Using Deep Learning to detect Deepfakes (GAN manipulated data)

Chandrahas Reddy Gurram. UbID : cgurram
Praneeth Posina. UbID : Pposina

***Abstract*—Deepfakes are AI-powered, realistic and deceptive videos and images that are generated by GANs (Generative Adversarial Networks) – advanced neural networks that focus on generating compelling content with the features of the source material. GANs are actually one AI algorithm where two neural networks compete each other and at the end produce new artificial data that can easily mimic the original one. Deepfakes have evolved in a sophisticated manner, to a state where it is difficult to distinguish between the real and fake, and this poses a real problem right from news dissemination, legal evidence, and personal safety. The detection of deepfakes is a must for ensuring the validity of information released through the media and defending citizens from being misinformed, defrauded, and influenced. As the creation of such synthetic images and videos increases, so does the need for proper identification and flagging of them as digital content holders go on to publish verified information.**

Keywords: Deepfakes , Generative Adversarial Networks (GAN's)

I Introduction

Generative Adversarial Networks (GANs) are a form of artificial intelligence algorithm structured around two competing neural networks: Generator and discriminator models are the core of this process. These correspond in a way such that a generator strives to produce the data which is almost undetectable, whereas a discriminator evaluates whether the data is authentic or generated by artificial method. The adversarial nature of GANs enables their high-quality fake outputs- they can cover all types of data from images and videos to text and sounds. GANs have evolved into a controversial tool, when it is employed to create the advanced machine learning techniques of deepfakes, where video and audio recordings are manipulated to be highly realistic.

The DeepFake Detection Challenge (DFDC) dataset is a prominent resource developed to train machine learning models to detect deepfakes and other synthetic media. This dataset comprises an extensive collection of videos that have been specifically altered using various methods, including GANs, to introduce realistic manipulations. These manipulations range from face swapping and lip syncing alterations to expression and entire facial transformation, aiming to simulate potential real-world deepfake scenarios. The DFDC dataset was compiled through a collaborative effort involving paid actors and advanced image processing techniques, ensuring a diverse and challenging environment for developing and testing deepfake detection models.

DeepFake Detection Challenge Dataset

II Literature Review

1) A Benchmark and Survey of Deepfake Technologies[1]

This section discusses the paper "Deepfake Generation and Detection: A Benchmark and Survey," which provides an overview of deepfake technologies, focusing on generative models and detection methods. The paper introduces a systematic classification of deepfake tasks, including face swapping, face reenactment, and talking face generation, which are driven by advances in generative adversarial networks (GANs), variational autoencoders (VAEs), and diffusion models. It highlights how these models have evolved to produce highly realistic and convincing deepfakes.

The survey examines various detection strategies that use both image-level and sequence-level analyses to identify manipulations. Technological usages include the effectiveness of different network architectures such as CNNs, the use of transformer models, and the integration of multimodal data sources to enhance detection accuracy. The challenges of adapting these technologies to the continuously improving quality of deepfakes are also discussed, emphasizing the need for ongoing research in developing robust and adaptable deepfake detectors.

2) State-of-the-art, Open Challenges, and Way Forward in Deepfake Detection [2]

The paper titled "Deepfakes Generation and Detection: State-of-the-art, open challenges, countermeasures, and way forward" offers an extensive review of current methodologies and tools for generating and detecting deepfakes. It outlines the utilization of deep learning techniques, particularly Generative Adversarial Networks (GANs), to create audio-visual manipulations. The paper elaborates on various detection approaches, including convolutional neural networks (CNNs) for spatial analysis and recurrent neural networks (RNNs) for temporal features, which have shown significant promise in identifying sophisticated deepfakes.

Furthermore, the paper addresses the critical challenges faced by deepfake detection technologies, such as the need for large, diverse datasets that mimic real-world scenarios and the development of methods that can detect deepfakes in different media types, including texts and audios. It suggests a multifaceted approach combining technological advancements, legal measures, and public awareness to combat the malicious use

of deepfakes. Future directions are proposed, focusing on enhancing the interpretability of detection models and increasing their resilience against adversarial attacks.

