**Deep Fake Image Detection Using Deep Learning**

**Abstract**

*This paper focuses on the ability to detect images of people, known as deep fakes, which has raised concerns about the potential for spreading misinformation and non-consensual content across social media platforms. We collected 6000 deep fake videos featuring celebrities from a widely used dataset. From these videos, we extracted individual frames to build a training dataset for our detection model. To ensure consistency, we resized these frames and fed them into a pre-trained model Our approach includes multiple deep learning model architectures and techniques using tensorflow, and keras, including pre-trained CNN models like ResNext called ResNet50 for classification, distinguishing between real and fake images. Our methodology involved several key steps: gathering deep fake videos, extracting frames, resizing images for uniformity, and employing ResNet50 for classification. By utilizing these techniques, we aimed to enhance the detection of deep fake images within social media content, contributing to the ongoing efforts to combat misinformation and promote online authenticity.*

1. **Introduction**

In an era where digital media has become ubiquitous, the ability to generate highly realistic synthetic images, known as deep fakes, has emerged as a concerning challenge. Our journey into this realm takes the form of an innovative project—the Deep Fake Image Detection System. At the heart of our endeavor is the utilization of computer vision and deep learning algorithms to distinguish real and fake images.

Our approach involves generating images as a first step, we considered a Celeb-DF dataset, which contains over 6000 real or fake videos, and from these videos, we have extracted images at an interval of 3 seconds using OpenCV and generated face cropped images using MTCNN with the help of in-built functions named VideoCapture and imwrite to process each frame and extracted a .png file from .mp4 file. Second, we performed some analysis to make images ready to ingest them for training so as part of preprocessing we resized the image to 128x128 to make sure all images have the same dimensions also we labeled the real images and fake images as 0 and 1 using encoding to label the images and split the images and its labels into training and testing data of 80% and 20%.

For the detection process, we employed a pre-trained deep learning model named ResNet50 which consists of 50 layers. These layers include a combination of convolutional layers, pooling layers, fully connected layers, and shortcut connections, commonly used in image processing tasks. These layers perform operations such as feature extraction and pattern recognition by applying a filter (kernel) to input images to produce feature maps. While ResNet50 primarily employs Conv2D layers, it also includes global average pooling and fully connected layers toward the end of the network for classification tasks.

In the final stages, the prepared training and testing datasets were fed into the model through an Image Data Generator, which normalizes pixel values and applies transformations like rotation and flipping to enhance the training process. After training, the model was used to evaluate the preprocessed validation data to determine the authenticity of the images, classifying them as either real or fake. This paper outlines our comprehensive approach to addressing the challenges posed by deep fakes in the realm of digital media.

1. **Exploratory Data Analysis**

In this project, we are utilizing the images that we extracted from the Celeb-DF(v2) video dataset available in the public source named paperswithcode.com. This dataset consists of 6000 videos including real videos of 600 taken from sources like YouTube and fake videos of a total of 5400 which are synthesized. Now as we are detecting the images whether real or fake, we have utilized the cv2 package to perform analysis on the videos. As a first step, we have extracted the first frame from each video file. Using the “cv2.VideoCapture” function, we accessed the videos stored in a local directory. For each video, the function attempts to read the first frame.

Once the frames of each video are successfully captured, it's saved as a JPEG file in a designated output folder. This involves changing the file extension from '.mp4' to '.png' to reflect the change in file type. We have performed batch processing of multiple videos through a loop that scans every file in the video directory. It filters for files ending with '.mp4', ensuring that only video files are processed. This generation of images serves as the foundation for our project.

1. **Methodology**

****

Based on the above Figure, the process involves several major tasks and subtasks, involving specific packages and machine learning algorithms.

* 1. **Video Processing**

This workflow is initiated by processing video files to extract still images, a pivotal step in training a machine-learning model. Utilizing the MTCNN (Multi-Task Cascaded Convolutional Networks) face detection model, it systematically tracks and identifies extracted faces, focusing on detecting counterfeit ones. Video frames undergo conversion to RGB format and subsequent face detection. Throughout the process, it monitors the number of counterfeit faces detected, halting once a predetermined limit is reached. The workflow encompasses two primary stages: one for extracting and cropping faces from individual frames, and another for managing the analysis of multiple video files. Detected faces are precisely cropped using bounding box coordinates and saved individually as PNG files in the specified output folder. Importantly, the process adapts seamlessly to the varying frame rates of the input videos.

