Convolutional Neural Network for DeepFake Detection

Abhinav Madabhushi, Pierce Ohlmeyer-Dawson

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1 Introduction

DeepFake classification is the process of identifying whether an image is real or AI-generated. According to some statistics, almost 0.1 percent of the images online are already fake, and if this goes on, the majority of the images on the internet will become fake. To moderate the fake images on the internet, it is important to have a model to classify whether a given image is fake or not, a task that is becoming more complicated by the day. Using technologies and generative models, it is extremely easy to produce an image similar to real life, and the human eye will not be able to tell the difference. As the models for generating AI images are growing, it is important to have content moderation, especially on social media websites, that recognizes and deletes these AI-generated images. Other applications of deepfake detection include news and media verification, law enforcement and digital forensics, identity verification, and cybersecurity and fraud detection, amongst many other applications.

There is also ongoing research that with repeated use of synthetic data (AI-generated images) in models, there is eventually going to be an irrecoverable model collapse, which would mean that models such as ChatGPT that have been trained for several years to reach their current state will have all its progress removed. This is due to the fact that every time synthetic data is generated, there are small perturbations to the image, and over time, the image degrades to an irrecoverable state. A lot of solutions have been proposed to mitigate this problem, out of which one solution is to create accurate deepfake detection models. Advanced convolutional neural network models are required for deepfake classification, especially for the fake images currently being made.

2 Background

2.1 Model Overview

Input: $x \in \mathbb{R}^{H \times W \times C}$ (An RGB image with height H, width W, and $C = 3$)
Output: $y \in \mathbb{R}$ (0 or 1 output for binary classification)

2.2 CNN Forward Propagation

Convolution Layer: For each output channel c' and spatial position (x, y), compute

$$z_{c'}(x,y) = h\left(\sum_{c=1}^{C}\sum_{u=0}^{k-1}\sum_{v=0}^{k-1} x_c(x+u, y+v)\, w_{c',c}(u,v) + b_{c'}\right),$$

where $h(\cdot)$ is a nonlinear activation function such as ReLU.

Pooling Layer: Reduce the dimensions of the data by taking:

$$p_{c'}(X,Y) = \begin{cases} \max\limits_{0\le u,v<p} z_{c'}(pX+u, pY+v) & \text{(max pooling)} \\ \dfrac{1}{p^2}\displaystyle\sum_{0\le u,v<p} z_{c'}(pX+u, pY+v) & \text{(average pooling)} \end{cases}$$

Fully Connected Layer: Flatten the pooled features into a vector p_d and compute:

$$z_j^{(\text{fc})} = h\left(\sum_{d=1}^{D} w_{jd}^{(\text{fc})} p_d + b_j^{(\text{fc})}\right).$$

Output Layer:

$$y = \sigma\left(\sum_{j=1}^{M} w_j^{(\text{out})} z_j^{(\text{fc})} + b^{(\text{out})}\right),$$

where σ is the sigmoid activation function for binary classification.

2.3 CNN Backpropagation

Error Computation: At the output layer, the error is given by

$$\delta_k^{(\text{out})} = y_k - t_k,$$

where t_k is the target.

Gradient Computation:

Fully Connected Layer:

$$\delta_j^{(\text{fc})} = h'\left(\sum_{d=1}^{D} w_{jd}^{(\text{fc})} p_d + b_j^{(\text{fc})}\right) \sum_k w_{kj}^{(\text{out})}\, \delta_k^{(\text{out})},$$

with the gradient of the weight computed as:

$$\frac{\partial E}{\partial w_{jd}^{(\text{fc})}} = \delta_j^{(\text{fc})}\, p_d.$$

Pooling Layer: The gradient for the pooling layer is given by:

$$\tilde{\delta}_{c'}(pX+u, pY+v) = \begin{cases} \delta_{c'}^{(p)}(X,Y), & \text{if using max pooling and } z_{c'}(pX+u,pY+v) = p_{c'}(X,Y), \\ 0, & \text{if using max pooling and } z_{c'}(pX+u,pY+v) \neq p_{c'}(X,Y), \\ \dfrac{1}{p^2}\,\delta_{c'}^{(p)}(X,Y), & \text{if using average pooling.} \end{cases}$$

Convolutional Layer: Backpropagate the gradient as:

$$\delta_{c'}(x,y) = h'\left(\sum_{c=1}^{C}\sum_{u=0}^{k-1}\sum_{v=0}^{k-1} x_c(x+u,y+v)\, w_{c',c}(u,v) + b_{c'}\right)\tilde{\delta}_{c'}(x,y),$$

and compute the gradient of the weight as:

$$\frac{\partial E}{\partial w_{c',c}(u,v)} = \sum_{x,y} \delta_{c'}(x,y)\, x_c(x+u,y+v).$$

Weight update: Update weights using gradient descent

$$w^{(\text{new})} = w^{(\text{old})} - \eta\,\frac{\partial E}{\partial w},$$

where η is the learning rate.