III Proposed Approach

We applied a deep learning to detect deepfakes and we started the research work with an extensive survey and categorization of existing methods. We refined our data by utilizing the latest preprocessing technologies comprising; Haar Cascades for face detection and Histogram of Oriented Gradients (HOG) to help in feature enhancement. Our model is composed of a convolutional neural network and bidirectional LSTM network to capture spatial and temporal characteristics of the data. We were learnt on a very carefully segmented dataset that is used to ensure a full evaluation over the training, validation and testing stage of the whole process. This strategic methodology ensured the emergence of a stable framework for the deepfake detection.

III-A Data Preprocessing

For our project, we aim to detect deep fakes (images manipulated by GAN's) using deep learning, training a dataset composed of both real and synthetic videos. Our huge dataset, totaling 477 GB which consists of 141,000 videos out of which 104,000 are labeled fake and remaining as real and, is divided into 10 GB chunks for efficient handling. We have utilized one chunk containing 828 videos—242 real and 586 fake—for training our model. This dataset selection although unbalanced like the original was optimised to enhance our model's capability to accurately distinguish between authentic and manipulated content.

sample fake data sample real data

Fig. 1: Sample from dataset

We chose only GPU during the dataset preprocessing and the neural network training to reduce the time of processing our algorithm. It was a necessity for us to apply GPU as it was the only way to get high computational rate, which enhanced the model performance significantly. Yet, taking into account GPUs as an alternative also meant we had to compromise on our dataset sizes; product of the memory limits of the GPUs. This was vital in giving us accurate reporting and high productivity which enables us to detect deepfakes easily when needed.

We broke down every video into videos of 20 frames each, each taken at various intervals of the video and making it a total of 16560 images for our model to train with.

Our project processing section utilizes the face-recognition library to isolate and detect faces in video frame by several advanced technical methods. This process is initiated by the implementation of Haar Cascade classifiers, a machine learning detection method for objects, which are defined through a cascade of predefined features. Haar Cascades are particularly effective for face detection because they can quickly identify facial structures at different scales, making them ideal for handling diverse video content where face sizes can vary significantly.

Once potential faces are detected using Haar Cascades, the system often applies more methods such as Histogram of Oriented Gradients (HOG) to further refine the face detection process. HOG feature descriptors help in capturing edge or gradient structure that is characteristic of a human face, regardless of lighting conditions. The orientation of these gradients is particularly useful as it helps form a detailed spatial pattern that enhances the ability of the model to distinguish between a face and non-face objects.

In general these methods aid a lot if you want to extract a particular face in a crowd. Such methods help us zoom in specifically or enlarge regions of most importance containing facial traits most useful for distinguishing genuine and fake videos. It results in a dataset that is aligned with what our deep learning model is supposed to struggle the most with, leading to decent detection accuracy.

Through this observation we found out that almost 20 videos were corrupted, and this in fact could decrease the quality of training of our deep learning model. To increase the accuracy and usefulness of our data we deleted those file that were problematic. Placing a major emphasis on the absence of errors was important to accomplish the primary goal of maximizing the model's correctness.

III-B Model training and Architechture

We divided our dataset into three parts: 80% is reserved for training, 15% for validation, and remaining 5% is dedicated to testing. Through the same preprocessing pipeline transformations across the training and validation sets have been maintained so as to ensure the uniformity of the data-processing. Transformations include using Pillow library in Python for converting the pictures to PIL format. Uniform size is achieved through resizing in the same manner. Lastly, the synthesized pictures are transformed into tensors that are set by the predefined means and standard deviations.

For model training and evaluation, we set up DataLoader instances for each dataset subset. These DataLoaders were configured with a batch size of four and utilized parallel processing with two workers to enhance data loading efficiency. The shuffling of data in training and validation phases ensures

better generalization and prevents the model from learning unintended patterns from the data sequence.

During the initial phase of our project we experimented with a Custom built CNN Model with Three convolutional layers with 3x3 kernels and ReLU activation. Each convolutional layer is followed by max pooling.This is followed by two fully connected layers and then Global Average Pooling. The output is a Binary classification for 2 classes Real and Fake. This gave us our preliminary results.

For the actual deep learning model used we've employed a neural network architecture suited for deep fake detection, using the ResNeXt101 for robust feature extraction. This variant uses split-transform-merge strategy, which utilizes parallel processing paths or 'cardinality' to learn more diversified features from input data. It consists of 101 layers, including numerous convolutional layers, ReLU activation functions, and pooling layers—far surpassing the 50 layers in ResNet-50. This increased depth allows for more detailed analysis, improving the detection of subtle video manipulations. It makes ResNeXt101 an effective backbone for our model.

