Neural Network Analyses of Image Modifications

Applications of Machine Learning to Identification of Image Forgeries

Eunpa Chae

Spring 2021

Contents

1a. Introduction

In this everything-digital age electronic images are ubiquitous. The average internet viewer may not be aware of the numerous ways to modify digital images. Although altering an image might seem innocuous – it could be motivated by a sense of humor – there are serious repercussions that could qualify as slander or illegal actions. In fact an increasingly web-centric mindset also brings greater attention to the importance of defending one's online reputation. Anonymous modifications of electronic images might seem untraceable to most individuals. Professionals with a greater stake in monitoring their online reputation as represented by electronic images include models, actors, athletes, celebrities, politicians, artists, pop stars, musicians, etc.

Depending on the extent of modifications to electronic images the owner could take legal action based on an interpretation of defacement as a type of modern graffiti or prove a link between defacement and financial loss or additional harm.

2. Possible applications

The advancement of data science algorithms and machine learning applications also brings with it an increasing chance that those with criminal intentions may hijack technology. Possible targets include methods that alter electronic images, seemingly without leaving a trace, including Generative Adversarial Networks (aka GANs). GANs were developed to train algorithms based on 'worst-case-scenario' test exercises. In this 'proven in enemy territory' sense GANs are one of the most rigorous types of machine learning applications in existence. This characteristic, combined with the fact that GANs are a relatively new type of algorithm, makes it less detectable. It is this virtual untraceability that mainly attracts nefarious activity including competitors who have to resort to sabotage in order to win.

An extension of this negative effect is that GANs have seized advertising market share. Product promotion and additional lucrative options were once the exclusive domain of professionals whose income is linked to maintaining their image, both online and at in-person events. As it is possible to quantify the financial value of superior visual presentation it could be interpreted as an equivalent of intellectual property. Going one step futher with this analogy a possible future analysis might cover an application to verify validity of live models via analyses of DNA. There is additional relevance of DNA-based analyses to verify real faces and identity of individuals induced to have elective surgical procedures. By definition any type of elective surgery would be sabotage of professionals with top model features. By contrast those who voluntarily elected to have plastic surgery enhancements include models who gamed a high-barrierto-entry system with very selective criteria via procedures that enhanced their features. Although it might be feasible to devise some type of licensing agreement between these high-profile professionals and technologists applying GANs to sell corporations less expensive advertising options in the end fake-image generators may get the better part of the deal. It is obvious that, currently, the public might not have the technology to differentiate between real and fake images. Yet given a choice between buying a product based on a marketing campaign with a real, unmodified, model and viewing an ad with a fake model it is probable that the majority prefers to view real celebrities.

3a. Preprocessing Methods

The source of data is 140k Real and Fake Faces | Kaggle The designer of this dataset applied GANs to generate 70000 fake faces. These data were supplemented by 70000 real faces. As the nature of processing digital images is very memory-intensive I chose to subset these images to 1000 in each training group and 100 in each testing group in the initial models. Subsequent models trained on 20000 images and tested on 200 images. Each image was standardized to 100x100 pixels and separated into grayscale and color lists. The arrays were processed into numeric Python arrays and stored as tensors.

The neural network code is based on https://keras.io/getting_started/intro_to_keras_for_engineers/

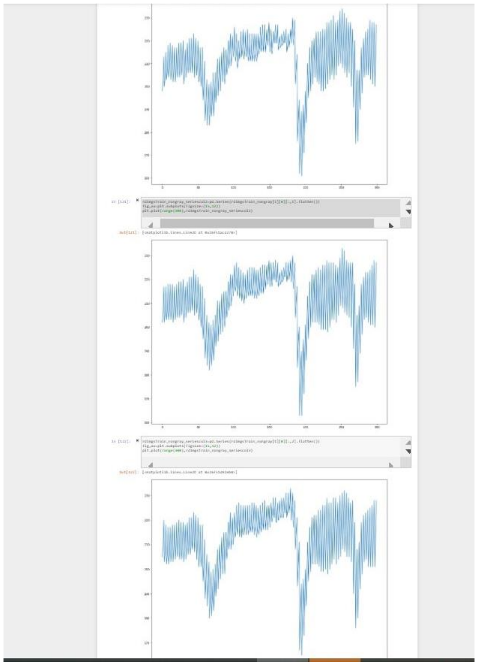
Preprocessing

A function was written to read images into arrays labeled to specify real or GANs and gray or color. The skimage library was applied to change color to grayscale to efficiently utilize memory and compare neural network analyses on both types of images. Then the actual images were processed with matplotlib.pyplot The lists with .jpg images were changed to numeric Python arrays as final preparation to enter data in a neural network models. In these models the data were processed as tensors with tensorflow and keras libraries. An additional function was written to standardize this preprocessing.

