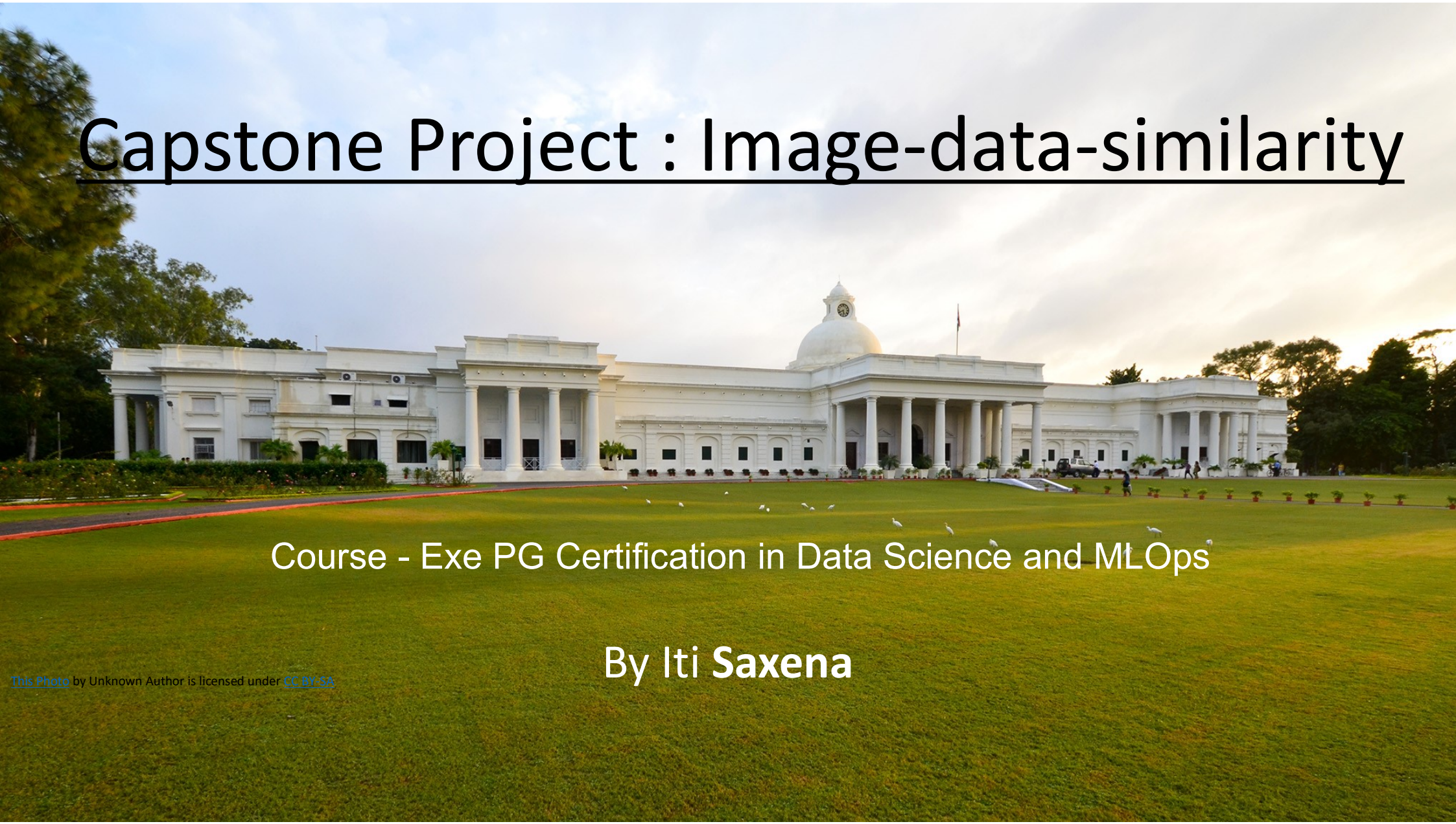
Capstone Project : Image-data-similarity



Course - Exe PG Certification in Data Science and MLOps

By Iti **Saxena**

Content

- Objective
- Approach to solving the problem
- Model summary
- Results
- Inference
- References
- Other comments (which may include how to further fine-tune the model, etc.)

