***A Generative Adversarial Network (GAN) for the classification of faces from video data.***

Deep Learning – April 20, 2022

Authors: Kevin Goodman - m20210701, Alberto Parenti - m20211304, Leonardo de Figueiredo - m20210667, Ricardo Arraiano - m20210995, Leonor Candeias - m20210990

Abstract

-right now I’m calling the inputs video data but that could change in the future if we just use images, I’m also framing it as a Russian Putin thing but that can easily be changed later on, this whole thing is still fairly vague and approximate because we haven’t built everything yet.

Introduction

With the rise of Machine Learning (ML) and neural networks (NNs) has come a time where information can be synthetically made to deceive and mislead. This information can be encoded as images, audio, text or combined into video formats. One outlet of these so called “Deepfakes” is to make a highly influential figurehead appear to be saying something they didn’t and thus sway public opinion. Concordantly the need for discrimination on the part of the viewer has risen. This issue has recently resurged with political events involving Russia and Ukraine. The Western media portrays itself as a beacon of truth and virtue, only interested in the common good. Meanwhile the Russian media has likewise labelled itself as a ‘liberator’ of Ukraine and condemned the expansion of NATO as overly hostile; labelling the United States as hypocritical. Both sides have partially or fully censored the media of the other side. The average citizen is left to themselves to determine which information is real and which is false.

In 2014 Goodfellow et al. (ref) developed a new type of NN whereby two networks – a generator model (G) and a discriminator model (D) – compete against one-another using a training set of real images. The discriminator model works to determine which input image is real and which is fake while the generator model tries to “trick” the discriminator model by improving its ability to generate images. This results in a zero-sum game where

The process proceeds as follows: the discriminator model is built and trained using real images with an input space representing the dimensions of the image by number of pixels and the colour-scale to be used. Each value represents the intensity of colour for that pixel. These training examples are labelled as unity to refer to the fact that they are real images. Because this is a binary classification problem, the output layer is typically uses a sigmoid activation function where any value greater than 0.5 signals a real image while any value below 0.5 represents a fake image. Initial examples of fake images are generated using the generator model whose first attempt is made by random initialization of weights. This image will be assigned a target value of zero and fed to the discriminator model for training whereupon the weights of the discriminator are updated using backpropagation via the chain rule. The error function of the discriminator takes the form –ln(1-x) (Eqn. 1), while the error function for the generator model’s error function follow –ln(x) where x is the prediction. In this way the models have opposing binary targets and the generator model must improve its output to better resemble the real images in the training data.

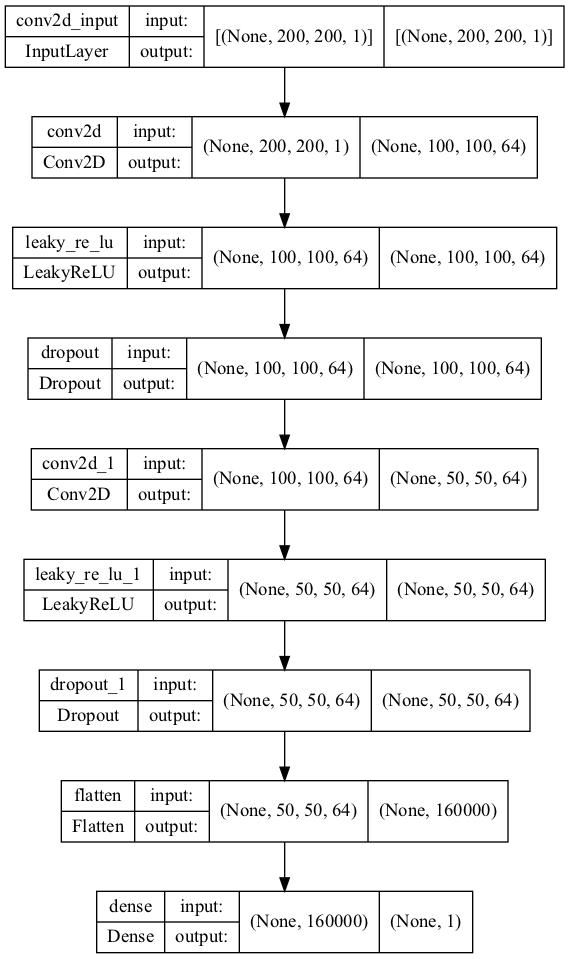
-eqn for loss functions here from ian goodfellow

In this work the authors have attempted to build such a Generative Adversarial Network (GAN) to apply to video data with the application of discrimination of real versus fake video footage of Vladimir Putin during this time of Russian conflict.

Methods

Our initial step was to build a discriminator model (D). This involved the use of two convolutional neural network (CNN) hidden layers with sixty-four nodes each outputting their result to a single node dense layer that utilizes the sigmoid activation function (eqn for sigmoid here if you want) to establish a probability of real (1) or fake (0) identity in the image.

-building of our GAN, kernel = 3, stride = 2, padding = ‘same’, solver = Adam, learning rate = 2e-4, momentum = 0.5, error = binary cross-entropy

**Fig. 1 –** format for D showing two CNN layers, two leaky ReLu activation layers, two dropout layers (alpha = 0.4), a flatten layer to convert the output to a single dimension, followed finally by a single-neuron dense layer for output.

-training and testing the GAN

Results and Discussion

Conclusions

References

-ian goodfellow 2014