3b Analysis of Pixel Variation

Although the mechanics of neural network analyses are not as transparent as some models it is possible to infer the type of process applied to arrive at these results. Detection of images synthesized via GANs compared with real photographs requires identification of a pattern in the mosaic method with which GANs generates images. This might be possible by unnatural or abrupt changes in pixel variation at edges or other anomalies. A quick analysis of variation in pixels by row and column of a sample training image reveals there is no obvious pattern of variance between real and GANs images.

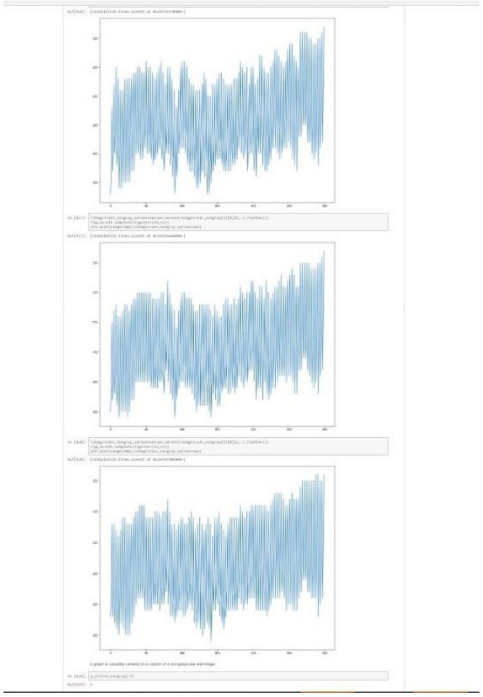
Pixel variation in three adjacent columns of nongrayscale GANs image



```
In [121]: rdimgstrain_nongray_seriescol2=pd.Series(rdimgstrain_nongray[1][0][:,1].flatten())
          fig,ax=plt.subplots(figsize=(15,12))
          plt.plot(range(300),rdimgstrain_nongray_seriescol2)
Out[121]: [<matplotlib.lines.Line2D at [illegible]>]
```

```
In [122]: rdimgstrain_nongray_seriescol3=pd.Series(rdimgstrain_nongray[1][0][:,2].flatten())
          fig,ax=plt.subplots(figsize=(15,12))
          plt.plot(range(300),rdimgstrain_nongray_seriescol3)
Out[122]: [<matplotlib.lines.Line2D at [illegible]>]
```

Pixel variation in three adjacent rows of nongrayscale GANs image

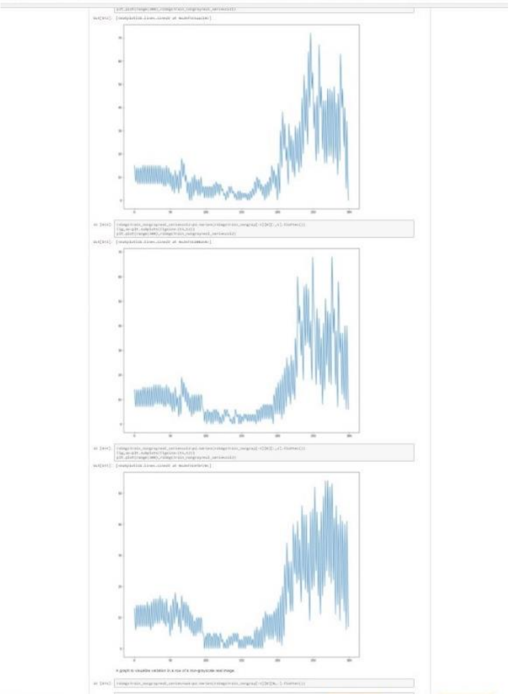


```
Out[126]: [<matplotlib.lines.Line2D at [illegible]>]
In [127]: [illegible]
Out[127]: [<matplotlib.lines.Line2D at [illegible]>]
In [128]: [illegible]
Out[128]: [<matplotlib.lines.Line2D at [illegible]>]
```

