

Deepfake Detection Using Deep Learning

Kannika Lakshmi D G, Hemanth S Vas, Shiva Balan K, Sinchana K K, Sonika M B

Rajeev Institute of Technology, Hassan, 573201, Karnataka

Approved by AICTE, New Delhi and Affiliated to

Visvesvaraya Technological University, Belagavi, 590018, Karnataka

Email: kannika.dg@gmail.com, hemanthsrinivas18@gmail.com, laxmi60097@gmail.com,

sinchanak865@gmail.com, sonikathriveni8@gmail.com

Abstract - Deepfake technology has advanced rapidly with the development of deep learning and generative models, enabling the creation of highly realistic manipulated images and videos. While these techniques have useful applications, their misuse threatens privacy, security, and information credibility. This paper presents a deep learning-based framework for detecting deepfake facial media using the FaceForensics++ dataset. The proposed system performs preprocessing through frame extraction, resizing, normalization, and data augmentation to improve robustness. A Convolutional Neural Network (CNN) is employed to automatically extract spatial features and identify subtle artifacts such as unnatural textures, distorted facial boundaries, and lighting inconsistencies. The model performs binary classification to distinguish real and fake content. Experimental results demonstrate strong detection performance evaluated using accuracy, precision, recall, and F1-score. The findings indicate that CNN-based approaches are effective for deepfake detection and suitable for real-world media verification applications.

Keywords- DeepFake Detection, CNN, image processing, FaceForensics++, Keras, TensorFlow.

I. INTRODUCTION

In recent years, the fast growth of artificial intelligence and deep learning methods has led to the rise of highly realistic synthetic media, known as deepfakes. These are digitally altered images, audio, or videos where a person's likeness is convincingly changed. While this technology has exciting uses in entertainment, film production, and virtual reality, it also raises significant issues about misinformation, identity theft, and trust in digital media.

Deepfakes are usually created using deep learning models like Generative Adversarial Networks (GANs)

and autoencoders, which can learn and imitate human facial features with great accuracy. As these models keep improving, it becomes harder to tell the difference between real and altered content, even for trained professionals. This presents a major risk in areas like social media, politics, journalism, and cybersecurity, where authenticity matters.

A notable example of deepfake manipulation is the videos where Nicolas Cage's face was superimposed on characters in movies he never appeared in. These videos attracted a lot of attention on platforms like YouTube, showing how convincingly someone's identity can be changed. These examples underline both the technical skills of deepfake systems and the potential for misuse in spreading misleading or fake content.

The increasing availability of deepfake tools has made the problem worse, allowing even those with little expertise to create realistic fake media. This has created an urgent need for reliable methods to detect slight inconsistencies that arise during the manipulation process. These inconsistencies can include unusual facial textures, mismatched lighting, and temporal artifacts across video frames.

In this situation, deep learning-based detection methods, especially Convolutional Neural Networks (CNNs), have shown promising results in spotting fake media. By learning different features from large datasets, these models can effectively categorize content as real or fake. However, because deepfake creation techniques keep evolving, building robust and adaptable detection systems remains difficult.

This paper which aims to explore the effective methods for detecting deepfake content using machine learning techniques, contributing broader effort to ensure authenticity and trust in digital media.



Figure 1: Deepfake Video of Nicholas Cage

<https://www.youtube.com/watch?v=BU9YAHigNx8>

II. RELATED WORKS

In recent years, spotting deepfakes has really gotten a lot of attention because more and more fake media is popping up, and it can be used for bad things. Experts have come up with different ways to find them, from old-school image tricks to newer, smart computer programs.

At first, people trying to find deepfakes mostly looked for little visual mistakes or things that just didn't quite add up in the altered images. For example, they'd check for weird skin textures, eyes that blinked strangely, or lighting that didn't make sense. But these older methods, which relied on spotting specific features by hand, often couldn't catch the really good deepfakes made with newer, more sophisticated tools.

Then, when deep learning got better, something called Convolutional Neural Networks, or CNNs, became the go-to way to find deepfakes. These CNN models are really good at automatically figuring out the important visual details in pictures and videos. That makes them super effective at telling what's real and what's fake. Many studies have shown how well different CNN designs, like XceptionNet, ResNet, and MobileNet, can catch these fakes with a lot of accuracy.

