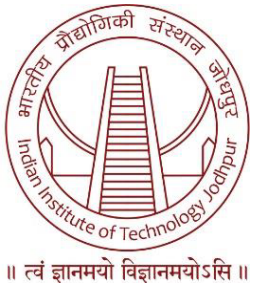
Mapping RGB face-video data to core body temperature at runtime (RAKSHAK)

Mentors: Dr. Romi Banerjee
Dr. Debarati Bhunia Chakraborty

Department of Computer Science and Engineering
Indian Institute of Technology, Jodhpur

Bachelor of Technology, 3rd Year Project



Submitted by-
Rituraj Kulshresth (B18CSE046)
Saurabh Burewar (B18CSE050)

1. Introduction

In these Covid times, as much as staying at home is important, it has also become very important to monitor people who step outside. The best way to recognize possible carriers is to measure body temperature. Normally, body temperature is something that would be taken by a thermometer. Here we are trying to build a solution that could make this process easier.

The normal human body temperature range is typically stated as 36.5-37.5 °C (97.7-99.5 °F). However, human body temperature is variable and dependent upon one's sex, age, time of day, exertion level, health status (i.e. illness, menstrual cycle in females), the location in/on the body in which the measurement is being taken, the subject's state of consciousness (waking, sleeping or sedated), as well as emotional state. Body temperature is maintained within normal range by thermoregulation whereby the lowering or raising of temperature is triggered by the central nervous system.

Oral thermometers are an easy tool to measure the temperature of a body since they give a very accurate and convenient method to get the core temperature of the body. (For this report we are assuming that the core body temperature is measured most accurately using rectal temperature measurement). Oral thermometers report a temperature which is 1-2 degree lower than the core body temperature. Since using thermometers is invasive and requires contact with the body of a person it has to be avoided in the current times. Instead an easier and quick solution is needed which can give accurate results quickly. We can use a thermal camera to get the accurate temperature of the surrounding and a person. However, a thermal camera is very costly. Using the infrared thermometers gives a better solution to the problem. infrared thermometers are cheaper but may not be available to all the public due to the exclusivity. Therefore another approach is needed to find temperature noninvasively. Since each of us have a camera with us at all times we have tried to find the temperature of a body using a regular RGB camera.

We are using an RGB camera to detect the temperature of the core body. For this we have approached this problem using two methods, first is by checking for a direct relation between the rbg images of the face. For this we have used convolutional neural networks and experimented with various networks to find a suitable mapping.

Since there is no direct relation between the RGB images and temperature of a body we have also tried using NIR images of the body to get a mapping between the images and the core body temperature. NIR has a wavelength of above 800 nm so we also tried to generate

the images in a specific range of wavelengths. Since the NIR image of a body is related to the temperature this can be used to find a mapping between the images and temperature

2. Objective

This project is divided into 2 parts

1. To find a relation between the temperature of a human body and the image taken by a camera. This is to be done using both RGB and NIR images of the face of a person
2. The second part of the project is where we try to generate the NIR image from RGB images of the human faces.

3. Challenges / Issues

1. The biggest challenge that is present in this project is the unavailability of data due to the ongoing pandemic and no proper source for reliable RGB NIR image pair along with the temperature of the body
2. In order to cope up with the challenge the projects' first part has been completed on multiple different datasets where one dataset has been collected from home with due permission of the involved party and the other dataset has been collected at the institute. Another dataset that has been used is the epfl RGB-NIR Scene Dataset.

4. Data Collection

One of the main challenges of this project was the lack of data in order to train the model. The data to be collected includes video or image of the face and corresponding core body temperature at the time.

We tried collecting data in our respective hometowns, with due permission of the involved party, by taking RGB images of the forehead of the participants and taking their core body temperature using a IR thermometer/thermal gun. However, due to the covid restrictions, our data collection was limited to a few participants causing the data to be skewed towards the normal body temperature of about 97 / 98 °F which caused problems in training the model with the data.

Further, data is also being collected on IITJ campus which includes RGB and thermal videos of participants using a thermal camera, their core body temperature and factors that can

affect the data such as ambient temperature, activity prior to collection. The data collection is still in progress and thus far, the data we have still poses the same problem that we encountered before and data from more participants pertaining to different conditions is necessary in order to balance the dataset.

