

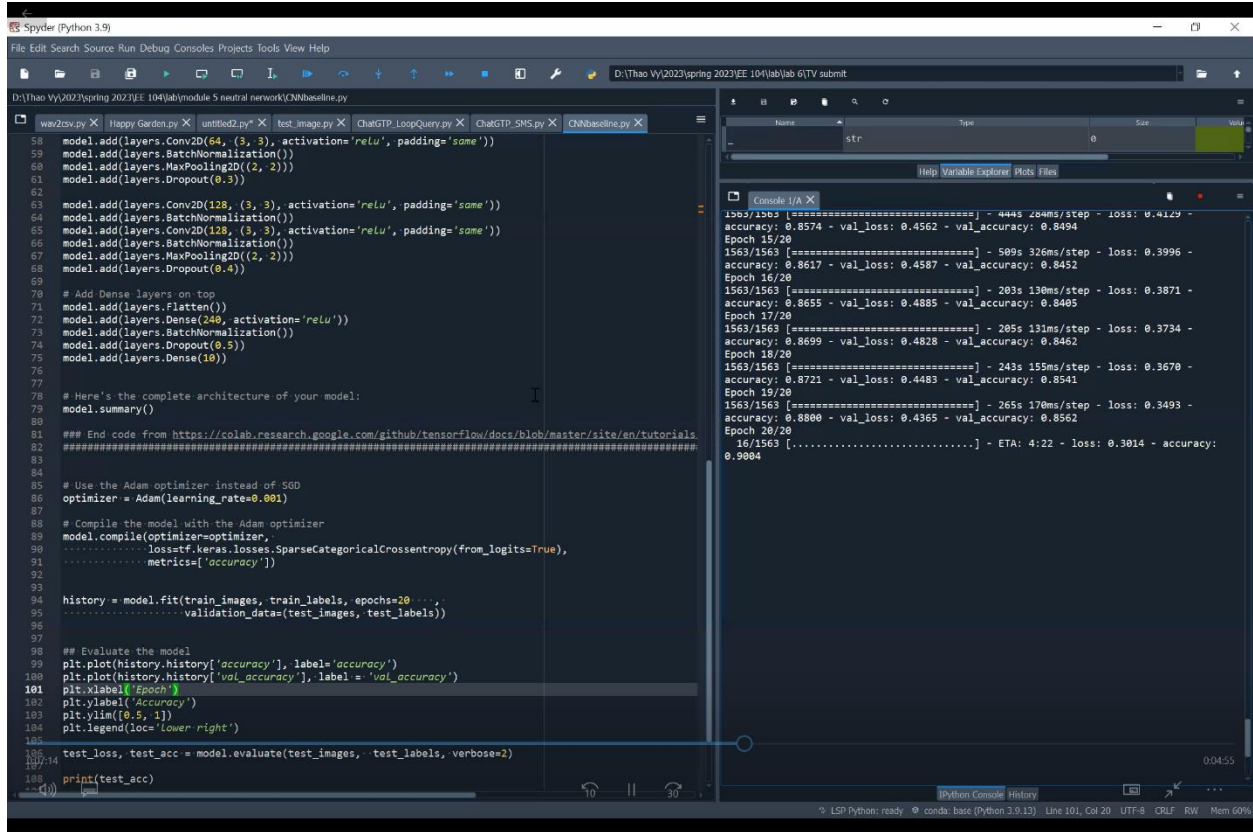
Name Thao Vy Tran

EE 104

Please watch this link for CNN: <https://youtu.be/zEFB9GVN1YU>

Link video: <https://youtu.be/Axn6WqiNRaA>

## 1. CNNbaseline



The screenshot shows the Spyder Python IDE with a file named 'CNNbaseline.py' open. The code defines a CNN architecture with three convolutional layers, two max pooling layers, and three dense layers. It uses the Adam optimizer and Keras for training and evaluation. The console output shows the training progress over 20 epochs, including accuracy, loss, and validation metrics.