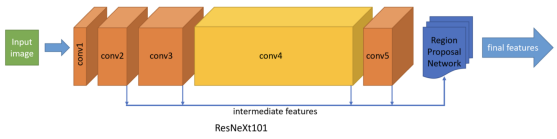


Fig. 2: ResNext-101

The core of our model features a two-layer bidirectional LSTM (Long Short-Term Memory) network, which processes video sequences in both forward and reverse directions. This capability is important for extracting complex temporal features and spotting inconsistencies in video dynamics, essential for detecting deepfakes.

We also improve the model's performance by using the ReLU activation function, which it is fitting better for images data as it introduces non-linearity, and the dropout layer of 0.5 just to prevent overfitting. Next comes a long short-term memory block, which is then followed by an adaptive average pooling layer which reduces the output size in a neat way, passing that to the next fully connected layer. This level is in charge of classifying the processed features and controlling the extent to which installation and real visual difference. This coombination of Deep CNNs, Bidirectional LSTM and selected regularization makes our model the best fit to correctly sense the features of the deepfake.

IV Results and Evaluation

IV-A Experimental setup

Pytorch is used to carry out the experiments in all the proposed work. This is a stem of its adaptability, advanced computing architecture, and great neural network implementation. Thus, PyTorch is extensively being used in deepfake detection applications. That being so, it greatly influences trainings and identification.

In this experiment a dataset consisting of videos of each frame 112x112 pixels and containing 20-frame sequences was employed instead. The dataset was divided into three subsets: Training, Validation, and Testing. The Training Set included 172 real videos and 490 fake videos, but the Validation Set contained only 33 real and 86 fake videos. The Test Set which was made up of 11 real and 31 fake videos included. The training utilized a batch size of 4, 10 or 20 epochs and Cross-Entropy Loss. Performance evaluation was conducted on the Test Set using accuracy, precision, recall, and F1-score. We aim to identify our image in the following manner that highlights the manipulated areas.

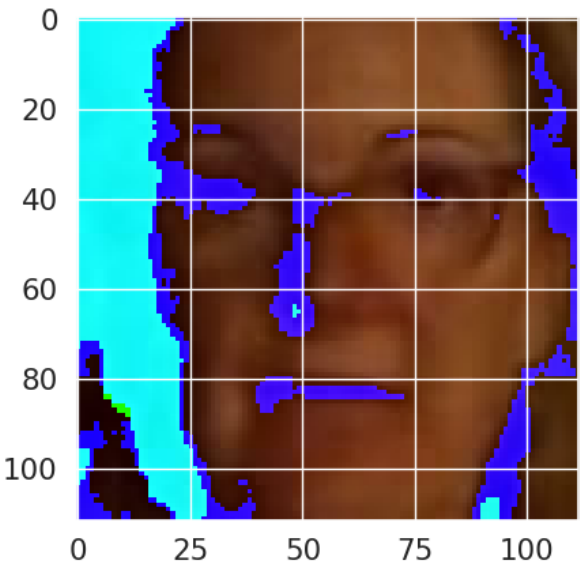


Fig. 3: Highlights of manipulated area.

IV-B Preliminary Results

CNN model achieved a training loss of 0.4095 and a validation loss of 0.4290. The training accuracy was measured at 80.43%, while the validation accuracy reached 84.06%. The network exhibited a 60% accuracy on the test videos. These results indicate that the model is performing well in terms of validation accuracy. However, the accuracy on the test videos is comparatively lower, suggesting some level of overfitting during the training phase. Further refinement and optimization of the model were necessary to improve its generalization performance which are performed in the latest model.

Metric	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
Value	0.3370	0.5671	85.33%	76.81%

TABLE I: Model Performance Metrics

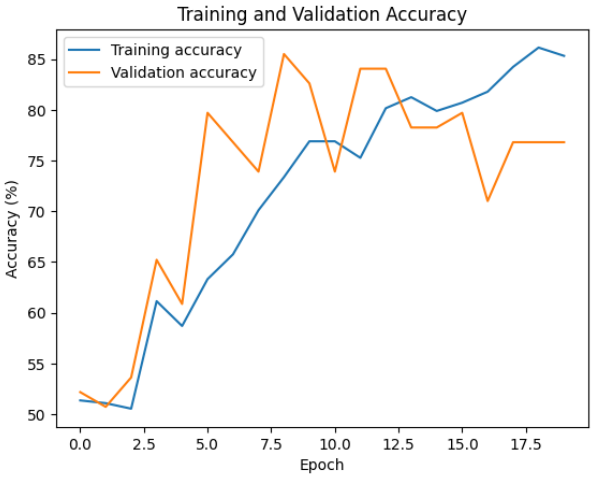


Fig. 4: Training and validation accuracy for CNN Model.