The collections of data used to teach and test these deepfake detectors are super important. Take FaceForensics++ for example; it's one of the most common ones. It has tons of real and fake videos, made with all sorts of face-changing tricks. This allows researchers to really put different detection methods to the test, seeing how they perform in different situations and with varying video quality.

Besides just using CNNs, some new ideas combine different kinds of information – what things look like in one moment (spatial) and how they change over time (temporal). For example, models that mix CNNs with LSTMs can look at what's in each single video frame and also spot weird changes or inconsistencies as the video plays out. This makes them better at finding fakes. These combined methods are especially good for catching tiny, subtle movement errors that you might not even notice if you just looked at one picture from the video.

More recently, people have also been trying out even more advanced designs, like Vision Transformers and models that mix CNNs with transformers. The idea behind these is to pick up both the small details and the bigger picture, which helps them work better on new, unseen data and be tougher against different kinds of fakes. Plus, a method called 'transfer learning' has become pretty common. It helps make detection better while also cutting down on the time and computer power needed to train these systems.

Even with all this progress, finding deepfakes is still a tough nut to crack. A lot of the current models have trouble working well when they encounter data they haven't seen before or new ways deepfakes are made. On top of that, people are also coming up with new tricks, like 'adversarial techniques' and 'anti-forensic methods,' to intentionally fool detection systems. This means we constantly need to build smarter, more flexible models that can keep up.

So, all in all, even though deep learning has shown a lot of promise in spotting deepfakes, we still really need systems that can work better in general, do it faster, and also let us understand how they figure things out.

III. DATASETS

How well deepfake detection systems work really comes down to the quality and variety of the data they learn from. Luckily, over the past few years, a bunch of public datasets have popped up, helping researchers in this area. These datasets have both real media and fake ones, made with different methods.

Take FaceForensics++ for example; it's one of the most popular datasets out there. It comes with tons of real videos, plus faked versions made with techniques like DeepFakes, Face2Face, FaceSwap, and NeuralTextures. What's neat is that it also offers videos with different levels of compression, which lets

researchers see how well their models hold up in real-world situations, like when videos get crunched on social media.

Then there's the DeepFake Detection Challenge (DFDC) Dataset. Facebook put this one out, working with universities. It's packed with thousands of videos, showing all sorts of people, in various lighting, and different places. The whole point of this dataset is to mimic real-life situations, so it's good for training models to work well no matter the environment.

The Celeb-DF dataset is pretty special because its deepfake videos look really good, with fewer obvious flaws than older datasets. It focuses on celebrities and offers fakes that seem much more real, which naturally makes them harder to spot. Researchers often use this one to see how good their models are at catching advanced deepfake methods.

Even CASIA-WebFace, though it was first made for face recognition, finds its use in deepfake detection. People often use it to pre-train models. It has a huge collection of labeled face pictures, and this helps deep learning models get better at picking out important features.

Then there's DeeperForensics-1.0. This dataset throws in real-world issues like different lighting, compression, and noise. It's built specifically to check how tough detection systems are when faced with actual everyday problems.

So, for this project, we can use a mix of these datasets to train and test our model. Combining different datasets helps make the model more adaptable and ensures it does a good job handling various fakes and real-world situations.

IV. METHODOLOGY

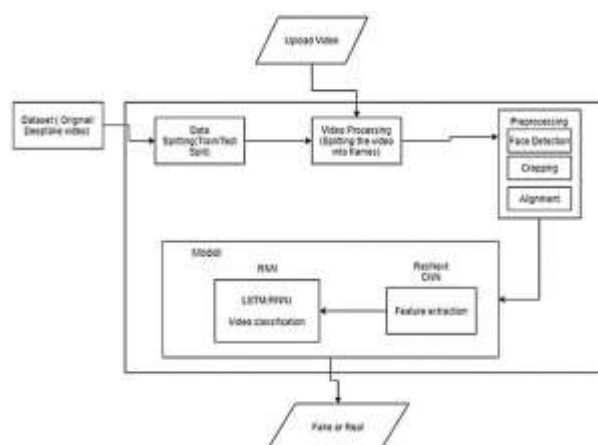


Figure 2: Overall process FlowChart

The idea is to spot deepfake pictures and videos by using a type of AI called deep learning, particularly something called Convolutional Neural Networks (or CNNs). We've set this up as a step-by-step process: it takes the raw information, pulls out important bits, and then decides if the content is real or fake. This works because deep learning models are good at automatically picking up on the complicated signs and oddities that show up in fake media.