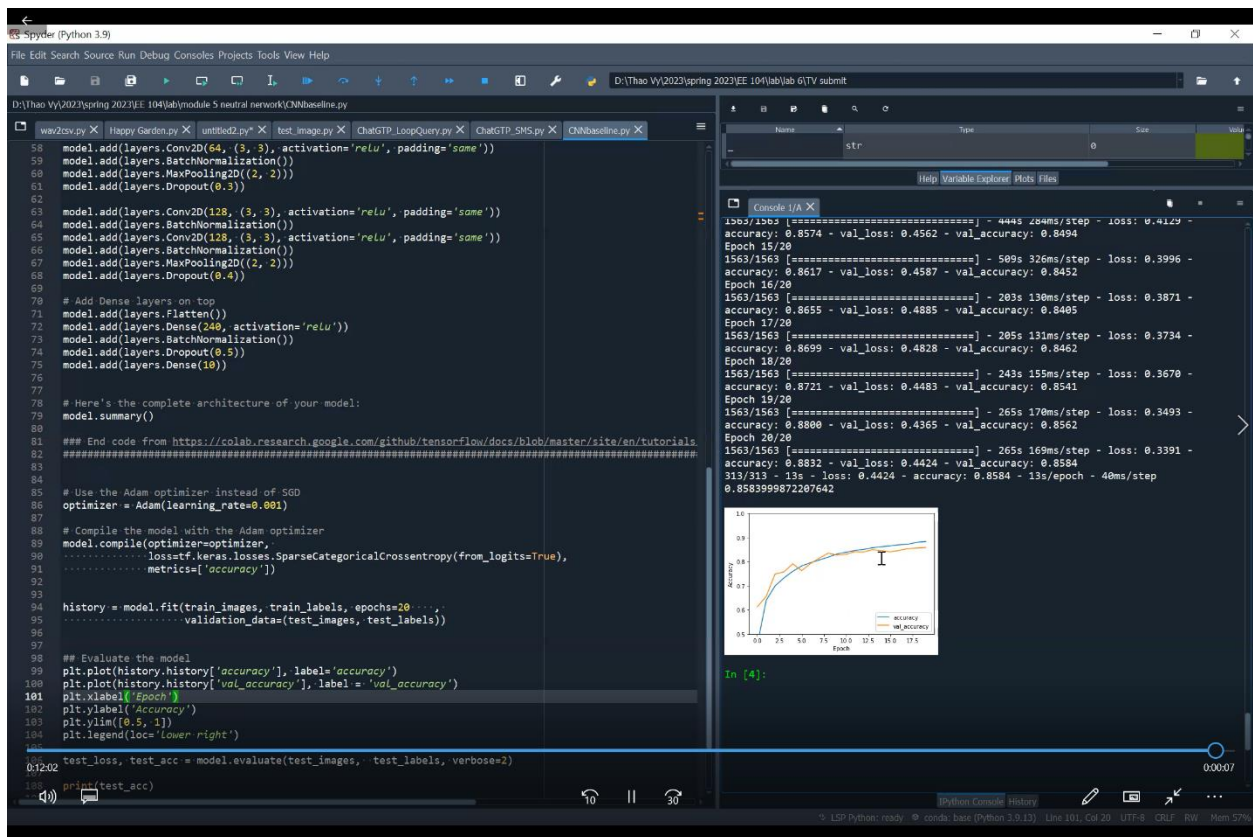
```
58 model.add(layers.Conv2D(64, (3, 3), activation='relu', padding='same'))
59 model.add(layers.BatchNormalization())
60 model.add(layers.MaxPooling2D((2, 2)))
61 model.add(layers.Dropout(0.4))
62
63 model.add(layers.Conv2D(128, (3, 3), activation='relu', padding='same'))
64 model.add(layers.BatchNormalization())
65 model.add(layers.Conv2D(128, (3, 3), activation='relu', padding='same'))
66 model.add(layers.BatchNormalization())
67 model.add(layers.MaxPooling2D((2, 2)))
68 model.add(layers.Dropout(0.4))
69
70 # Add Dense layers on top
71 model.add(layers.Flatten())
72 model.add(layers.Dense(256, activation='relu'))
73 model.add(layers.BatchNormalization())
74 model.add(layers.Dropout(0.5))
75 model.add(layers.Dense(10))
76
77
78 # Here's the complete architecture of your model:
79 model.summary()
80
81 ## End code from https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials
82 #####
83
84
85 # Use the Adam optimizer instead of SGD
86 optimizer = Adam(learning_rate=0.001)
87
88 # Compile the model with the Adam optimizer
89 model.compile(optimizer=optimizer,
90               loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
91               metrics=['accuracy'])
92
93
94 history = model.fit(train_images, train_labels, epochs=20,
95                    validation_data=(test_images, test_labels))
96
97
98 ## Evaluate the model
99 plt.plot(history.history['accuracy'], label='accuracy')
100 plt.plot(history.history['val_accuracy'], label='val_accuracy')
101 plt.xlabel('Epoch')
102 plt.ylabel('Accuracy')
103 plt.ylim([0.5, 1])
104 plt.legend(loc='lower right')
105
106 test_loss, test_acc = model.evaluate(test_images, test_labels, verbose=2)
107 print(test_acc)
```

