

UE18CS390B - Capstone Project Phase - 2

SEMESTER - VII

END SEMESTER ASSESSMENT

Project Title : DeepFake Detection Using Machine Learning

Project ID : PW22AV01

Project Guide : Prof. A. Vinay

Project Team : VISHRUTH L PES1201800362

VIVEK N PES1201801077

NIPUN BHAT PES1201801857

PARAS S KHURANA PES1201801158

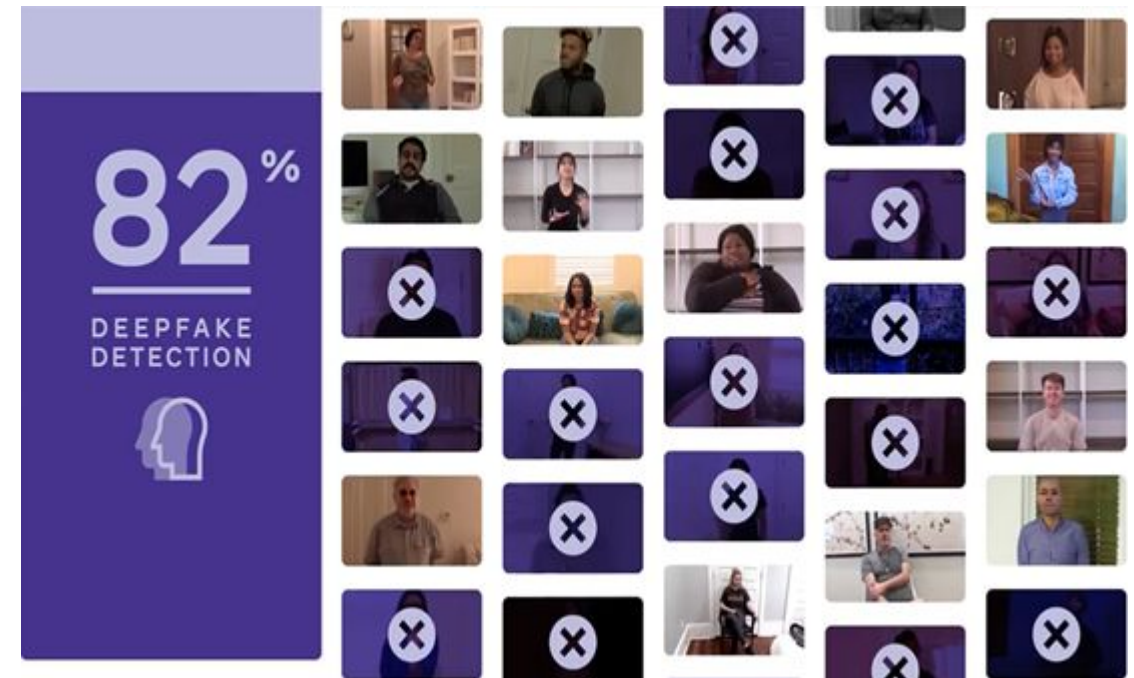
Abstract

- DeepFakes are synthetic media in which a person in an existing image or video is replaced with someone else's likeness.
- DeepFakes have been used to misrepresent well known politicians in videos.
- DeepFakes can be used to generate blackmail materials that falsely incriminate a victim.



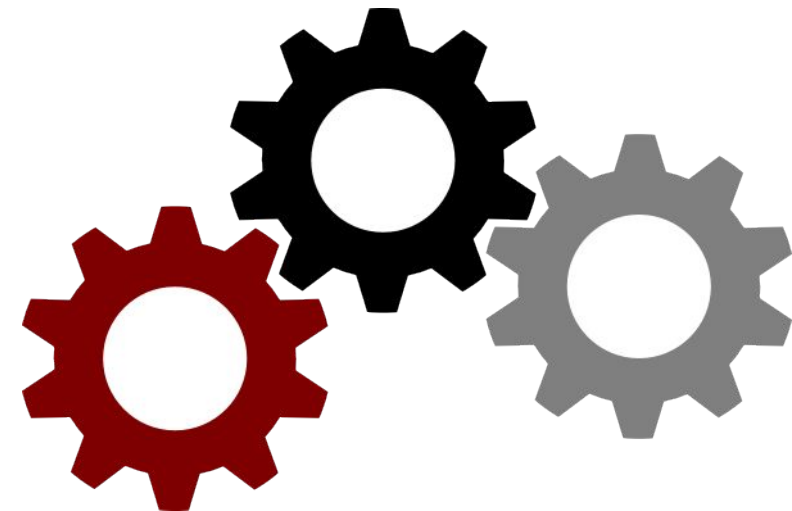
Motivation & Scope

- Misinterpretation of forged facial media in today's age of social networking
- Deep Fakes are ranked as most serious AI crime threats (University College London)
- The recent DeepFake detection challenges by Kaggle and Facebook



State of Research

- Deep Fake detection is a zero sum game in which both the detector and generator algorithms improve constantly.
- We have done an extensive literature survey and have come across various methods in detection of Deep Fakes.
- Most of them have a CNN and LSTM approach. This is not susceptible to the zero sum game.



State of Research

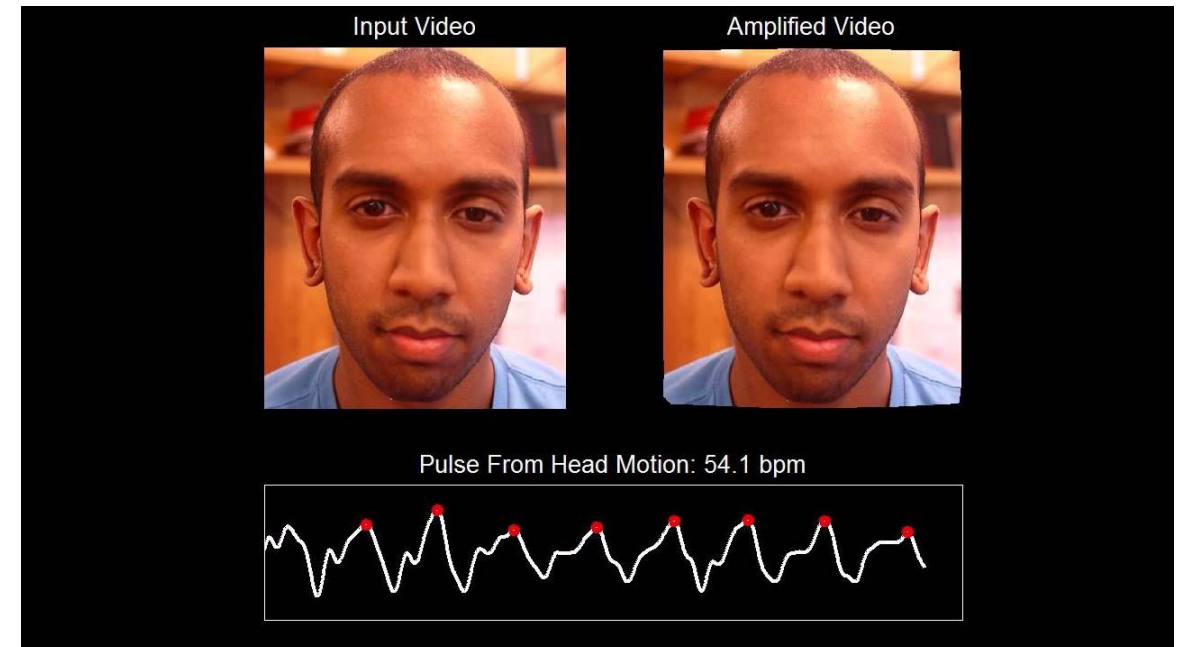


- Hence we have chosen to do further research using this approach. Biological signals (heart rate analysis) is listed as one of the best methods to detect Deep Fakes.
- When a video is deepfaked, the natural heart rate gets disrupted and thus can be used for detection of deepfaked video.



Summary of Methodology

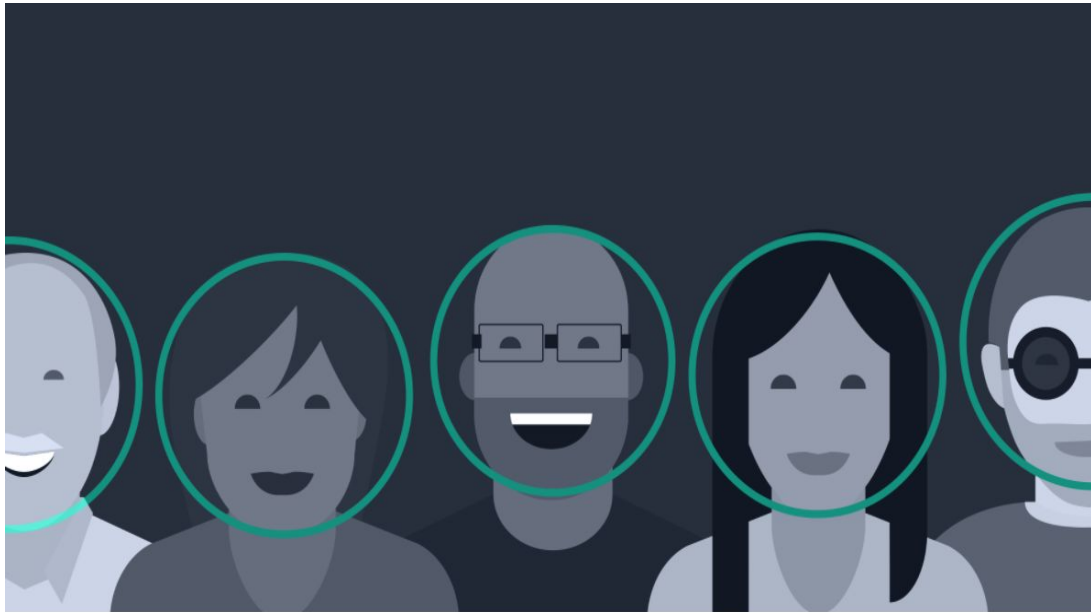
- DeepFake detection is a zero sum game.
- We are using the heart-rate analysis method for the detection of deep-fakes.



Summary of Methodology

Intuition: When deepfake is used on a video, the natural heart rate gets disrupted and thus can be used for detection of deepfaked video.

1. Detection, resizing and alignment of faces and reconstruction of face video.
2. Generating Motion Magnified Video from Face Video.



Summary of Methodology



3. Generate Motion Magnified Spatial Temporal Representation (X).
 - a. Divide the face areas of all frames into N non-overlapping ROI blocks.
 - b. Perform average pooling on each block and each color channel for each frame.
 - c. Each row of MMSTR Map represents the motion-magnified temporal variation of one block.