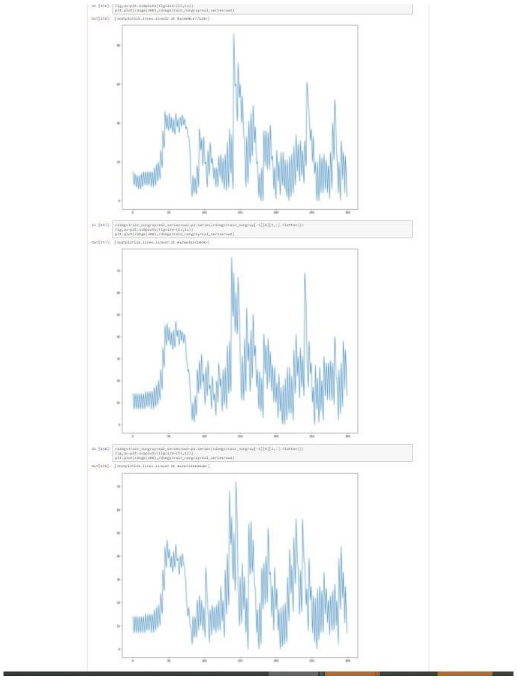
A graph to visualise variation in a column of a non-grayscale real image

```
In [129]: [illegible]
Out[129]: [illegible]
```

Pixel variation in three adjacent columns of nongrayscale real image



Pixel variation in three adjacent rows of nongrayscale real image



Thus the sophistication of neural network models might detect unnatural changes at certain key parts of images.

4a. Modeling

Neural networks were applied as referenced in the keras library of tensorflow. A model with five layers was applied in which data were scaled and passed through layers with decreasing number of nodes tapering to an endpoint in this binary classification.

Layer (type)	Output Shape	Param #
Input_2	[(None, 100, 100)]	0
Rescaling_1	(None, 100, 100)	0
Flatten_1	(None, 10000)	0
Dense_3	(None, 128)	1280128
Dense_4	(None, 64)	8256
Dense_5	(None, 2)	130
Total params: 1,288,514		
Trainable params: 1,288,514		

After the model was built, it was compiled with optimizers , loss functions, and metrics specified as follows:

The model was compiled with 'adam' optimizer, sparse_categorical_crossentropy loss function (requires y_variable to be one-hot-encoded – in this instance 0 code meant GANs image and 1 meant real), and SparseCategoricalAccuracy metric. The fact that validation_accuracy started at 63.5% on epoch 1 is encouraging as this may mean there is generalizability of the model trained on 2000 color images.

The model was trained and validation data passed to assess accuracy. When model is trained on 2000 images the maximum validation accuracy was 67% which is a bit better than maximum accuracy on validation data passed to a model trained on 1000 epochs with 100% accuracy. This decrease in validation accuracy at 100% training accuracy may reflect overfitting at 1000 iterations. At a training accuracy of about 85%, when model is trained at 300 iterations, the maximum validation accuracy is achieved.

