MARQUETTE UNIVERSITY
College of Business Administration

INSY 4054: Emerging Technologies

Week 5-2

AI Project

Learning Objectives

- To practice the concepts we learned previously
- To learn a new model through the project
- To gain confidence to work with new models

Project Description

- In this project, we want to create two main models.
- The first model can be used to denoise noisy images [we call this autoencoder_one]
- The second model can be used to compress images on a device and decompress them on another device, decreasing the bandwidth needed to send images over the network [we call this autoencoder_two]

Datasets

- To start, open the Colab Notebook below, import the required libraries, and run the first four cells designed to load and view the datasets.
  - https://colab.research.google.com/github/saffarizadeh/INSY4054/blob/main/AI_Project_Student_Version.ipynb
- We use x_train_noisy as input and x_train as output to train the autoencoder_one
- We use x_train as both input and output to train autoencoder_two

autoencoder_one

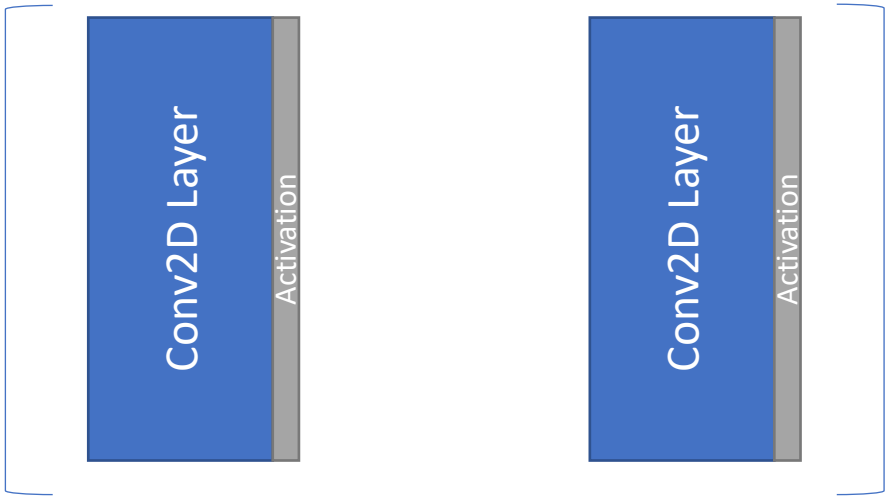
autoencoder_one

- The goal is to create a model that can remove noise from images
- To do so, we train a model that receives noisy images and outputs denoised images
- To achieve this, we need to create three models:
  1. A model called `encoder_one`
  2. A model called `decoder_one`
  3. A model called `autoencoder_one`
- On the next three slide, you can see how each model should look like

encoder_one

Number of filters: 16
Kernel size: (3,3)
Activation: 'relu'
Padding: 'same'
Strides: 2

Number of filters: 8
Kernel size: (3,3)
Activation: 'relu'
Padding: 'same'
Strides: 2



decoder_one

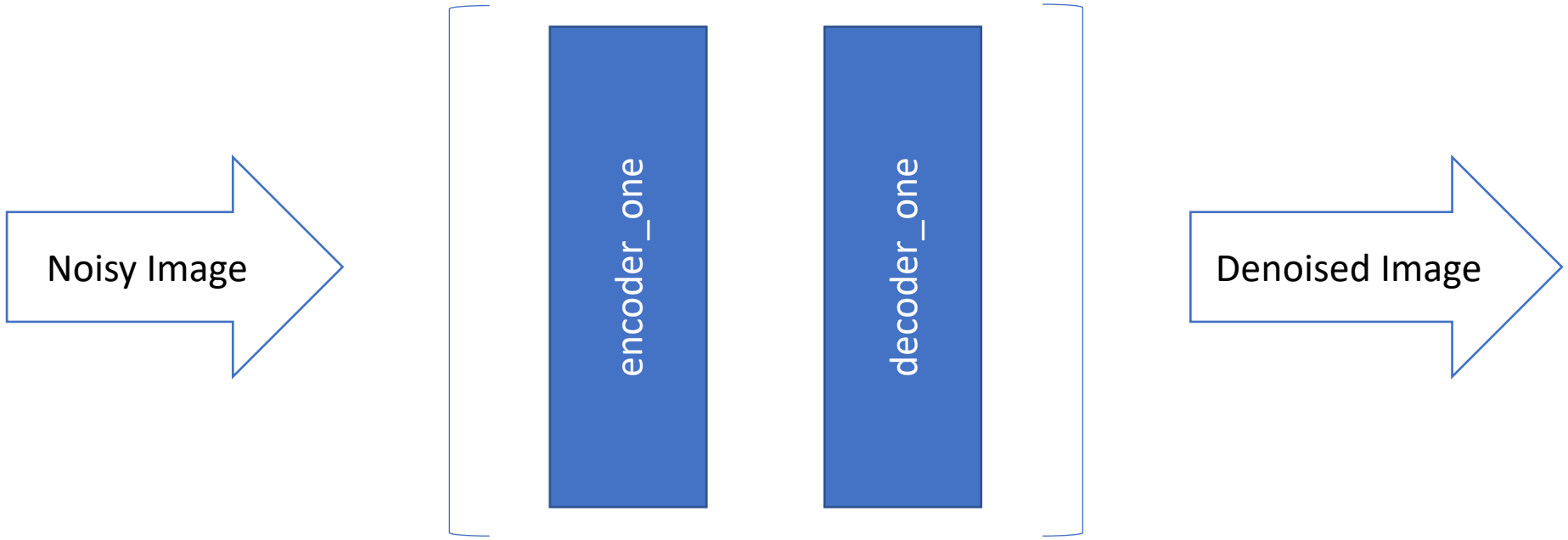
```
tf.keras.layers.Conv2DTranspose(16, (3,3),
        activation='relu', padding='same', strides=2)
```

