XAI for DeepFake Detection

Tanisha Khurana **200483138**

Kaushik Ram Selvaraj **200474371**

Vikram Pande **200473855**

(Team - 7)

Abstract:

The proliferation of AI-generated fake images, or DeepFakes, has become a serious issue as they can spread misinformation and abusive content rapidly through modern media platforms. To address this problem, we present a two-phase learning architecture for detecting DeepFake images. The first phase uses model XceptionNet which is a depthwise convolution based model to identify unique features that identify fake images from real ones. The second phase is more about learning interpretability of the model, utilizing Explainable AI (XAI) models such as LIME and Grad-CAM to understand the interpretation of the model and why it makes the decisions. Datasets used were FaceForensic++[6] and Celef-DF[7] benchmark datasets.

Introduction:

Due to advancements in Applied Computer Graphics and AI, it is now possible to create realistic videos of people saying fictional things. With the help of architectures such as GANs (Generative Adversarial Networks) and AEs (AutoEncoders), it has become very common to generate fakes. This impacts the legitimacy of information presented online and such content may be used maliciously as a source of manipulation, harassment, misinformation and persuasion. Faces are of peak interest as subject to manipulation techniques. Also, face detection, face feature tracking, face reconstruction are well researched fields in Computer Vision. In this project, we try to classify between fake and real images with the help of XceptionNet and try to gain insights of the explainability of the model with LIME and Grad-CAM.

Literature Review:

Deepfake detection by manual efforts is an extremely difficult task and thus deep learning approaches are more dependable. The earliest generation of work focused on non-deep learning approaches for detecting manipulated images before the rise of GANs, and included analyzing low-level features in images such as JPEG compression artifacts and chromatic aberrations [12]. The paper “Deep Learning for Deepfakes Creation and Detection: A Survey"[4] presents a survey of algorithms used to create deep fakes and more than that, proposes methods to detect deep fakes. It discusses the challenges, trends and directions related to deepfake technology. It mainly focuses on the architectures of GANs and AutoEncoders.

Another approach "DeepFake Detection Through Deep Learning"[5] discusses the two state-of-the-art architectures XceptionNet[1] and MobileNet[11] as classification models on FaceForensic++ dataset. The accuracy range of these models is in between 91%-98% depending on the type of fakes in FFPP. They have also developed a voting mechanism that can be used to detect fake videos using the aggregation of all four methods instead of only one.

Dataset Description:

Dataset	Description
Celeb-DF[6]	This dataset includes 590 original videos collected from YouTube with subjects of different ages, ethnic groups and genders with 5639 corresponding DeepFake videos.
FaceForensic++[7]	FaceForensic++ is a forensic dataset consisting of 1000 original video sequences that have been manipulated with four automated face manipulation methods: DeepFakes, Face2Face, FaceSwap and NeuralTextures. The data has been collected from 977 YouTube Videos and all videos contain a trackable, mostly frontal face which enables automated tampering methods to generate realistic forgeries.

Methodology:

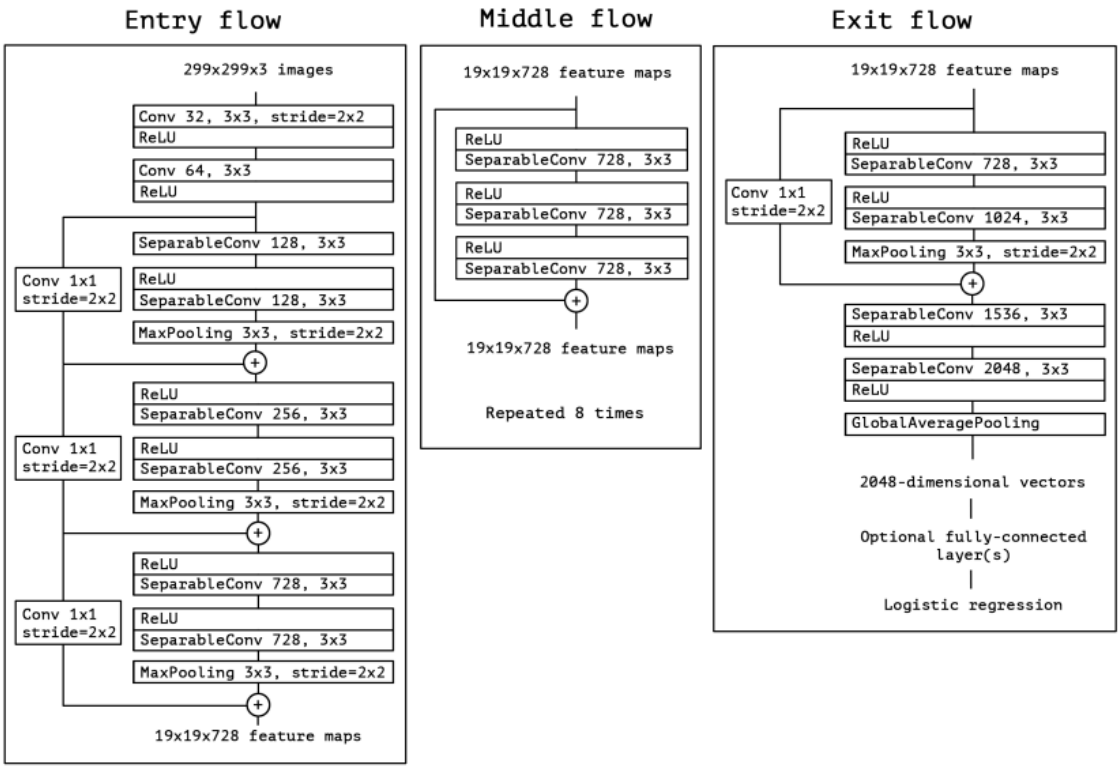
XceptionNet:

We have used XceptionNet as a classifier to classify between Real and Fake images. XceptionNet *(Extreme Inception)* is an architecture presented by Francois Chollet in 2017. It focuses on the reduction of memory by introducing Depthwise Separable Convolution Operation(a depthwise convolution followed by a pointwise convolution) inspired by Inception architecture[1]. The main motive of these architectures was to train models efficiently with fewer parameters with richer results. These architectures are known to be convolutional feature extractor architectures and are best to gain the insights of how each layer of model exactly works and how they differ from a regular convolution. The definition of a depthwise separable convolution from the original XceptionNet paper[1], commonly called "separable convolution" in deep learning frameworks is nothing but a model, consists in a depthwise convolution, i.e. a spatial convolution performed independently over each channel of an input, followed by a

pointwise convolution, i.e. a 1x1 convolution, projecting the channels output by the depthwise convolution onto a new channel space. This is not to be confused with a spatially separable convolution, which is also commonly called “separable convolution” in the image processing community. In a depthwise separable convolution, the convolution operation is first applied to each input channel separately using depth-wise convolution. Then, a point-wise convolution is applied to combine the output of the depthwise convolution operation across channels, as in a traditional convolutional operation. This approach allows the model to reduce the computational cost and the number of parameters needed, hence this makes it an excellent model which can be used in places where computational resources are limited.