Overall, we have simplified the task of processing video data for face analysis, making it accessible even to those unfamiliar with the technicalities of the underlying code. It underscores the importance of automated tools in streamlining data processing tasks, such as face detection, within the realm of computer vision applications.

* 1. **Image Loading and Preprocessing**

Once the images are extracted, the next step is to load and preprocess these images to prepare them for the model training. This is conducted through a function that iterates over each image in the specified folders (for both 'real' and 'fake' images). Each image is read, converted to the RGB color space, resized to a uniform dimension (128x128 pixels), and normalized. The normalization and resizing are crucial preprocessing steps to ensure that the images are in a suitable format for efficient learning by the neural network. Subsequently, data from both real and fake image folders are combined, forming the dataset for further processing. Following data combination, the dataset is split into training and testing sets, facilitating model evaluation. The preprocess\_input function from the TensorFlow.keras.applications.resnet50 module is used to normalize pixel values appropriately for the ResNet50 model, ensuring that the model receives input data that is on a similar scale. As a final step, we have labeled all the real and fake images to 0 and 1 using encoding to train and validate the model

* 1. **Model Preparation & Training**

The core of this process involves setting up and training a convolutional neural network to classify images as real or fake. The ResNet50 model, a pre-trained deep learning model provided by the Keras library, is employed here. This model is well-known for its effectiveness in image recognition tasks due to its deep architecture and the use of residual connections. In this implementation, the top layers of the model are replaced with new layers that are specifically tuned for the binary classification task (real vs. fake). The model is then compiled with a binary cross-entropy loss function and an Adam optimizer, which is a method for stochastic optimization. The training process involves using data augmentation techniques provided by the ImageDataGenerator class to improve the robustness of the model by artificially expanding the training dataset through random transformations.

* 1. **Model Validation & Prediction**

After training, the model's performance is validated using a separate set of images that were not included in the training dataset. The validation process involves preprocessing new images in the same way as the training images and then feeding them to the model to predict whether they are real or fake. The predictions are made using the trained model, and the results indicate the effectiveness of the model in distinguishing between real and synthetic imagery.

Overall, the script utilizes a combination of Python packages such as MTCNN for image extraction, TensorFlow and Keras for modeling and training, and NumPy for data manipulation. The methodological approach exemplifies a typical machine learning pipeline for image classification tasks, leveraging both deep learning techniques and traditional image processing methods.

1. **Analysis**

Our primary objective was to evaluate the efficacy of machine learning algorithms in the detection of deep fake images, with a focus on comparing various models' performance against a baseline. For our baseline model, we chose the ResNet50 architecture, a robust convolutional neural network that has demonstrated high effectiveness in various image recognition tasks. ResNet50 was selected due to its deep residual learning framework, which helps in training deeper networks by addressing the vanishing gradient problem. This makes it an ideal starting point for our comparisons.

The data handling and preparation phase was critical for ensuring the consistency and quality of the inputs fed into our machine learning models. Images, extracted as the first frame from video files using MTCNN, were batched and preprocessed uniformly to maintain consistency across the dataset. These images were resized to a standard dimension of 128x128 pixels, converted to RGB color space, and normalized using the preprocessing function specific to ResNet50. Data augmentation techniques such as rotation, width, height shifting, and horizontal flipping were applied using the ImageDataGenerator from TensorFlow's Keras API to enhance the model's generalization capabilities. For model training and validation, the dataset was split into 80% for training and 20% for testing, with images batched in sizes of 32 during model training to optimize memory usage and computational efficiency.

Hyperparameter tuning was an essential part of refining our models. For the ResNet50 model, the learning rate, number of epochs, and batch size were the primary hyperparameters adjusted. The learning rate was initially set to 0.0001, utilizing an Adam optimizer to facilitate efficient convergence. The model was trained over 10 epochs to balance between adequate learning and computational efficiency. The performance evaluation of our models was systematically carried out by comparing accuracy and loss metrics on the validation set. Binary cross-entropy was used as the loss function, which is suitable for the binary classification tasks at hand. Additionally, the accuracy metric provided a direct measure of model performance in correctly classifying the images as real or fake.

In conclusion, the analytical approach adopted in this research provided a detailed examination of the effectiveness of the baseline model and facilitated a structured comparison with other machine learning algorithms. The rigorous data preprocessing, systematic hyperparameter tuning and comprehensive performance evaluation established a robust framework for assessing the capabilities of deep learning models in the context of deep fake detection. This methodological rigor ensures that the findings are reliable and can serve as a benchmark for future research in the field.