First off, all the data we gather goes through some initial clean-up steps. This makes everything consistent and helps the model work better. If we're dealing with videos, we pull out individual frames (like still pictures) at steady intervals. This turns the video information into something the system can treat like a bunch of images. Then, we use face detection tools to find just the face, since that's usually where the deepfake changes happen. These face pictures are then made into a standard size, usually 224 by 224 pixels. This keeps things uniform throughout all the data. We also adjust the color values of the pixels so they fall within a set range. This makes the training process more stable and quicker. To make the model tougher and better at handling new, unfamiliar deepfakes, we use tricks like rotating, flipping, and changing the brightness of the images. This helps it learn more broadly.

Once the data is ready, it goes into the Convolutional Neural Network. Here, it picks out key features and decides what category they belong to. Inside the CNN, there are several layers that specialize in learning different kinds of details from the images. Think of it learning everything from simple edges and textures to more complex things like overall facial shapes. Special functions, like something called ReLU, help the model see more intricate connections in the data, not just straight-line ones. There are also 'pooling' layers that essentially shrink down these detail maps. This makes the computing less heavy without losing the crucial information. Finally, these learned features go through other layers that make the ultimate decision about whether something is real or fake.

To make it work even better and train faster, we can use a trick called 'transfer learning.' This means taking models that have already been trained on huge amounts of data, like XceptionNet or ResNet. Doing this really boosts how accurate the system is, especially when we don't have a ton of our own data to start with.

When we train the model, it learns to tell real from fake by looking at examples that are already labeled. We use something called a 'loss function' (like binary cross-

entropy) to see how far off its guesses are from the actual labels. Then, other tools, like the Adam algorithm, help it adjust its internal settings to get better. We usually split our data into a 'training' part and a 'validation' part. This lets us keep an eye on how well the model is doing as it learns and stops it from becoming too specialized to just the training examples.

Finally, the trained model looks at each new image or video and gives it a score, basically telling us how likely it thinks it is a deepfake. It uses a 'sigmoid' function at the very end to give a number between 0 and 1. If the number is close to 1, it's probably fake; if it's close to 0, it's likely real. We check how well the model works using common measurements like accuracy, precision, recall, and F1-score. These give us a full picture of how good it really is.

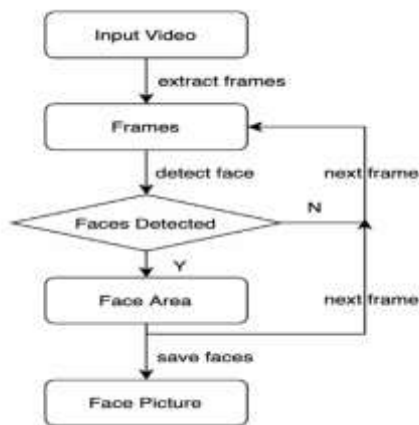


Figure 3: Pre-processed Flow Chart

On the left side of the picture, you'll see a deepfake image. This is where someone's face has been digitally changed and put onto another person's body. These images often look pretty real, but if you look closely, you might spot small errors in the face's shape, its skin, or how it's lit. The right side, on the other hand, shows the same image after it's been prepared for a computer. Here, the system finds the face, cuts it out, and then adjusts it to be consistent for the computer model to process. When we do this preparation, the face is separated from the background, made into a standard size, and then its lighting and pixel intensity are tweaked. This whole process gets rid of any extra information and makes sure the computer model focuses only on the most important facial details. After all that, this prepared image is fed into a Convolutional Neural Network (CNN), which then works to pull out specific features and sort things out.



Figure 4: Extracted Frames (Left) & Processed Images (Right)

4.1 DEEP LEARNING MODELS

Deep learning models have become the main way modern deepfake detection systems work. That's because they can automatically figure out the complicated details from images and videos. Unlike older methods where people had to manually pick out specific clues, deep learning can automatically spot things like small mistakes, odd textures, or unusual facial patterns in fake videos or pictures. These models are really good at dealing with huge amounts of data, and they can even learn to handle new ways deepfakes are made.

For deepfake detection, people have tried out several kinds of deep learning models. Each one has its own strengths, depending on the data it's working with and what problem it's trying to solve.

4.1.1 Convolutional Neural Networks (CNNs):

When it comes to spotting deepfakes in images, Convolutional Neural Networks, or CNNs, are what people use most often. These models are built to handle image data, which is like a grid, and they're really good at picking out things like edges, textures, and even how faces are put together. A CNN is made up of different parts—convolutional layers, pooling layers, and fully connected layers—all working together to pull out important details and then sort them.