Note: The detailed explanation of the implementation are presented in the following slides

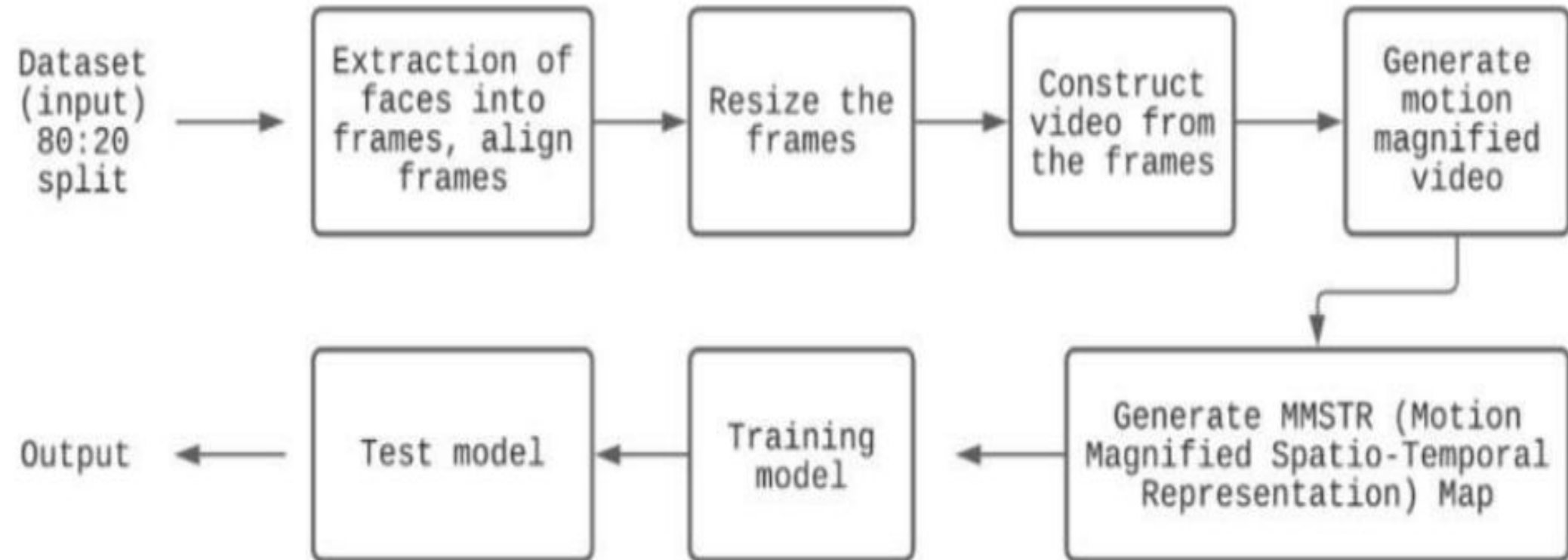
Summary of Methodology



- 4) Using the MMST Map we generate s and t vectors, which assign spatial and temporal weights using pre-trained MesoNet and LSTM.
- 5) The product of the s and t vectors give us weight matrix (W).
- 6) The weights (W) are multiplied element wise with the MMST map (X) as $A = X * W^T$
- 7) The weighted MMST map is passed on to the detection stage.

Note: The detailed explanation of the implementation are presented in the following slides

Design Description



DataSet

We are using the Celeb-DF DataSet for our project.

- Celeb-DF includes **590 original** videos collected from YouTube
- Subjects of different ages, ethic groups and genders
- **5639** corresponding DeepFake videos.



DataSet



Samples From Celeb-DF.

DataSet

UADFV



- Low resolution
- Visible facial parts of original face

DeepFake-TIMIT



- Visible artifacts in face boundary

FF++ / DF



- Color inconsistency
- Visible facial parts of original face

Celeb-DF

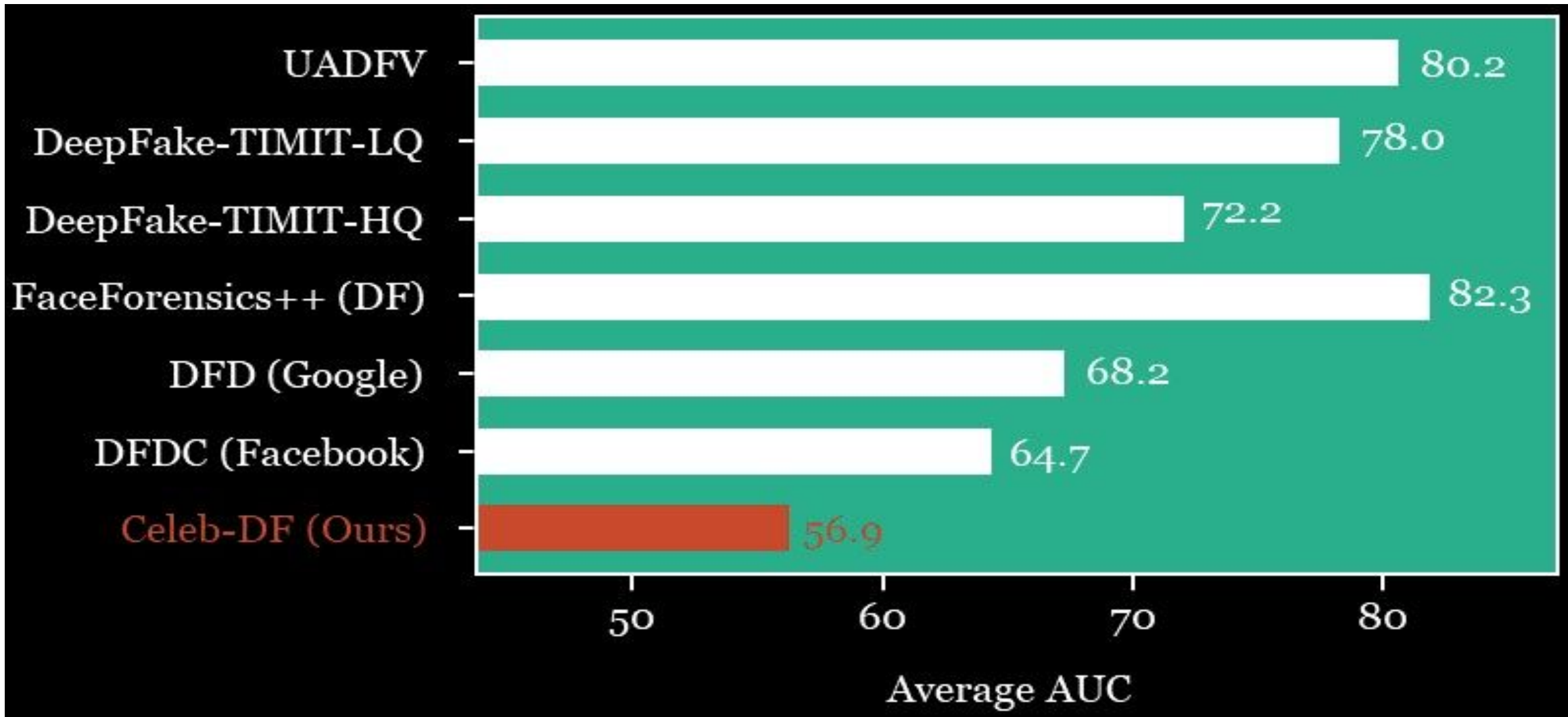


DeepFake Videos From Internet



Comparison between Celeb-DF and other datasets

DataSet



Comparison between Celeb-DF and other datasets

Data Collection:

The Celeb-DF Dataset was acquired from the following link:

<https://drive.google.com/file/d/1iLx76wsbi9itnkxSqz9BVB14ZvnbIazj/view>

Download access was granted after filling a google form:

<https://docs.google.com/forms/d/e/1FAIpQLScoXint8ndZXyJi2Rcy4MvDHkkZLyBFKN431TeyiG88wrG0rA/viewform>

Note: Upon downloading the dataset, it was already divided into DeepFake and Original videos into separate folders.

Project Demonstration : Step 1



Data Pre-processing:

After downloading the dataset, the data is preprocessed before training. The data has gone through the following stages for preprocessing

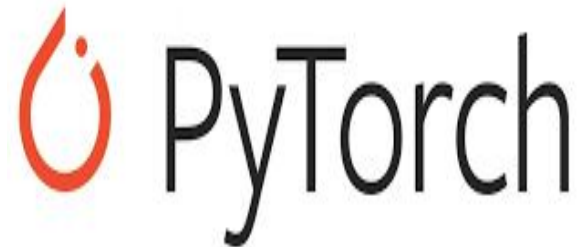
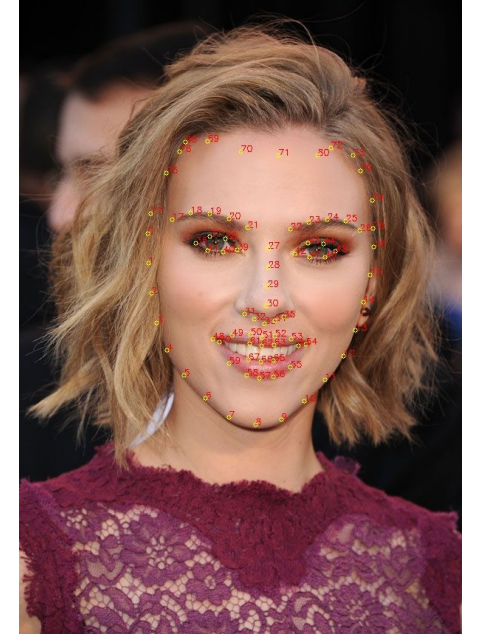
1) Extraction of faces from videos:

In this stage the faces from each frame have been extracted along with the facial landmarks and are stored

Implementation Details : Step 1

Step 1: Aligning face with MTCNN

Technology Used: pytorch, MTCNN (facenet_pytorch) , numpy, cv2, dlib



Project Demonstration : Results from Step 1

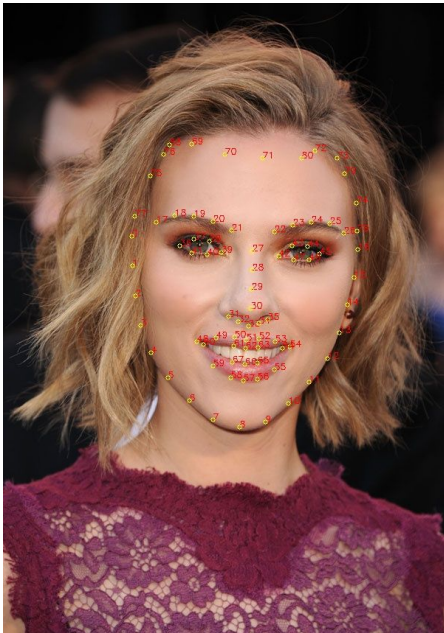


Implementation Details : Step 1 Algorithm

Using the dlib detector and predictor and MTCNN library, we obtain the facial structure.

Using the 81 facial landmarks predictor, we isolate the face

The angle of the isolated face is adjusted so that faces are evenly aligned across all the frames



Finally, we store the final preprocessed image along with the facial landmarks.

We then manipulate the facial structure by blackening the eye regions and inner region of the mouth.

Implementation Details : Step 2

2) Resizing of frames:

In this stage the isolated faces are resized to equal dimensions and saved

Technology Used: numpy, cv2



Project Demonstration : Results from Step 2



Implementation Details : Step 2 Algorithm



The output from Step I is the input in this step. These files are all of different dimensions. The aim of this step is to make all the frames of equal dimensions

The original dimensions of each jpg file representing a frame are resized to the dimensions $300 * 300$ using `cv2.resize`.

The resized frames are now saved as jpg files. (Each jpg image from a given video are stored in sequential order in a given folder representative of that video)

Project Demonstration : Step 3



3) Alignment and creation of face video:

In this stage, the resized faces are combined to generate a video of the isolated faces only

Technology Used: numpy, cv2, ffmpeg



Project Demonstration : Results from Step 3



Implementation Details : Step 3



We iterate through the folders containing frames of the aligned faces.

Using the ffmpeg tool, we now combine these resized and aligned frames to generate a face video

These face videos are now stored

Project Demonstration : Step 4

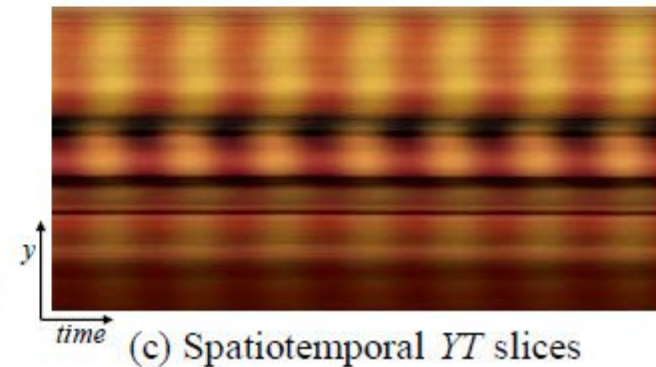
Motion Magnified Video from Face Video.