In our case, we did transfer learning by changing the last classifier layer of Xception with 2 as the number of classes. Weights that are used are ImageNet weights. We used the HuggingFace XceptionNet model.

Architecture:



Loss Function

As this is a binary classification task, real/fake, the loss function used is Binary Cross Entropy Loss/Log Loss.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

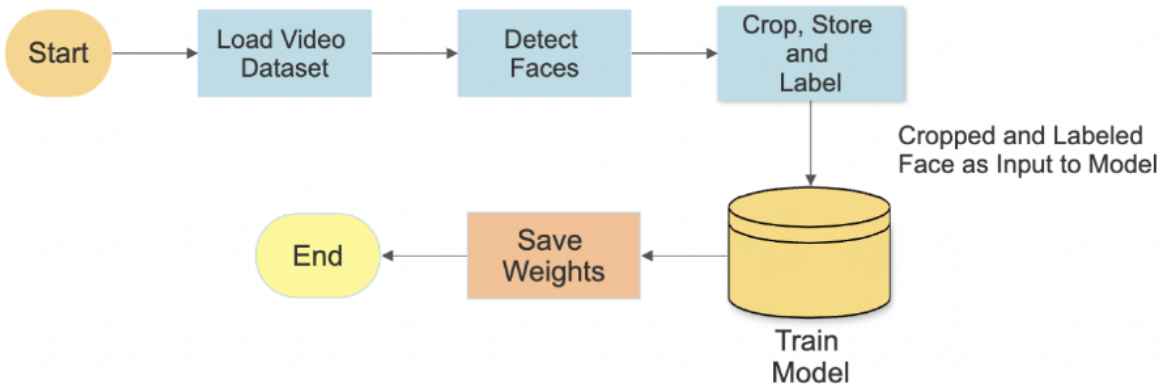
The Binary Cross Entropy describes how likely a model is and the error function of each data point. It will compare the predicted and real values and tell us how diverged our results are from the original data using Kullback-Leibler Divergence[3]

Optimizer

Optimizer used is ADAM (ADAptive Moment Estimation) optimizer, which is most common nowadays as it is a combination of RMSprop and Stochastic Gradient Descent with momentum. Adaptive Moment Estimation, (ADAM) as the name suggests, as it uses the estimations of the first as well as second moments of the gradient to adapt the existing learning rate for each weight of the neural network. It uses squared gradients to scale the learning rate (RMSprop) in this case and the moving average of the gradient.[8] Learning rate was set at 0.001.

Training Workflow

Ran the model for 10 epochs for both datasets and saved the model and weights again to use in LIME and Grad-CAM.



Explainable AI

LIME:
LIME (Local Interpretable Model-Agnostic Explanations) is an algorithm used for explaining the predictions of machine learning models, including image classification models.

The basic idea behind LIME for images is to explain the prediction of an image classifier by highlighting the regions of the image that were most important in making the prediction. This can help us understand why the model made a certain prediction, and can provide insights into how the model is working.

LIME works by generating a set of "perturbed" versions of the original image, each of which is slightly modified in some way (for example, by adding or removing pixels). It then passes each perturbed image through the image classifier and observes the changes in the predictions. Based on this information, LIME can estimate the importance of different regions of the image in making the final prediction.

Once LIME has identified the important regions of the image, it can generate a "heatmap" that visually highlights these regions. This can be a useful tool for interpreting the results of an image classification model and understanding the features that are most important in determining the model's predictions.

Grad-CAM:
GradCAM (Gradient-weighted Class Activation Mapping) is an algorithm used for visualizing and interpreting the predictions of deep convolutional neural networks (CNNs) in image classification tasks.

The basic idea behind GradCAM is to generate a heatmap that highlights the regions of the image that the CNN was most sensitive to in making its prediction. This can help us understand why the CNN made a certain prediction and can provide insights into the features that the CNN is using to make its predictions.

To generate the heatmap, GradCAM uses the gradients of the predicted class with respect to the feature maps of the last convolutional layer of the CNN. These gradients are then used to weight the feature maps, highlighting the regions that are most relevant to the predicted class.

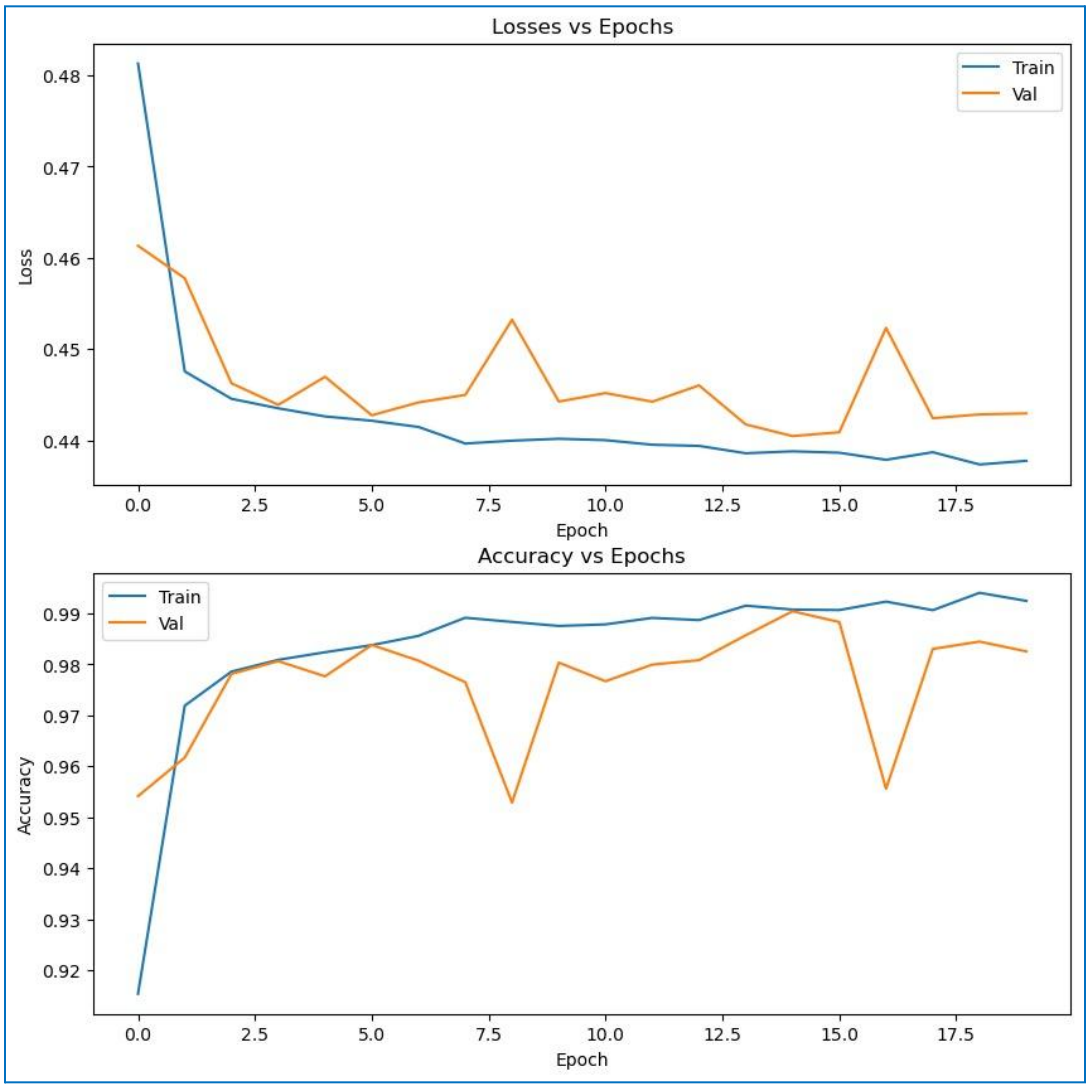
Once the feature maps have been weighted, GradCAM applies a global average pooling operation to generate a one-dimensional vector of weights for each feature map. These weights

