, SAVE_DATASET_IMAGES_TEST_FILENAME))
    self._labels_test = np.load(join(SAVE_DIRECTORY, SAVE_DATASET_LABELS_TEST_FILENAME))
    self._images      = self._images.reshape([-1, SIZE_FACE, SIZE_FACE, 1])
    self._images_test = self._images.reshape([-1, SIZE_FACE, SIZE_FACE, 1])
    self._labels      = self._labels.reshape([-1, len(EMOTIONS)])
    self._labels_test = self._labels.reshape([-1, len(EMOTIONS)])

  @property
  def images(self):
    return self._images

  @property
  def labels(self):
    return self._labels

  @property
  def images_test(self):
    return self._images_test

  @property
  def labels_test(self):
    return self._labels_test

  @property
  def num_examples(self):
    return self._num_examples
```

Python Tab Width: 8 Ln 29, Col 12 INS

CODES (Cont.)


constants.py (~/Downloads/emotion-recognition-neural-networks-master) - gedit
Open
Save
.txt
CASC_PATH = './haarcascade_files/haarcascade_frontalface_default.xml'
SIZE_FACE = 48
EMOTIONS = ['angry', 'disgusted', 'fearful', 'happy', 'sad', 'surprised', 'neutral']
SAVE_DIRECTORY = './data/'
SAVE_MODEL_FILENAME = 'Gudi_model_100_epochs_20000_faces'
SAVE_DATASET_IMAGES_FILENAME = 'data_set_fer2013.npy'
SAVE_DATASET_LABELS_FILENAME = 'data_labels_fer2013.npy'
SAVE_DATASET_IMAGES_TEST_FILENAME = 'test_set_fer2013.npy'
SAVE_DATASET_LABELS_TEST_FILENAME = 'test_labels_fer2013.npy'
Python
Tab Width: 8
Ln 16, Col 52
INS

CODES (Cont.)

poc.py (~/Downloads/emotion-recognition-neural-networks-master) - gedit

Open Save

```
# Proof-of-concept
import cv2
import sys
from constants import *
from emotion_recognition import EmotionRecognition
import numpy as np

cascade_classifier = cv2.CascadeClassifier(CASC_PATH)