Console 1/A X

```
1563/1563 [#####] - 444s 284ms/step - loss: 0.4129 -
accuracy: 0.8574 - val_loss: 0.4562 - val_accuracy: 0.8494
Epoch 15/20
1563/1563 [#####] - 509s 326ms/step - loss: 0.3996 -
accuracy: 0.8617 - val_loss: 0.4587 - val_accuracy: 0.8452
Epoch 16/20
1563/1563 [#####] - 203s 130ms/step - loss: 0.3871 -
accuracy: 0.8655 - val_loss: 0.4885 - val_accuracy: 0.8485
Epoch 17/20
1563/1563 [#####] - 205s 131ms/step - loss: 0.3734 -
accuracy: 0.8699 - val_loss: 0.4828 - val_accuracy: 0.8462
Epoch 18/20
1563/1563 [#####] - 243s 155ms/step - loss: 0.3670 -
accuracy: 0.8721 - val_loss: 0.4483 - val_accuracy: 0.8541
Epoch 19/20
1563/1563 [#####] - 265s 170ms/step - loss: 0.3493 -
accuracy: 0.8800 - val_loss: 0.4365 - val_accuracy: 0.8562
Epoch 20/20
1563/1563 [#####] - ETA: 4:22 - loss: 0.3014 - accuracy:
0.9004
```

Python Console History

LSP Python: ready @ conda: base (Python 3.9.12) Line 101, Col 20 UTF-8 CRLF RW Mem 60%



## 2. Testimage.py

Spyder (Python 3.9)

File Edit Search Source Run Debug Consoles Projects Tools View Help

D:\Thao Vy\2023\spring 2023\EE 104\lab\lab 6\TV submit\CNN\test\_image.py

```
32 img = img.reshape(1, 32, 32, 3)
33 img = img / 255.0
34 return img
35
36
37 # load the trained CIFAR10 model
38 model = load_model('MyGroup_CIFARmodel.h5')
39
40 # get the image from the internet
41 URL = "https://ichef.bbci.co.uk/news/976/cr
42 picture_path = tf.keras.utils.get_file(or
43 img = load_image(picture_path)
44 result = model.predict(img)
45
46 # show the picture
47 image = plt.imread(picture_path)
48 plt.figure()
49 plt.imshow(image)
50
51 # show prediction result.
52 print('\nPrediction: This image most likel
53
54
55 # get the image from the internet
56 URL = "https://upload.wikimedia.org/wikipe
57 picture_path = tf.keras.utils.get_file(or
58 img = load_image(picture_path)
59 result = model.predict(img)
60
61 # show the picture
62 image = plt.imread(picture_path)
63 plt.figure()
64 plt.imshow(image)
65
66 # show prediction result.
67 print('\nPrediction: This image most likel
68
69
70 # get the image from the internet
71 URL = "https://www.kbb.com/articles/wp-con
72 picture_path = tf.keras.utils.get_file(or
73 img = load_image(picture_path)
74 result = model.predict(img)
75
76 # show the picture
77 image = plt.imread(picture_path)
78 plt.figure()
79 plt.imshow(image)
80
81 # show prediction result.
82 print('\nPrediction: This image most likel
83
```

Name	Type	Size	Value
	str	0	
	str	0	


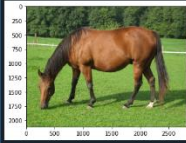

Console 20/A X

Prediction: This image most likely belongs to bird  
1/1 [=====] - 0s 33ms/step

Prediction: This image most likely belongs to horses  
1/1 [=====] - 0s 49ms/step

