CONVOLUTIONAL NEURAL NETWORKS FOR REAL AND FAKE FACE CLASSIFICATION

Natalija Perišić*,
Radiša Jovanović

Faculty of Mechanical Engineering,
Belgrade, Serbia

Abstract:

This paper deals with the problem of classifying images of real and fake faces as it is impossible to distinguish them with the bare eye. Two different convolutional neural networks architecture models are applied. The first one is pre-trained VGG16 model, where transfer learning method is applied on our dataset. The architecture of the second model is based on VGG16 and represents its smaller and lighter version. Techniques such as learning rate decay, dropout and batch normalization was applied in training process. Comparison of obtained results of both models is made.

Keywords:

Convolutional Neural Network, Deep Learning, Fake Face Image Classification, Transfer Learning, VGG16.

INTRODUCTION

Artificial intelligence (AI) represents scientific filed that constantly improves. Deep neural networks (DNN) have purpose in many fields such as medicine, pharmacy, automatic control, robotic, entertainment, language processing, etc. Special type of technology, that uses deep learning for creating fake videos, images, texts or events is called 'deepfakes' [1]. Term 'deepfakes' was first used in 2017. on Reddit and since than the use of deepfakes increased. Usually, there are two ways to create deepfakes. The first one is called face-swapping method. Basically, this method revolves replacing person's face from the input image with another face, usually from large set of faces, like in [2]. The second method is by using generative adversarial network (GAN). GAN was firstly proposed in [3] and represents two artificial neural networks (ANN) that compete to each other in order to give the best solution, which is, in this case, the most realistic fake face. The first ANN generates new image from random instances contained in dataset while the role of the second ANN is to evaluate generated image for authenticity. As a result of technology improvement, high resolution images and videos, and development of AI algorithms created deepfakes look very realistic and it is almost impossible to determine are they real or not.

Correspondence:
Natalija Perišić

e-mail:
nperisic@mas.bg.ac.rs

DOI: 10.15308/Sinteza-2022-29-35

Even though nowadays deepfakes can be used for good purposes, such as in movie and chemical industries, material science, medicine, entertainment, etc. which is better explained in [4], they also represent a threat that can have serious consequences. Many scientists of different science fields worn of the danger of malicious use of deepfakes. For example, GAN created images can be used to deceive facial recognition system or for hiding identity on social networks. Video, voice or image manipulation can be used for blackmailing people and there is no need to explain what kind of political consequences can be caused by abusing deepfake technology. There is obvious need to recognize deepfakes to prevent their abuse.

Research about visual processing by observing primary visual cortex of a cat, where it was shown that changing the angle of a line causes activation of different neuron groups and that the same group of neurons is in charge of edge detection regardless of their position, served as an inspiration for creating first ANN that was used for pattern recognition. That ANN is called neocognitron, proposed by Kunihiko Fukushima in 1980. [5]. Neocognitron was the basis of the further research that led to the creation of the first convolutional neural network (CNN). In 1989. backpropagation algorithm was used to train CNN in order to recognize handwritten numbers [6]. This was a prototype for LeNet architecture of CNN. The huge step forward in researching CNN was achieved by developing AlexNet [7] in 2012. This type of CNN won the ImageNet Large Scale Visual Recognition Challenge in the same year with achieved error of 15.3% on test set which was significantly less than error of other competitors. Nowadays we are familiar with many different CNN architectures. One of the most famous and most commonly used architecture is VGG16 [8]. VGG16 was created as a result of the research how network depth affects accuracy on large-scale image dataset. With novelties such as depth of 16 weight layers and convolutional filters sized 3 x 3, this network was used at the ImageNet Challenge 2014, where it achieved 7.5% top-5 error. It is important to mention that VGG16 is large, heavy model, sized around 530MB. That is caused by many weight parameters and leads to slow training process.