4.1.2 Recurrent Neural Networks (RNNs) and LSTM:

Recurrent Neural Networks, especially something called Long Short-Term Memory (LSTM) networks, are good for looking at video data because they can keep track of how things change over time. Deepfake videos sometimes have odd little changes from one frame to the next, so LSTMs are useful for checking things like facial movements and expressions as they happen. You often see these models teamed up with CNNs to create more complex systems.

4.1.3 Transfer Learning Models:

Transfer learning is basically about taking models that have already been taught using huge amounts of data—like XceptionNet, ResNet, and VGG—and then adjusting them a bit for spotting deepfakes. Doing this saves a lot of training time and often makes them work better. They're particularly handy if you don't have a ton of your own data to start with.

4.1.3 Autoencoders:

Autoencoders are a type of neural network that learn on their own, often by trying to rebuild the data they're given. For deepfake detection, they can learn what real faces look like, and then point out anything weird when a fake image doesn't match those learned patterns. It's also worth noting that these are often used to create deepfakes in the first place.

4.1.4 Generative Adversarial Networks (GANs):

GANs, or Generative Adversarial Networks, are made up of two networks that basically compete with each other: one that creates things (the generator) and one that judges them (the discriminator). Even though GANs are mostly known for making deepfakes, the judging part—the discriminator—can actually be tweaked to help spot them instead. Knowing what kind of flaws GANs tend to leave behind can really help us build better ways to catch deepfakes.

4.1.6 Transformer-Based Models:

Models based on 'Transformers,' like Vision Transformers (or ViTs), use something called 'attention' to look at how different parts of an image relate to each other overall. Unlike CNNs that might just focus on small bits, transformers take in the whole picture, which makes them really good at finding tricky, complicated fakes. The downside is they need a lot of data and powerful computers to work well.

4.1.7 Image processing:

Image processing is super important for deepfake detection. It's all about getting the input images ready and making them better before we feed them to our deep learning models. Since deepfake changes mostly happen on faces, image processing techniques help us zero in on those areas, clean them up, and make them consistent. This makes it easier for the models to spot fakes.

Basically, these methods cut down on clutter, bring out key details, and make sure all the images look similar enough for the deep learning models to do a better job of picking out important stuff. In our system, we use a few different image processing tricks to make sure the

data we put in is good and useful.

A. Face Detection and Cropping: First up is face detection. This is where we find the face in an image or video frame and pull it out. Tools like Haar Cascade and MTCNN are often used because they're pretty good at finding faces exactly where they are. By cutting out just the face, we get rid of anything in the background that isn't important, letting the model concentrate only on what matters.

B. Image Resizing: Once we've got the faces, we make all the images the same size—say, 224 by 224 pixels. This keeps everything uniform across our collection of images. It's a crucial step because deep learning models work best, and fastest, when all their input images are the exact same size.

C. Normalization: Normalization means we adjust the brightness and color values of the pixels so they all fall within a certain range, usually from 0 to 1. This helps the model learn more steadily and reach its goal faster when we're training it.

D. Noise Reduction: We also use things like Gaussian filtering to get rid of any unwanted fuzziness or weird marks in the image. This makes the facial features clearer and means the model is less likely to pick out the wrong things.

E. Image Enhancement: Then there's image enhancement, where we might tweak things like contrast or equalize the histogram. The goal here is to make important features stand out more. These methods can really help us see tiny little oddities in deepfake images, like strange lighting or textures that just don't look right.

F. Data Augmentation: Finally, we do something called data augmentation. This involves making a bunch of slightly different versions of our images—by rotating them, flipping them, zooming in or out, or changing the brightness. We do this to make our dataset more diverse. It helps the model learn to recognize fakes even when they're presented in slightly different ways, and stops it from getting too fixated on only the exact images it saw during training.

V. LIMITATIONS

Even though this deepfake detection system looks promising, there are still some limits we need to talk about. A big problem is that the model really relies on how good and varied its training data is. If the training data doesn't properly cover different kinds of deepfakes, lighting, and face types, the model might struggle to recognize new, unfamiliar deepfakes.

Another big issue is that it's hard to catch really advanced deepfakes made with the latest methods. Since deepfake makers keep getting better, their creations look more and more real, with hardly any tells. That makes it harder for even our advanced detection systems to catch them, creating a constant race between those making deepfakes and those trying to find them.