Prediction: This image most likely belongs to automobile  
1/1 [=====] - 0s 30ms/step

Prediction: This image most likely belongs to airplane



IPython Console History

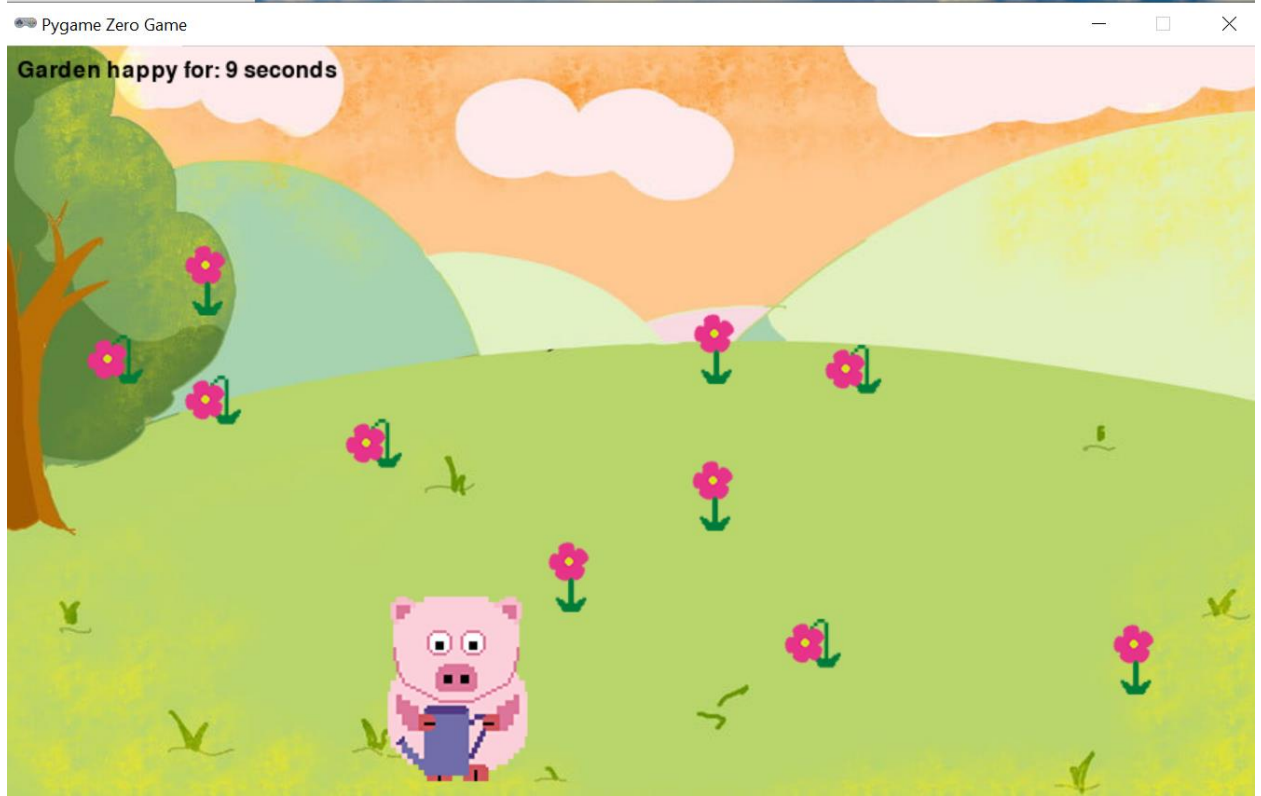
LSP Python: ready conda: base (Python 3.9.13) Line 57, Col 1 UTF-8 CRLF RW Mem 62%