Similarly, back propagation is also performed on the bias term.

2.4 Conclusion

Our CNN architecture integrates multiple convolutional and pooling layers for feature extraction and dimensionality reduction, followed by fully connected layers, and concludes with an output layer using a sigmoid activation for binary classification. [1]

3 Dataset

The dataset [2] used in this project is a Kaggle dataset containing 60,000 real images from the CIFAR-10 dataset and 60,000 AI-generated synthetic images generated by Stable Diffusion version 1.4, a text-to-image generation model. The images are equivalent to those in the CIFAR-10 dataset, making them difficult to distinguish as Real or Synthetic. The real images from the CIFAR-10 dataset are widely used in academia and research, and the dataset is publicly released under specific terms that permit free use for non-commercial research. However, the fake images obtained from the stable diffusion model were trained on images scrapped from the internet, which could spark ethical issues of data privacy. Still, we are only using the output of that model for our model, which is not a major concern. There might be certain licensing and trademark issues with synthetic images replicating logos or distinctive product designs. There also might be bias in both the CIFAR-10 dataset and the stable diffusion data, as the dataset might contain only certain breeds of dogs and cats, for example, which might lead to the model not recognizing other breeds as fake or real. All these privacy, bias, and ethical concerns were taken into account before creating our CNN model.

Several preprocessing steps were taken for the dataset. Firstly, the data transformations and augmentations were done. Images in the training dataset had random horizontal flips with a 50 percent chance of getting flipped. Along with this, there were small random rotations of approximately 10 degrees and random alterations to brightness, contrast, saturation, and hue. This was done to give the training data diversity and to prevent overfitting. Secondly, data normalization and transformation were done. Each image in the dataset was made into a 32 by 32 image size with red, blue, and green (RGB) channels. The image's pixel values were normalized to a mean and standard deviation of 0.5. Then, the PIL images were converted to PyTorch tensors. Finally, the data was split into train, validation, and test datasets with an 80-10-10 split.

Basic exploratory data analysis was done on the data to observe what the real and fake images looked like. Five images were randomly chosen from the dataset and plotted using matplotlib. As shown in Figure 1, the data is of extremely low quality, and this is because CIFAR-10 was

Figure 1: Visual inspections of images from the dataset

derived from the "80 million tiny images" dataset, and researchers at that time chose 32×32 pixels in size because it significantly reduces storage and computational requirements, making it efficient to conduct research.
Next, we analyzed the principal components of the features of the images. To do this, the images were run through a pre-trained model named VGG-16 to extract the features. We randomly sampled 160 images from the dataset for analysis and extracted its features. Then, the PCA was imported from the sklearn package in Python, and the principal components were analyzed. As

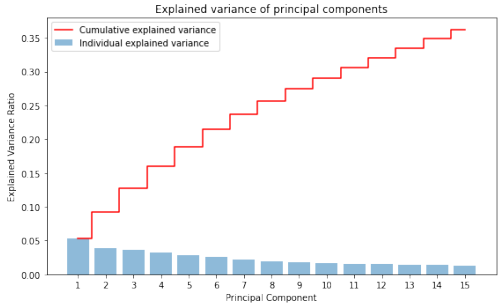

Figure 2: Principal Component Analysis

shown in Figure 2, even 15 components only explain approximately 35 percent of the variance in the dataset. Therefore, we decided to use all the features and instead used multiple pooling layers in our CNN architecture to reduce the number of dimensions.

4 Model

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

5 Methodology

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

6 Results

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

7 Conclusion

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

8 References

[1] I. Goodfellow, Y. Bengio, and A. Courville, "Convolutional Networks," in *Deep Learning*. Cambridge, Massachusetts: MIT Press, 2016. [Online]. Available: `https://www.deeplearningbook.org/contents/convnets.html`

[2] J. J. Bird, "CIFAKE Real and AI-Generated Synthetic Images," *Kaggle*, 2023. [Online]. Available: `https://www.kaggle.com/datasets/birdy654/cifake-real-and-ai-generated-synthetic-images`