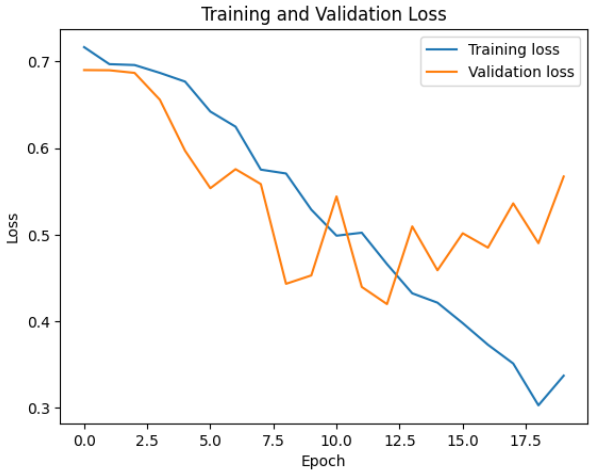


Fig. 5: Training and validation loss for CNN Model.

IV-C Secondary Results

We have tested our model for two scenarios where in one, we have a total of 828 videos with real : the fake ratio of words to pictures as in the original dataset is 172 : 490. To draw this situation as accurately as possible, ResNeXt101 model turned out to be good. While being trained, it reached a training loss of 0.0820, which being very little and provides an amazing training rate of 97.73%. The model continued showing good performance on testing samples, where the accuracy reached 86.29%. The shown result provide evidence that the model ResNeXt101 is well-suited for separating of real and fake videos. The model's excellent results on both instances or validation sets and also test set depict this model as one that has the capability to generalize for detection by deepfake applications.

The train and validation loss and accuracies for this model are :

And another scenario where we wanted to take the ideal dataset for the model with similar ratio of real : no longer I

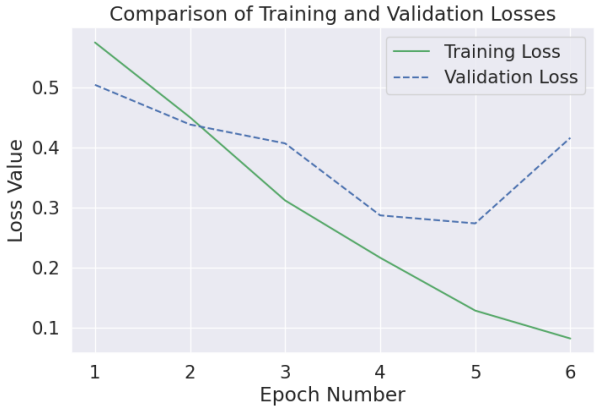


Fig. 6: ResNext_101 Training and validation accuracy for skew data.

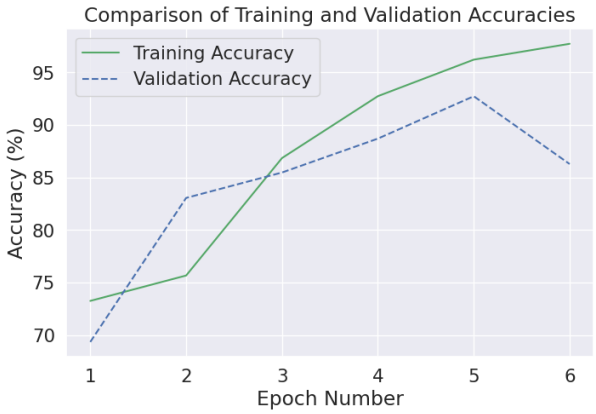


Fig. 7: ResNext_101 Training and validation loss for skew data.

on the planet fitted in, . In the case of this scenario, performing at it's best, the ResNeXt101 model. During an optimized training cycle, it got a very low training loss of 0.01811 and saw a training accuracy of 99.73%, which is quite high. On validation set of the model, the accuracy rate still remained stable at 91.3%, and loss remained low at 0.2371.

