CAPSTONE 3

Synthetic images compared with real photographs

Eunpa Chae

Springboard 2020-2021

DIGITAL IMAGES ARE EASILY MODIFIABLE

- With everything and everyone online a web presence is increasingly vulnerable to interference from hackers, frenemies, competition, etc.
- This is of even greater importance when intruders deface images of professionals whose income is linked to keeping superiority of appearance intact as per original features, etc.
- Intellectual property analogy applies to faces of professionals whose career and reputation is mainly centered on visual presentation maximizing attractive nature of their features
- Several methods of modifying electronic images
  - Copy-Move forgery (copy and paste regions)
  - Copy-Move forgery with modification of copied region (blur, smear, etc.)
  - GANs (Generative Adversarial Networks)

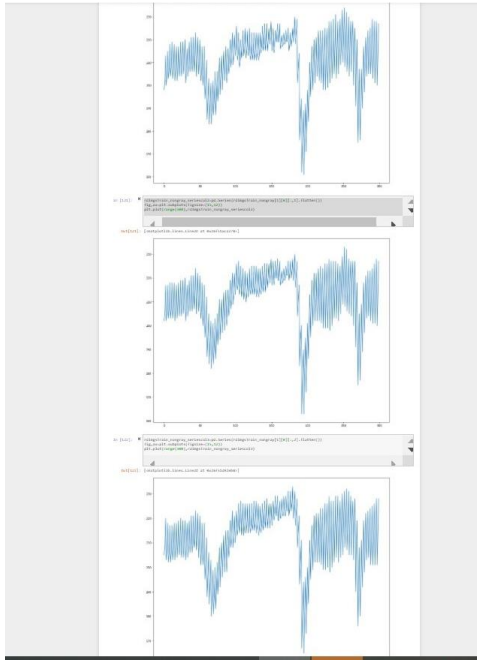
PREPROCESSING OF IMAGES

- Source of data https://www.kaggle.com/xhlulu/140k-real-and-fake-faces
- All 20200 images were standardized to 100x100 pixels in size
- Images were converted to grayscale via skimage's color.rgb2gray
- Datasets labeling each image with right classification were built and converted to numeric Python arrays then tensors via tensorflow.convert_to_tensor

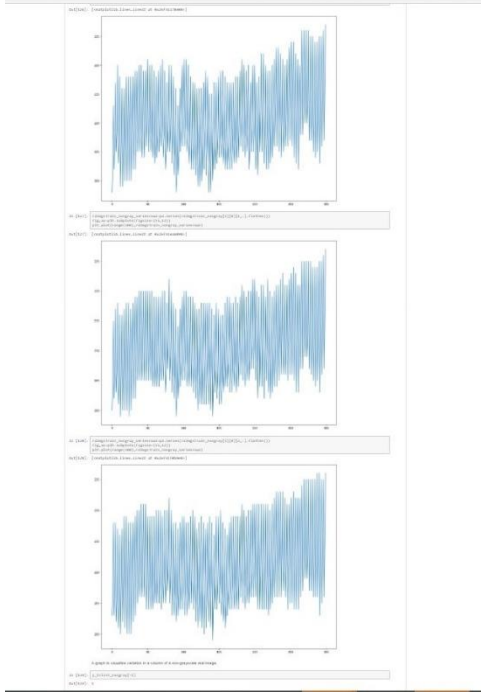
PIXEL VARIATION ANALYSIS

- 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 detection of unnatural or abrupt change 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 change in pattern of variance between real and GANs images.

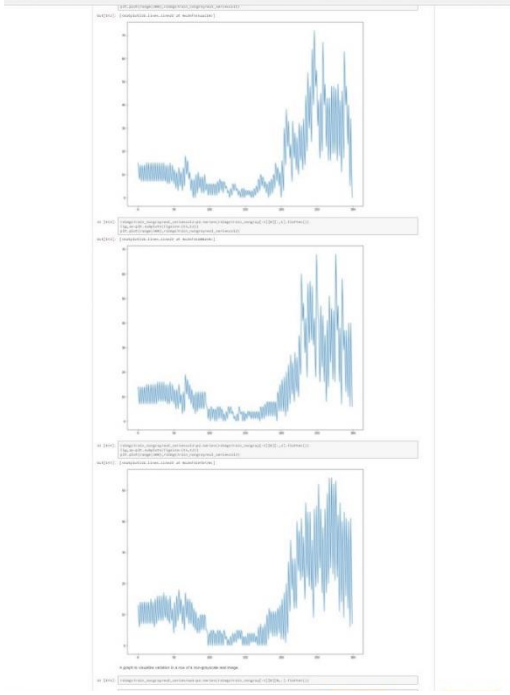
Pixel variation in three adjacent columns of nongrayscale GANs image