(a) Input



(b) Magnified



(c) Spatiotemporal YT slices

Credits : <https://www.youtube.com/watch?v=fHfhorJnAEI&t=135s>

Project Demonstration : Step 4



Source

(Courtesy of Winchester Hospital. Do not copy)



EKG

Hospital monitor

Bandpass signal +
peaks (pulse)

Estimated heart rate



156 bpm

35.2 =
Age - 19.2



Color-amplified (x150)

With Dr. Donna Brezinski and
the Winchester Hospital staff

clideo.com

Credits : <https://www.youtube.com/watch?v=fHfhorJnAEI&t=135s>

Project Demonstration : Step 4



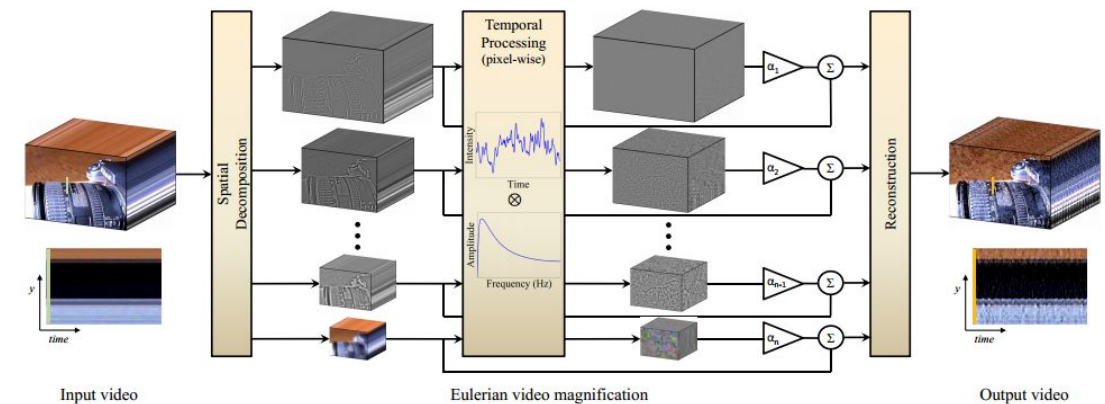
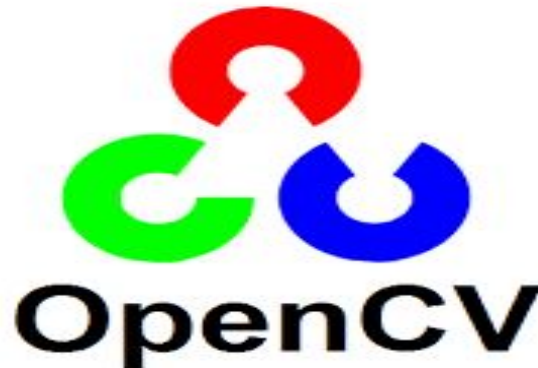
4) Generating Motion Magnified Video from Face Video:

In this stage, the face video from the previous stage is used to generate a Motion Magnified Video

Technology Used: numpy, cv2, python_eulerian_video_magnification (PyEVM)



NumPy



Project Demonstration : Results from Step 4

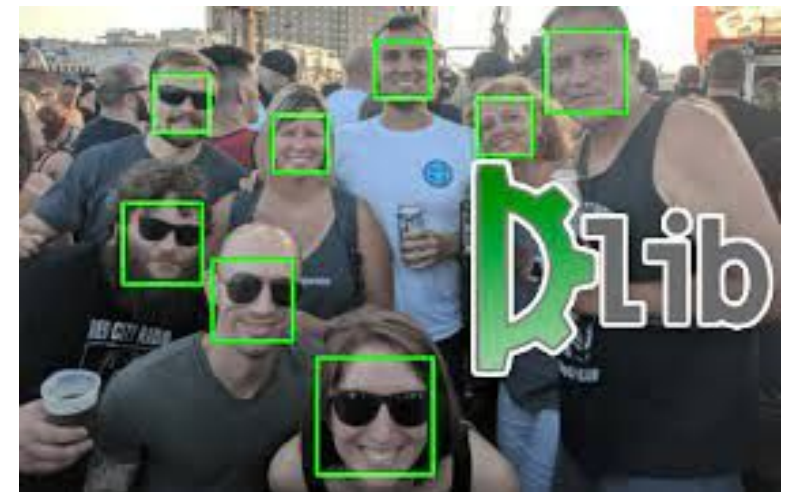


Project Demonstration : Step 5

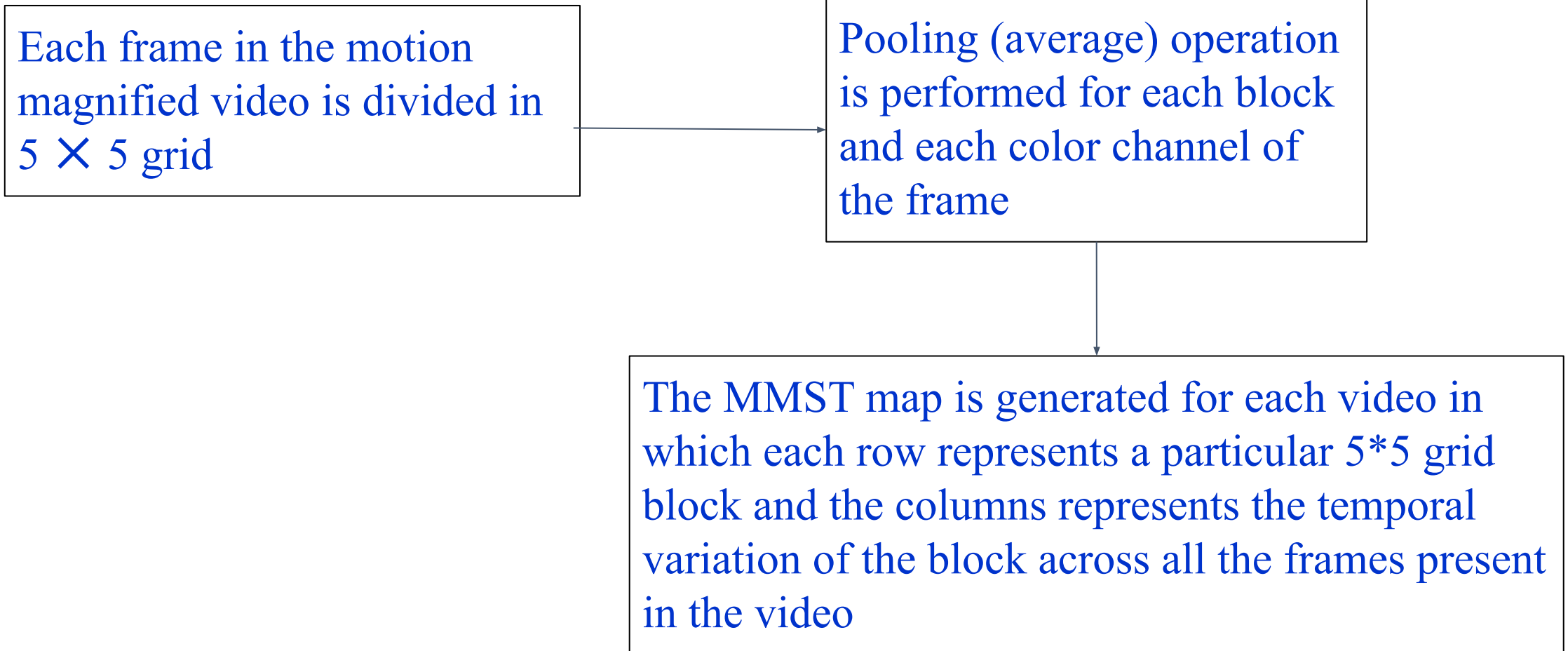
5) Generate MMST Map:

The Motion Magnified Video from the previous stage is used to generate the MMST Map for each video. (Stored as a numpy array)

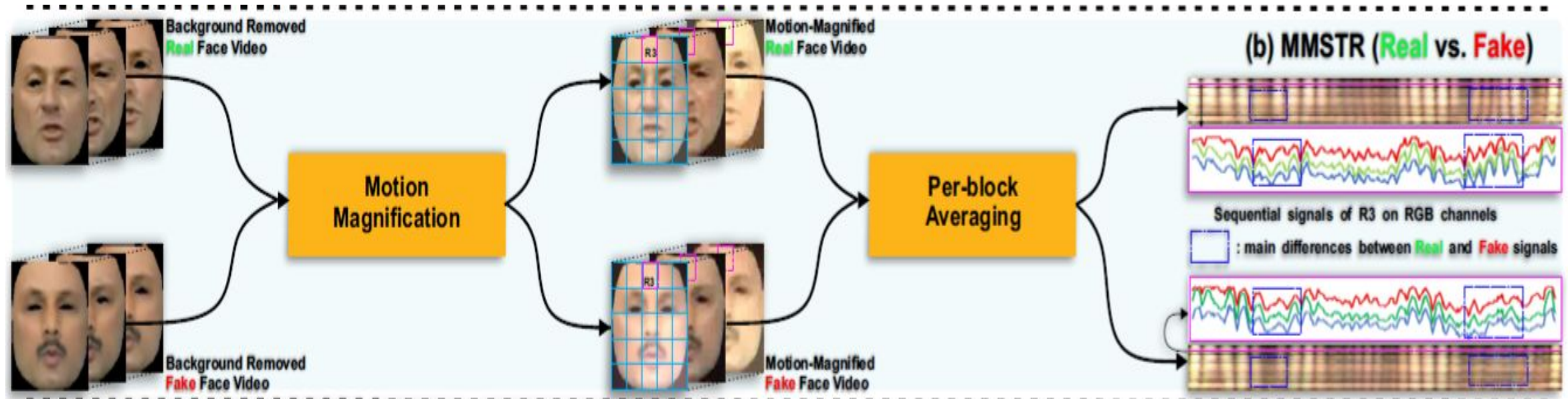
Technology Used: numpy, cv2, dlib



Implementation Details : Step 5



Project Demonstration : Step 5

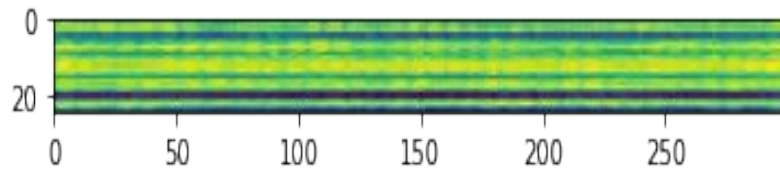


Credits : **Deep Rhythm: Exposing Deep Fakes with Attentional Visual Heartbeat Rhythms**

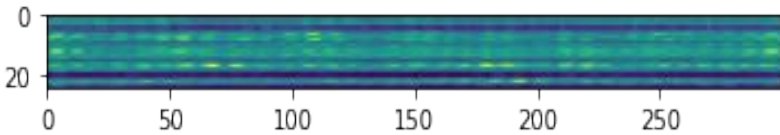
Project Demonstration : Results from Step 5