Thus, its given that ResNeXt101 is a very powerful architecture for the purpose of distinguishing real videos from the memorized ones. The model's high accuracy not always on the train and validation set, but also on the test set, indicate its capability of generalization well. This means that the model could be used in deepfake detection tasks.

Metric	Training Loss	Validation Loss	Training Accuracy	Validation Accuracy
Value	0.01811	0.2371	99.73%	91.30%

TABLE II: Performance Metrics of ResNeXt101 Model

We compared our models against two notable sota architectures: Xception and Efficient-Net view at two advanced CNNs. Xception weighs its performance on depthwise separable convolutions, making it a top choice for the complex image classification problems like detecting deepfakes. While EfficientNet realizes the outputs completion of accuracy and

efficiency through scalable architecture that can reduce depth, width or resolution of the network. Its gentle scaling is perfectly suited for task oriented image analysis, revealing made-up media as it was designed to.

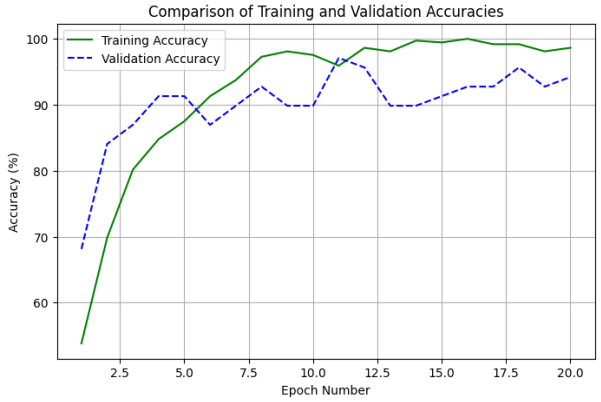


Fig. 8: Training and validation accuracy for ResNeXt-101 (processed data).

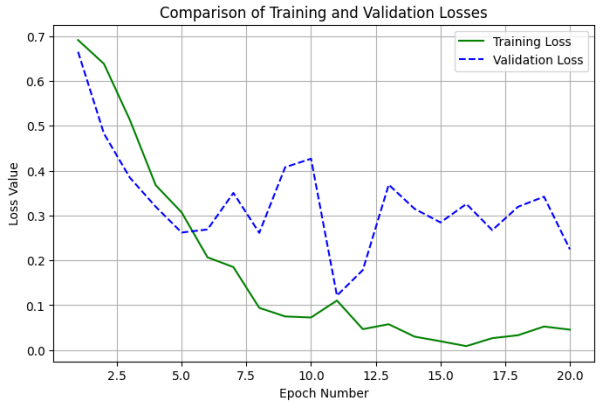
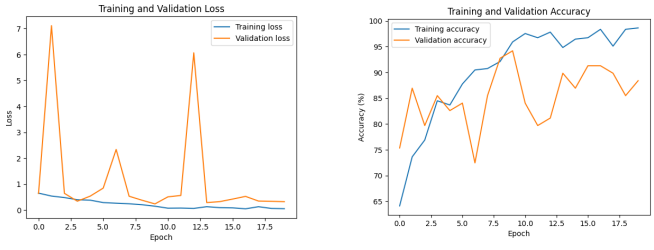


Fig. 9: Training and validation loss for ResNeXt-101 (processed data).

V Comaprision against SoTA models

To properly compare against our model results against the current SoTA models we ran the SoTA models on the same dataset as ours to properly form a comparision between them. First we ran the EfficientNet and the



EfficientNet Loss EfficientNet accuracy

Fig. 10: EfficientNet over the same chunk

The above are the Training and Validation loss of the EfficientNet model over the same chunk of data we used followed by the training and validation accuracy of the same.

The various metrics of the EfficientNet model over the validation dataset are shown in the table below.