def brighten(data,b):
     datab = data * b
     return datab

def format_image(image):
  if len(image.shape) > 2 and image.shape[2] == 3:
    image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
  else:
    image = cv2.imdecode(image, cv2.CV_LOAD_IMAGE_GRAYSCALE)
  faces = cascade_classifier.detectMultiScale(
      image,
      scaleFactor = 1.3,
      minNeighbors = 5
  )
  # None is we don't found an image
  if not len(faces) > 0:
    return None
  max_area_face = faces[0]
  for face in faces:
    if face[2] * face[3] > max_area_face[2] * max_area_face[3]:
      max_area_face = face
  # Chop image to face
  face = max_area_face
  image = image[face[1]:(face[1] + face[2]), face[0]:(face[0] + face[3])]
  # Resize image to network size
  try:
    image = cv2.resize(image, (SIZE_FACE, SIZE_FACE), interpolation = cv2.INTER_CUBIC) / 255.
  except Exception:
    print("[+] Problem during resize")
    return None
  # cv2.imshow("Lol", image)
```

Python Tab Width: 8 Ln 56, Col 27 INS

CODES (Cont.)

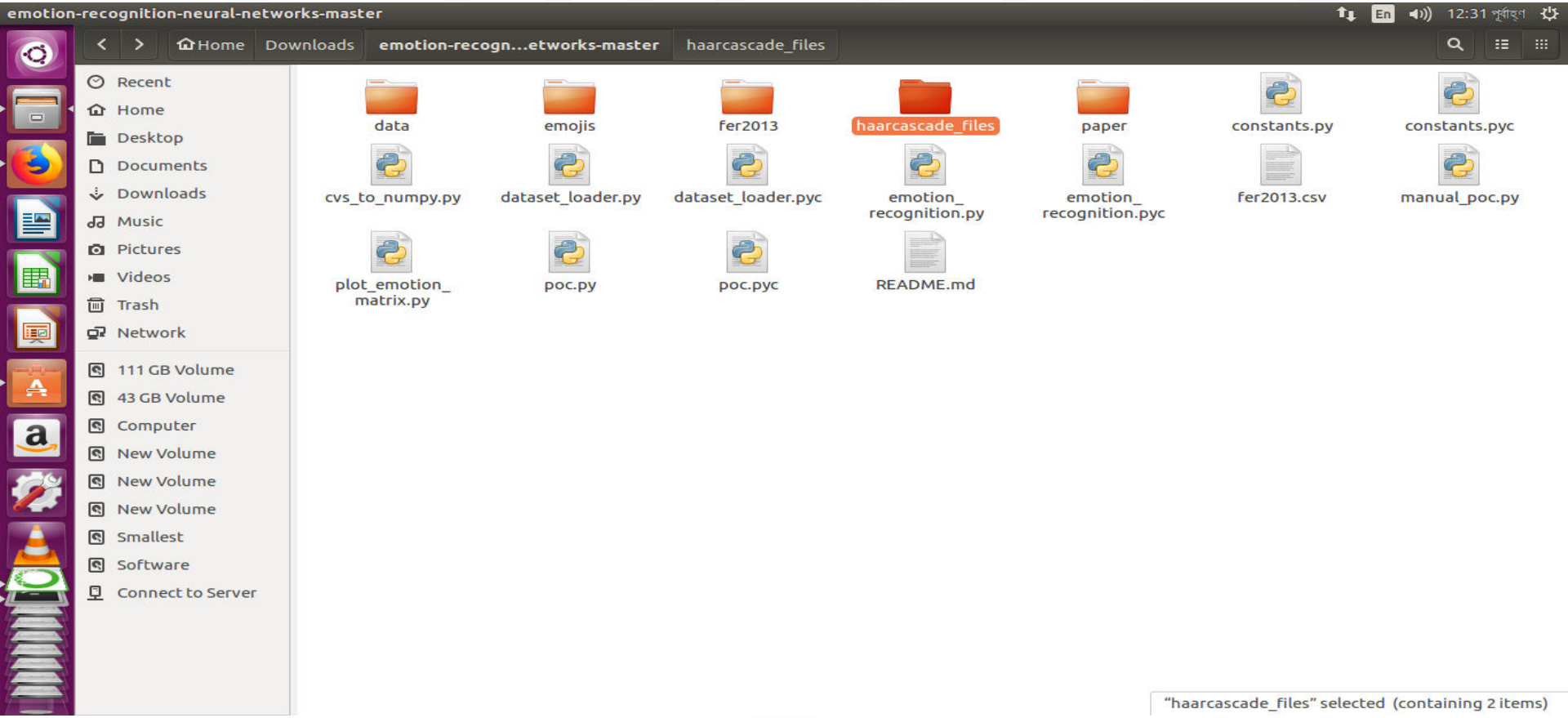
cvs_to_numpy.py (~/Downloads/emotion-recognition-neural-networks-master/data) - gedit

Open Save

```
def emotion_to_vec(x):
    d = np.zeros(len(EMOTIONS))
    d[x] = 1.0
    return d

def flip_image(image):
    return cv2.flip(image, 1)

def data_to_image(data):
    #print data
    data_image = np.fromstring(str(data), dtype = np.uint8, sep = ' ').reshape((SIZE_FACE, SIZE_FACE))
    data_image = Image.fromarray(data_image).convert('RGB')
    data_image = np.array(data_image)[:, :, ::-1].copy()
    data_image = format_image(data_image)
    return data_image

FILE_PATH = 'fer2013.csv'
data = pd.read_csv(FILE_PATH)

labels = []
images = []
index = 1
total = data.shape[0]
for index, row in data.iterrows():
    emotion = emotion_to_vec(row['emotion'])
    image = data_to_image(row['pixels'])
    if image is not None:
        labels.append(emotion)
        images.append(image)
        #labels.append(emotion)
        #images.append(flip_image(image))
    else:
        print "Error"
    index += 1
    print "Progreso: {}/{} {:.2f}%".format(index, total, index * 100.0 / total)