This system also needs a lot of computer power to get trained and then to actually work. Big learning models, especially complex ones or those that adapt what they've learned elsewhere, need powerful computers and lots of memory. That could be a problem in places with limited resources or for things that need to happen instantly.

Plus, the model mainly checks visual stuff and might not catch odd things that happen over time in a video. If it doesn't analyze things frame by frame, some deepfake videos that look fine in each individual shot could get past it. This means the system isn't as good when dealing with fast-moving video situations.

It also seems quite sensitive to the quality of the image. Things like a compressed image, noise, low resolution, or bad lighting can really hurt how well it performs. In the real world, where media often gets compressed or tweaked when it's sent, the system might not be as accurate.

Finally, the system might sometimes make mistakes, calling real things fake or missing actual fakes, which affects how much you can trust it. If it wrongly labels real content as fake, people might lose faith in it. But if it fails to spot manipulated content, then misinformation can spread. Because of this, more work is needed to make the system stronger and more reliable.

VI. RESULTS AND DISCUSSIONS

Even though this deepfake detection system looks promising, there are still some limits we need to talk about. A big problem is that the model really relies on how good and varied its training data is. If the training data doesn't properly cover different kinds of deepfakes, lighting, and face types, the model might struggle to recognize new, unfamiliar deepfakes. Another big issue is that it's hard to catch really advanced deepfakes made with the latest methods.

Since deepfake makers keep getting better, their

$$\begin{aligned} \text{Real Video Detection Accuracy} &= \frac{TN}{TN + FN} \times 100\% \\ \text{Fake Video Detection Accuracy} &= \frac{TP}{TP + FP} \times 100\% \\ \text{Overall Detection Accuracy} &= \frac{TN + TP}{TP + TN + FP + FN} \times 100\% \end{aligned}$$

creations look more and more real, with hardly any tells. That makes it harder for even our advanced detection systems to catch them, creating a constant race between those making deepfakes and those trying to find them. This system also needs a lot of computer power to get trained and then to actually work.

We checked how well the model performed using True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) numbers. The "real video detection accuracy" tells us how well it correctly spots genuine content. The "fake video detection accuracy" tells us how well it finds manipulated stuff. The "overall detection accuracy" combines how well it did on both real and fake examples. These numbers give us a full picture of how reliable the system is.