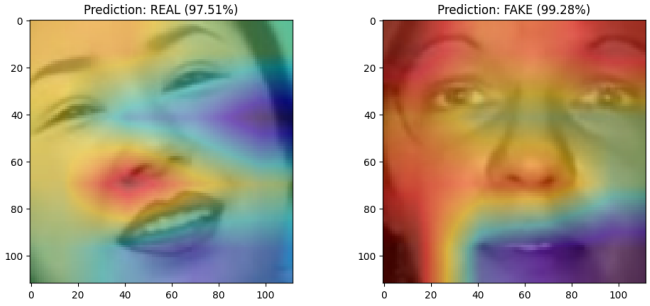
TABLE III: Network Performance Metrics of EfficientNet Model

Metric	Accuracy	Precision	Recall	F1 Score
Value	86%	0.917	0.846	0.880

EfficientNet is optimized by scaling across depth, width, and size independently with a specific formula providing a trade-off between accuracy and processing time, suitable for systems with a variety of computational limitations. In the case of ResNeXt, the strategy is ”split-transform-merge” like Inception networks, but with the method of grouped convolutions, that allows it to be suitable for work in the system with much model parallelism.

VI Prediction on unseen data.

The heat maps overlaid on the facial images provide insights into areas where the deepfake detection model focuses to determine authenticity. Common characteristics across these images include concentrated attention around key facial features such as the eyes, mouth, and nose—areas typically manipulated in deepfakes. The variations in color intensity within the heat maps, ranging from cool to warm tones, reflect the model’s confidence and suspicion levels, with warmer areas indicating higher likelihoods of manipulation. This visualization helps in understanding how deepfake detection models analyze facial cues and inconsistencies to classify images as real or fake.



Preditcted real image Preditcted real image

Fig. 11: Predictions with heatmaps

Therefore, by visual identification, not only do these assisted methods find potential tampering but the assistance also helps craft our deepfake detectors with high precision. Designing a system to distinguish between a high-probability scenario and a low-probability scenario provides an excellent input space for researchers and developers to work on the system in a more targeted manner, thereby laying the best possible groundwork for an efficient detection system.

Evaluation Metric	Average Training Loss	Average Training Accuracy	Average Validation Loss	Average Validation Accuracy	Average Precision	Average Recall	Average F1 Score
Value	0.2939321	87.084592%	0.3876297	84.274193%	0.8105040	0.8427419	0.8227764

TABLE IV: Evaluation Metrics for ResNeXt101 Model (Skewed_Data)

Model	Accuracy	Precision	Recall	F1 Score
Preliminary CNN Model	69%	68.75%	84.61%	75.86%
ResNeXt101 (processed_data)	86.95%	100%	76.92%	86.95%
ResNeXt101 (skewed_data)	87.08%	81.05%	84.27%	82.28%

TABLE V: Performance Evaluation Metrics of all the Models

Model	Validation Accuracy	Validation Loss	Test Accuracy	Test Loss
Xception	96.45%	0.1345	92.63%	0.2062
Efficient Net	95.62%	0.1624	90.86%	0.2172
ResNeXt101 (processed data)	91.30%	0.2371	86.96%	0.6619
ResNeXt101 (skewed_data)	84.27%	0.3876	86.29%	0.4265

TABLE VI: Raw Performance Comparison with SoTA Models

Model	Validation Accuracy	Validation Loss	Test Accuracy	Test Loss
Efficient Net	88.41%	0.3259	86%	0.421
ResNeXt101	94.20%	0.2245	86.96%	0.6716

TABLE VII: Performance Comparison with SoTA Models on same chunk of data.

Contribution

Chandrahas (50538624):

- Performed the pre-processing of the data and augmentation.
- Implemented the resnext proposed model.
- Performed the prediction against unseen data, plotting and visualised.

Praneeth (50540667):

- Implemented the preliminary model for the baseline.
- Implemented other SoTA models like efficient net.
- performed the model comparisons between the implemented and SoTA reference models.

References

1 - Deepfake Generation and Detection: A Benchmark and Survey

2 - State-of-the-art, Open Challenges, and Way Forward in Deepfake Detection

3 - DeepFake Detection Challenge Dataset

4 - Gupta, N.; Garg, H.; Agarwal, R. A robust framework for glaucoma detection using CLAHE and EfficientNet.

5 - Pan, D., et al., "Deepfake Detection through Deep Learning," IEEE BDCAT 2020.

6 - Malik, A., et al., "DeepFake Detection for Human Face Images and Videos: A Survey," IEEE Access, 2022.

7 - Access the journal article directly

8 - ResNeXt Model Overview on Papers with Code

9 - ResNeXt: Aggregated Residual Transformations for Deep Neural Networks