Since the data collection (on IITJ campus) is still in progress and not yet completely usable, we have tried using datasets available through various resources and have currently trained the model on one of them. For the second objective, in order to train and test the model to generate NIR images from RGB counterparts, we have used the EPFL RGB-NIR dataset. This dataset contains a set of RGB and NIR images of different scenery ranging from forest and lakes to cities and buildings. Thus, the results seen in this report are based on the said dataset.

5. Relation between temperature and RGB image (using self made dataset)

1. Dataset

The images for the temperature mapping consist of forehead region images for temperatures 96 °F, 97 °F, 98 °F, 99 °F and 100 °F. The images were collected in a room with artificial lighting. Temperature was measured for various times of the day. The temperature measurements were taken after various activities which were also included. The temperature was measured using an IR thermal gun at the forehead region.

To find any relation between the RGB image of a person and the temperature of the person we have used convolutional neural networks. The images have been divided based on their respective body temperature and various convolutional neural networks were experimented with to find any underlying mapping function.

The data was resized to 256 x 256 pixels for easy training.

Appropriate weights were added to each temperature class in order to adjust for the unequal distribution of data

Dataset image

2. Model

Since there was no previous research as to what kind of model would give proper results, We experimented with models consisting of 1 to 6 convolutional layers followed by 0 to 3 dense layers with layer sizes of 32, 64, 128 in all possible combinations. Each convolutional layer is followed by a ReLU activation layer and a Max pooling layer. Each dense layer is followed by a ReLU activation layer. The kernel size of the convolution kernels is the same for all convolutional layers. In order to compensate for the skewed data we weighted the dataset to reach a balanced data.

In our experiment we found that the best results were obtained with layer size of 64 along with 3 convolutional layers and 1 dense layer. This model gave us an accuracy of over 50% when we tested it with our data. However this result may not be absolutely correct due to the small size of the dataset that we used.

We used 'sparse categorical cross entropy' as a loss function for all the models as 'categorical cross entropy' gave extremely poor results. The optimizer used was 'adam'. Some models were also used with each single R, G, B channel. However no significant accuracy result was found so we went back to the complete RGB image.

3. Result

Model	Accuracy	Loss
1-conv-32-nodes-0-dense	19%	3.06
2-conv-32-nodes-0-dense	14% (varied)	2.72
3-conv-32-nodes-0-dense	9% (varied)	2.74

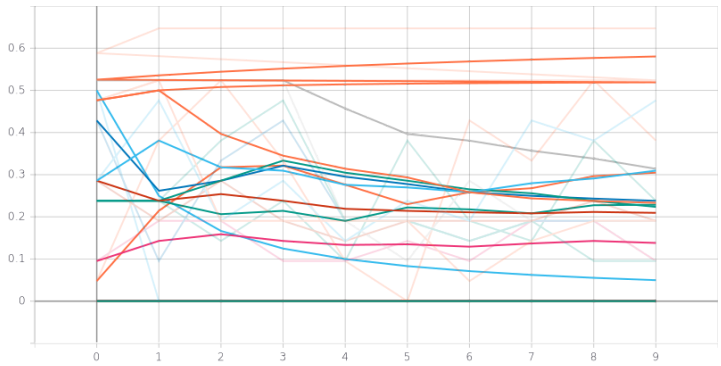
4-conv-32-nodes-0-dense	19%	2.74
5-conv-32-nodes-0-dense	14%	2.77
1-conv-64-nodes-0-dense	50% (varied)	2.77
2-conv-64-nodes-0-dense	9% (varied)	2.72
3-conv-64-nodes-0-dense	23%	2.86
4-conv-64-nodes-0-dense	19%	2.74
5-conv-64-nodes-0-dense	28%	2.73
1-conv-128-nodes-0-dense	19%	2.70
2-conv-128-nodes-0-dense	50% (varied)	2.73
3-conv-128-nodes-0-dense	19%	2.72
4-conv-128-nodes-0-dense	19%	2.75
5-conv-128-nodes-0-dense	50% (varied)	2.72
1-conv-32-nodes-1-dense	19%	2.77
2-conv-32-nodes-1-dense	19%	2.76
3-conv-32-nodes-1-dense	9% (varied)	2.85
4-conv-32-nodes-1-dense	9% (varied)	2.73
5-conv-32-nodes-1-dense	14%	2.75
1-conv-64-nodes-1-dense	47%	3.20
2-conv-64-nodes-1-dense	9% (varied)	2.72
3-conv-64-nodes-1-dense	52%	2.64
4-conv-64-nodes-1-dense	23%	2.74
5-conv-64-nodes-1-dense	23%	2.73
1-conv-128-nodes-1-dense	19%	2.73
2-conv-128-nodes-1-dense	19%	2.71