Solution for the problem of detecting fake faces is presented in [9], where new architecture of CNN, Local Binary Pattern-Net is designed and detection is based on the texture features of fake faces as it is different than the texture of real faces. Few different CNN models are used in [10] and it is concluded that deep-learning algorithms and models are propriate for recognizing fake faces. The best results are obtained by using VGG19 architecture. In [11] authors firstly use Kalman Filter for preprocessing images, then use amalgamation of fisherface algorithm for face recognizing with Local Binary Pattern Histogram for space dimension reduction of face. Deep Belief Network is used for final classification. This method is applied on four different datasets and the results showed that this method is very effective, and that executes very fast.

In this study we propose solution for classifying real and fake faces, created by GAN network, by using pretrained VGG16 model and custom VGG-like network, with smaller and lighter architecture.

2. DATASET

Dataset, that is used for the research consists of 140000 images of human faces. Half of the total number of data are images of real human faces, retrieved by Nvidia from the Flickr-Faces-HQ dataset [12], while the other half of data are images of fake faces. Fake face images are all generated by StyleGAN and they are part of huge dataset of 1 million fake faces [13]. Combined, they represent one of the largest datasets available online. All of the images in this dataset are resized to 256 × 256 pixels and divided into three folders – train, test and validation folder. Train folder contains 100000 images, while test and validation folder have 20000 images, each. The ratio between fake and real images in all folders is always 1:1. This dataset can be found in [14]. Few random data samples from both classes are shown in Figure 1. As it can be seen, it is almost impossible to find the difference between real and fake faces with bare eye.

Figure 1 – Random samples of fake and real faces from dataset.

Data augmentation, such as zooming, flipping, rotating images, etc. is avoided in this research, because there is already a large amount of data in this dataset.

3. APPLIED METHODS AND ARCHITECTURES OF CONVOLUTIONAL NEURAL NETWORKS

3.1. CUSTOM CONVOLUTIONAL NEURAL NETWORK

There are three types of layers that form CNN – convolutional, pooling and fully connected layers. Convolutional layer contains filter kernels whose weights need to be learned in learning process. Role of pooling layer is to reduce size of the output from convolutional layer which leads to reducing number of weights that should be learned and reducing computation process. Fully connected layer connects all outputs from one layer to all inputs from next layer. Usually, this type of layer is placed at the end of neural network in order to perform classification of flatten output from the last convolutional or pooling layer.

Input layer of the created CNN consists of RGB images with height and width resized to 224 pixels, which gives them dimension 224×224×3 pixels. There are three convolutional layers, one with 64, two with 128 filters. The size of all filters is 3×3. A Rectified Linear Unit (ReLU) is used after convolutional layers as activation function to eliminate all negative weights after filtering image and replace them with zero value. ReLU function can be described as

$$f(x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases}$$

Equation 1 – ReLU function

After every activation function, batch normalization method is applied. It was proposed in [15] and it is used for normalization of inputs in layers. According to [16], this method provides acceleration of training process because it allows using higher learning rate. Proposed CNN model contains three pooling layers that have pool size 2×2, and with stride 2. Maximum pooling operation is applied for reducing number of learning parameters. Output from the last pooling layer is flattened in order to convert data into vector. Following, there are two fully connected layers, one with 256 nodes, and one with 1 node. For predicting probability that input data belongs to certain class sigmoid activation function is used. Sigmoid function is determined as

$$f(x) = \frac{1}{1+e^{-x}}.$$

Equation 2 – Sigmoid function

Fully structure of designed CNN is shown in Figure 2.