are then used to compute a weighted sum of the feature maps, resulting in a single heatmap that highlights the regions of the image that were most important in making the prediction.

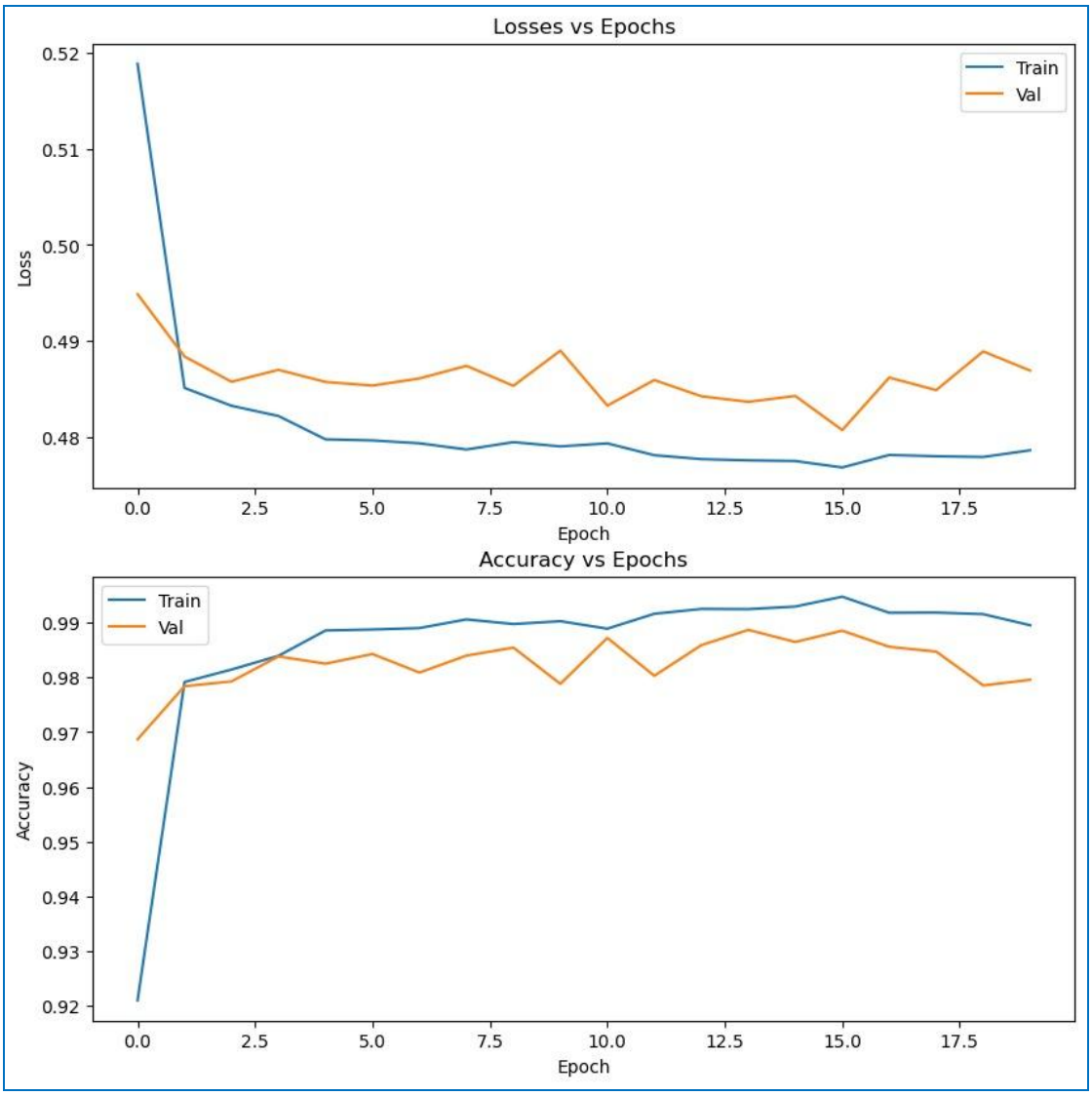
The heatmap generated by GradCAM can be overlaid onto the original image to create a visualization that highlights the important regions of the image. This can be a useful tool for interpreting the results of an image classification model and understanding the features that are most important in determining the model's predictions.

