Real and Fake Face Detection

J L
1234567
Master in Data and Computer Science
Person1@i1.org

X P
1234567
Master in Data and Computer Science
Person1@i2.org

R C
1234567
Master in Data and Computer Science
Person1@i2.org

H K
1234567
Master in Data and Computer Science
Person1@i2.org

Mentor: PLEASE DO NOT UPLOAD PERSONAL INFORMATION TO PUBLIC PLATFORM

1. Abstract

2. Introduction

In this project, our primary aim is to compile a comprehensive dataset consisting of genuine and altered human facial images. Subsequently, we seek to develop a convolutional neural network (CNN) capable of discerning fabricated images within this dataset. Our focus extends to evaluating the performance of our CNN model against established benchmarks, including the baseline ResNet model, MobileNetV2, and GramNet (with Gram Block). Through meticulous data collection and model development processes, we aim to provide insights into the efficacy of various CNN architectures for detecting manipulated images. Throughout model development, we experiment with various CNN architectures, exploring differences in depth, layer types, and regularization methods. We benchmark our CNN model against the baseline and other models using identical test datasets.

3. Related Work

4. Method

4.1. Datasets WIP

We adopt two different public datasets to train our model. The first dataset is a compact one with appropriate 1400 training images. The dataset contains expert-generated high-quality photoshopped face images, and images are composite of different faces, separated by eyes, nose, mouth, or whole face[4].

The second dataset is a large-scale dataset with appropriate 190K images. This dataset contains both manipulated images and real images, manipulated images are faces are generated by multiple ways, each image is a 256 x 256 jpg image of human face either real or fake[2, 3].

Cross-datasets Training WIP

4.1.1 Simple CNN

A simple CNN network can be used to test the correctness of program exectution and observe the flow of data. Because of its brief structure, it can significantly reduce the cost of computing resources. At the same time, it can alos be used as one of the benchmark performance indicators for comparison and analysis with other types of subsequent network models.

This convolutional neural network is designed for image classificaion, structured with an input layer which taks 3-channel RGB images, followed by three convolutional layers, each with a 3x3 kernal and padding of 1, progressively increasing the number of filters from 32 to 64 and finally 128. Each convolutional layer is followed by a ReLU activation function and a 2x2 max pooling layer. The output is flattened and passed throught two fully connected layers. The first FC layer transforms the feature map into 512 features, followed by another ReLU, and the second FC layer reduces it to 2 outputs for classification.

4.1.2 Improved CNN

The Improved CNN represents an enhancement of the previous 'SimpleCNN' model. it is designed to achieve better performance by adopting serveral architectural adjustments to increase the network's capacity and reduce overfitting.

Compared with the basic version of CNN, the improved version of convolutional neural network has been improved

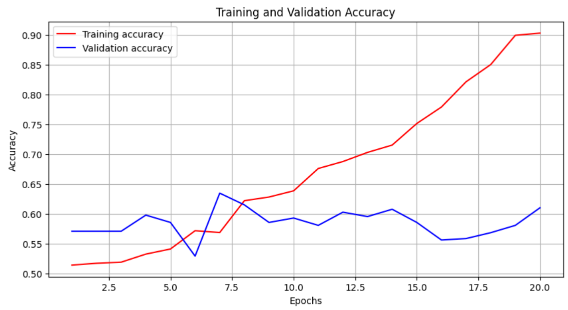

Figure 1. Training and Validation results of simple CNN with the compact dataset

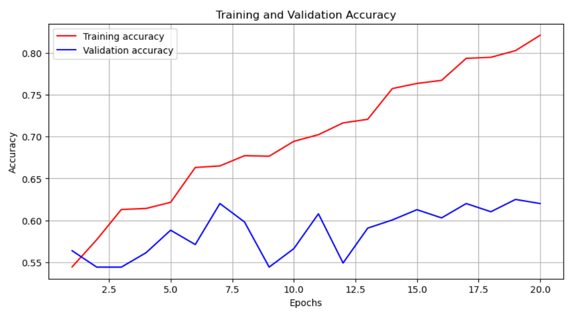

Figure 2. Training and Validation results of improved CNN with the compact dataset

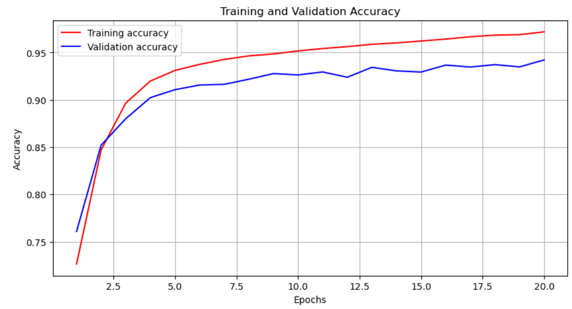

Figure 3. Training and Validation results of simple CNN with the large-scale dataset

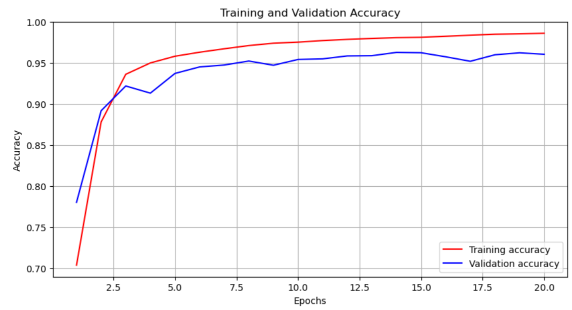

Figure 4. Training and Validation results of improved CNN with the large-scale dataset

in the following aspects: An additional convolutional layer has been introduced; The depth is increased with 256 filters; Each convolutional layer is now followed by a batch normalization layer; A dropout layer with a rate is introduced before the first fully connected layer; Also an increased fully connected layer capacity, transforms the feature maps into a larger dimensional space.