3-conv-128-nodes-1-dense	14%	2.77
4-conv-128-nodes-1-dense	19%	2.77
5-conv-128-nodes-1-dense	19%	2.73
1-conv-32-nodes-2-dense	23%	2.68
2-conv-32-nodes-2-dense	4%	2.76
3-conv-32-nodes-2-dense	9%	2.83
4-conv-32-nodes-2-dense	19%	2.72
5-conv-32-nodes-2-dense	19%	2.76
1-conv-64-nodes-2-dense	19%	2.91
2-conv-64-nodes-2-dense	14%	2.99
3-conv-64-nodes-2-dense	19%	2.77
4-conv-64-nodes-2-dense	19%	2.74
5-conv-64-nodes-2-dense	28%	2.76
1-conv-128-nodes-2-dense	19%	2.76
2-conv-128-nodes-2-dense	28%	2.75
3-conv-128-nodes-2-dense	14%	2.71
4-conv-128-nodes-2-dense	19% (varied)	2.75
5-conv-128-nodes-2-dense	19% (varied)	2.73

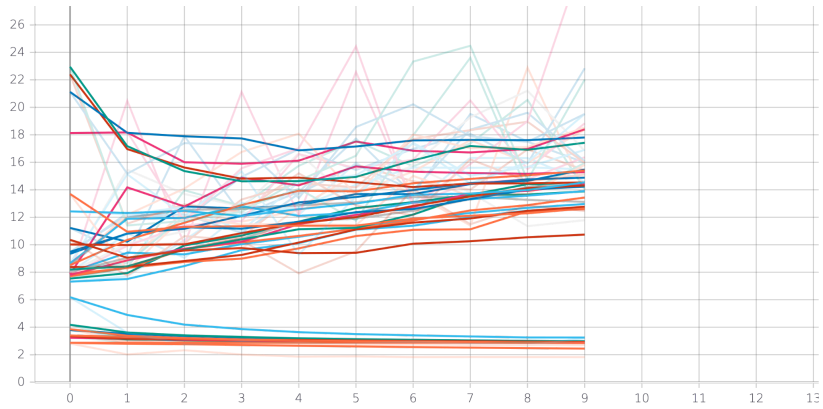
Similarly results for all the combinations with 1 2 3 dense layers, 1, 2, 3, 4, 5 convolutional layers of layer size 32, 64, 128 were near the accuracy range of 20 % to 50 percent with losses in the range of 2 to 4.

Among these,
3-conv-1-dense-64-nodes gave best result of 52% accuracy with 2.64

The trend for the accuracy and of various models and the trend followed is shown below



Sparse categorical Accuracy result



Loss

6. Generation of NIR images from RGB (using EPFL dataset)

The second part of the project includes generation of NIR images using RGB images using a autoencoder deep learning model.

1. Dataset

The dataset used here is the EPFL RGB-NIR dataset which contains 477 RGB-NIR pairs of images. The images were captured using separate exposures from modified SLR cameras, using visible and NIR filters. The cutoff between the two filters is approximately 750nm. The images were further processed by EPFL using equal weights for per band for NIR followed by averaging of the channels. The images were registered by extracting SIFT features at approximately 1500x2000 resolution

(50% scale) and using RANSAC to find a similarity transform. The final transformation was recomputed via least squares from the inliers and used to resample both images in a common coordinate frame.