1. **Machine Learning Model**

In this project, we are using a pre-trained machine learning model, ResNet50 which is a type of convolutional neural network (CNN). ResNet50 is well-known for its architecture that includes 50 layers deep, incorporating residual connections that make it possible to train such deep networks effectively. The residual connections help in addressing the vanishing gradient problem that typically occurs with deeper networks by allowing gradients to flow through a shortcut connection across layers.

We employed ResNet50 for the task of detecting deep fake images, which has been adapted to this binary classification task by modifying the top layers. Specifically, after the base ResNet50 layers, which are kept non-trainable to retain the learned features from pre-training on ImageNet, a GlobalAveragePooling2D layer is added to reduce the spatial dimensions of the output from the convolutional base. This is followed by a dense layer with 1024 units and 'relu' activation to introduce non-linearity and the capacity to learn complex patterns in the data. Finally, a dense output layer with 2 units and a 'softmax' activation function classifies the inputs into two categories: real or fake.

This model is compiled with the Adam optimizer and a binary cross-entropy loss function, which is typical for binary classification tasks. The adjustments to the original ResNet50 architecture and the specific compilation settings are tailored to optimize performance for the specific challenge of distinguishing real images from AI-generated fake images.

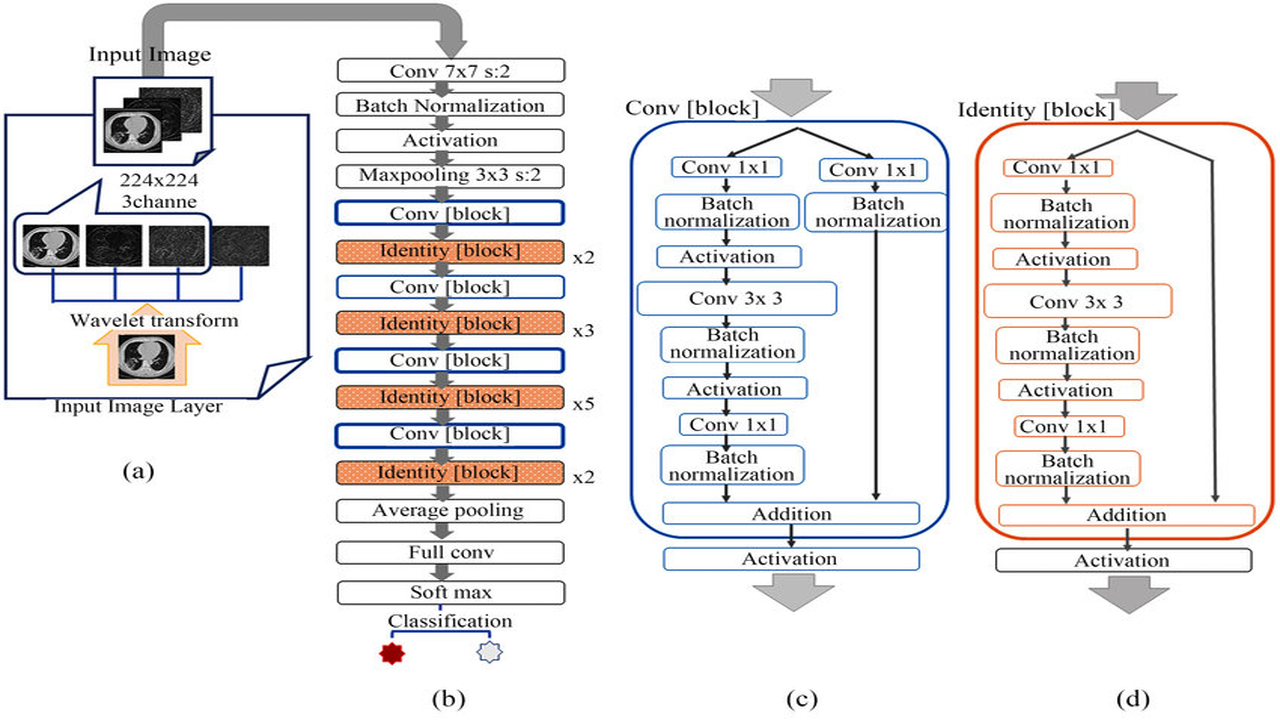
* 1. **ResNet50**

The ResNet50 model architecture is then outlined as follows:

An initial convolutional layer with a 7x7 kernel and a stride of 2, followed by batch normalization and an activation function (usually ReLU), is used to extract low-level features from the image.