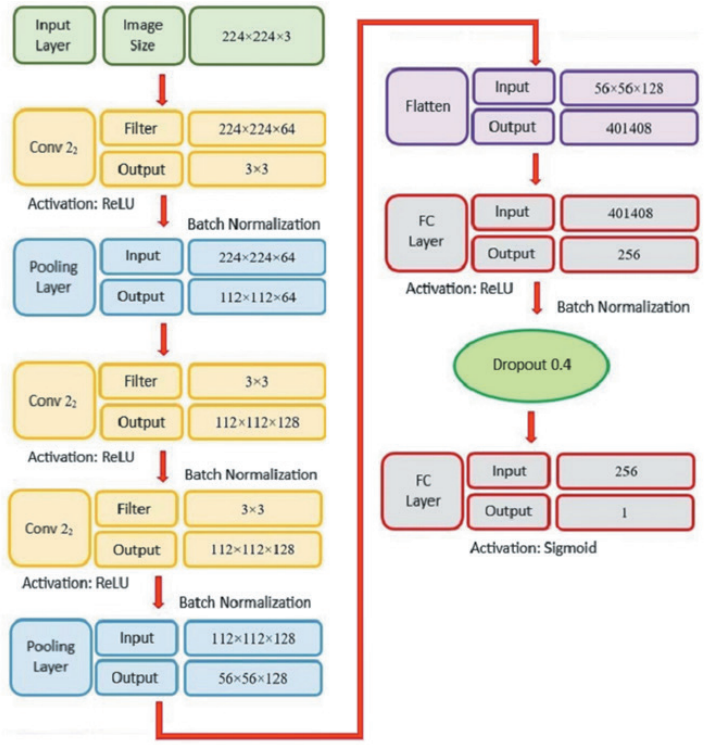

Figure 2 – Structure of proposed, VGGNet inspired CNN.

Regularization covers few techniques that can help avoid overfitting and getting higher accuracy when ANN works with unknown data. As form of regularization, dropout method is applied. This method implies randomly selecting and ejecting some neurons and its weights from learning process according to previously determined probability level. By applying dropout, it is possible to prevent high dependence between certain neurons that leads to activation of only few neurons for solving problem. In this research dropout technique is used after first fully connected layer with probability 0.4.

3.2. PRE-TRAINED VGG16 MODEL

Transfer learning is a method in deep learning which implies using already trained model for solving new but similar problem, [16]. Fine-tuning is technique in transfer learning that allows changing model in order to adjust it to new task. It means that some of pre-trained parameters are frozen or non-trainable, but some of them should be trained in learning process. The structure of pre-trained VGG16 model on ImageNet dataset is presented in [8]. All layers are kept and all parameters were frozen, except for the last three fully connected layers that are removed from original structure. Flatten was added after last pooling layer, and two new fully connected layers are added, first with 512 nodes and second with 1. Between them, dropout was applied with probability 0.4. Images in input layer are resized to 224×224 pixels in order to match with the size of images that was originally used for training VGG16 model.

4. TRAINING OF THE DESCRIBED MODELS

For implementing, evaluating and training our two models, Python programming language was used with Keras library.

As we are dealing with the binary classification problem, which implies determining the affiliation of a sample to the class, binary cross entropy was defined as loss functions for both models. Loss function calculates the distance between target value of model's output and obtained output's value. In order to find minimum of the loss function Adaptive Moment Estimation (ADAM optimizer) is used. This optimization technique compute learning rate for every parameter and does not require a lot of memory, so it is suitable for large dataset. Number of training examples per iteration defines batch size. In training of both models, 100 images are processed in one iteration and after each iteration weights were updated.

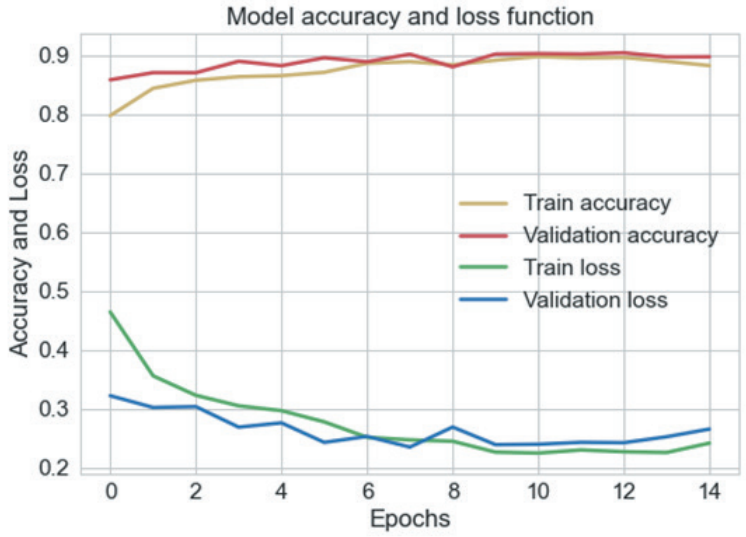

Figure 3 – Accuracy and loss function obtained in training and validation of pre-trained VGG16 model.