Results:

Accuracy vs Epochs and Loss vs Epochs plots for both datasets



Celeb-DF

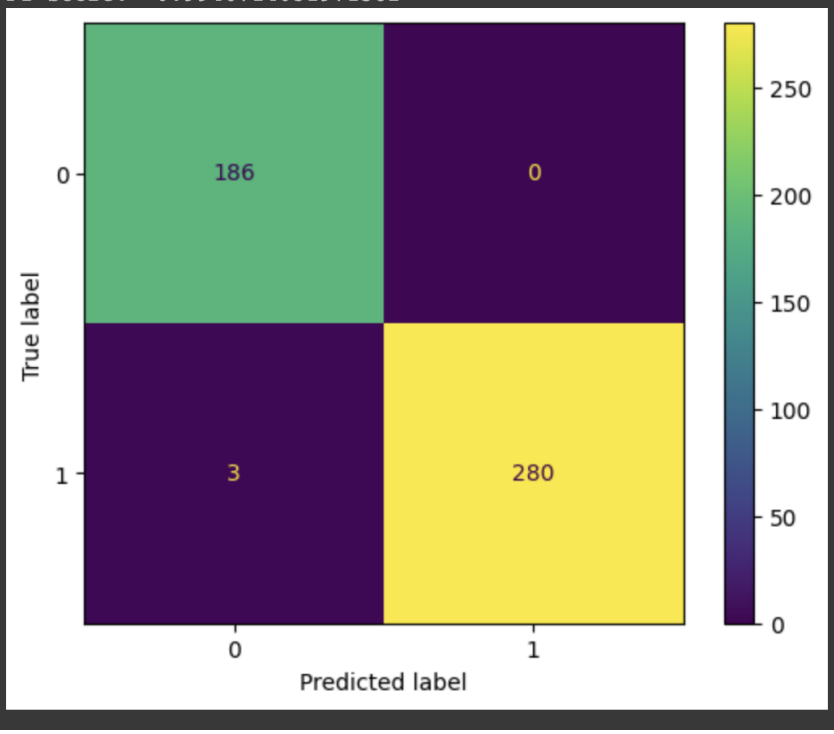


FaceForensic++

Metrics and confusion matrix for CelebDF dataset

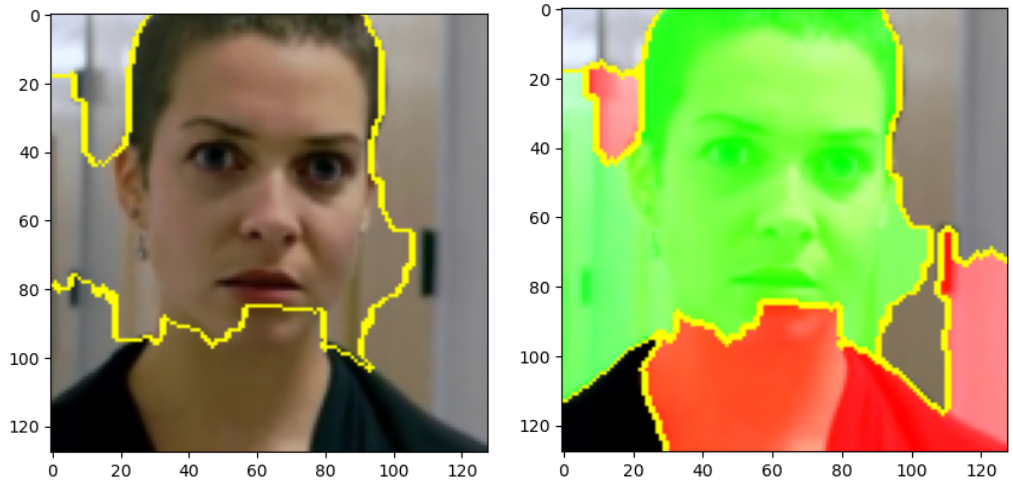
	precision	recall	f1-score	support
0	0.98	1.00	0.99	186
1	1.00	0.99	0.99	283
accuracy			0.99	469
macro avg	0.99	0.99	0.99	469
weighted avg	0.99	0.99	0.99	469

```
Precision:  1.0
Recall:  0.9893992932862191
F1 score:  0.9946714031971582
```

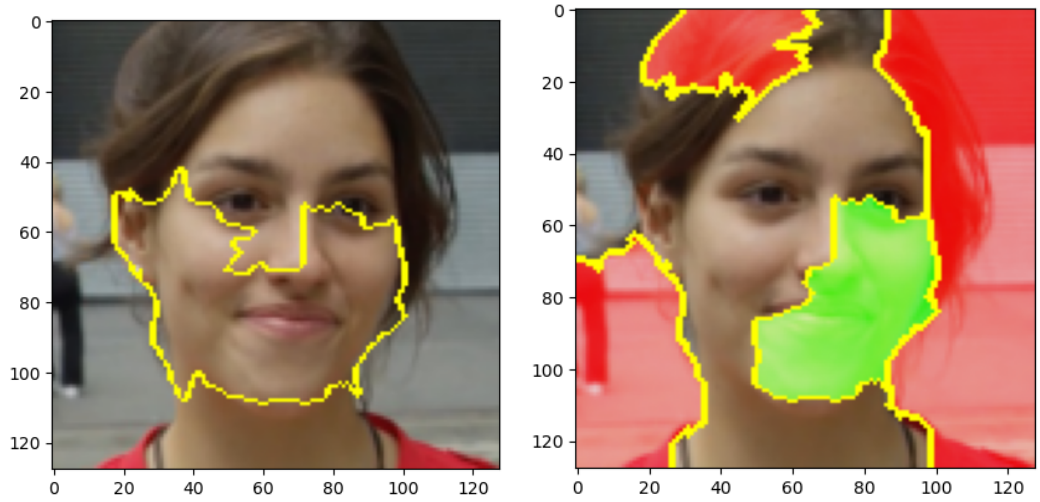


LIME results on fake and real images

In the Figure below, we have considered a fake image and displayed the LIME output. From the figure we can conclude that the model mainly takes the face into account and highlights the portion which positively helps the prediction in green and the portion which negatively affects the prediction in red. One reason for this could be that a fake image was pasted on another real image hence the model considers the red part as a section from a real image.