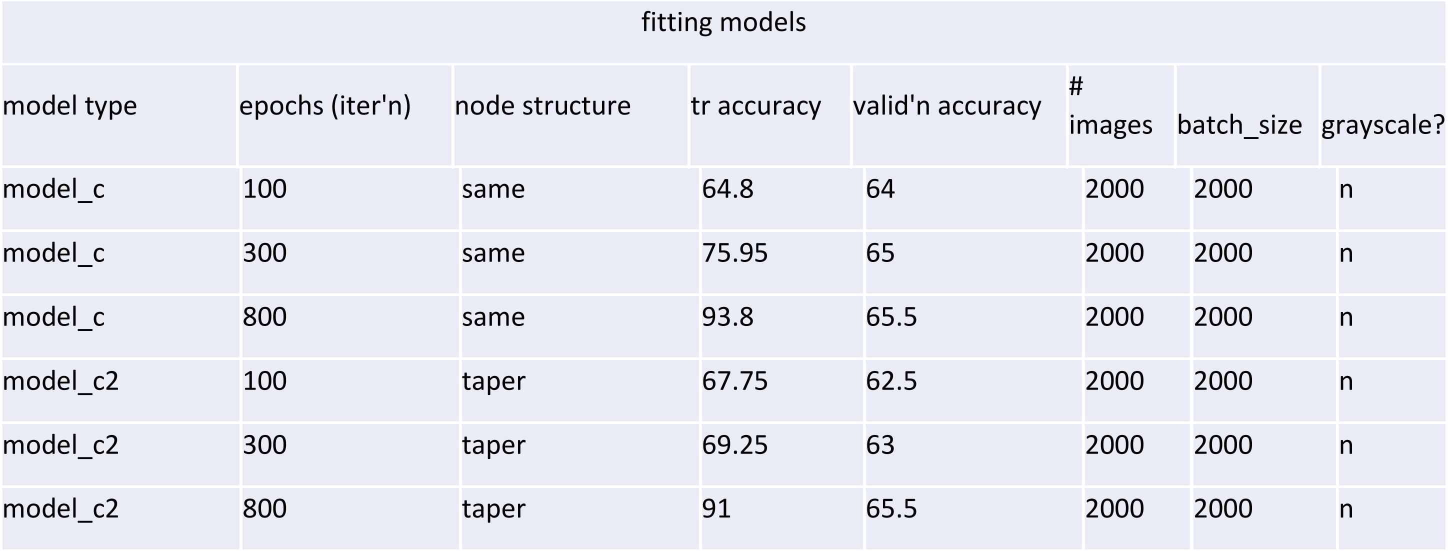
fitting models

model type	epochs (iter'n)	node structure	tr accuracy	valid'n accuracy	# images	batch_size	grayscale?
model_g	1	same	57.55	-	2000	2000	y
model_g2	1	taper	50.25	-	2000	2000	y
model_g3	1	double	50.25	-	2000	2000	y
model_c	1	same	50	-	2000	2000	n
model_c2	1	taper	53.2	-	2000	2000	n
model_g	300	same	-	68	2000	2000	y
model_g2	12	taper	59.05	-	2000	2000	y
model_g2	100	taper	69.05	65.5	2000	2000	y
model_g2	300	taper	83.85	67	2000	2000	y
model_g2	500	taper	92.95	67.5	2000	2000	y
model_g2	800	taper	97.8	66	2000	2000	y
model_g2	1000	taper	99.65	65	2000	2000	y

This accuracy level may reflect my naivete with neural networks or verisimilitude of GANs-generated images with real images although at greater number of iterations the accuracy improves.

In determining whether analyses of grayscale images affect neural network results because of loss of information compared with color images the same models were processed on non-grayscale images.

A 5-layer neural network model with similar structure was processed on 2000 color images and the validation accuracy turned out to be a bit lower at an estimated 65%.



fitting models							
model type	epochs (iter'n)	node structure	tr accuracy	valid'n accuracy	# images	batch_size	grayscale?
model_c	100	same	64.8	64	2000	2000	n
model_c	300	same	75.95	65	2000	2000	n
model_c	800	same	93.8	65.5	2000	2000	n
model_c2	100	taper	67.75	62.5	2000	2000	n
model_c2	300	taper	69.25	63	2000	2000	n
model_c2	800	taper	91	65.5	2000	2000	n

To determine whether number of images was affecting accuracy of neural network models analysis was applied on 20000 training images to compare with 2000-image analysis as presented above.

A 5-layer neural network model trained on 20000 grayscale images processed similarly as a grayscale model trained on 2000 as above gives slightly better validation accuracy at 71.5%

Fitting models							
Model type	Epochs	Node structure	Tr accuracy	Valid'n accuracy	# images	Batch_size	Grayscale?
model_g2	300	taper	65.43-71.35	67.5-70.1	20000	10000	y
model_c	300	same	70.91	72	20000	10000	n

A 5-layer neural network model with same number of nodes at each layer trained on 20000 color images processed similarly as grayscale model above gives best of this series with validation accuracy at 73.5%

Fitting models

Model type	Epochs	Node structure	Tr accuracy	Valid'n accuracy	# images	Batch_size	Grayscale?
model_c	800	Same	82.25	73.5	20000	10000	n

5. Final thoughts and Future analyses

To sum up the analyses a Keras TensorFlow neural network model was applied to distinguish between synthetically-generated GANs images and real images. The model was trained on 2000 grayscale and color images and validated on 200 test images. A subsequent model was trained on 20000 grayscale and color images. The structure of the model included five layers with preprocessing standardization of images and application of rectified linear activation function. The model was compiled with ‘adam’ optimizer, sparse_categorical_crossentropy loss function, and SparseCategoricalAccuracy metric. The fitting of models revealed accuracy on validation data reaching a maximum of 73.5% on a model trained on 20000 non-grayscale images. As expected best accuracy was achieved on models trained on a greater number of images with non-grayscale data which retains maximum amount of information.

Time-permitting future avenues of research include convolutional neural networks (CNN) and watermarking processes. It would be interesting to supplement analyses with data on images going beyond pixel-variation to include watermarks. In addition the fact that CNN involves frames that scan all possible regions of a specified size to analyze patterns might lead to a better accuracy percentage.