Real

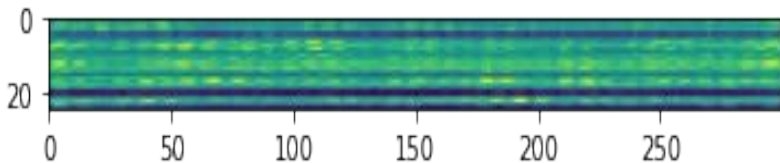
**Blue
Channel**



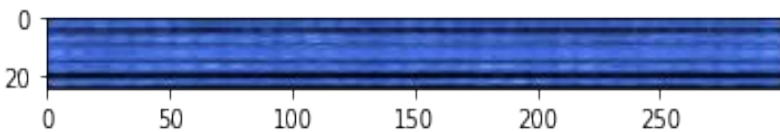
**Green
Channel**



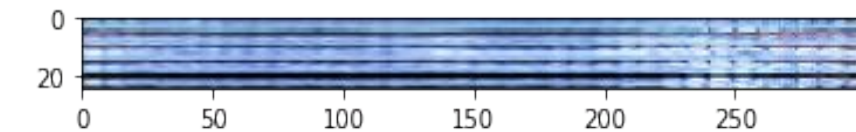
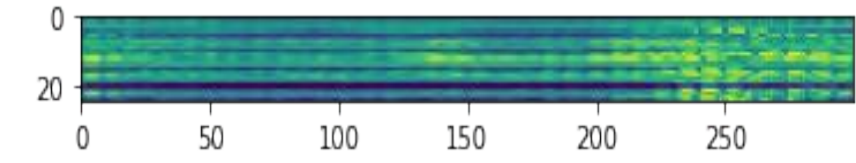
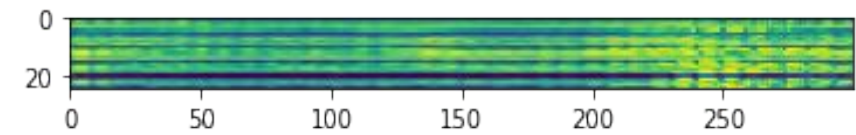
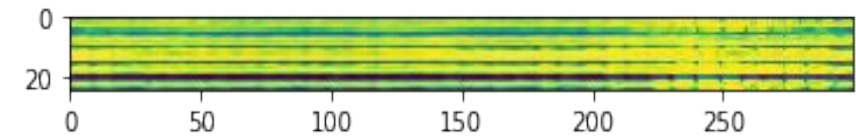
**Red
Channel**



Combined

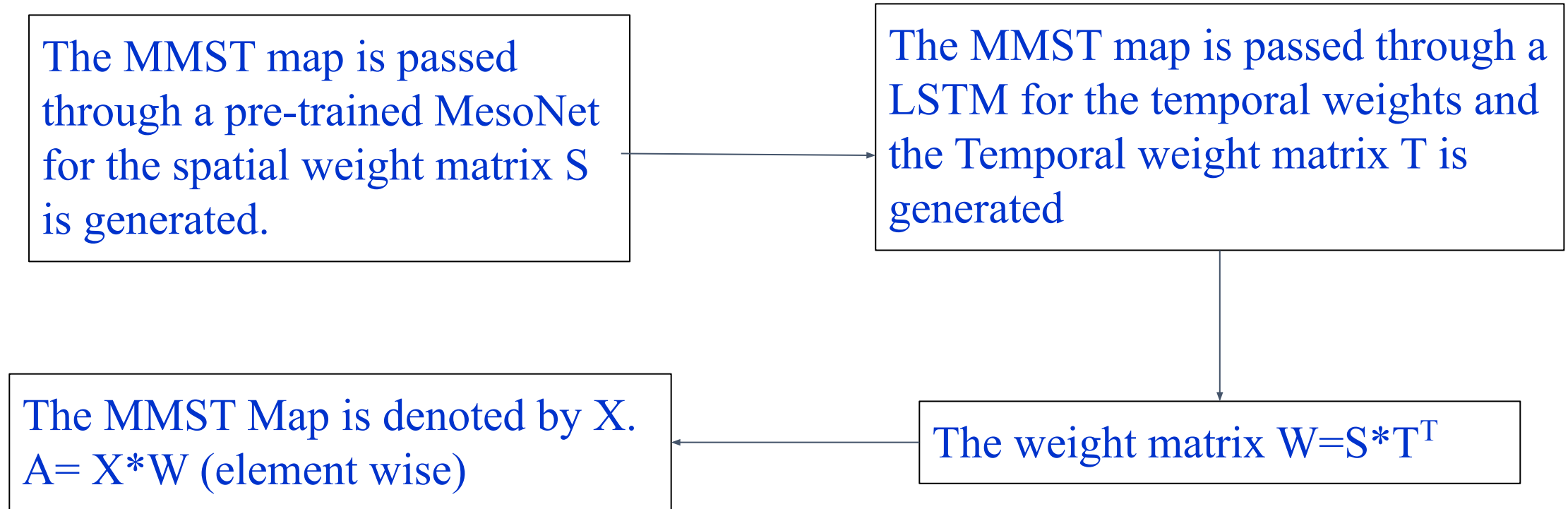


Synthesized

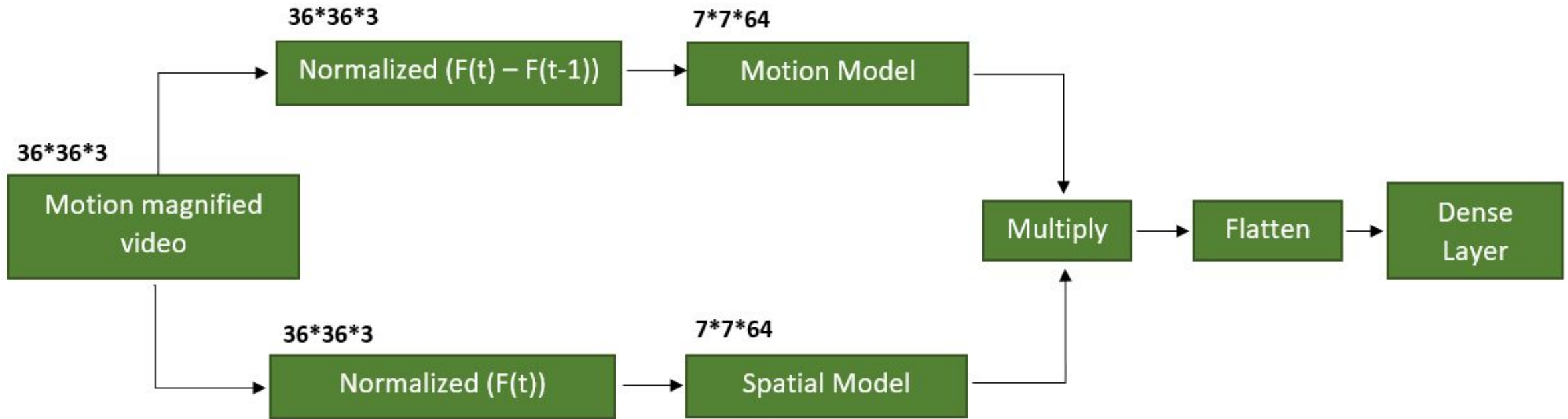


Note: MMST Map for each motion magnified video is saved as a numpy file

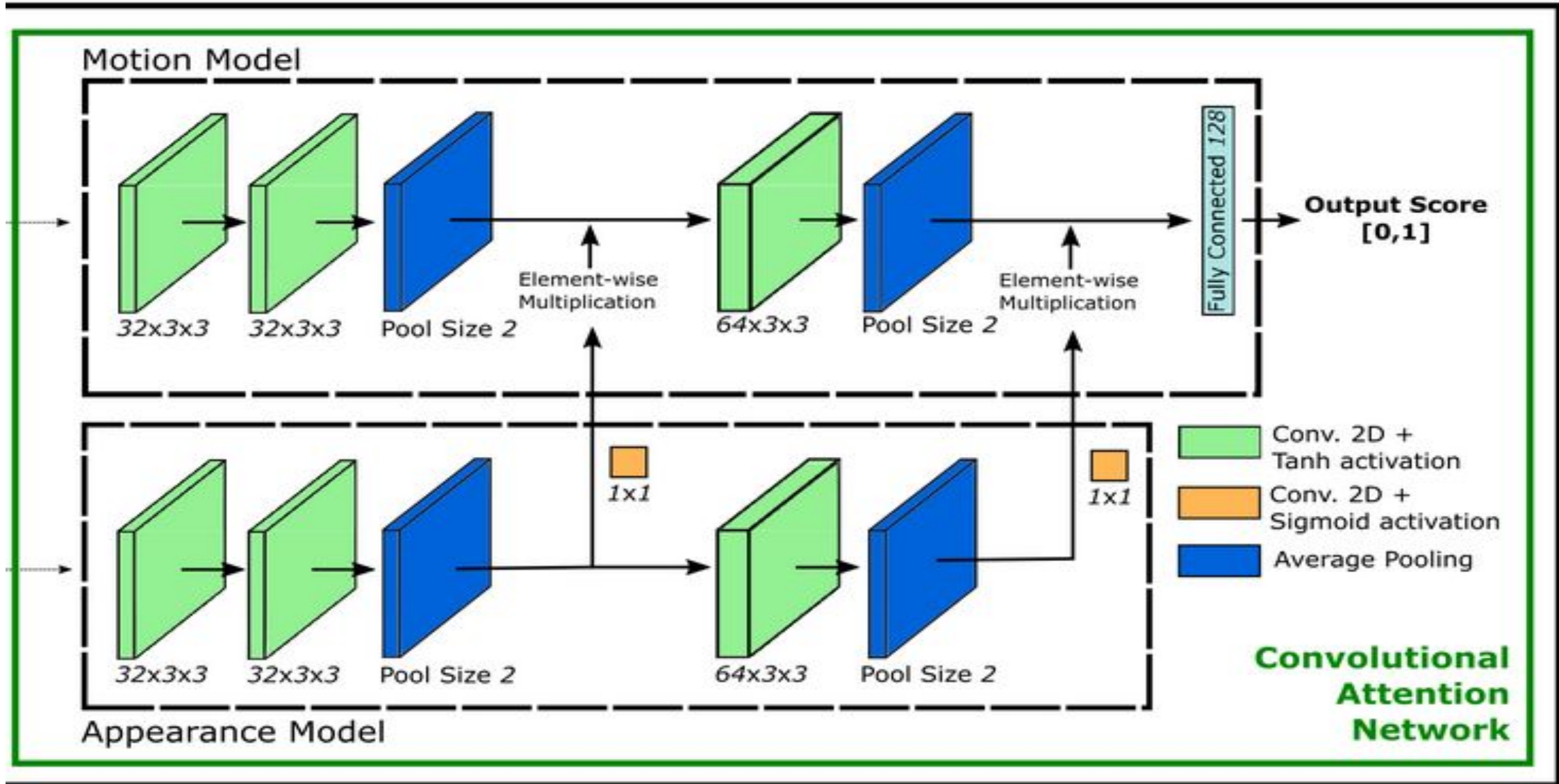
Implementation Details : Algorithm



Architecture Design 1



Architecture Design 1



Architecture Design 1

Normalised Frames:

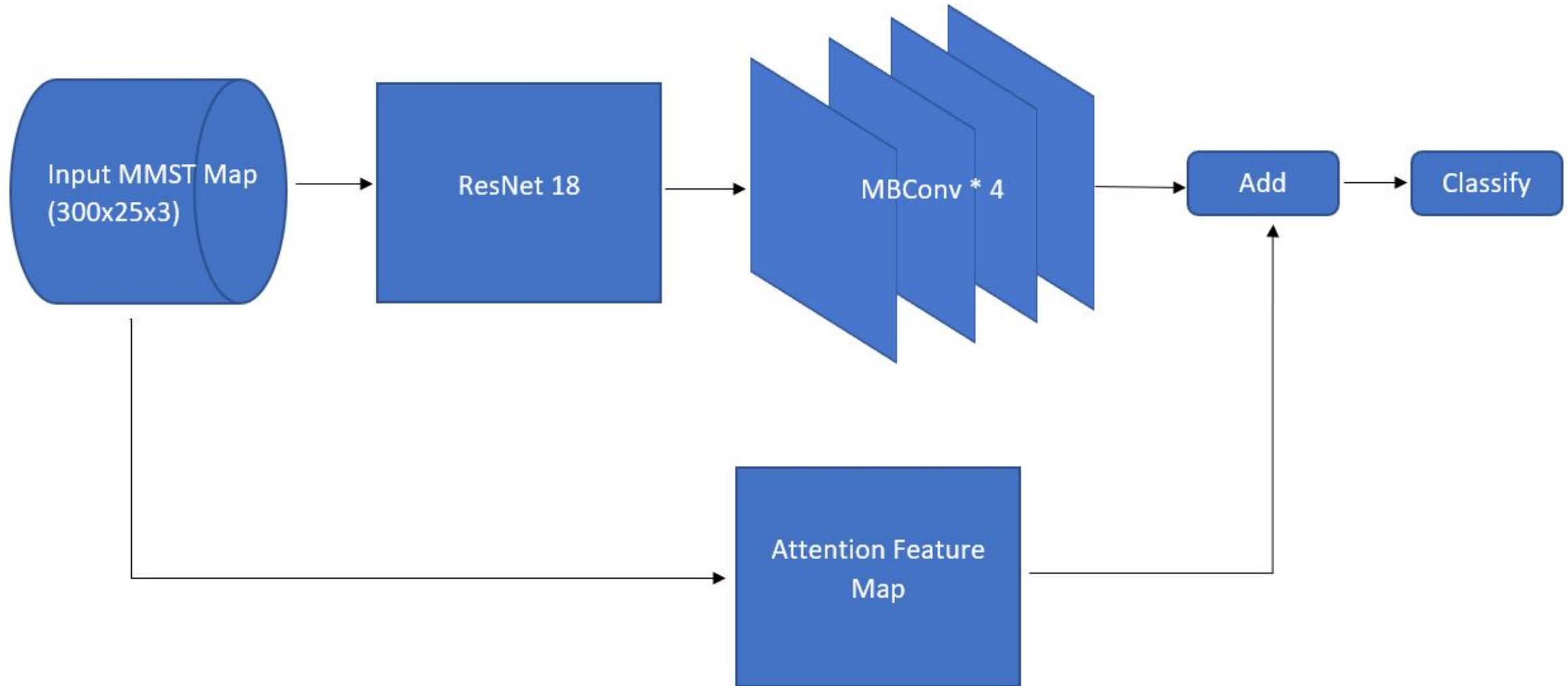


Normalised($f(t)-f(t-1)$)

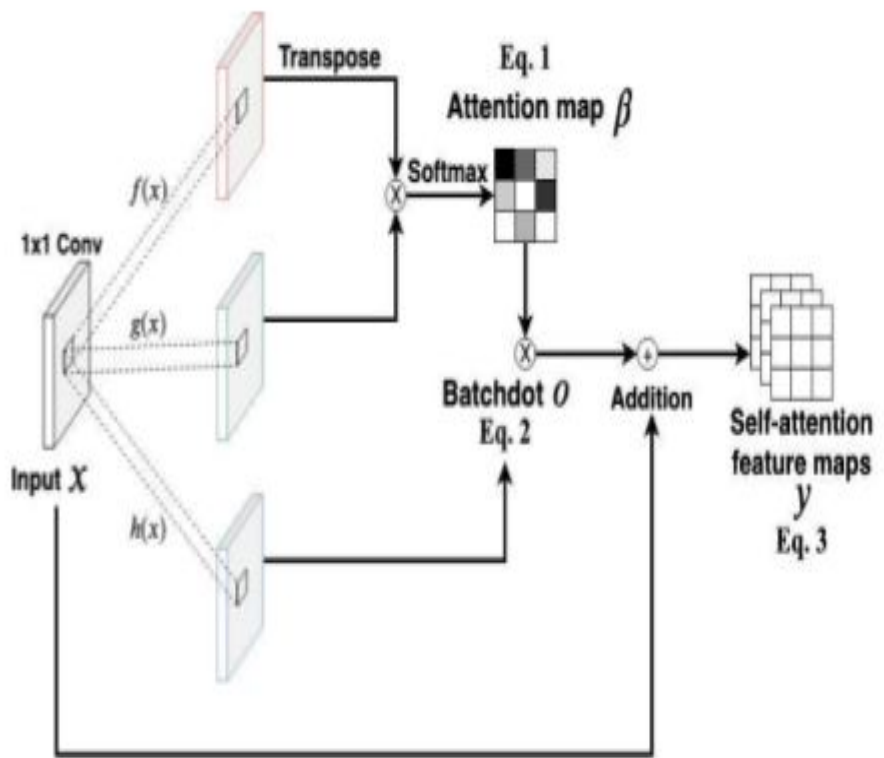


Normalised($f(t)$)

Architecture Design 2



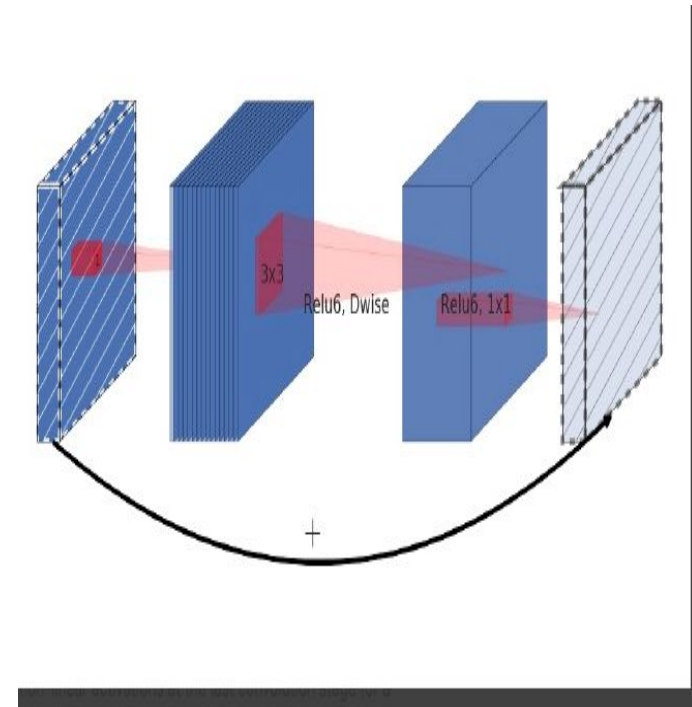
Architecture Design 2



ATTENTION FEATURE MAP

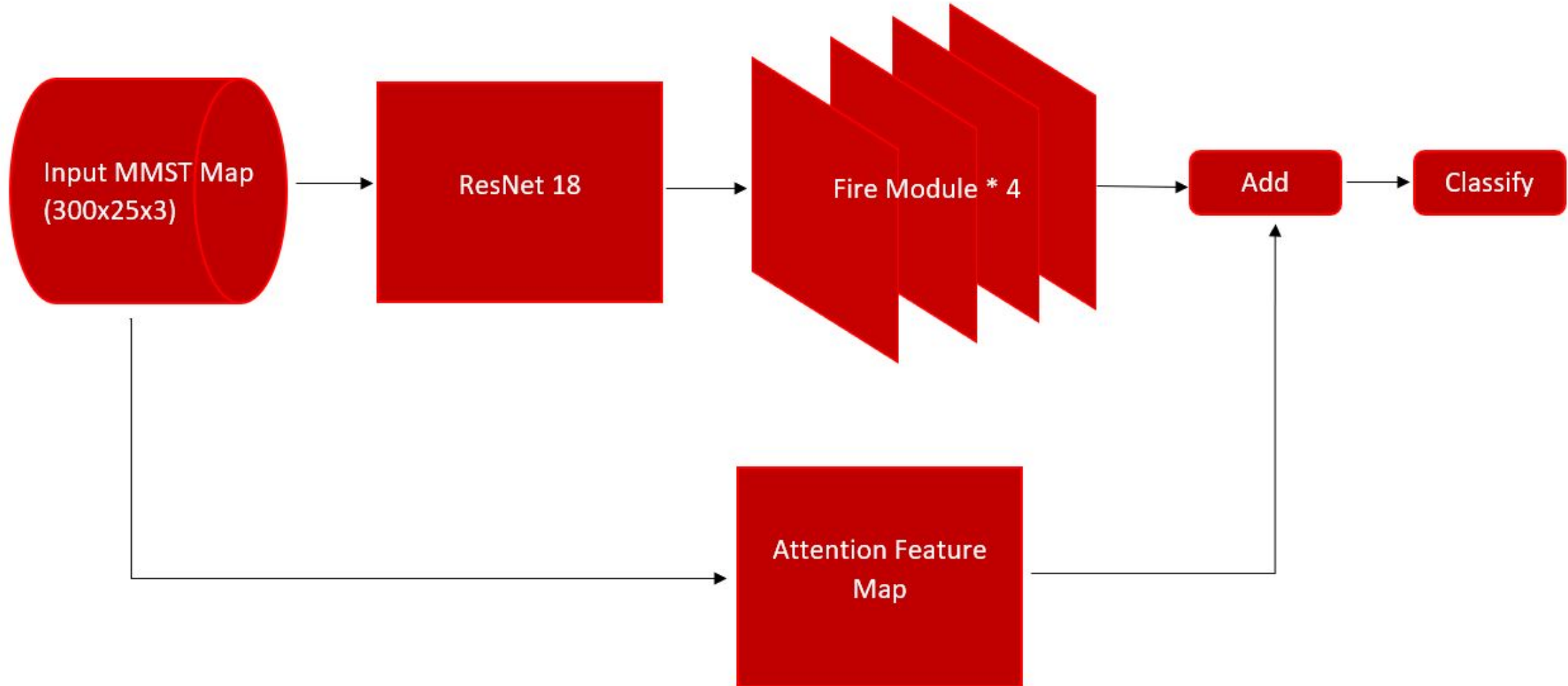
Input size	Operation	Num. Parameters	Output dim.	Stride
$64 \times 64 \times 3$	3×3 DConv	123	32	2
$32 \times 32 \times 32$	BN	128	32	-
$32 \times 32 \times 32$	ReLU	0	32	-
$32 \times 32 \times 32$	1st Stage self-attention	1,321	32	-
$32 \times 32 \times 32$	3×3 DConv	2,336	64	2
$16 \times 16 \times 64$	BN	256	64	-
$16 \times 16 \times 64$	ReLU	0	64	-
$16 \times 16 \times 64$	2nd Stage self-attention	5,201	64	-
$16 \times 16 \times 64$	3×3 DConv	8,768	128	2
$8 \times 8 \times 128$	BN	512	128	-
$8 \times 8 \times 128$	ReLU	0	128	-
$8 \times 8 \times 128$	3rd Stage self-attention	20,641	128	-
$8 \times 8 \times 128$	1×1 Conv	73,728	576	1
$8 \times 8 \times 576$	BN	2,304	576	-
$8 \times 8 \times 576$	ReLU	0	576	-
$8 \times 8 \times 576$	GAP	0	576	-