This is followed by a max pooling layer with a 3x3 kernel and a stride of 2 to reduce the spatial dimensions.

A series of convolutional blocks and identity blocks follow, which form the core of the ResNet50 architecture. Convolutional blocks are used when the input and output dimensions do not match up, and they employ a convolutional layer at the shortcut connection. Identity blocks are used when the dimensions are the same and the shortcut can be directly added to the output of the block.



* + 1. **Convolutional Block**

In the convolutional block, the input goes through three convolutional layers:

The first and third convolutional layers use 1x1 kernels primarily for dimensionality reduction and expansion while keeping computational complexity in check.

The second layer uses a 3x3 kernel for extracting spatial features.

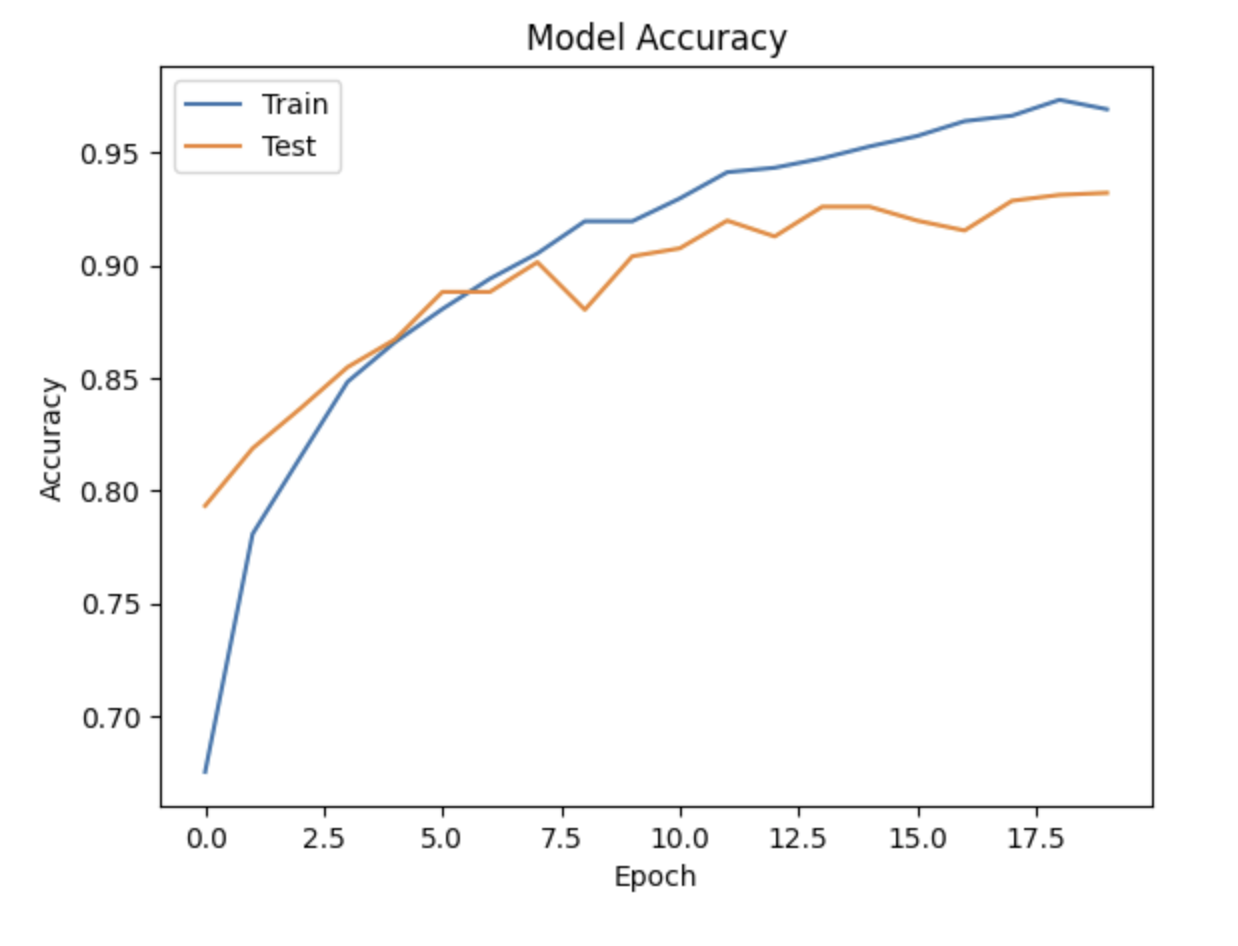
Each convolutional operation is followed by batch normalization and an activation function, with the output of the third layer being added to the original input after it has gone through its own 1x1 convolution.