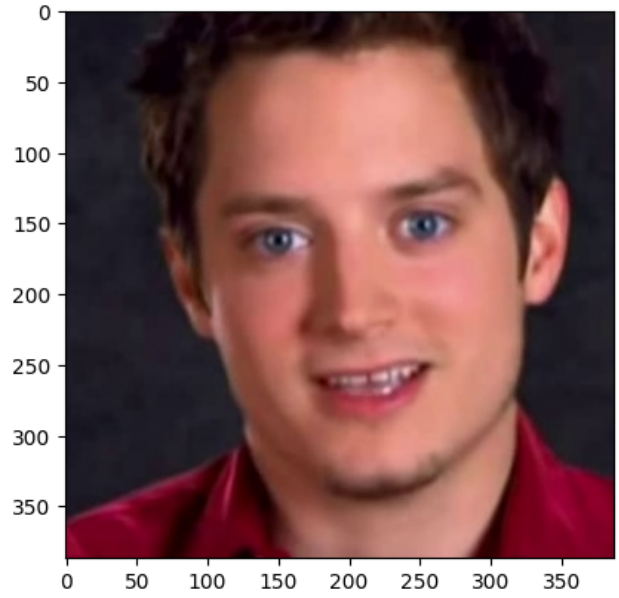


More Examples of LIME:



Grad-CAM results

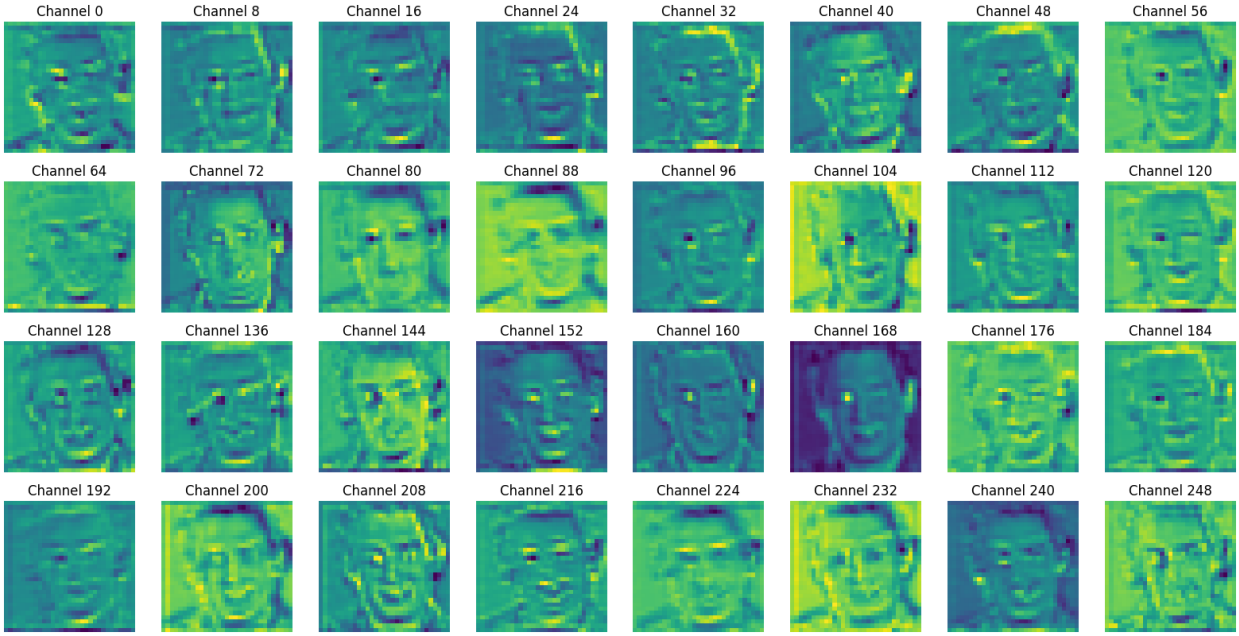
For Celeb- DF real image:



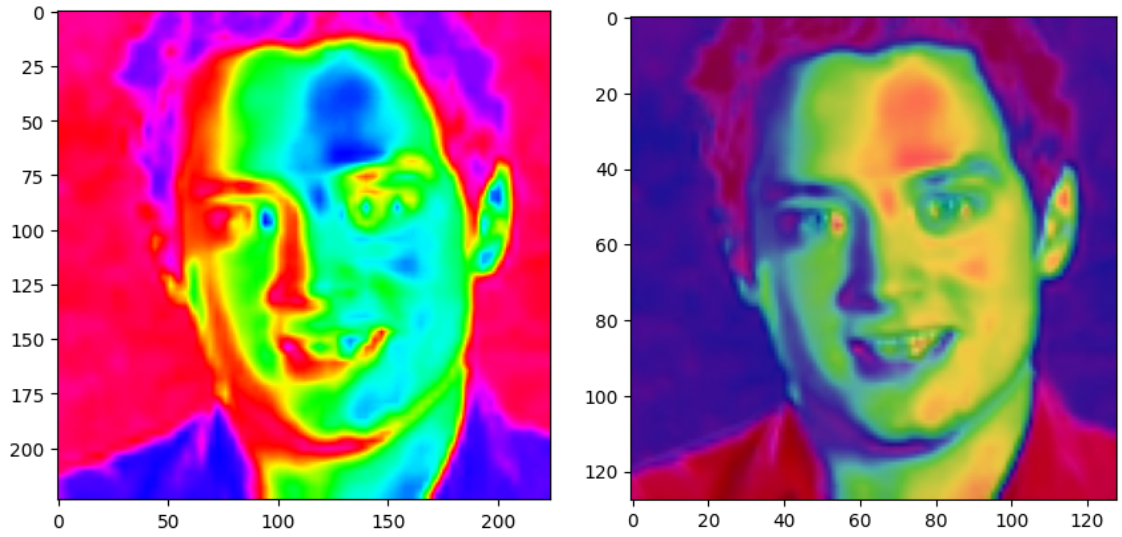
Original image

Activations for the first conv2D layer

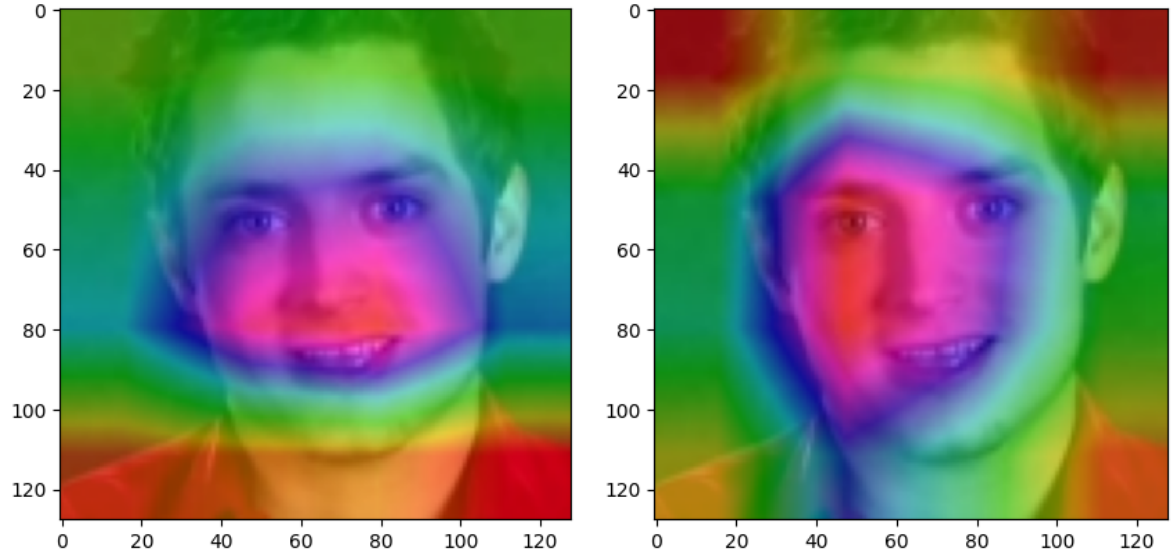
Activation for conv1 after applying Relu