Attention Map Operation

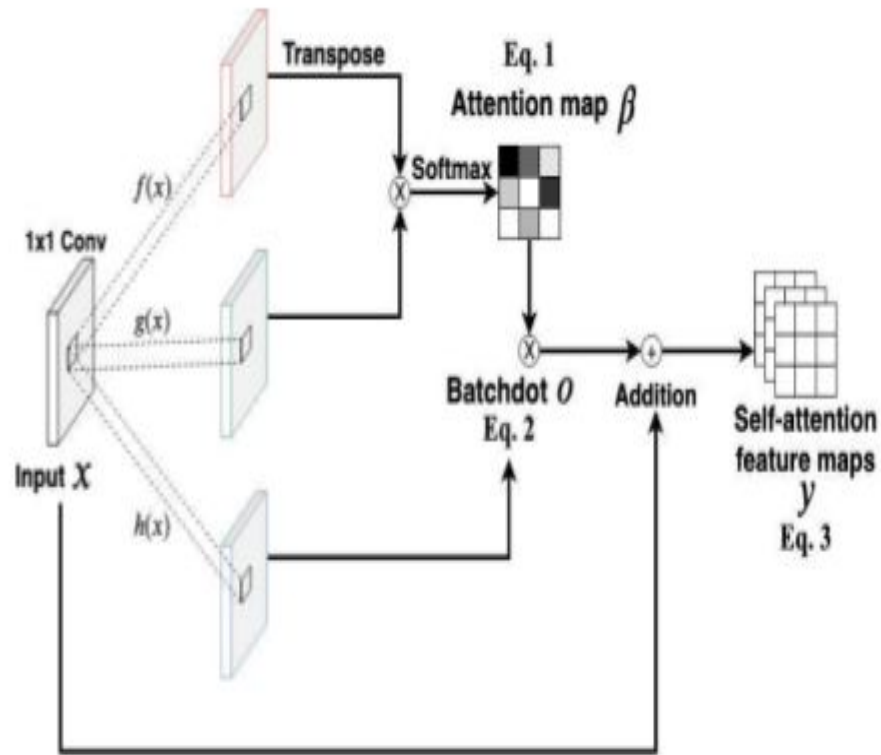


MB CONV MODULE

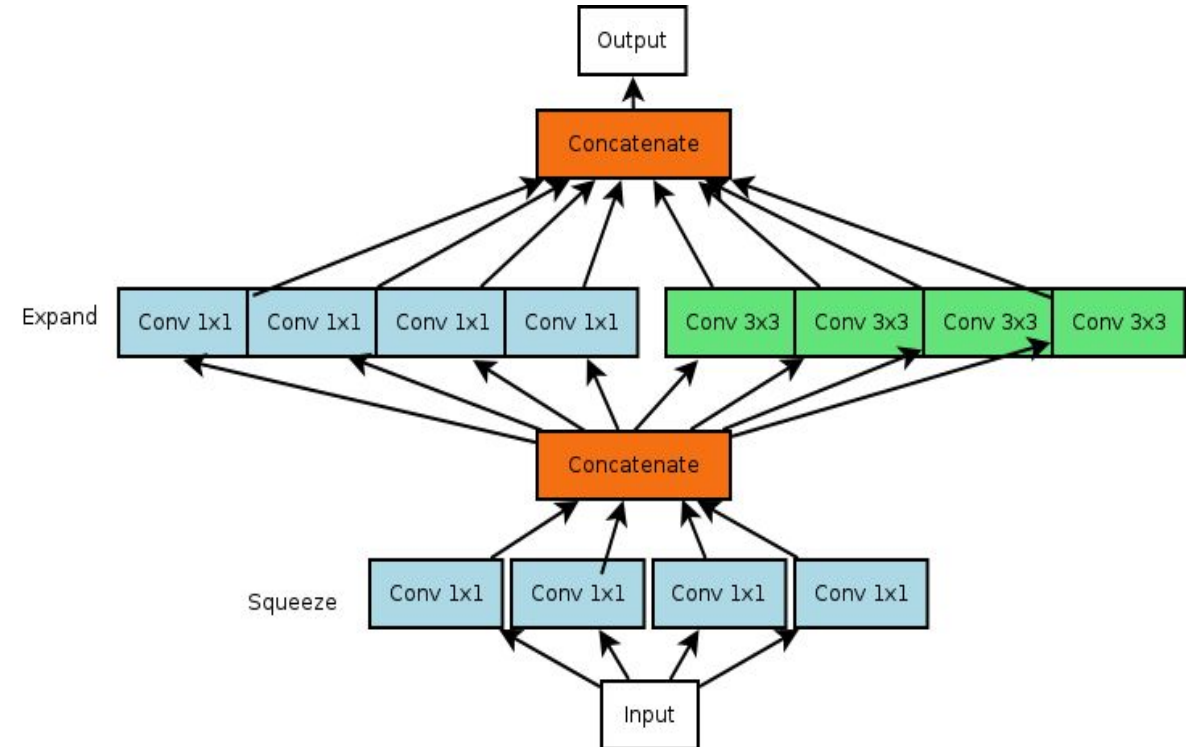
Architecture Design 3



Architecture Design 3

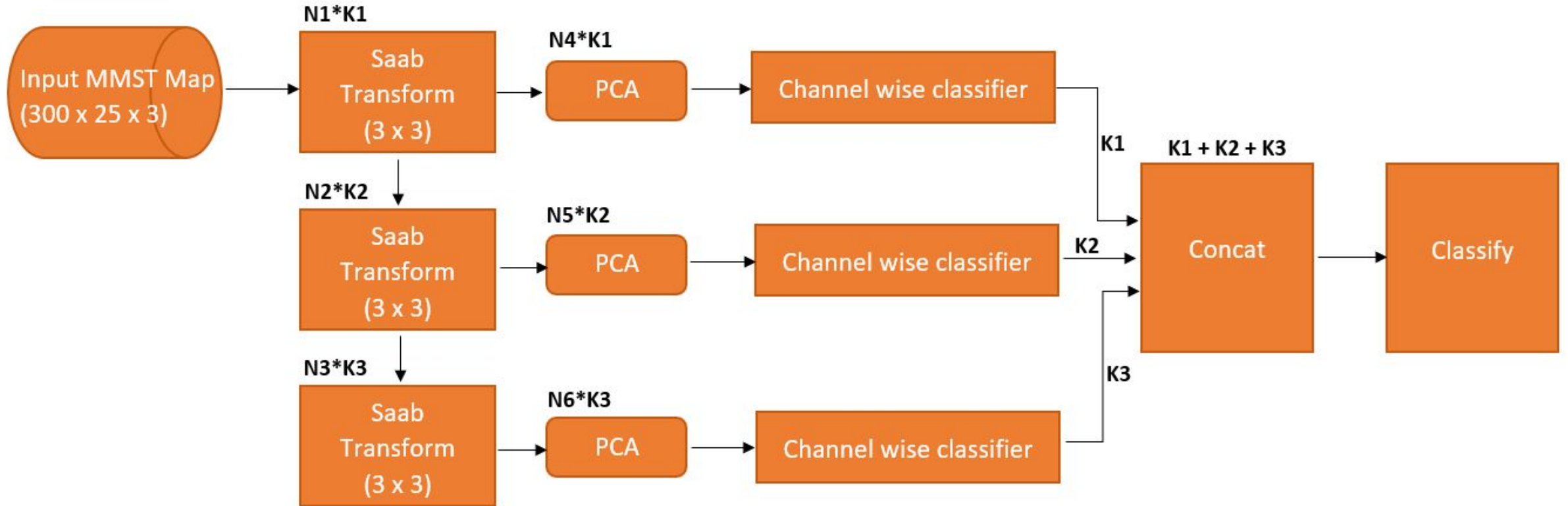


ATTENTION FEATURE MAP



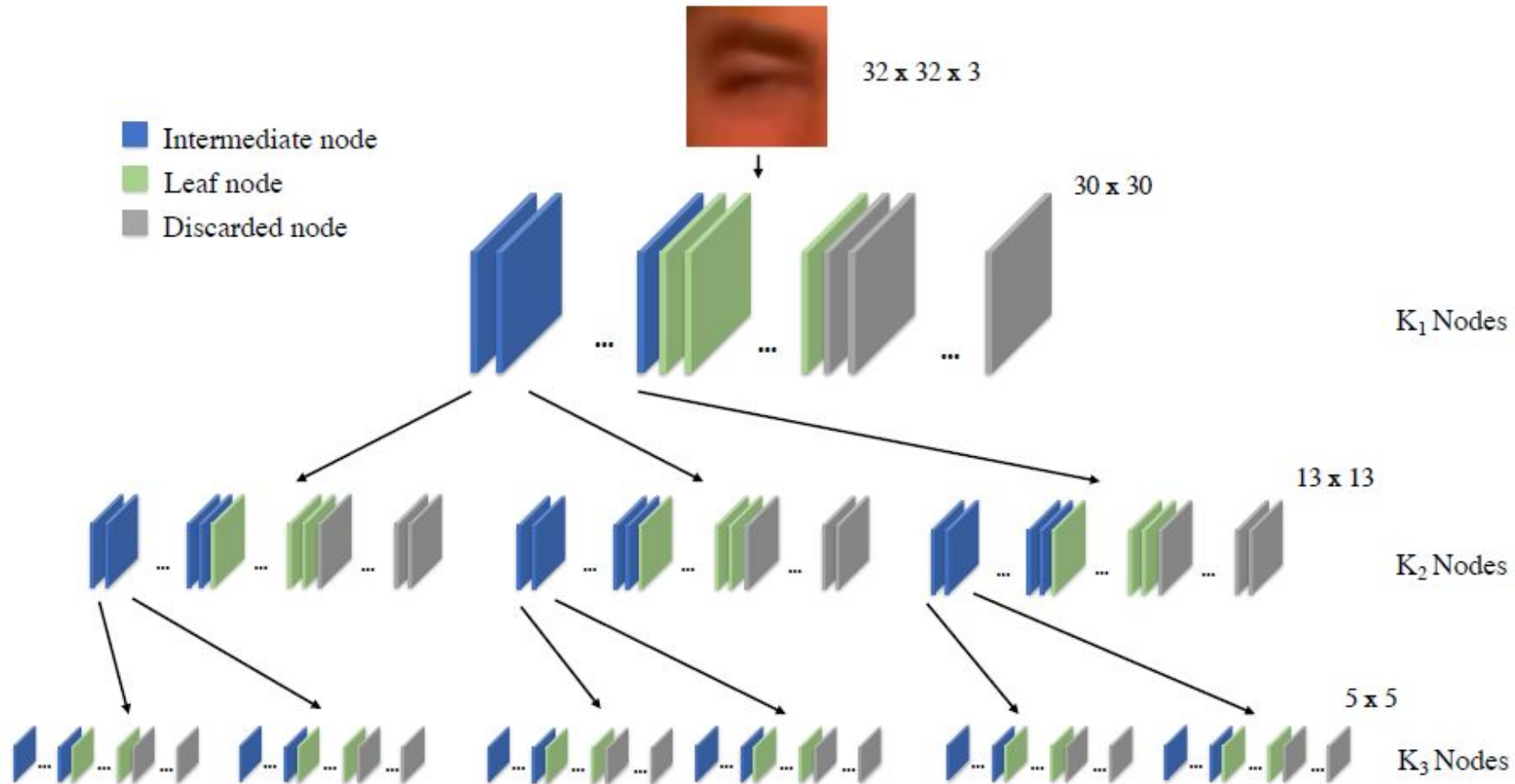
FireModule

Architecture Design 4



Note: $N1, N2, N3, N4, N5, N6$ are of dimensions less than 300×25

Architecture Design 4



Saab Transform

Results and Discussion

- The results obtained for the above architectures are for the entire dataset (Celeb-DF).
- The training, validation and testing splits are 80:10:10 respectively.