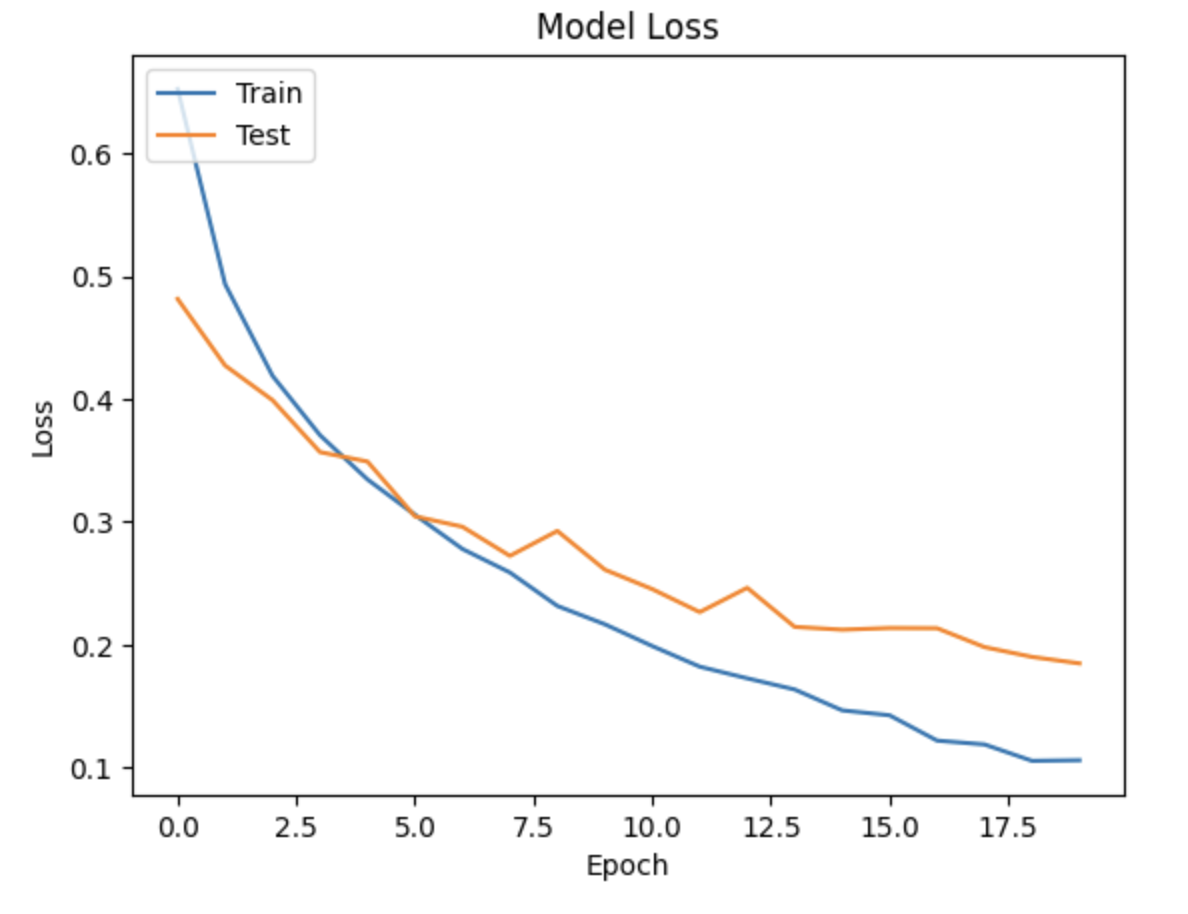
* + 1. **Identity Block**

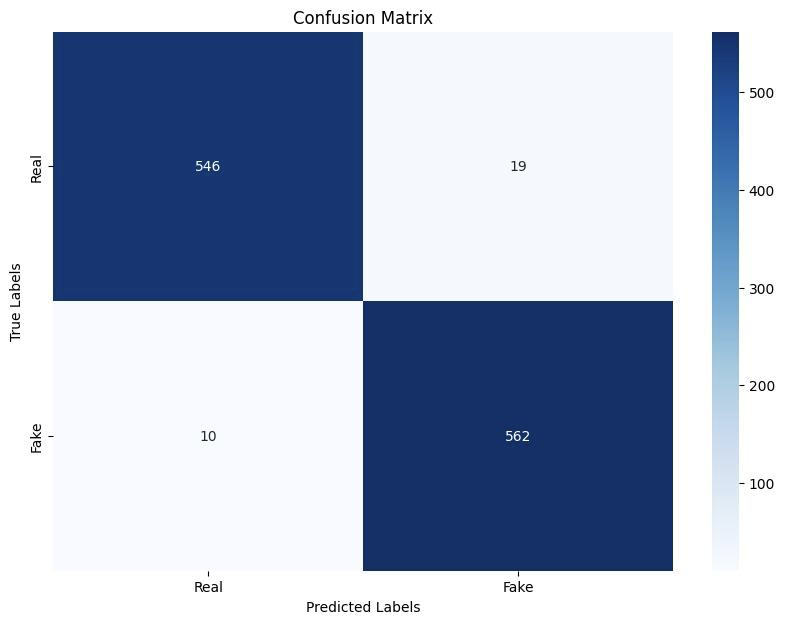
The identity block also has three convolutional layers with batch normalization and activation in between, similar to the convolutional block. However, here the input is added directly to the output of the third layer without any convolution, as the input and output dimensions are the same.

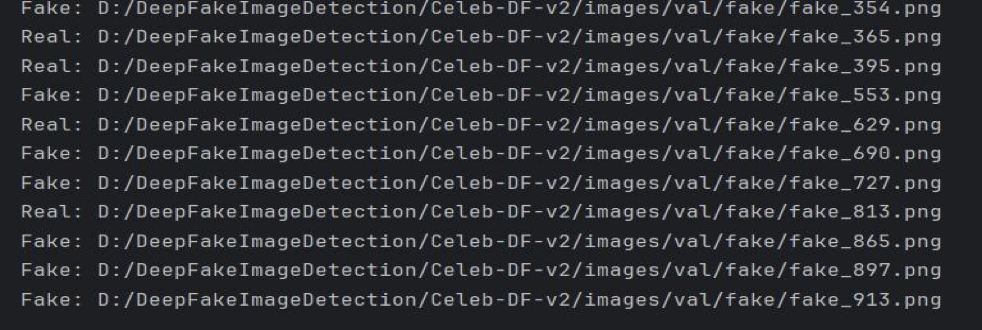
The final stages of the ResNet50 architecture involve an average pooling layer to reduce the feature map size. A fully connected layer (referred to as "full conv" in the diagram) to flatten the output and prepare it for classification. A softmax activation function is applied to the output layer to obtain the probabilities of each class (real or fake in this project). The classification result is determined by choosing the class with the highest probability.

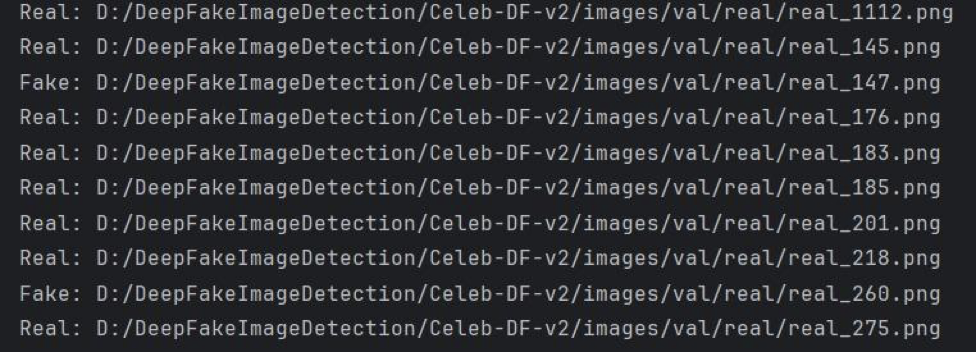
1. **Results**

****

****

****

****

****

1. **Conclusions**

In conclusion, this project presents a comprehensive approach to deep fake image detection using deep learning techniques, focusing on distinguishing between real and fake images with a specific emphasis on combating misinformation and promoting online authenticity.

Key components of the project include:

1. **Data Collection and Preprocessing:** Utilizing the Celeb-DF(v2) dataset, videos were processed to extract single frames using OpenCV, which were then resized and converted to JPEG format for efficient handling and training.
2. **Model Training and Evaluation:** The ResNet50 deep learning model was employed, leveraging its robust architecture for image classification. The model was fine-tuned with binary classification layers to distinguish between real and fake images.
3. **Data Augmentation and Hyperparameter Tuning:** Techniques like data augmentation through rotation and flipping were applied using TensorFlow's ImageDataGenerator to enhance model generalization. Hyperparameter tuning, including adjustments to learning rate and batch size, was conducted to optimize model performance.
4. **Performance Analysis:** The effectiveness of the trained model was evaluated using a separate validation set, assessing metrics such as accuracy and loss to gauge its ability to accurately classify real and synthetic imagery.

