COSC 4337

keras_fashion_graphviz

[1]:
```
pip install nnv
```

```
Requirement already satisfied: nnv in c:\users\rizkn\.conda\envs\tf1\lib\site-
packages (0.0.5)
Note: you may need to restart the kernel to use updated packages.
```

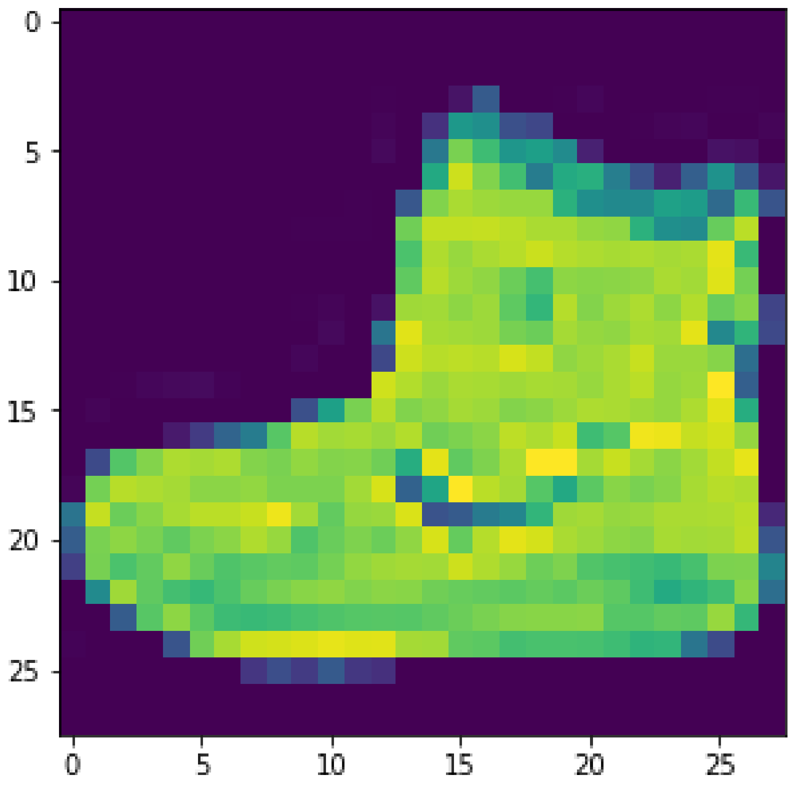
[2]:
```
import tensorflow as tf
from tensorflow import keras

import matplotlib.pyplot as plt
import numpy as np
```

[3]:
```
fashion_mnist = keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.
 ↪load_data()
```

[4]:
```
print(f"Train images dimensions: {train_images.shape}")
print(f"Test images dimensions: {test_images.shape}")
```

```
Train images dimensions: (60000, 28, 28)
Test images dimensions: (10000, 28, 28)
```

[5]:
```
plt.figure(figsize=(10,5))
plt.imshow(train_images[0])
plt.colormaps()
plt.show()
```



[6]:
```
train_images = train_images /255.
test_images = test_images / 255
```

[7]:
```
model = keras.Sequential([
    keras.layers.Flatten(input_shape=(28, 28)),
    keras.layers.Dense(128, activation=tf.nn.relu),
    keras.layers.Dense(10, activation=tf.nn.softmax)
])
```