Results and Discussion

- The above executions have been carried out in the same environment and hardware setup as used to simulate the base paper.
- The same dataset has been used for execution of our architectures and results are as shown



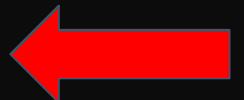
Base Paper Results on CelebDF



CelebDF - Testing accuracy = 0.6333 (80:10:10) - subset

Anaconda Powershell Prompt (anaconda3)

```
Epoch 164/200
8/8 [=====] - 19s 2s/step - loss: 0.0562 - accuracy: 0.9805 - val_loss: 0.9557 - val_accuracy: 0.8519
1/1 [=====] - 0s 379ms/step - loss: 2.9774 - accuracy: 0.3667
Epoch 165/200
8/8 [=====] - 19s 2s/step - loss: 0.0528 - accuracy: 0.9925 - val_loss: 4.2101 - val_accuracy: 0.3333
1/1 [=====] - 0s 396ms/step - loss: 11.7048 - accuracy: 0.2333
Epoch 166/200
8/8 [=====] - 19s 2s/step - loss: 0.2031 - accuracy: 0.9266 - val_loss: 0.6540 - val_accuracy: 0.9259
1/1 [=====] - 0s 378ms/step - loss: 3.3879 - accuracy: 0.4000
Epoch 167/200
8/8 [=====] - 19s 2s/step - loss: 0.1249 - accuracy: 0.9567 - val_loss: 0.6434 - val_accuracy: 0.9259
1/1 [=====] - 0s 385ms/step - loss: 2.7561 - accuracy: 0.4667
Epoch 168/200
8/8 [=====] - 19s 2s/step - loss: 0.0763 - accuracy: 0.9817 - val_loss: 0.7054 - val_accuracy: 0.9259
1/1 [=====] - 0s 377ms/step - loss: 2.4198 - accuracy: 0.6333
Epoch 169/200
8/8 [=====] - 20s 2s/step - loss: 0.0796 - accuracy: 0.9823 - val_loss: 1.9320 - val_accuracy: 0.5185
1/1 [=====] - 0s 391ms/step - loss: 3.0208 - accuracy: 0.4333
Epoch 170/200
8/8 [=====] - 19s 2s/step - loss: 0.0703 - accuracy: 0.9774 - val_loss: 2.0418 - val_accuracy: 0.5926
1/1 [=====] - 0s 389ms/step - loss: 3.0050 - accuracy: 0.5333
```



Base Paper Results on CelebDF



CelebDF - Testing accuracy = 0.9175 (80:10:10)

```
Epoch 70/100
77/77 [=====] - 48s 630ms/step - loss: 0.0604 - accuracy: 0.9788 - val_loss: 1.4128 - val_accuracy: 0.6252
5/5 [=====] - 1s 145ms/step - loss: 0.2695 - accuracy: 0.9038
Save model, acc= 0.9037800431251526
Epoch 71/100
77/77 [=====] - 48s 628ms/step - loss: 0.0278 - accuracy: 0.9895 - val_loss: 3.2452 - val_accuracy: 0.3364
5/5 [=====] - 1s 141ms/step - loss: 0.4000 - accuracy: 0.8763
Epoch 72/100
77/77 [=====] - 48s 629ms/step - loss: 0.0187 - accuracy: 0.9943 - val_loss: 1.7453 - val_accuracy: 0.6252
5/5 [=====] - 1s 137ms/step - loss: 0.2681 - accuracy: 0.9175
Save model, acc= 0.9175257682800293
```



Epochs = 100, batch_size = 64, Adam optimizer (lr = 0.001)

Fire Module + Attention Feature Map Results



CelebDF - Testing accuracy = 0.6999 (80:10:10) - subset

```
- accuracy: 1.0000 - val_loss: 4.2343 - val_accuracy: 0.5185
1/1 [=====] - 1s 523ms/step - loss:
4.5832 - accuracy: 0.4333
Epoch 99/100
8/8 [=====] - 24s 3s/step - loss: 0.0105
- accuracy: 1.0000 - val_loss: 5.1388 - val_accuracy: 0.4074
1/1 [=====] - 1s 516ms/step - loss:
5.1332 - accuracy: 0.4333
Epoch 100/100
8/8 [=====] - 24s 3s/step - loss: 0.0108
- accuracy: 0.9991 - val_loss: 5.1138 - val_accuracy: 0.4074
1/1 [=====] - 1s 523ms/step - loss:
5.1714 - accuracy: 0.4667
Best accuracy achieved 0.6999999988079071
```

Epochs = 100, batch_size = 32, Adam optimizer (lr = 0.001)

Fire Module + Attention Feature Map Results



CelebDF - Testing accuracy = 0.9588 (80:10:10)

```
Epoch 83/100
81/81 [=====] - 42s 516ms/step - loss: 0.0884 - accuracy: 0.9704 - val_loss: 0.6137 - val_accuracy: 0.8129
5/5 [=====] - 1s 120ms/step - loss: 0.4423 - accuracy: 0.8179
Epoch 84/100
81/81 [=====] - 42s 515ms/step - loss: 0.1205 - accuracy: 0.9575 - val_loss: 0.2706 - val_accuracy: 0.9126
5/5 [=====] - 1s 118ms/step - loss: 0.6101 - accuracy: 0.7491
Epoch 85/100
81/81 [=====] - 42s 514ms/step - loss: 0.0876 - accuracy: 0.9696 - val_loss: 0.3108 - val_accuracy: 0.9091
5/5 [=====] - 1s 118ms/step - loss: 0.2655 - accuracy: 0.8935
Epoch 86/100
81/81 [=====] - 42s 517ms/step - loss: 0.0628 - accuracy: 0.9785 - val_loss: 0.1692 - val_accuracy: 0.9895
5/5 [=====] - 1s 117ms/step - loss: 0.1378 - accuracy: 0.9588
Save model, acc= 0.9587628841400146
Epoch 87/100
81/81 [=====] - 42s 513ms/step - loss: 0.0708 - accuracy: 0.9776 - val_loss: 0.3379 - val_accuracy: 0.9353
5/5 [=====] - 1s 118ms/step - loss: 0.3156 - accuracy: 0.9003
```


Epochs = 100, batch_size = 64, Adam optimizer (lr = 0.001)

MBConv + Attention Feature Map Results



CelebDF - Testing accuracy = 0.6666 (80:10:10) - subset

```
1/1 [-----] - 1s 302ms/step - loss: 3.7722 - accuracy: 0.3007
Epoch 98/100
8/8 [=====] - 18s 2s/step - loss: 0.0158 - accuracy: 1.0000 - val_loss: 3.1451 - val_accuracy: 0.5556
1/1 [=====] - 0s 470ms/step - loss: 4.1120 - accuracy: 0.3667
Epoch 99/100
8/8 [=====] - 16s 2s/step - loss: 0.0123 - accuracy: 1.0000 - val_loss: 3.2814 - val_accuracy: 0.5556
1/1 [=====] - 0s 484ms/step - loss: 3.9743 - accuracy: 0.3333
Epoch 100/100
8/8 [=====] - 17s 2s/step - loss: 0.0082 - accuracy: 1.0000 - val_loss: 3.3245 - val_accuracy: 0.5556
1/1 [=====] - 0s 473ms/step - loss: 3.9330 - accuracy: 0.3000
Best accuracy achieved 0.666666865348816
```

A thick red arrow pointing to the left, highlighting the 'Best accuracy achieved' line in the terminal output.

Epochs = 100, batch_size = 32, Adam optimizer (lr = 0.001)

MBConv + Attention Feature Map Results



CelebDF - Testing accuracy = 0.9518 (80:10:10)

```
Epoch 96/100
77/77 [=====] - 39s 513ms/step - loss: 0.1019 - accuracy: 0.9643 - val_loss: 0.4520 - val_accuracy: 0.8702
5/5 [=====] - 1s 124ms/step - loss: 0.3821 - accuracy: 0.8247
Epoch 97/100
77/77 [=====] - 39s 512ms/step - loss: 0.0788 - accuracy: 0.9716 - val_loss: 0.3047 - val_accuracy: 0.8995
5/5 [=====] - 1s 122ms/step - loss: 0.3751 - accuracy: 0.8351
Epoch 98/100
77/77 [=====] - 40s 514ms/step - loss: 0.0588 - accuracy: 0.9791 - val_loss: 0.7224 - val_accuracy: 0.7697
5/5 [=====] - 1s 121ms/step - loss: 0.1386 - accuracy: 0.9519
Save model, acc= 0.9518900513648987
Epoch 99/100
77/77 [=====] - 40s 515ms/step - loss: 0.0649 - accuracy: 0.9818 - val_loss: 0.3641 - val_accuracy: 0.9031
5/5 [=====] - 1s 122ms/step - loss: 0.6527 - accuracy: 0.7320
Epoch 100/100
77/77 [=====] - 39s 512ms/step - loss: 0.0575 - accuracy: 0.9813 - val_loss: 0.7332 - val_accuracy: 0.7898
5/5 [=====] - 1s 120ms/step - loss: 0.1940 - accuracy: 0.9175
```

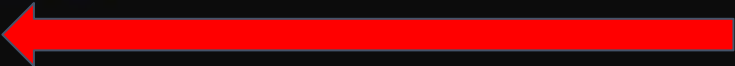
Epochs = 100, batch_size = 64, Adam optimizer (lr = 0.001)

Saab Transform + PCA

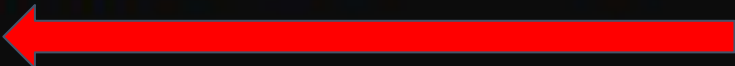


CelebDF - Testing accuracy = 0.3666 and 0.45 (80:20) - subset

```
=====Training Results=====
Accuracy  99.56709956709958
=====Testing Results=====
Accuracy  36.6666666666666664
```

A thick red arrow pointing from the right towards the testing accuracy value 36.6666666666666664.

```
=====Training Results=====
Accuracy  99.56709956709958
=====Testing Results=====
Accuracy  45.0
```

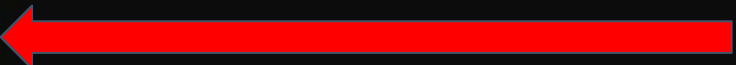
A thick red arrow pointing from the right towards the testing accuracy value 45.0.

Saab Transform + PCA

Dataset split - 80:20 (total of 2000 samples)

CelebDF - Testing accuracy = 0.8865 (80:20)

```
=====Training Results=====
Accuracy  99.2
=====Testing Results=====
Accuracy  88.65979381443299
```



Number of features	30	32	42	43	52
Training accuracy	0.9775	0.9825	0.992	0.993	0.9975
Testing accuracy	0.8384	0.8522	0.8865	0.875	0.868

Frame Normalization Method



Dataset split - 80:10:10 (total of 1300 samples)

CelebDF - Testing accuracy = 0.7538

Epoch 100/100

17/17 [=====] - ETA: 0s - loss: 0.3655 - accuracy: 0.8538 - val_loss: 1.0880 - val_accuracy: 0.4188

3/3 [=====] - 0s 16ms/step - loss: 0.6030 - accuracy: 0.7538

Save model, acc= 0.7538461685180664



Epochs = 100, batch_size = 64, Adam optimizer (lr = 0.001)

Comparative Study

Architecture	Attention Feature Map + MBConv	Attention Feature Map + FireModule	PCA + Saab Transform	Frame Normalization
Training Accuracy	0.9791	0.9785	0.992	0.8538
Testing Accuracy	0.9519	0.9588	0.8865	0.7538

Documentation



Show the evidences, status of the below documents:

- Project report:
<https://drive.google.com/file/d/1nyDxKgl9pBEa87AHeCds-WREs1dayEwM/view?usp=sharing>
- IEEE Paper:
<https://drive.google.com/file/d/1nyDxKgl9pBEa87AHeCds-WREs1dayEwM/view?usp=sharing>
- Add the Github repository link: <https://github.com/pk210700/PW22AV01.git>
- A3 size Poster of your project :<https://github.com/pk210700/PW22AV01.git>
- IEEE Plagiarism Report:
<https://drive.google.com/file/d/1aBFBT0d5Nv4OqWZMaTiIMkcJuPs3iy-h/view?usp=sharing>
- Project Report Plagiarism:
https://drive.google.com/file/d/1zTZZnrRxC5C32RT01j_vRxpWIJ_Tduzq/view?usp=sharing

Lessons Learnt

Following are some of the lessons we learnt along the way:

- Introduced to various methods of deepfake detection.
- Familiarised ourselves with the biological signal approach.
- Lots and lots of Debugging and error resolution.



Lessons Learnt

- By performing an extensive literature survey, we have been introduced to various methods such as Self Supervised Learning, N-shot Learning, Domain Adaptation, Reverse Engineering, Vision Transformer, Generalization.
- Working in a team and in a timely fashion.
- Writing an IEEE format paper and Report.
- Reaching out to peers and mentors for guidance.



Applications

Some major regions of application for deepfake detection

- 1) Social media / Video sharing sites (Ex: Twitter, Facebook, TikTok)
 - Screen videos for deep fake content.
 - Notify person whose likeness is being used.
- 2) Messaging applications. (Ex: Whatsapp, Telegram)
 - Screen rapidly spreading video content for deep fakes.
 - Notify users whenever deepfake videos are received by them.