Overall, the project demonstrates a systematic approach to addressing the challenges posed by deep fakes, emphasizing the importance of leveraging advanced machine learning techniques for detecting and mitigating the spread of misleading content in digital media. Future directions for this work could involve exploring additional datasets to further enhance model robustness, implementing ensemble learning techniques for improved accuracy, and deploying the trained model within real-world applications to combat the proliferation of deep fake images across social media platforms. Through continued research and development in this area, advancements in deep fake detection will play a crucial role in upholding the integrity and trustworthiness of digital media content.

**Relevant Industries and Research Areas:**

Our research findings have broad implications across various industries and domains, including media and entertainment, cybersecurity, digital forensics, and journalism. Media platforms and content-sharing platforms can leverage advanced image-based deep fake detection technologies to safeguard their platforms against the proliferation of manipulated content and misinformation. Legal and regulatory frameworks may need to be adapted to address the ethical, legal, and societal implications of image-based deep fake technology, particularly concerning privacy, intellectual property rights, and digital trust.

In summary, while our current focus has been on image-based deep fake detection, the expansion into video-based content represents a natural progression in our efforts to combat digital disinformation and safeguard the integrity of multimedia content. Continued research, collaboration, and interdisciplinary efforts will be crucial in addressing the evolving challenges posed by image-based deep fake technology and ensuring the responsible use of AI in digital media analysis.

1. **About the Authors**

**Arbazuddin Mohammad** is a first-year graduate student specializing in Big Data Analytics at San Diego State University. His journey into the world of data began during his undergraduate studies in Mechanical Engineering in India, where he gained over 2 years of valuable experience as a Data Engineer at Tata Consultancy Services. It was during this time that Arbaz developed a strong passion for data and analytics, ultimately leading him to pursue a master's degree in Big Data Analytics. Arbaz's career aspirations are centered around roles in data science, data engineering, or related fields, with a particular interest in the dynamic gaming industry. Outside of academics, Arbaz is an avid fitness enthusiast. You'll often find him hitting the gym, and engaging in cardio activities like badminton and swimming, which he believes helps maintain a balanced and focused mind. In his leisure time, He enjoys staying updated with emerging data trends, honing his analytics skills, and looks forward to contributing his expertise to real-world projects as he progresses in his academic and professional journey.

Email: [amohammad1049@sdsu.edu](mailto:amohammad1049@sdsu.edu)

**Susheel Chebrolu** is pursuing his Master’s in Big Data Analytics at San Diego State University, after building a strong foundation in computer science from his undergraduate studies. Soon after that, With multiple years of professional experience as a data analyst at Infosys, he honed his skills across a diverse set of SAP Technologies and various data analysis tools like Power BI and also developed workflows between disparate sources, ensuring efficient data transfer. Presently his focus lies on gaining more knowledge on developing recommendation models using Machine Learning algorithms and also learning more about cloud platforms like AWS, and GCP to tackle challenges posed by big data. His major focus is to stay at the forefront of data science & analytics and shaping the future of information technology.

Email: [schebrolu6405@sdsu.edu](mailto:schebrolu6405@sdsu.edu)

**Jani Miya Shaik** is currently embarking on his journey towards a Master’s degree in Big Data Analytics at San Diego State University, where he is delving deep into the realm of cutting-edge technologies. With a rich background in software engineering, spanning over two years, He has honed his problem-solving prowess to perfection. Through immersive internships at Cognifront and Technocolabs, he has meticulously sharpened his skills in the dynamic field of data science. Now, He is channeling his passion toward the realms of Deep Learning, Computer Vision, and Natural Language Processing, where he seeks to push the boundaries of innovation and make impactful contributions to the ever-evolving landscape of technology.

Email: [jshaik2452@sdsu.edu](mailto:jshaik2452@sdsu.edu)

**Aditya Bhor** is a first-year graduate student in Big Data Analytics at San Diego State University. With a background in Computer Science, Aditya is driven by a passion for leveraging data to drive innovation and solve complex problems. They specialize in Python programming, SQL database management, machine learning, and data visualization. As a future professional in the field, Aditya aspires to contribute to cutting-edge projects in data science and analytics.