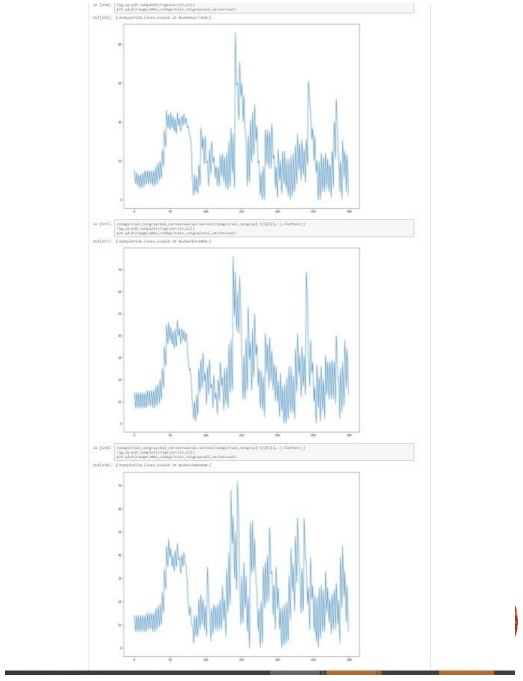
Pixel variation in three adjacent rows of nongrayscale GANs image



Pixel variation in three adjacent columns of nongrayscale real image



Pixel variation in three adjacent rows of nongrayscale real image



MODELS

- Keras neural network models were built with these specifications
  - Layer 1 rescales to standardize images
  - Layer 2 flattens the (100,100,3) structure of images
  - Layer 3 densely connected neural network layer with 128 nodes and relu activation function
  - Layer 4 densely connected neural network layer tapering to 64 nodes [variations with same 128 nodes and double 256 nodes] and relu activation function
  - Layer 5 output with 2 nodes, as this is binary classification, and softmax activation function

- Source of code https://keras.io/getting_started/intro_to_keras_for_engineers/

MODEL COMPILATION

- Adam optimizer
  - Stochastic gradient descent method with greater probability of finding global maximum
  - Applied with default learning rate of 0.001
- Sparse Categorical Cross Entropy loss function
  - Applied with >= 2 labels
  - Relevant with labels as integers
    - GANs images labeled with 0
    - Real images labeled with 1
- SparseCategoricalAccuracy metric
  - Relevant to Sparse Categorical Cross Entropy loss function
  - Calculates percentage of accurately labeled validation images

MODEL TRAINING

This Keras neural network model was trained on 2000 standardized grayscale images and 200 validation test images.

After some experimentation with number of epochs 300 was chosen to prevent overfitting while keeping a reasonable training accuracy level ~85%

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
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

OVERFITTING IN TRAINING OF MODELS

- Given enough iterations some models plateau on a level of accuracy while some models reach 100% training_accuracy
- In this instance 1000 epochs (iter'n) was sufficient to achieve perfect accuracy on training data
- The 'number of iterations' parameter in fitting of models was fine-tuned to 300 at which level training accuracy is a reasonable percentage in 80s while overfitting is minimized

fitting models							
model type	epochs (iter'n)	node structure	tr accuracy	valid'n accuracy	# images	batch_size	grayscale
model_g2	800	taper	97.8	66	2000	2000	y
model_g2	1000	taper	99.65	65	2000	2000	y

GENERALIZABILITY OF MODEL

- As validation_accuracy of model started at 63.5% right away on first iteration there is significant generalizability of model trained on color images without extensive processing

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 FITTING

- A Keras neural network fitted on 2000 grayscale training images and validated on 200 test images resulted in a maximum validation_accuracy of ~67.5%

- A Keras neural network fitted on 20000 training images and validated on 800 test images resulted in a maximum validation_accuracy of ~73.5%
- Increasing number of images seems to increase validation_accuracy!

fitting models							
model type	epochs (iter'n)	node structure	tr accuracy	valid'n accuracy	# images	batch	grayscale
model_c	800	same	82.25	73.5	20000	10000	n

FUTURE AVENUES OF RESEARCH

- Watermarking
- Convolutional Neural Networks
- Etc.