Activations for the 4th separable conv2D layer in the second block

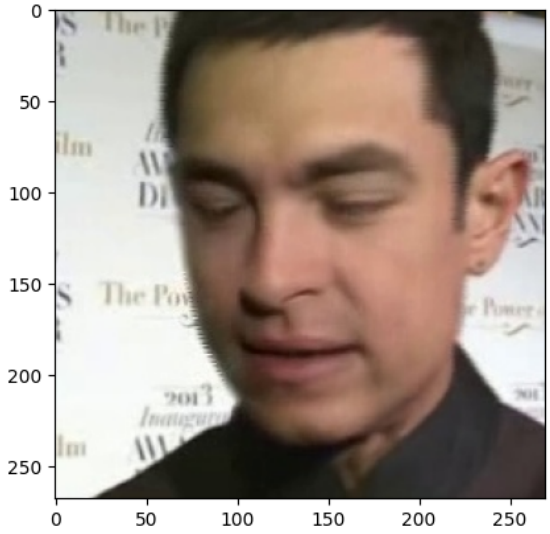


Grad cam visualization results for conv1 layer

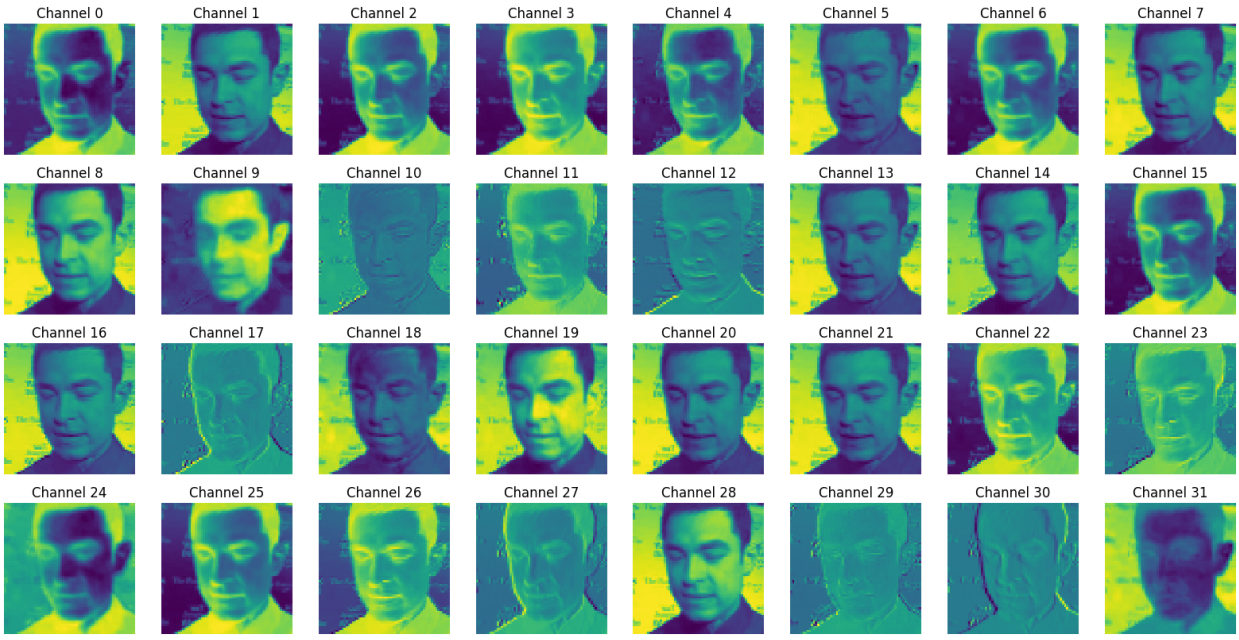


Grad cam visualization results for last two conv2D layers

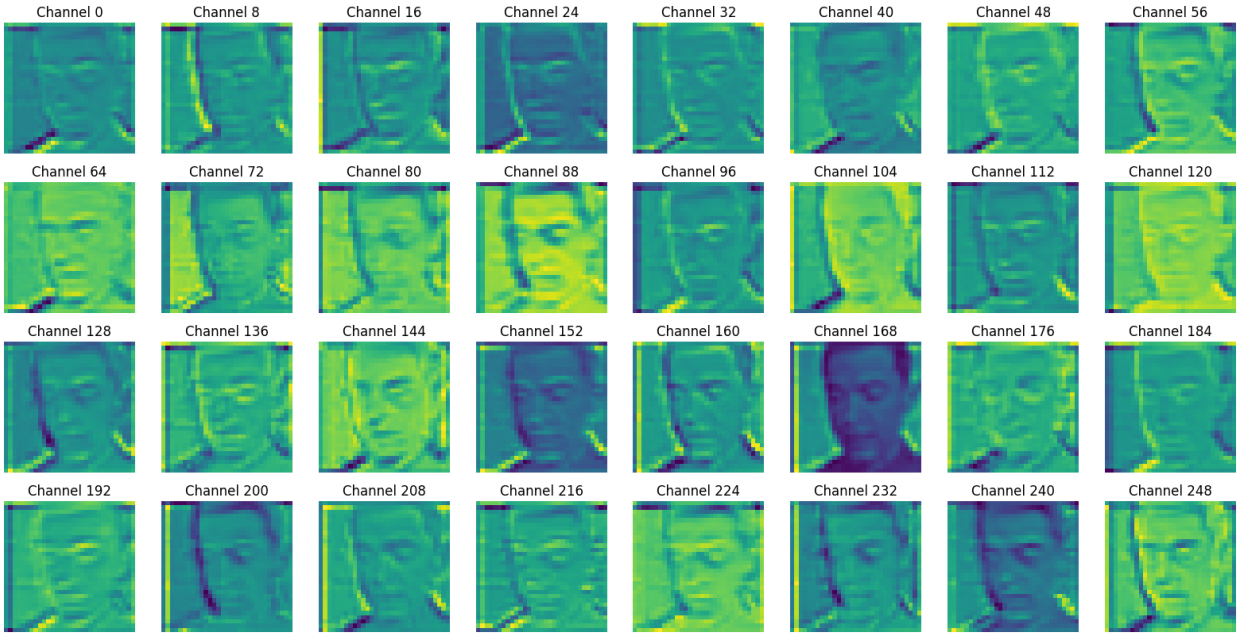
For Celeb- DF fake image:



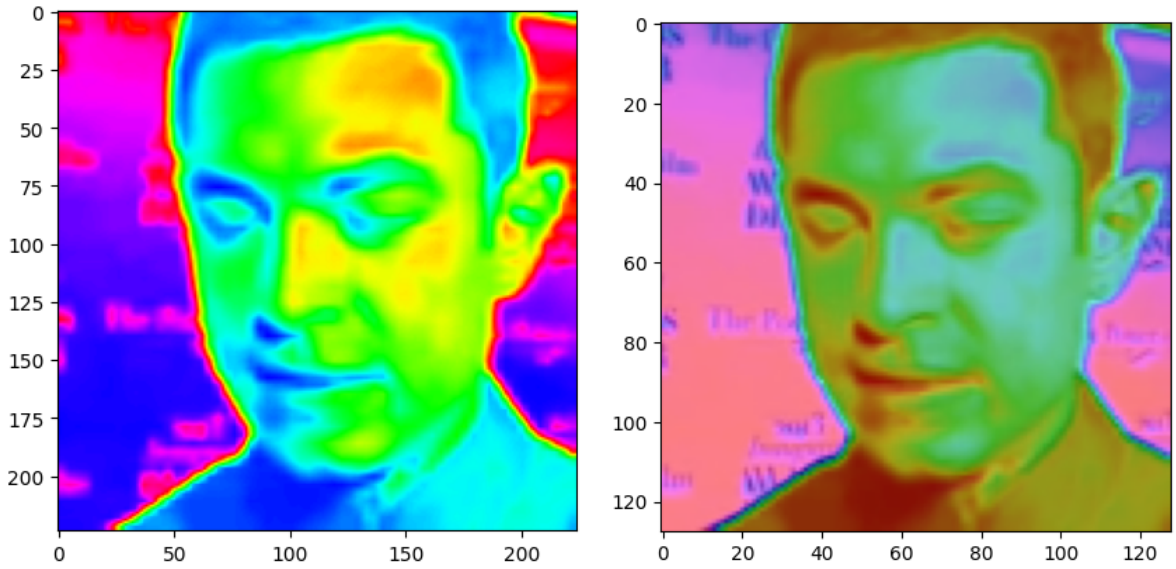
Original image



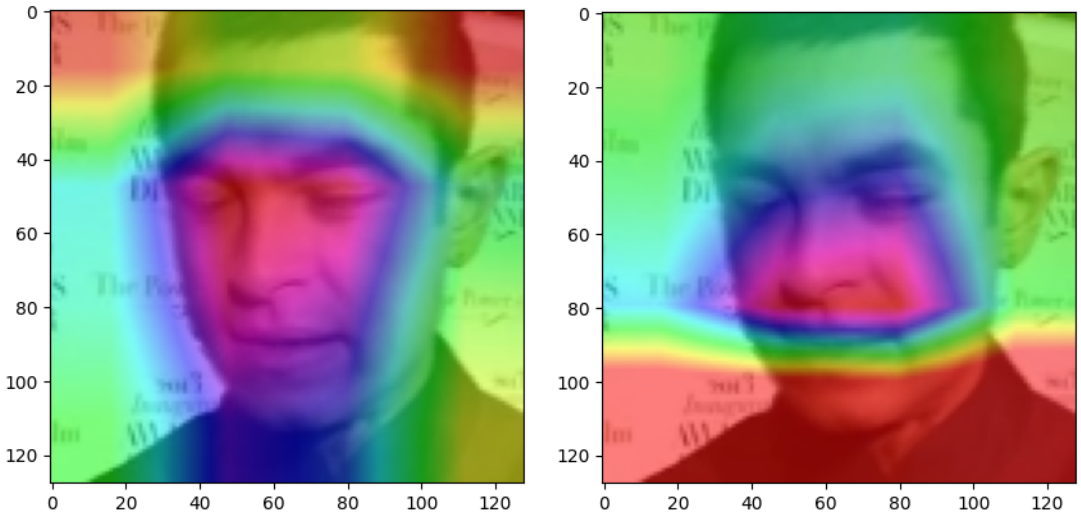
Activations for the first conv2D layer



Activations for the 4th separable conv2D layer in the second block



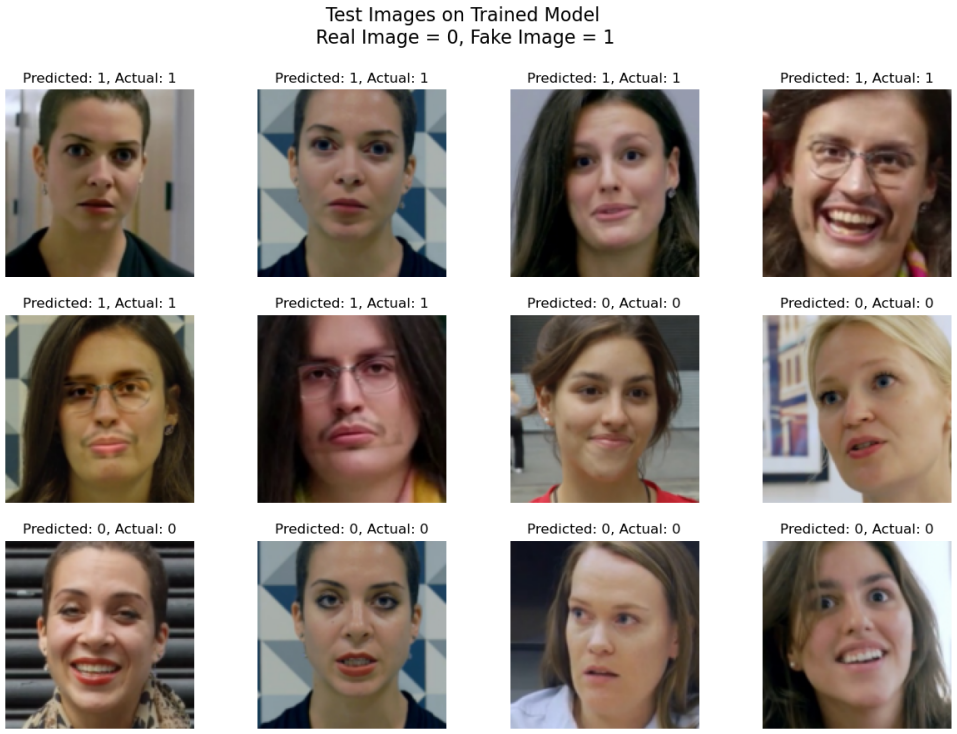
Grad cam visualization results for conv1 layer