Sample images from the dataset

2. Pre Processing

The processing includes two components. The first is the reading and resizing of the image in 1:1 ratio, in order to make training easier. The second is augmentation of the data in order to increase the dataset. This includes rotating, flipping and varying pixel brightness to create various variants of an image to increase the data for training.

Original RGB image

Original NIR image

3. Model

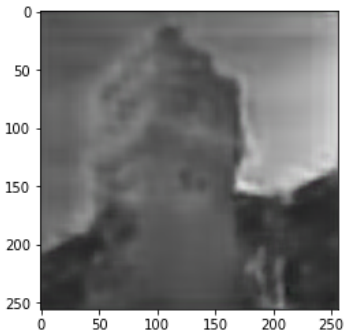
For this problem we used autoencoder neural networks to generate the NIR images from the RGB images. Our initial approach to this problem was to solve with domain adaptation technique where the image is first reduced to very less data and reconstructed to get a new image. This approach is useful in the problems where autoencoders are used to generate new data from some sample of existing data. However these models require a lot of training data and huge resources.
Since we did not have a large dataset to train the model and due the restraint on the computational power that we had access to, the images were not of good quality.
Hence we dropped this approach as the images generated by these networks was extremely poor and no useful property could be extracted from it

The Initial model used a convolutional layer of size 256-128-128-64 for the downsampling followed by 64-128-128-256 for upsampling. They were trained for 1000 or 100 epochs. Each convolutional layer in the downsampling part had a max pooling layer following it. The following image is the final image generated by this model.
Other models that gave better results were,

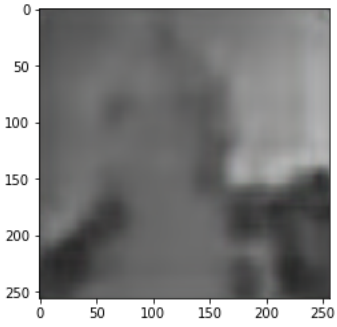
- 16_8_8 for 1000 epochs
- 64-128-128-256-256-256-128 in downsampling and reverse in upsampling for 1000 epochs
- 128-64-32-16 in downsampling and reverse in upsampling for 100 epochs

- 128-64-32-16 in downsampling and reverse in upsampling for 100 epochs
- 256-128-128 in downsampling and reverse in upsampling for 100 epochs
- 256-128-128 for 1000 epochs

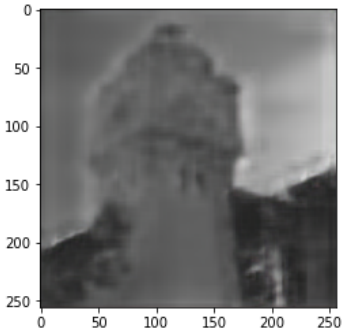
And many other models of the combination 256, 128, 64, 32 for 100 epochs which gave even poorer results. Note: Each input image was of the size 256 x 256
Some results with the domain adaptation approach where the pooling was done are:
(models are write in the form <Convolutional layer sizes> <epochs>)