Email - [abhor5014@sdsu.edu](mailto:abhor5014@sdsu.edu)

**Aditya Desale** is pursuing his Master's in Big Data Analytics, after building a strong foundation in engineering from his undergraduate studies. Soon after that, with two years of professional experience as a data engineer at Tata Consultancy Services, he honed his skills across data engineering, data analytics, and data reporting. Presently his focus lies on delving deeper into the realm of data and learning advanced techniques in data science and machine learning, which can be implemented to help derive data-based decisions. His ultimate goal is to leverage his expertise in driving strategic decision-making and enabling organizations to unlock the true potential of their data assets.

Email - [adesale1896@sdsu.edu](mailto:adesale1896@sdsu.edu)

1. **References**
2. Matern F., Riess C., Stamminger M. Exploiting visual artifacts to expose deepfakes and face manipulations; Proceedings of the Winter Applications of Computer Vision Workshops (WACVW) 2019; Seoul, Korea. 27 October–2 November 2019; pp. 83–92. [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Proceedings+of+the+Winter+Applications+of+Computer+Vision+Workshops+(WACVW)+2019&title=Exploiting+visual+artifacts+to+expose+deepfakes+and+face+manipulations&author=F.+Matern&author=C.+Riess&author=M.+Stamminger&pages=83-92&)]
3. Mo H., Chen B., Luo W. Fake faces identification via convolutional neural network; Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security 2018; Innsbruck, Austria. 20–22 June 2018; pp. 43–47. [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Proceedings+of+the+6th+ACM+Workshop+on+Information+Hiding+and+Multimedia+Security+2018&title=Fake+faces+identification+via+convolutional+neural+network&author=H.+Mo&author=B.+Chen&author=W.+Luo&pages=43-47&)]
4. Huang G., Liu Z., Van Der Maaten L. Densely connected convolutional networks; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017; Honolulu, HI, USA. 21–26 July 2017; pp. 4700–4708. [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Proceedings+of+the+IEEE+Conference+on+Computer+Vision+and+Pattern+Recognition+2017&title=Densely+connected+convolutional+networks&author=G.+Huang&author=Z.+Liu&author=L.+Van+Der+Maaten&pages=4700-4708&)]
5. He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2016; Las Vegas, NV, USA. 26 June–1 July 2016; pp. 770–778. [[Google Scholar](https://scholar.google.com/scholar_lookup?journal=Proceedings+of+the+IEEE+Conference+on+Computer+Vision+and+Pattern+Recognition+2016&title=Deep+Residual+Learning+for+Image+Recognition&author=K.+He&author=X.+Zhang&author=S.+Ren&author=J.+Sun&pages=770-778&)]
6. Jeon, H.; Bang, Y.; Woo, S.S. FDFtNet: Facing Off Fake Images Using Fake Detection Fine-Tuning Network. *IFIP Adv. Inf. Commun. Technol.* 2020, *580*, 416–430. [[Google Scholar](https://scholar.google.com/scholar_lookup?title=FDFtNet:+Facing+Off+Fake+Images+Using+Fake+Detection+Fine-Tuning+Network&author=Jeon,+H.&author=Bang,+Y.&author=Woo,+S.S.&publication_year=2020&journal=IFIP+Adv.+Inf.+Commun.+Technol.&volume=580&pages=416%E2%80%93430&doi=10.1007/978-3-030-58201-2_28)] [[CrossRef](https://doi.org/10.1007/978-3-030-58201-2_28)]
7. Wang, S.-Y.; Wang, O.; Zhang, R.; Owens, A.; Efros, A.A. CNN-Generated Images are Surprisingly Easy to Spot… for Now. *arXiv* 2020, arXiv:1912.11035. [[Google Scholar](https://scholar.google.com/scholar_lookup?title=CNN-Generated+Images+are+Surprisingly+Easy+to+Spot%E2%80%A6+for+Now&author=Wang,+S.-Y.&author=Wang,+O.&author=Zhang,+R.&author=Owens,+A.&author=Efros,+A.A.&publication_year=2020&journal=arXiv)]
8. Vadsola, M. The Math behind GANs (Generative Adversarial Networks). Available online: <https://towardsdatascience.com/the-math-behind-gans-generative-adversarial-networks-3828f3469d9c> (accessed on 10 May 2020).
9. **Appendix**

**Github Link:** [**https://github.com/schebrolu6405/Deep\_fake\_video\_detection**](https://github.com/schebrolu6405/Deep_fake_video_detection)