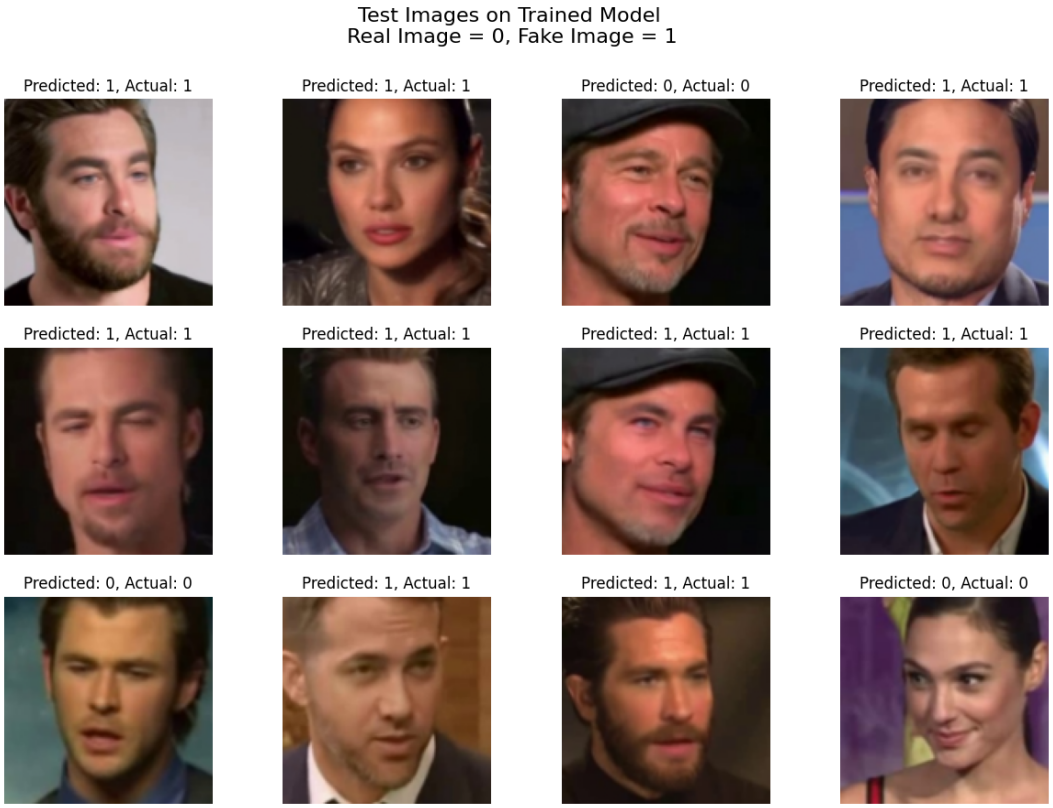
Grad cam visualization results for last two conv2D layers

Predictions:

Predictions on validation dataset for FaceForensics ++:



Predictions on validation dataset for CelebDF dataset:



Hardware/Software Resources:

Pytorch framework was used to train the model. All the steps such as loading data with data loader, transforming and training the data were implemented using PyTorch documentation[9]. Hardware used was Advanced Workstation in Dataspace, Hunt library with specifications of 128 GB RAM and 64 GB NVIDIA GPU.

Conclusion:

From the implementation, we can conclude that by using strong pretrained models such as Xception, best results can be obtained in terms of loss, accuracy and predictions. XceptionNet achieved a validation accuracy of 98.2% on the CelebDF dataset and 97.5% on the FaceForensic++ dataset. LIME identified the pixels in the input image that were most influential in prediction and generated the heatmap to highlight regions of the image contributed most to the prediction which is helpful to understand what features the model focused on when making the prediction. With Grad-CAM, Gradient information is obtained and used to highlight the regions of the input image that are most relevant to the output class and thereby explaining why CNN made a particular prediction.

References:

[1] B. Malolan, A. Parekh and F. Kazi, "Explainable Deep-Fake Detection Using Visual Interpretability Methods," 2020 3rd International Conference on Information and Computer Technologies (ICICT), San Jose, CA, USA, 2020, pp. 289-293, doi: 10.1109/ICICT50521.2020.00051.

[2] Chollet, François. "Xception: Deep learning with depth wise separable convolutions." In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251-1258. 2017.

[4] "Deep Learning for Deepfakes Creation and Detection: A Survey" https://arxiv.org/pdf/1909.11573.pdf

[5] D. Pan, L. Sun, R. Wang, X. Zhang and R. O. Sinnott, "Deep fake Detection through Deep Learning," *2020 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT)*, Leicester, UK, 2020, pp. 134-143, doi: 10.1109/BDCAT50828.2020.00001

[6] FaceForensic++ Dataset - https://github.com/ondyari/FaceForensics

[7] CelebDFv2 Dataset - https://github.com/yuezunli/celeb-deepfakeforensics

[8] ADAM Optimizer - https://optimization.cbe.cornell.edu/index.php?title=Adam#:~:text=Adam%20optimizer%20is%20the%20extended,was%20first%20introduced%20in%202014.

[9] PyTorch Training a Classifier - https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

[11] MobileNetv2: Inverted Residuals and Linear Bottlenecks https://arxiv.org/abs/1801.04381

[12] H. Farid, "Image forgery detection," IEEE Signal Processing Magazine, vol. 26, no. 2, pp. 16–25, 2009. https://farid.berkeley.edu/downloads/publications/spm09.pdf

[13]Wahidul Hasan Abir "Detecting Deepfake Images Using Deep Learning Techniques and Explainable AI Methods" https://file.techscience.com/ueditor/files/iasc/TSP_IASC-35-2/TSP_IASC_29653/TSP_IASC_29653.pdf in Intelligent Automation & Soft Computing, 2022

[14] Suk-Young Lim , "Detecting Deepfake Voice Using Explainable Deep Learning Techniques" https://www.mdpi.com/2076-3417/12/8/3926, Department of Artificial Intelligence, Hanyang University, Seoul 04763, Korea, 2022