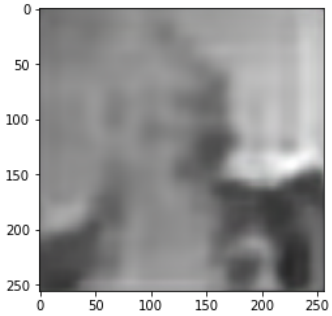
16-8-8 for 1000 epochs



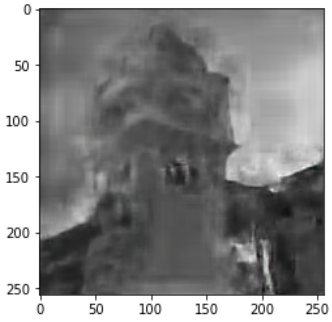
64-32-16-8 for 100 epochs



32-8-8 for 1000 epochs

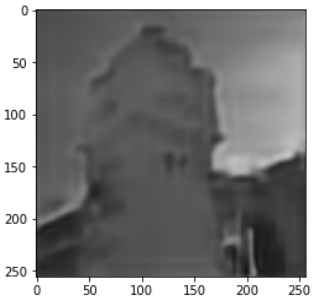


128-64-32-16 for 100 epochs

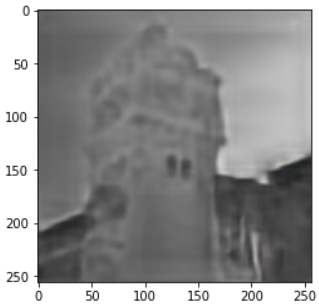


256-128-128-64 for 1000 epochs

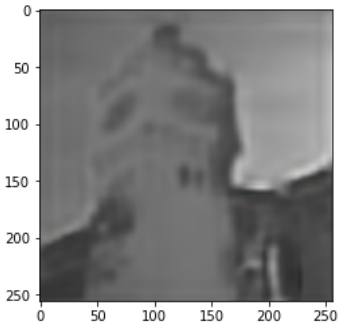
Thus we shifted to Image recolorization approach models. Here we did not use the pooling layer to reduce the data size and instead used different combinations of layer sizes from 256 to 64 or 32 size convolutional layers.
Some results with the Image colorization approach where the pooling was done are: (models are write in the form <Convolutional layer sizes> <epochs>)



64-128-128-256-256-256-128 for 100 epochs



512-256-256 for 100 epochs



256-128-128 for 100 epochs

Some other prominent models which gave good results were,

- 256-256-128-128-64 for 1000 epochs
- 128 64 64 32 for 100 epochs
- 128 128 64 64 64 32 for 1000 epochs

Finally the best results came with the model 128-128-128-64-64-64-64 size convolutional layers. The image shown in the following part was generated using the above model.

For all our models we used 'adam' optimizer and mean squared error as loss function. This approach also had issues with the shadow regions.

4. Inference

The processed data is fed through the convolutional layers. There are a total of 7 convolutional layers with no pooling layers, in upsampling and downsampling to prevent the loss of data. The activation function of the convolutional layers is the ReLU function. The kernel size of the convolution kernels is the same for all convolutional layers. After the model is trained, we get a result as follows which is further processed to give the final output.

Original RGB image

Image generated by Conv net

5. Post Processing

After the image is generated by the convolutional network. The image is passed through a joint bilateral filter in order to remove the haze around the edges. Joint bilateral filter is an edge preserving and image smoothing filter. The original output image shows high noise in the edges due to the sub correlation property of the convolutional layer which causes the edges to lose the sharpness. In order to compensate for this the joint bilateral filter is used. We use a gaussian distribution for the weights of the nearby regions.

Due to the shadowy regions where the RGB image and NIR image have significantly different brightness the final output also has a reduced contrast in the shadowy regions. This is adjusted by increasing the contrast of the images.

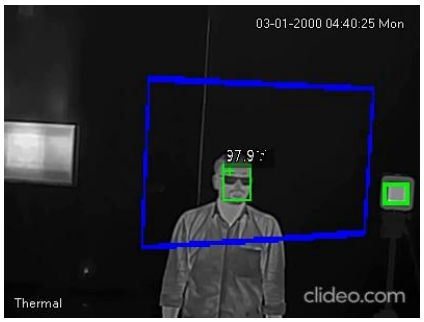
Filtered Image

Contrast adjusted Image

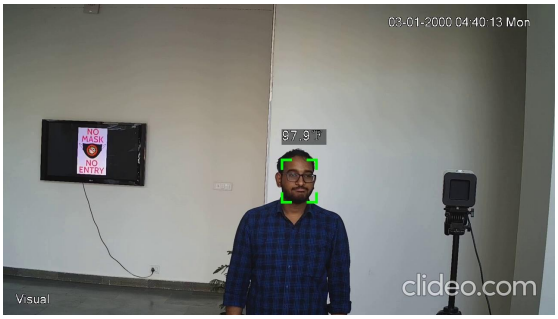
7. Using the Institute Dataset

After the collection of data by the institute was done and a significant number of images were present we used the institute dataset and reran the models with new data. The new data had temperatures in the 96 F 97 F 98 F range so training was also done in this range only.