print "Total: " + str(len(images))
np.save('data_kike.npy', images)
np.save('labels_kike.npy', labels)
```

Python Tab Width: 8 Ln 1, Col 1 INS

CODES (Cont.)


emotion-recognition-neural-networks-master
12:31
Home
Downloads
emotion-recogn...etworks-master
haarcascade_files
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Network
111 GB Volume
43 GB Volume
Computer
New Volume
New Volume
New Volume
Smallest
Software
Connect to Server
data
emojis
fer2013
haarcascade_files
paper
constants.py
constants.pyc
cvs_to_numpy.py
dataset_loader.py
dataset_loader.pyc
emotion_recognition.py
emotion_recognition.pyc
fer2013.csv
manual_poc.py
plot_emotion_matrix.py
poc.py
poc.pyc
README.md
"haarcascade_files" selected (containing 2 items)

EXPECTED RESULTS

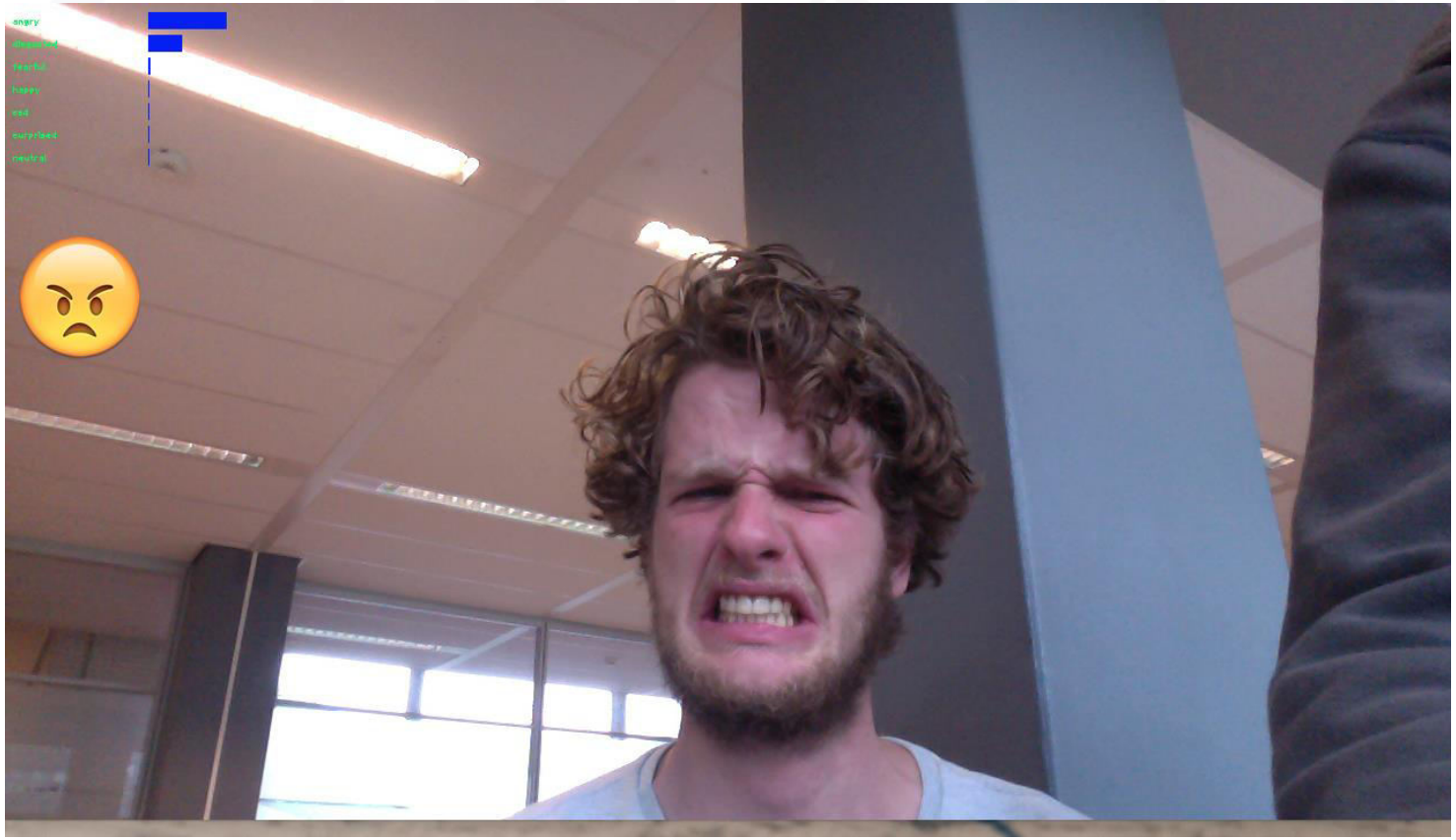


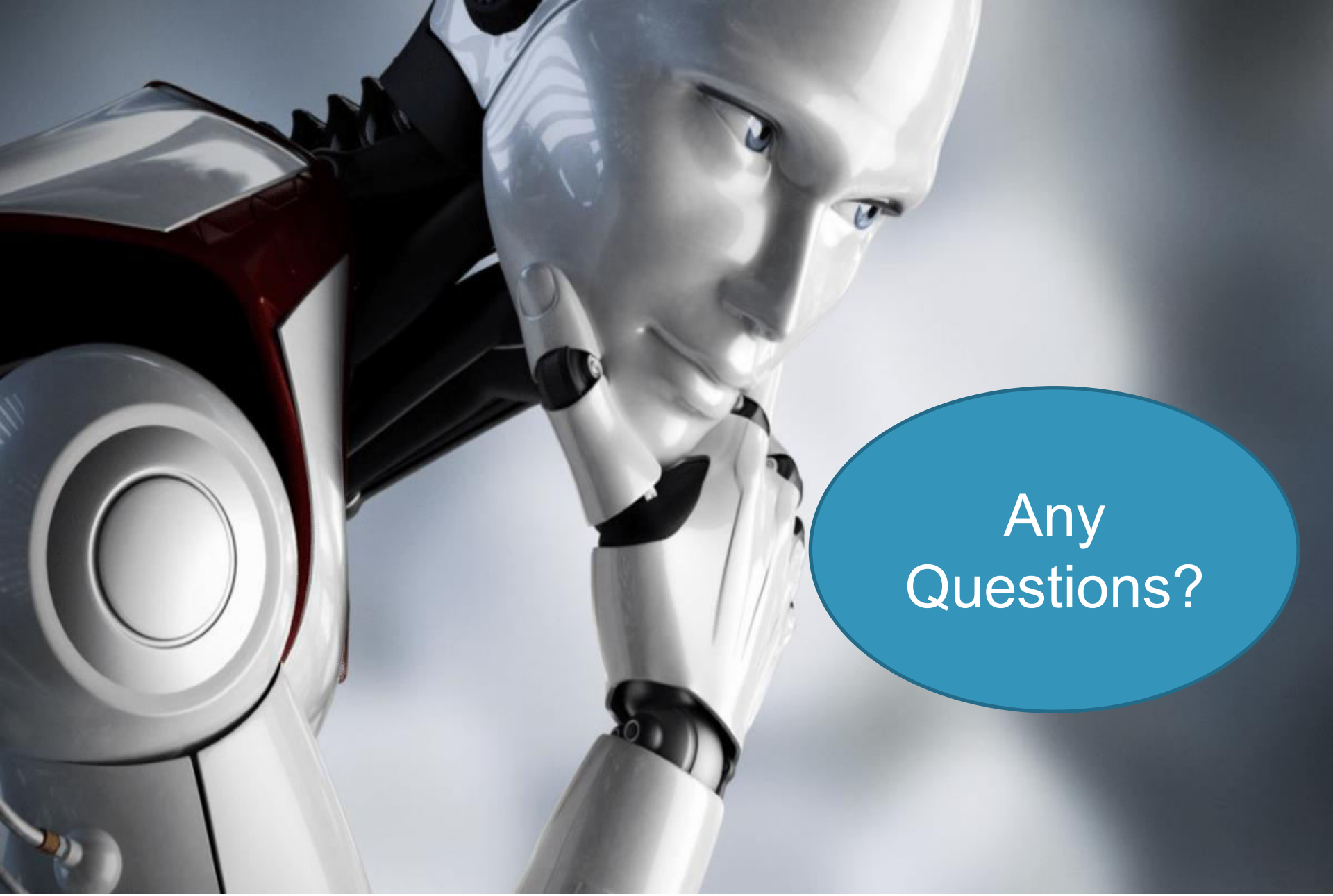
Image 4

CONCLUSION

- Emotion is abstract
- Face expression is not always the emotion
- Real Time
- Real Challenges

REFERENCES

- Kaggle. Challenges in representation learning: Facial expression recognition challenge, 2013.
- A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images, 2009.
- OpenSourceComputerVision. Face detection using haar cascades. URL http://docs.opencv.org/master/d7/d8b/tutorial_py_face_detection.html.


Any
Questions?