Applications

3) Sites sharing pornographic content.

- Identify deep fake content, prevent them from being uploaded

4) In police investigations and court proceedings

- To prevent manipulation of decisions made by the court or police by identifying false evidence



Conclusion

- Since biological signals cannot be forged, it is a very effective method in detection of DeepFakes.
- The use of Attention Feature Map helped increase the accuracy on the baseline model.
- We trained 4 models of which 2 of them are recommended by us



Future Work

1. Modifications in preprocessing stage
2. Improvise on
Self Supervised Learning approach
3. Reduce Overfitting
4. Graphical Representation



Individual Contribution



Vishruth L

SRN - PES1201800362

- Literature Survey
- PPT preparation and presentation
- Alignment of faces and generation of face video
- Implementation of Attention Feature map, Normalized frame Model
- Generation of MMST Map
- IEEE Paper and Project Report



Individual Contribution

Vivek N

SRN - PES1201801077

- Literature Survey
- PPT preparation and presentation
- Resizing of frames
- Implementation of Fire Module
- Spatio-temporal weights generation
- IEEE Paper and Project Report



Individual Contribution

Nipun Bhat

SRN - PES1201801857

- Literature Survey
- PPT preparation and presentation
- Generation of Motion Magnified Video
Implementation of MBConv, Normalized
Frame Model
- Spatio-temporal weights generation
- IEEE Paper and Project Report



Individual Contribution

Paras S Khurana

SRN - PES1201801158



- Literature Survey
- PPT preparation and presentation
- Detection and extraction of face from each frame of video
- Implementation of PCA model and Fire Module
- Training on Celeb-DF and completion of base-paper implementation
- IEEE Paper and Project Report

References



- [1] Umur Aybars Ciftci, Ilke Demir, Lijun Yin , “How Do the Hearts of Deep Fakes Beat? DeepFake Source Detection via Interpreting Residuals with Biological Signals”, published in the proceedings of 2020 IEEE/IAPR International Joint Conference on Biometrics (IJCB), 26 Aug 2020.

- [2] Hua Qi, Qing Guo, Felix Juefei-Xu, Xiaofei Xie, Lei Ma, Wei Feng, Yang Liu, Jianjun Zhao “DeepRhythm: Exposing DeepFakes with Attentional Visual Heartbeat Rhythms”, published in the proceedings of ACM-MM 2020, 13 June 2020.

- [3] Steven Fernandes, Sunny Raj, Eddy Ortiz, Iustina Vintila, Margaret Salter, Gordana Urosevic, Sumit Jha, “Predicting Heart Rate Variations of Deepfake Videos using Neural ODE”, in the proceedings of 2019 IEEE Conference on Computer Vision and Pattern Recognition/CVF International Conference on Computer Vision Workshop (ICCVW) , 27-28 October 2019.

References



- [4] Ipek Ganiyusufoglu, L. Minh Ngô, Nedko Savov, Sezer Karaoglu, Theo Gevers, “Spatiotemporal Features for Generalized Detection of Deepfake Videos”, published in the proceedings of Computer Vision and Image Understanding (CVIU), 22 Oct 2020.
- [5] Ekraam Sabir, Jiaxin Cheng, Ayush Jaiswal, Wael AbdAlmageed, Iacopo Masi, Prem Natarajan, “Recurrent Convolutional Strategies for Face Manipulation Detection in Videos”, published in the proceedings of workshop on Applications of Computer Vision and Pattern Recognition to Media Forensics at CVPR 2019, 2 May 2019.
- [6] Lingzhi Li, Jianmin Bao, Ting Zhang, Hao Yang, Dong Chen, Fang Wen, Baining Guo, “Face X-ray for More General Face Forgery Detection”, published in the proceedings of CVPR 2020, 31 Dec 2019.
- [7] Pavel Korshunov, Sebastien Marcel, “DeepFakes: a New Threat to Face Recognition? Assessment and Detection”, 20 Dec 2018.

References



- [8] Yuezun Li, Xin Yang, Pu Sun, Honggang Qi, Siwei Lyu, “Celeb-DF: A Large-scale Challenging Dataset for DeepFake Forensics”, published in the proceedings of 2019 IEEE Conference on Computer Vision and Pattern Recognition/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 27 Sep 2019.
- [9] Chih-Chung Hsu, Chia-Yen Lee, Yi-Xiu Zhuang, “Learning to Detect Fake Face Images in the Wild”, published in the proceedings of 2018 IEEE IS3C Conference (IEEE International Symposium on Computer, Consumer and Control Conference), 24 Sep 2018.
- [10] Xinsheng Xuan, Bo Peng, Wei Wang, Jing Dong, “On the generalization of GAN image forensics”, published in the proceedings of CCBR 2019, 27 Feb 2019.
- [11] Lakshmanan Nataraj, Tajuddin Manhar Mohammed, Shivkumar Chandrasekaran, Arjuna Flenner, Jawadul H. Bappy, Amit K. Roy-Chowdhury, B.S. Manjunath, “Detecting GAN generated Fake Images using Co-occurrence Matrices”, 15 Mar 2019.

References



[12] Huy H. Nguyen, Fuming Fang, Junichi Yamagishi, Isao Echizen, “Multi-task Learning For Detecting and Segmenting Manipulated Facial Images and Videos”, Published in Proceedings of the IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS) 2019, Florida, USA, 7 Jun 2019.

[13] Run Wang, Felix Juefei-Xu, Lei Ma, Xiaofei Xie, Yihao Huang, Jian Wang, Yang Liu, “FakeSpotter: A Simple yet Robust Baseline for Spotting AI-Synthesized Fake Faces”, published in the proceedings to IJCAI 2020, 16 Jul 2020.

References



[14] David Güera, Edward J. Delp, “Deepfake Video Detection Using Recurrent Neural Networks”, Published in the proceedings of 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 14 Feb 2018.

[15] Peng Zhou, Xintong Han, Vlad I. Morariu, Larry S. Davis, “Two-Stream Neural Networks for Tampered Face Detection”, Published in the proceedings to 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 26 July 2017.

References



[16] Darius Afchar, Vincent Nozick, Junichi Yamagishi, Isao Echizen, “MesoNet: a Compact Facial Video Forgery Detection Network”, Published in the proceedings of 2018 IEEE International Workshop on Information Forensics and Security (WIFS), 4 Sep 2018.

[17] Huy H. Nguyen, Junichi Yamagishi, Isao Echizen, “Capsule-Forensics: Using Capsule Networks to Detect Forged Images and Videos”, Published in the proceedings of ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 26 Oct 2019.

References



- [18] Thanh Thi Nguyen, Cuong M. Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, Saeid Nahavandi, “Deep Learning for Deep Fakes Creation and Detection: A Survey”, 25 Sep 2019.

- [19] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, Matthias Nießner, “Face Forensics++: Learning to Detect Manipulated Facial Images”, Published in the proceedings to 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 25 Jan 2019.

References



- [20] Sowmen Das, Arup Datta, Md. Saiful Islam, Md. Ruhul Amin, “Improving DeepFake Detection Using Dynamic Face Augmentation”, Published in the proceedings of 2021 ICCVPR International Conference on Computer Vision and Pattern Recognition, 22 May 2021.
- [21] Shivangi Aneja, Chris Bregler, Matthias Nießner, “COSMOS: Catching Out-of-Context Misinformation with Self-Supervised Learning”, Published in the proceedings of 2021 ICCVPR International Conference on Computer Vision and Pattern Recognition, 21 April 2021.

References



- [22] Xinlei Jin , Dengpan Ye , and Chuanxi Chen , “Countering Spoof: Towards Detecting DeepFake with Multidimensional Biological Signals” ,22 Apr 2021
- [23] Roberto Caldelli ,Leonardo Galteri ,Irene Amerini , Alberto Del Bimbo, “Optical Flow based CNN for detection of unlearnt deepfake manipulations”, 8 March 2021
- [24] Patrick Kwon, Jaeseong You, Gyuhyeon Nam, Sungwoo Park, Gyeongsu Chae ,
“KoDF: A Large-scale Korean DeepFake Detection Dataset”, Published in the proceedings of ICCV 2021, 18 Mar 2021

References



- [25] Bojia Zi1, Minghao Chang, Jingjing Chen, Xingjun Ma , Yu-Gang Jiang , "Wild Deepfake: A Challenging Real-World Dataset for Deepfake Detection” , 5 Jan 2021
- [26] Baoying Chen and Shunquan Tan, “Feature Transfer: Unsupervised Domain Adaptation for Cross-Domain Deepfake Detection”,7 June 2021
- [27] Zhi Wang, Yiwen Guo, and Wangmeng Zuo, “ DeepFake Forensics via An Adversarial Game” , 25 Mar 2021

References



- [28] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, Aythami Morales, Javier, Ortega-Garcia , “Deepfakes and beyond: A Survey of face manipulation and fake detection”, Published in the proceedings of Biometrics and Data Pattern Analysis conference 2020 , 1 Jan 2020
- [29] Ali Borji, “Pros and Cons of GAN Evaluation Measures” , Published in the proceedings of , published in the proceedings of Computer Vision and Image Understanding ,17 Mar 2021
- [30] Minha Kim, Shahroz Tariq, Simon S. Woo,”FReTAL: Generalizing Deepfake Detection using Knowledge Distillation and Representation Learning”, Published in the Workshop on Media Forensics 2021 28 May 2021

References



- [31] Shivangi Aneja, Matthias Nießner, “Generalized Zero and Few-Shot Transfer for Facial Forgery Detection” , 21 Jun 2020
- [32] Haoqiang Fan *, Erjin Zhou, “Approaching human level facial landmark localization by deep learning”, 30 November 2015
- [33] Yuezun Li, Cong Zhang, Pu Sun, Honggang Qi, Siwei Lyu,”DeepFake-o-meter: An Open Platform for DeepFake Detection”, Published in the proceedings of SAPDE 2021, 2 Mar 2021
- [34] João Paulo Meneses, CECs, Portugal,”Deepfakes and the 2020 US elections: what (did not) happen”, 22 Jan 2021

References



- [35] Vineet Mehta, Parul Gupta, Ramanathan Subramanian, Abhinav Dhall, “FakeBuster: A DeepFakes Detection Tool for Video Conferencing Scenarios”, 9 Jan 2021
- [36] Xuan HauNguyenaThai SonTranaVan ThinhLebKim
DuyNguyencDinh-TuTruong,”Learning Spatio-temporal features to detect manipulated facial videos created by the Deepfake techniques “ ,5 January 2021
- [37] Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, Maurizio Tesconi, “TweepFake: about Detecting Deepfake Tweets”, 31 Jul 2020

References



- [38] Sowmen Das, Selim Seferbekov, Arup Datta, Md. Saiful Islam, Md. Ruhul Amin, “Towards Solving the DeepFake Problem : An Analysis on Improving DeepFake Detection using Dynamic Face Augmentation”, 18 Feb 2021
- [39] Ramin Skibba , “Accuracy Eludes Competitors in Facebook Deepfake Detection Challenge”,
- [40] Shahroz Tariq, Sowon Jeon, Simon S. Woo, “Am I a Real or Fake Celebrity? Measuring Commercial Face Recognition Web APIs under Deepfake Impersonation Attack”, 1 Mar 2021

References



- [41] Shivangi Aneja, Chris Bregler, Matthias Nießner , “COSMOS: Catching Out-of-Context Misinformation with Self-Supervised Learning”, 15 Jan 2021
- [42] Vivek Sharma★, Rainer Stiefelhagen,” Self-Supervised Face-Grouping on Graphs Veith Röthlingshöfer “, 21 October 2019
- [43] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn,Xiaohua Zhai,Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby , “AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE “, Published in the proceedings of ICLR 2020, 22 Oct 2020



Thank You