Number of filters: 8
Kernel size: (3,3)
Activation: 'relu'
Padding: 'same'
Strides: 2

Number of filters: 16
Kernel size: (3,3)
Activation: 'relu'
Padding: 'same'
Strides: 2

Number of filters: 1
Kernel size: (3,3)
Activation: 'sigmoid'
Padding: 'same'

autoencoder_one



Optimization Information

- We only need to train the autoencoder_one model
  - Use adam optimizer
  - Use mean_squared_error as the loss function

autoencoder_two

autoencoder_two

- The goal is to create a model that can compress and decompress images
- To do so, we train a model that receives some images as input and outputs the same images
- To achieve this, we need to create three models:
  1. A model called `encoder_two`
  2. A model called `decoder_two`
  3. A model called `autoencoder_two`
- On the next three slide, you can see how each model should look like

encoder_two

Number of nodes: 64
Activation: 'relu'

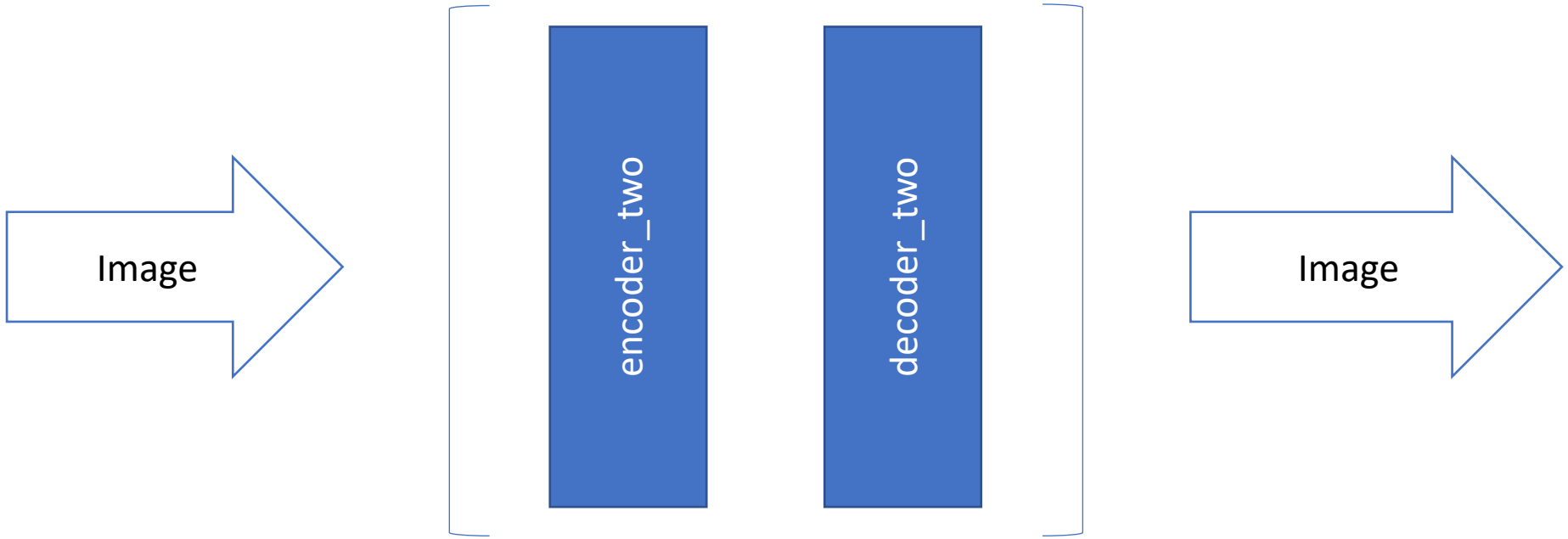
decoder_two

```
tf.keras.layers.Reshape((28,28))
```

Number of nodes: 784
Activation: 'sigmoid'

Reshape: (28,28)

autoencoder_two



Optimization Information

- We only need to train the autoencoder_two model
  - Use adam optimizer
  - Use mean_squared_error as the loss function

Submit the notebook

- After you finished the project, from the top menu bar click on File and then “download .ipynb”
  - Make sure you download “.ipynb” and not “.py”
- Submit the downloaded file

Forum Discussion

- Discuss on the forum:
  - How do you think the first autoencoder works [starting point: pay attention to how the dimensions change at each layer and compare the dimensions of input and output for each model]
  - Why do you think the second autoencoder can be used to compress images?
  - Can you think of any use-cases for these models?
- Reminder:
  - Read one of the TensorFlow case studies mentioned last session and share your takeaways on the forum
  - Take a look at other students' takeaways as well