A frame from the NIR video and RGb video along with the face detected in the RGB video is given below:



NIR Frame

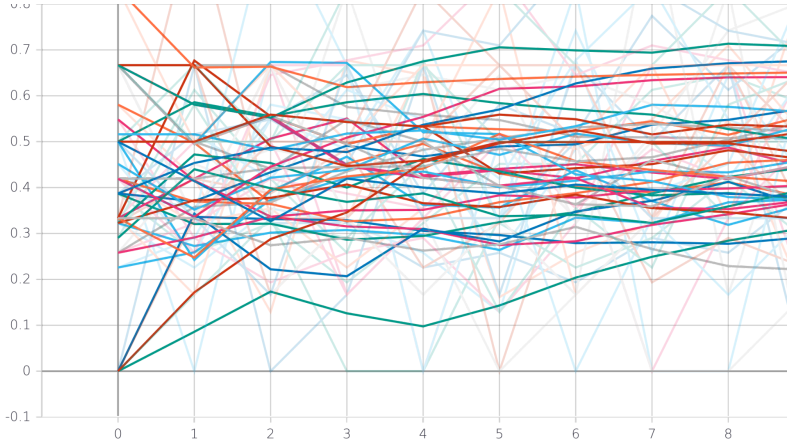


RGB Frame

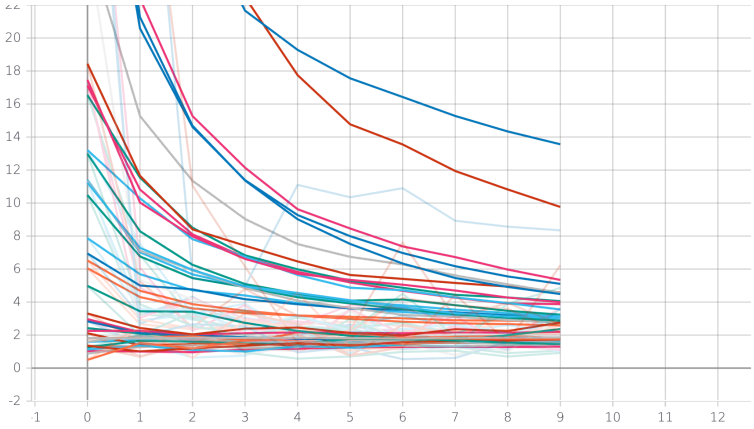
Detected face used in the below model to predict temperature

For the first approach we used the RGB videos and extracted frames from the videos. Then the faces were detected using openCV and cropped out. These images were then sorted according to the temperature captured by the IR thermal gun and similar to the first approach accuracy and loss was calculated for each of the models of CNN with 1 to 6 convolutional layers followed by 0 to 3 dense layers with layer sizes of 32, 64, 128 in all possible combinations. The data can be found in the following table.

The trend for the accuracy and of various models and the trend followed is shown below:



Sparse categorical Accuracy result



Loss

The training data is as follows:

Model	Accuracy	Loss
1-conv-32-nodes-0-dense	100%	0

2-conv-32-nodes-0-dense	100%	3.5380e-04
3-conv-32-nodes-0-dense	70%	1.50
4-conv-32-nodes-0-dense	83%	0.95
5-conv-32-nodes-0-dense	58%	1.73
1-conv-64-nodes-0-dense	100%	0
2-conv-64-nodes-0-dense	100%	0.05
3-conv-64-nodes-0-dense	64%	1.11
4-conv-64-nodes-0-dense	45%	1.49
5-conv-64-nodes-0-dense	38%	2.83
1-conv-128-nodes-0-dense	100%	0
2-conv-128-nodes-0-dense	19% (varied)	3.20
3-conv-128-nodes-0-dense	58%	2.06
4-conv-128-nodes-0-dense	22%	2.15
5-conv-128-nodes-0-dense	54%	3.60
1-conv-32-nodes-1-dense	83%	4.16
2-conv-32-nodes-1-dense	96%	0.15
3-conv-32-nodes-1-dense	48%	3.27
4-conv-32-nodes-1-dense	29% (varied)	2.59
5-conv-32-nodes-1-dense	16% (varied)	2.32
1-conv-64-nodes-1-dense	100%	9.1443e-06
2-conv-64-nodes-1-dense	100%	6.8134e-04
3-conv-64-nodes-1-dense	80%	1.16
4-conv-64-nodes-1-dense	74%	2.08
5-conv-64-nodes-1-dense	41% (varied)	2.13