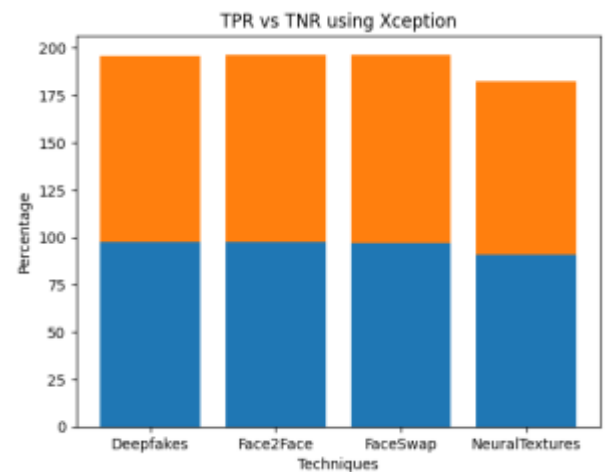


Figure 5: TPR vs TNR using Xception

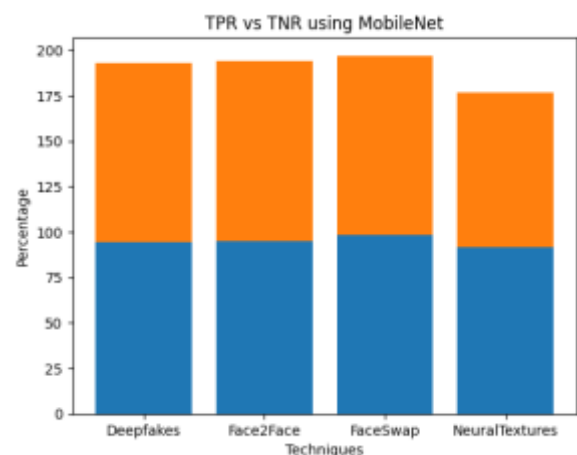


Figure 6: TPR vs TNR using MobileNet

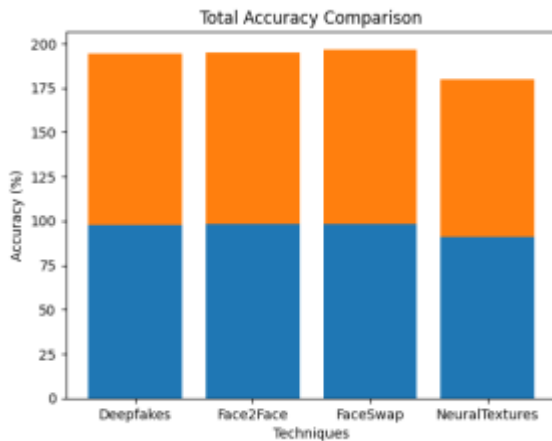


Figure 7: Total Accuracy Comparison

The charts show how the Xception and MobileNet systems performed with different deepfake types, like Deepfakes, Face2Face, FaceSwap, and NeuralTextures. You can see that the Xception system almost always had a better True Positive Rate (meaning it found real things correctly) and True Negative Rate (meaning it correctly identified fakes) than MobileNet.

This tells us Xception is better at correctly spotting both real and fake images. Also, when we looked at the overall accuracy, Xception beat MobileNet in every category, especially with tricky fakes like FaceSwap.

But both systems struggled a bit with NeuralTextures, which probably means this kind of fake is harder to spot because it looks so real. So, generally, the results show that more complex computer setups like Xception are better at pulling out important details, which helps them do a better job at finding deepfakes.

Also, the good balance between precision and recall means the system isn't favoring one type of image over another. It reliably sorts both real and fake pictures. This kind of balance is really important in real-world situations, because getting either a false positive or a false negative can cause big problems. All in all, what we've talked about shows that even though our system works well and is dependable, finding deepfakes is still a tough and changing problem.

In the future, we should aim to make it even stronger, more adaptable, and better at handling new ways of making fakes.

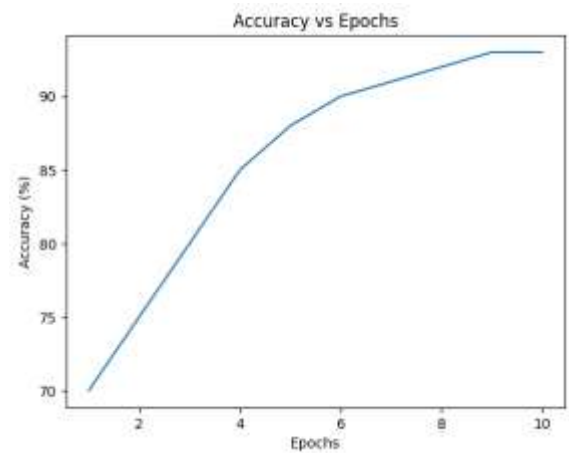


Figure 8: Accuracy vs Epochs

On top of that, while the system is quite good at catching common deepfake methods, it struggles with really advanced fakes that look almost exactly like real faces. This shows we need to keep updating the system and add smarter ways to analyze things, like looking at changes over time or combining different kinds of information, to make it even more accurate.

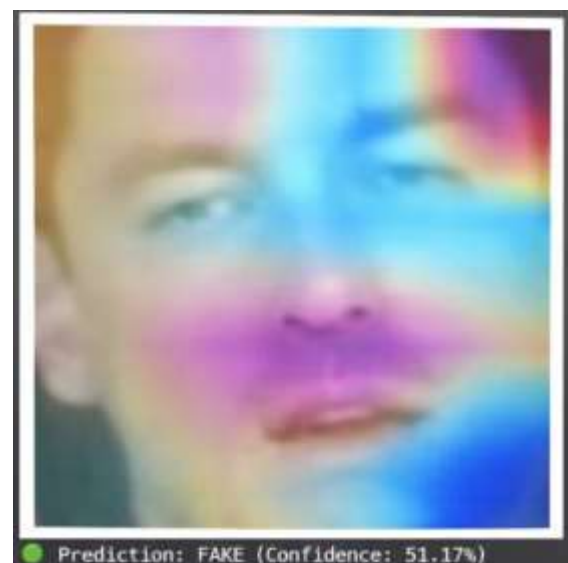


Figure 9: Heatmap visualization of deepfake detection showing important facial regions influencing the model's prediction.