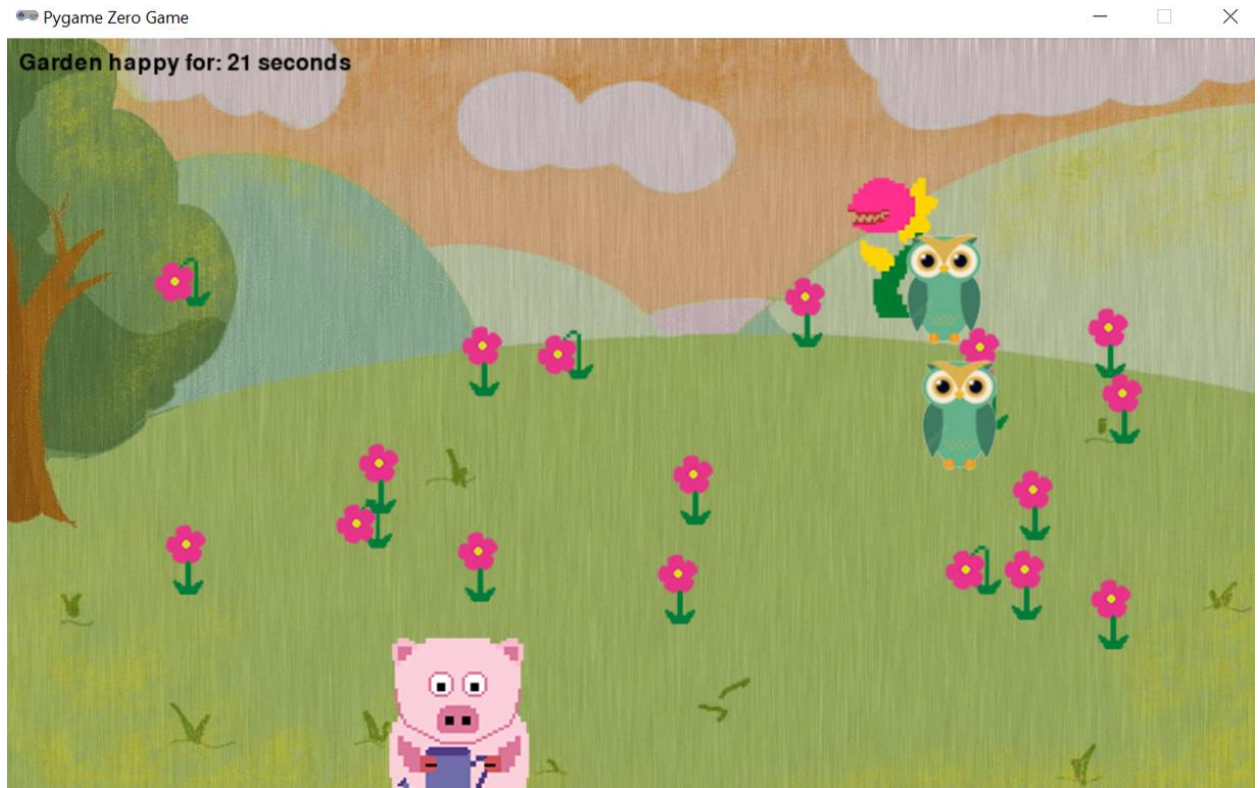
Type here to search

60°F 3:38 PM 4/7/2023

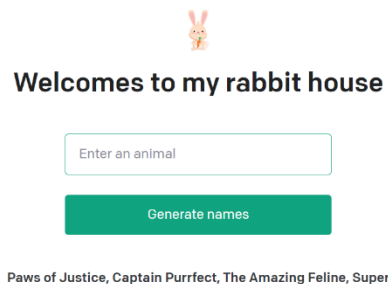
### 3. Happy Garden game

When not raining,





#### 4. Hello World to OpenAI



---

#### 5. Hello World to ChatGPT

```
Console 45/A X
Python 3.9.13 (main, Aug 25 2022, 23:51:50) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.31.1 -- An enhanced Interactive Python.

In [1]: runfile('D:/Thao Vy/2023/spring 2023/EE 104/lab/lab 6/ChatGTP_LoopQuery.py', wdir='D:/Thao
Vy/2023/spring 2023/EE 104/lab/lab 6')

What you want to ask ChatGPT: Type the question (or type 'quitme' to quit):
describe a dog

Loyal, friendly, and playful are a few words that come to mind when thinking about dogs. They
provide us with endless hours of amusement and are always down for a good time. From big to small,
there's a dog out there for everyone.

What you want to ask ChatGPT: Type the question (or type 'quitme' to quit):
```