Learning rate decay is method in model training where the learning rate slowly decreases at the beginning of each epoch. In training of custom CNN model learning rate was set to 0.01 with decay that is equal to the quotient of the initial learning rate and number of epochs.

Defined number of epochs for both models is set to 15, and during training process function for saving best results was used. It means that best weights are saved and used in model testing which is a great way to avoid overfitting and bad generalization.

5. RESULTS AND DISCUSSION

The best way to analyse training process is to observe change of the loss function and accuracy during epochs.

In Figure 3, accuracy and loss function for training and validation of VGG16 model during 15 epochs training are shown. It is clearly that minimizing loss and increasing accuracy were successful until 7th epoch and after that loss started to grow in validation process. In other words that is the moment when overfitting started. Graph that shows same functions for second, custom CNN is given in Figure 4. In this case, model make progress in learning during first 6 epochs right before overfitting started.

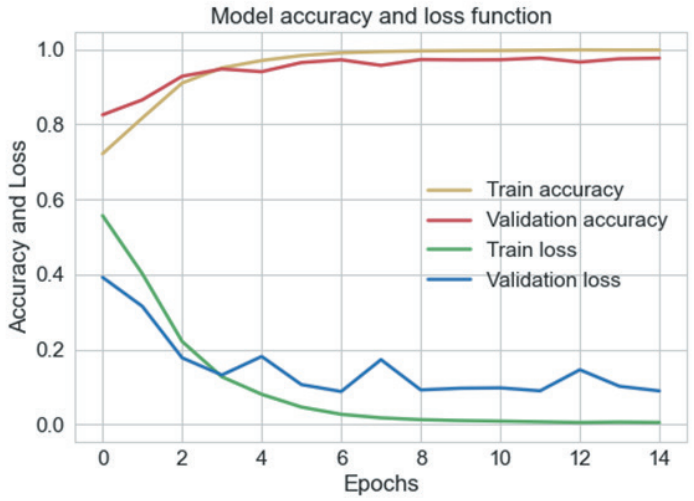

Figure 4 – Accuracy and loss function obtained in training and validation of proposed, custom model.

Trained models are tested on set of images that was excluded from training, which means that they are unknown for them. Criteria for comparison and obtained results are given in the below.

In case of binary classification, predicted output can be included in one from the following categories:

- True positive (TP), if predicted fake face class is correct for the input image,
- True negative (TN), if predicted real face class is correct for the input image,
- False positive (FP), if predicted fake face class is incorrect for the input image,
- False negative (FN), if predicted real face class is incorrect for the input image.

For performance evaluation few parameters are considered – accuracy, precision, recall and F1 score. Accuracy is performance measure that shows ratio between correctly classified samples and total number of samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

Equation 3 – Accuracy

Precision can be calculated as:

$$Precision = \frac{TP}{TP + FP}.$$

Equation 4 – Precision

Recall or sensitivity measures the model's capability to correctly classify true positives. It can be represented by following expression:

$$Recall = \frac{TP}{TP + FN}.$$

Equation 5 – Recall

Finally, F1 score is measure of model's performance that combines precision and recall into following equation:

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}.$$

Equation 6 – F1 Score

Obtained performance measures, rounded on four decimal places are given in Table 1.

	Pre-trained VGG16	Proposed model
Accuracy	0.8998	0.9718
Precision	0.9038	0.9789
Recall	0.8949	0.9644
F1 Score	0.8993	0.9715

Table 1 – Obtained results

As it can be seen from Figures 3 and 4, in training and validation process less loss and higher accuracy are obtained by proposed, custom model. In testing, both models scored acceptable and high values of the selected parameters for performance measurement. However, custom model showed better performance in testing, which means that more of the samples were correctly classified, so it achieved significantly greater all values – accuracy, recall, precision and F1 Score. Although training process time is shortened by using transfer learning strategy, for this particular case, smaller, shallower ANN that was proposed in this paper, undoubtedly represents better choice.