Objective

- Business Scenario– For a fashion app/website, a smart recommendation module has to be developed that creates personalized experiences for customers every time FAI (Fashion AI) recommends items to order based on what customers are already ordering and increases basket size.
- Image-data-similarity solutions compare items/images and returns a value that tells how visually similar they are. Similar looking items then can be recommended to the app/website user.

Approach to solving the problem

- Approach – **Convolutional_AutoEncoder_Model**
- Step 1- Generate vectorized form of images. (OpenCV and numpy)
- Step 2- Model creation : Implementated Convolutional Autoencoder Model. In which we have created many convolution layers.
- Step 3- Layers are the building blocks of the convolution neural network.
  - Conv2D(filters, kernel_size, activation = 'reLu'): The kernel_size is the height and width of the 2D convolution window.
  - The padding specifies what to do when the filter does not fit the input image well. padding='valid' means dropping the part of the image when the filter does not fit; padding='same' pads the picture with zeros to fit the picture.
- Step 4- So the input for the model will be the image that we are looking for and the output that the model will give the list of top images similar to that of input images.

Model Summary

```
Model: "Convolutional_AutoEncoder_Model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Encoding_Conv2D_1 (Conv2D)   (None, 224, 224, 64)      1792
_________________________________________________________________
Encoding_MaxPooling2D_1 (Max (None, 112, 112, 64)      0
_________________________________________________________________
Encoding_Conv2D_2 (Conv2D)   (None, 112, 112, 128)     73856
_________________________________________________________________
Encoding_MaxPooling2D_2 (Max (None, 56, 56, 128)       0
_________________________________________________________________
Encoding_Conv2D_3 (Conv2D)   (None, 56, 56, 256)       295168
_________________________________________________________________
Encoding_MaxPooling2D_3 (Max (None, 28, 28, 256)       0
_________________________________________________________________
Encoding_Conv2D_4 (Conv2D)   (None, 28, 28, 512)       1180160
_________________________________________________________________
Encoding_MaxPooling2D_4 (Max (None, 14, 14, 512)       0
_________________________________________________________________
Encoding_Conv2D_5 (Conv2D)   (None, 14, 14, 512)       2359808
_________________________________________________________________
```

```
_________________________________________________________________
Decoding_Upsamping2D_1 (UpSa (None, 14, 14, 512)       0
_________________________________________________________________
Decoding_Conv2D_2 (Conv2D)   (None, 14, 14, 512)       2359808
_________________________________________________________________
Decoding_Upsamping2D_2 (UpSa (None, 28, 28, 512)       0
_________________________________________________________________
Decoding_Conv2D_3 (Conv2D)   (None, 28, 28, 256)       1179904
_________________________________________________________________
Decoding_Upsamping2D_3 (UpSa (None, 56, 56, 256)       0
_________________________________________________________________
Decoding_Conv2D_4 (Conv2D)   (None, 56, 56, 128)       295040
_________________________________________________________________
Decoding_Upsamping2D_4 (UpSa (None, 112, 112, 128)     0
_________________________________________________________________
Decoding_Conv2D_5 (Conv2D)   (None, 112, 112, 64)      73792
_________________________________________________________________
Decoding_Upsamping2D_5 (UpSa (None, 224, 224, 64)      0
_________________________________________________________________
Decoding_Output (Conv2D)     (None, 224, 224, 3)       1731
=================================================================
Total params: 10,180,867
Trainable params: 10,180,867
Non-trainable params: 0
_________________________________________________________________
```

Results

- Used early_stopping for model training and at 17th epoch, test loss and validation loss became absolutely similar ie, ~0.0100

```
Epoch 16/25
237/237 [==============================] - 23s 96ms/step - loss: 0.0102 - val_loss: 0.0106
Epoch 17/25
237/237 [==============================] - 23s 95ms/step - loss: 0.0100 - val_loss: 0.0105
Epoch 00017: early stopping
```

Inference

- The idea of Convolution autoencoder model is to train a model with image data as the inputs, and their respective similare image data as the outputs.

Other comments

- Convolutional_AutoEncoder_Model is not the only way to solve this problem, solutions like cosine similarity, embeddings can also be used.

Thank You!!