1-conv-128-nodes-1-dense	96%	0.02
2-conv-128-nodes-1-dense	38% (varied)	1.60
3-conv-128-nodes-1-dense	58%	1.95
4-conv-128-nodes-1-dense	45%	2.09
5-conv-128-nodes-1-dense	16%	2.98
1-conv-32-nodes-2-dense	96%	0.46
2-conv-32-nodes-2-dense	96%	0.16
3-conv-32-nodes-2-dense	64%	1.67
4-conv-32-nodes-2-dense	51%	2.08
5-conv-32-nodes-2-dense	16% (varied)	2.36
1-conv-64-nodes-2-dense	100%	2.8879e-06
2-conv-64-nodes-2-dense	90%	0.41
3-conv-64-nodes-2-dense	38% (varied)	1.86
4-conv-64-nodes-2-dense	64%	1.66
5-conv-64-nodes-2-dense	48% (varied)	2.60
1-conv-128-nodes-2-dense	100%	6.1370e-04
2-conv-128-nodes-2-dense	6% (varied)	2.60
3-conv-128-nodes-2-dense	12%	3.09
4-conv-128-nodes-2-dense	41%	3.05
5-conv-128-nodes-2-dense	25%	2.62

For the generation of NIR images from RGB images we used the video data provided by the institute. The rgb and NIR videos were taken from different perspectives and the nir videos were of very low resolution. Moreover there were inbuilt features of the camera device used which made face detection in the NIR images very difficult and inaccurate. So for the

generation of Nir images from Rgb we used the whole frames of the videos. The RGb images were of different aspect ratio so it was adjusted to match the Nir frames. Then After resizing the images for both RGB and NIR a new dataset was created.

In order to Train the model the images were resized to 256 x 256 pixels. The nir frames consisted of some colored boxes and text due to the camera features so the NIR image was first adjusted to Grayscale and a 3 channel image was generated from it for training. Then we used various training models to get the NIR images. Some of the results are given below.

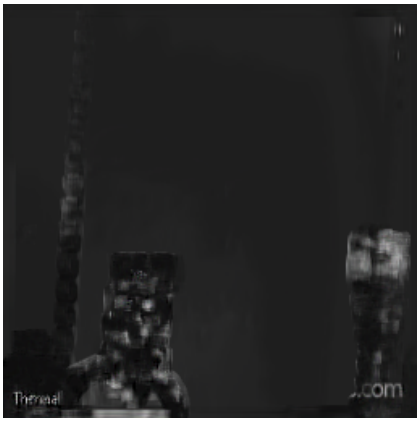
64-128-128-256-256-256-128-64 for 100 epochs

128-128-128-128-64-64-64 for 100 epochs

128-128-128-128-64-64-64 for 10 epochs

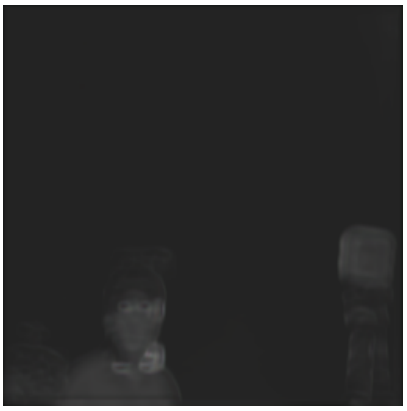
256-128-128-64-128-128 for 100 epochs

256-256-256-256-128-128-128 for 100 epochs



128-128-128-128-64-64-64 for 1000 epochs

256-256-256-256-256-256-256 for 100 epochs



256-128-128-64-128-128-256 for 100 epochs

Result

Due to the very small number of videos present in the rgb dataset the rgb image to temperature mapping gave results that were highly inaccurate and misleading. Some of the models gave results which were 100% accurate from the start and had zero loss throughout the training.