Figure [9] shows what our deepfake detection system puts out. You'll see a heatmap laid over the face we put in. This heatmap points out the parts of the face that really help the model decide. Hotter colors mean those spots are more important, and cooler colors mean they're less important. You can see that the model mostly looks at key facial features like the eyes, nose, and mouth, which are often changed when someone makes a deepfake.

Besides just showing you the picture, the system also tells you its guess and how sure it is. For instance, here,

the model says the image is "FAKE" with about 51% certainty, meaning it's somewhat sure of its choice. This kind of output helps us understand not just what the model decided, but also why it decided that way. It makes the model easier to understand.

The heatmap makes things clearer, showing us how the deep learning model spots weird bits in different parts of the face. This way of looking at things is really helpful for checking if the model is working right and making sure it's basing its decisions on important features, not just random patterns.

VII. CONCLUSIONS

We came up with a way to detect deepfakes using deep learning. Our method combines Convolutional Neural Networks (CNNs) with some useful image processing steps. This system first pulls out important details from faces. It does this by finding the faces, making them uniform (normalizing), and adjusting their size. Then, a specially trained deep learning model checks what they are. Using transfer learning also makes the model work even better and cuts down on how much effort it takes to train it.

Our tests showed that this system is really good at telling the difference between real and fake media, and it worked well on different sets of data. We used measurements like accuracy, precision, recall, and F1-score, and these all confirm that the model is dependable and does what it's supposed to. Plus, we used tools like heatmaps to see how the model makes its decisions. This helps us understand it better and trust it more.

But our study also pointed out some issues. For example, it didn't work as well on images that were heavily compressed, and it struggled to catch some of the more advanced deepfake methods. Even with these shortcomings, though, our method turned out to be a strong and effective way to find deepfakes.

Since fake media is becoming more and more common, we really need good detection systems to keep digital content authentic and trustworthy. Our method helps with this goal, and it lays the groundwork for more research to make detection even more accurate and useful in everyday situations.

VIII. REFERENCES

- [1] A. Rössler and colleagues presented their work "FaceForensics++: Learning to Detect Manipulated Facial Images" at the IEEE International Conference on Computer Vision (ICCV) in 2019.
- [2] B. Dolhansky et al. put out "The DeepFake Detection Challenge (DFDC) Dataset" as an arXiv preprint, reference arXiv:2006.07397, in 2020.
- [3] Y. Li, M.-C. Chang, and S. Lyu shared their paper "In Ictu Oculi: Exposing AI Generated Fake Face Videos by Detecting Eye Blinking" at the IEEE International Workshop on Information Forensics and Security (WIFS) in 2018.
- [4] H. Nguyen, J. Yamagishi, and I. Echizen wrote "Capsule-Forensics: Using Capsule Networks to Detect Forged Images and Videos," which appeared in ICASSP, 2019.
- [5] F. Chollet introduced "Xception: Deep Learning with Depthwise Separable Convolutions" at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) in 2017.
- [6] K. He, X. Zhang, S. Ren, and J. Sun published "Deep Residual Learning for Image Recognition" in CVPR, 2016.
- [7] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen presented "MesoNet: A Compact Facial Video Forgery Detection Network" at WIFS in 2018.
- [8] T. Karras and others talked about "Analyzing and Improving the Image Quality of StyleGAN" at CVPR in 2020.
- [9] I. Goodfellow et al. introduced "Generative Adversarial Nets" at the Advances in Neural Information Processing Systems (NeurIPS) conference in 2014.
- [10] D. Kingma and J. Ba described "Adam: A Method for Stochastic Optimization" in an arXiv preprint, reference arXiv:1412.6980, back in 2014.
- [11] K. Simonyan and A. Zisserman shared "Very Deep Convolutional Networks for Large-Scale Image Recognition" as an arXiv preprint, reference arXiv:1409.1556, in 2014.
- [12] J. Deng et al. presented "ImageNet: A Large-Scale Hierarchical Image Database" at CVPR in 2009.
- [13] D. Güera and E. J. Delp published "Deepfake Video Detection Using Recurrent Neural Networks" at the IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS) in 2018.
- [14] H. Li and S. Lyu wrote about "Exposing DeepFake Videos By Detecting Face Warping Artifacts" for the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) in 2019.
- [15] X. Yang, Y. Li, and S. Lyu discussed "Exposing Deep Fakes Using Inconsistent Head Poses" at ICASSP in 2019.