6. CONCLUSION

In this research the problem and potential treat of malicious use of deepfake technology is described. Potential solution for identifying fake faces that was created by GAN networks, is found by using CNN.

Two different strategies in CNN training were applied. The first one was transfer learning, and the second one was custom CNN with lighter and smaller structure. Some of the optimization and regularization methods were applied in order to obtain the best possible results in the training process.

Learning process of both models took 15 epochs and it was shown that overfitting started at similar moment. Also, custom CNN showed better results after learning process, so it was expected that it gives better result in testing as well.

The goal of this paper was to compare performance of these two models in testing, were they had task to classify data samples that was unknown for them. As it was expected, second CNN achieved incomparably better results, with accuracy of around 97%. This is valid argument to recommend using this type of CNN for solving this particular problem.

The next step in research is to try applying different optimization methods and regularization techniques.

Also, changing defined parameters such as learning rate, dropout probability may result in a change in model's performance.

7. ACKNOWLEDGEMENTS

This research was financially supported by Ministry of Education, Science and Technological Development of the Serbian Government, MPNTR RS under contract 451-03-68/2022-14/200105, from date 4.2.2022.

This work was supported by the Science Fund of the Republic of Serbia, grant No. 6523109, AI- MISSION4.0, 2020-2022.

8. REFERENCES

[1] L. Pupillo, S. Fantin, A. Ferreira, and C. Polito, "AI Malicious Uses," in *Artificial Intelligence and Cybersecurity, Brussels, Centre for European Policy Studies*, 2021, pp. 30-36.

[2] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur and S. K. Nayar, "Face Swapping: Automatically Replacing Faces in Photographs," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 1-8, 2008.

[3] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems,* Montreal, 2014.

[4] M. Westerlund, "The Emergence of Deepfake Technology: A Review," *Technological Innovation Management Review*, vol. 9, no. 11, pp. 39-52, 2019.

[5] K. Fukushima, "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position," *Biological Cybernetics,* vol. 36, no. 4, pp. 193-202, 1980.

[6] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R.E. Howard, W. Hubbard and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541-551, 1989.

[7] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Network," in *Proceedings of the 25th International Conference on Neural Information Processing Systems*, Lake Tahoe, 2012.

[8] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," April 2015. [Online]. Available: https://arxiv.org/abs/1409.1556. [Accessed 25 3 2022].

[9] Y. Wang, V. Zarghami and S. Cui, "Fake Face Detection Using Local Binary Pattern and Ensemble Modeling," in *2021 IEEE International Conference on Image Processing*, Anchorage, 2021.

[10] M. Taeb and H. Chi, "Comparison of Deepfake Detection Techniques Through Deep Learning," *Journal of Cybersecurity and Privacy*, vol. 2, no. 1, pp. 89-106, 2022.

[11] ST Suganthi, M. U. A. Ayoobkhan, K. Kumar V, N. Bacanin, K. Venkatachalam, S. Hubalovsky and P. Trojovsky, "Deep Learning Model for Deep Fake Face Recognition and Detection," *PeerJ Computer Science*, vol. 8, p. e881, 2022.

[12] P. datasets, "70k Real Faces," [Online]. Available: https://www.kaggle.com/c/deepfake-detection-challenge/discussion/122786. [Accessed 23 3 2022].

[13] P. Datasets, "1 Million Fake Faces on Kaggle," [Online]. Available: https:// www.kaggle.com/c/deepfake-detection-challenge/discussion/121173. [Accessed 23 3 2022].

[14] P. Datasets, "140k Real and Fake Faces," [Online]. Available: https://www.kaggle.com /datasets/xhlulu/140k-real-and-fake-faces,. [Accessed 23 3 2022].

[15] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proceedings of the 32nd International Conference on Machine Learning Research*, Lille, 2015.

[16] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345-1359, 2010.