In the RGB to NIR recolorization part the videos taken by the NIR camera were distorted due to the camera features and had a different perspective. Moreover the images were in very low resolution due to which face detection could not be done properly. The temperature data by the NIR camera was also present at the forehead region which made face detection very difficult. Hence we couldn't separate out the face only for recolorization and had to use the full frame. Moreover since the dataset was very small the training done is still inaccurate. Due to the videos being

of different perspectives and the built in features of the camera covering the face region of the image the results were poor.

8. Conclusion and Future work

The prediction of the temperature in the first part is accurate more than 50 percent of the time, however this accuracy may be misleading due to the small dataset that we have used. In the first experiment the results obtained by the convolutional network are very similar also due to the insufficient dataset present, Also this leads to very high loss function values.
Similarly the dataset provided by the institute was very small for proper prediction of the temperature and mostly consisted of 97F. Due to which the training was bad and as a result the accuracy was misleading.

This experiment should be repeated with a larger dataset The second experiment where we generated NIR images from RGB images gave sufficiently accurate results. However these results were taken of sceneries and the results may vary for faces. Though it was unable to construct the regions with sudden changes and sharp boundaries. This can be corrected by training the model for a longer time. Although the results generated were good due to the unavailability of any NIR images for the human face and subsequent temperatures.
The dataset provided by the institute is very small currently for a proper result. When the dataset being collected by the institute is sufficiently large the above results can be applied to those datasets albeit with some changes and proper NIR images of the face can be obtained. The resulting images from this small dataset are very poor.
After the NIR images are generated the NIR images can then be used to find an indirect mapping of image to core body temperature.
We also worked with hyperspectral images to generate results of images in particular channels belonging to the wavelength range 800 nm to 1200 nm. We approached the hyperspectral images in a similar manner as we approached the NIR images. We separated each channel of the hyperspectral image, however due to the extremely large image sizes and due to unavailability of data it was dropped. This part can also be picked up in future to generate images of specific wavelength ranges.

9. References

1. Infrared Colorization Using Deep Convolutional Neural Networks https://arxiv.org/pdf/1604.02245.pdf

2. Autoencoder for colorization
    a. https://github.com/bnsreenu/python_for_microscopists/blob/master/090a-autoencoder_colorize_V0.2.py
    b. https://xiangyutang2.github.io/auto-colorization-autoencoders/#7.-Coloring-Images
3. EPFL Dataset https://ivrlwww.epfl.ch/supplementary_material/cvpr11/index.html
4. ICVL Dataset http://icvl.cs.bgu.ac.il/hyperspectral/
5. FLIR Dataset https://app.box.com/s/1z1wn8e5b9ff0vvcc589l1m5529smrnx
6. https://www.spiedigitallibrary.org/conference-proceedings-of-spie/11399/2557856/Mobile-application-for-monitoring-body-temperature-from-facial-images-using/10.1117/12.2557856.full?SSO=1
7. RGB to hyperspectral
    a. https://www.sfu.ca/~nsa84/projects/mmsys20.pdf
    b. https://openaccess.thecvf.com/content_cvpr_2018_workshops/papers/w13/Shi_HSCNN_Advanced_CNN-Based_CVPR_2018_paper.pdf
    c. https://github.com/KAIST-VCLAB/deepcassi
8. https://arxiv.org/abs/2102.12627
9. https://openaccess.thecvf.com/content_cvpr_workshops_2014/W19/papers/Yang_Feature_Regression_for_2014_CVPR_paper.pdf
10. https://www.sciencedirect.com/science/article/abs/pii/S0165168412000345
11. https://paperswithcode.com/newsletter/3
12. https://people.csail.mit.edu/mrub/evm/
13. https://github.com/RichardEvans/apperception
14. https://openai.com/blog/dall-e/
15. https://osf.io/fdrbh/wiki/home/
16. https://iopscience.iop.org/article/10.1088/0967-3334/35/5/807/pdf
17. Learning more with less data https://blog.floydhub.com/n-shot-learning/
18. Stack Overflow for doubts https://stackoverflow.com/