HEART RATE ANALYSIS USING CONVOLUTIONAL NEURAL NETWORK

Model Architecture :

VIDEOS --------> Generate two inputs a) RGB frame b) Normalised frame --------> Convlutional layer (100 filters , kernel size 5*5) --------->Convolutional layer(120 , 3*3) -----------> Average Pooling (size 2*2) -------------> Convolutional layer (120 ,3*3) --------> Convolutional layer (90 , 3*3) -------->Average pooling (size 2*2) ----------> Flattened ----------> Dense (512 Neuron) ----------> Dropout (0.3) -------> Dense (32) ----------> Dropout (0.5) ----------> Check with fitted data(HR value) and arranged accordingly ----------> HR reading correspond to each frame

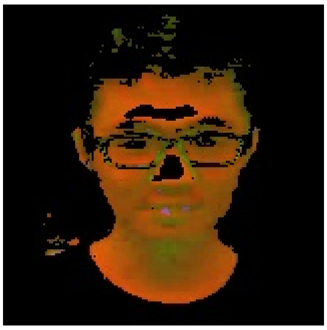
Approach :

1) We used VIPL-HR dataset (Stable Scenerio) . these are 96 videos . Split into train_data and test_data. train_data contain 1 to 70 videos and test_data contain 71 to 96 videos. Similarly create two csv file train_csv contain average HR reading of training data. And test_csv contain average HR reading of testing data.

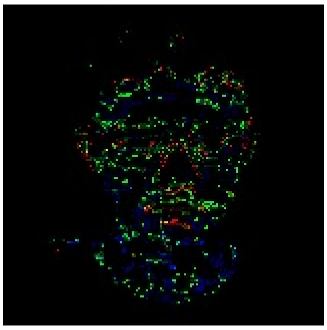
2) As per paper. We need two input image frame one is original RGB frame and second is normalised frame. (extracted using implemented code video_filter.py attached)

a) RGB frame – we create face mask with upper and lower HSV values to remove unwanted data from images. (for apperance purpose)

b) Normalised frame – It shows changes in motion between two consecutive frame.



Generated RGB frame



Generated normalised frame

Generated 100 + RGB frame and Normalised frame of each video.

3) After preprocessing data , we need to convert dataset into suitable format to input into neural network model for training and testing purpose. For same we need two input image frame and corresponding heart rate reading. So we made pair of 3 (RGB

frame ,normalised frame , and corresponding HR reading) and stored it in csv file using implemented code (arrange_dataset.py attached for reference) and similarly used this csv file to create neural network model and for training purpose.

4) Reshape all frames of size 72*72. 36*36 mentioned in paper but we found it very less

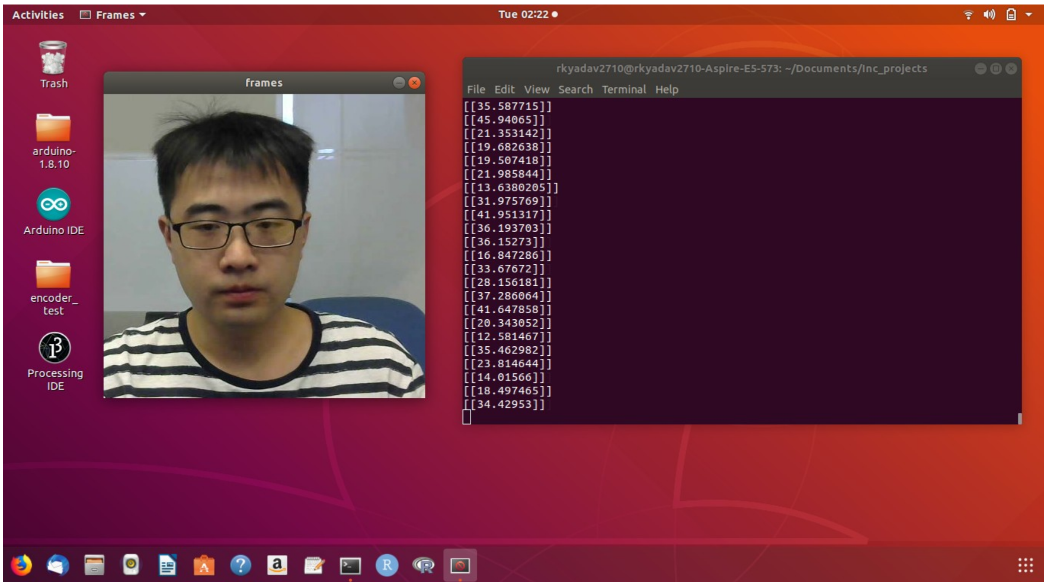
5) we divided dataset into two part x_train and x_test on colab to create validation set.

6) Neural Network Model :

a. To create neural network model we used VGG style CNN (16 layer) as given in paper.
b. we used average pooling layers instead of max pooling. The reasoning was that for physiological measurement, combining an important feature with a less important feature can often produce a higher signal-to-noise ratio than using the more important feature alone.
c. we structured the model so that those two inputs generate a number and in model fit we give HR reading (y_train) as input so that it will correct output.
d. we used very less dataset so we give epoch = 20 and batch size = 15
e. we are not getting any accuracy because it might be detect actual HR with no error. Suppose our actual HR is 75 and estimated HR is 75 in such cases it may give accuracy (JUST PREDICTED NOT SURE).
f. Atlast we stored trained model architecture in .json file and model weights value in .h5 file. (automatically it will be stored in drive)

7) Now, to test trained model we implement code model_demo.py. And extracted .json file .h5 file as a model and gave input frame to predict Heart rate reading corresponding to each frame . And average HR is estimated as final HR.

Sample snapshot of model_demo,py


Activities
Frames
Tue 02:22
Trash
arduino-1.8.10
Arduino IDE
encoder_test
Processing IDE
frames
rkyadav2710@rkyadav2710-Aspire-E5-573: ~/Documents/Inc_projects
File Edit View Search Terminal Help

```
[[35.587715]]
[[45.94065]]
[[21.353142]]
[[19.682638]]
[[19.507418]]
[[21.985844]]
[[13.6380205]]
[[31.975769]]
[[41.951317]]
[[36.193703]]
[[36.15273]]
[[16.847286]]
[[33.67672]]
[[28.156181]]
[[37.286064]]
[[41.647858]]
[[20.343052]]
[[12.581467]]
[[35.462982]]
[[23.814644]]
[[14.01566]]
[[18.497465]]
[[34.42953]]
```