5. Experiments

5.1. Dataset 1 WIP

Our experienments starts with the small-scale dataset 1, wiit simpleCNN structure, as in Figure 1, after 20 rounds of iterations, the validation accuracy remains fluctuating within a range and has not increased significantly. The accuracy is about 60%.

The ImprovedCNN brought slight better performance in the later iterations of training. As shown in Figure 2, the validation accuracy range came to between 60% to 65%, but it was very unobvious and thus did not bring significantly improvement.

5.2. Dataset 2 WIP

The situation changes when large-scale data sets are adopted. With the support of the training set of 140k+ images, even the network structure of simpleCNN has achieved a huge leap in verification accuracy. As shown in Figure 3, it reached about 94% after 20 rounds of iterations. Compared with the case of using the simplified data set (Figure 1), an improvement of more than 30% is achieved.

The performance of the improved CNN network structure in large-scale datasets is shown in Figure 4. Although it has been greatly improved, the advantage over simpleCNN is very little. after 20 epochs, it reached about 96% validation accuracy.

5.3. Cross-datasets Training WIP

During the experiment and testing process, we found that the models trained on the two data sets can achieve better results when tested on their own data sets. However, when performing cross-testing, when testing on another data set, the performance was disappointing.

In order to achieve better generalization ability of the model, we adopted the cross-data training method.

An ImprovedCNN structure has no advantage in capturing and managing features from cross-datasets, as in Figure 5, the performance is quite poor in absolute terms, with only about 51% accuracy. It is also very unstable in relative terms, thus it is basically unusable.

WIP

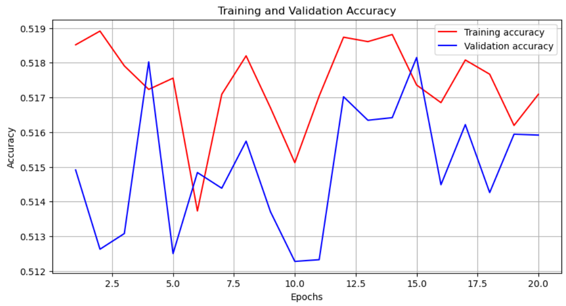

Figure 5. Training and Validation results of improved CNN with the cross-datasets

Figure 6. Example of caption. It is set in Roman so that mathematics (always set in Roman: $B \sin A = A \sin B$) may be included without an ugly clash.

5.4. Dataset generated by GAN network

6. Findings

Make sure to update the paper title and paper ID in the appropriate place in the tex file.

All text must be in a two-column format. The total allowable width of the text area is $6\frac{7}{8}$ inches (17.5 cm) wide by $8\frac{7}{8}$ inches (22.54 cm) high. Columns are to be $3\frac{1}{4}$ inches (8.25 cm) wide, with a $\frac{5}{16}$ inch (0.8 cm) space between them. The top margin should begin 1.0 inch (2.54 cm) from the top edge of the page. The bottom margin should be 1-1/8 inches (2.86 cm) from the bottom edge of the page for 8.5×11-inch paper; for A4 paper, approximately 1-5/8 inches (4.13 cm) from the bottom edge of the page.

Please number all of your sections and any displayed equations. It is important for readers to be able to refer to any particular equation.

Wherever Times is specified, Times Roman may also be used. Main text should be in 10-point Times, single-spaced. Section headings should be in 10 or 12 point Times. All paragraphs should be indented 1 pica (approx. 1/6 inch or 0.422 cm). Figure and table captions should be 9-point Roman type as in Figure 6.

List and number all bibliographical references in 9-point Times, single-spaced, at the end of your response. When referenced in the text, enclose the citation number in square brackets, for example [1]. Where appropriate, include the name(s) of editors of referenced books.

6.1. Illustrations, graphs, and photographs

All graphics should be centered. Please ensure that any point you wish to make is resolvable in a printed copy of the response. Resize fonts in figures to match the font in the body text, and choose line widths which render effectively in print. Many readers (and reviewers), even of an electronic copy, will choose to print your response in order to read it. You cannot insist that they do otherwise, and therefore must not assume that they can zoom in to see tiny details on a graphic.

When placing figures in LaTeX, it's almost always best to use `\includegraphics`, and to specify the figure width as a multiple of the line width as in the example below

```
\usepackage[dvips]{graphicx} ...
\includegraphics[width=0.8\linewidth]
                {myfile.eps}
```

7. Conclution

References

[1] FirstName LastName. The frobnicatable foo filter, 2014. Face and Gesture submission ID 324. Supplied as additional material `fg324.pdf`. 3

[2] Trung-Nghia Le. Deepfake and real images dataset. https://www.kaggle.com/datasets/manjilkarki/deepfake-and-real-images, 2022. 1

[3] Trung-Nghia Le, Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. Openforensics: Large-scale challenging dataset for multi-face forgery detection and segmentation in-the-wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 1

[4] Seonghyeon Nam, Hyolim Kang, Dongyoung Kim, Sejong Yang, et al. Real and fake face detection. https://www.kaggle.com/datasets/ciplab/real-and-fake-face-detection, 2020. Computational Intelligence and Photography Lab, Department of Computer Science, Yonsei University. 1