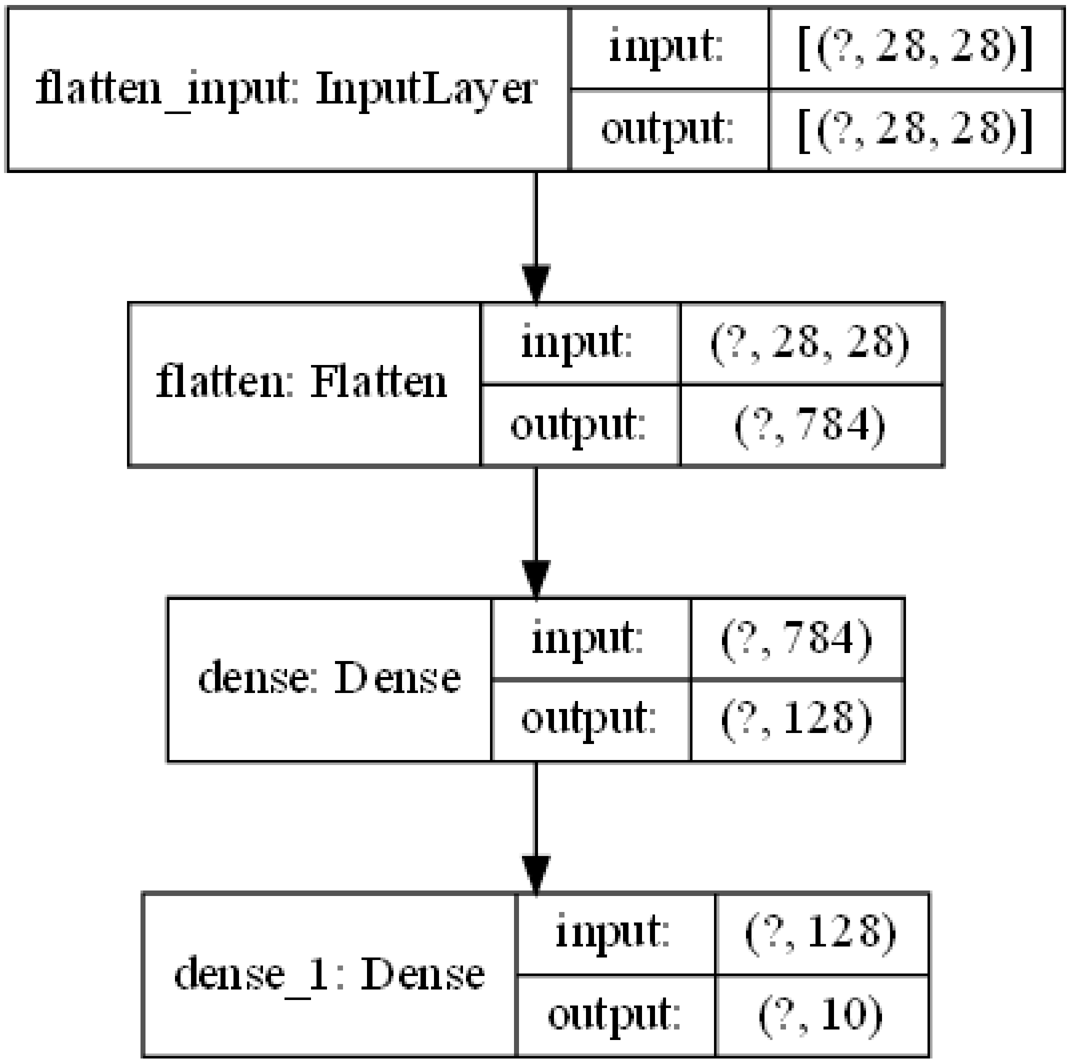
```
WARNING:tensorflow:From C:\Users\RizkN\.conda\envs\tf1\lib\site-
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
If using Keras pass *_constraint arguments to layers.
```

[8]:
```
#import pydot_ng as pydot
```

[9]:
```
from tensorflow.python.keras.utils.vis_utils import plot_model
```

[10]:
```
plot_model(model, to_file='model.png', show_shapes=True)
```

[10]:

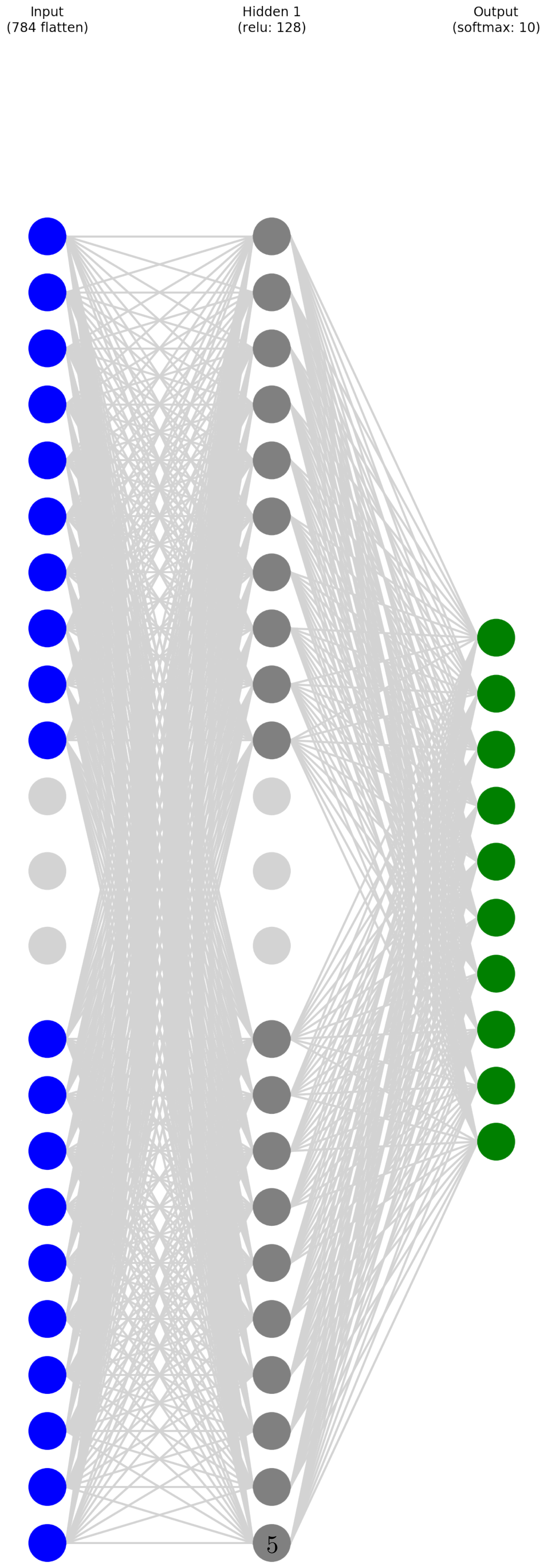


[11]:
```
import os
os.environ["PATH"] += os.pathsep + 'C:\Program Files\Graphviz\bin'
```

[12]:
```
#https://graphviz.org/download/
```

[13]:
```
#from keras.utils.vis_utils import plot_model
```

[14]:
```
from nnv import NNV
plt.rcParams["figure.figsize"] = 200,50

layersList = [
    {"title":"Input\n(784 flatten)", "units": 784, "color": "Blue"},
    {"title":"Hidden 1\n(relu: 128)", "units": 128},
    {"title":"Output\n(softmax: 10)", "units": 10,"color": "Green"},
]
NNV(layersList, spacing_layer=10, max_num_nodes_visible=20, node_radius=1,␣
 ↪font_size=24).render()
```


Input
(784 flatten)
Hidden 1
(relu: 128)
Output
(softmax: 10)

```
[14]: (<Figure size 14400x3600 with 1 Axes>,
       <matplotlib.axes._subplots.AxesSubplot at 0x16a9a337bc8>)
```

1 Train the model

```
[15]: model.compile(
          optimizer='adam',
          loss='sparse_categorical_crossentropy',
          metrics=['accuracy']
      )
```

```
[16]: # Begin Training
```

```
[17]: model.fit(train_images, train_labels, epochs=5)
```

```
Train on 60000 samples
Epoch 1/5
60000/60000 [==============================] - 2s 30us/sample - loss: 0.4994 -
acc: 0.8245
Epoch 2/5
60000/60000 [==============================] - 2s 32us/sample - loss: 0.3711 -
acc: 0.8648
Epoch 3/5
60000/60000 [==============================] - 2s 32us/sample - loss: 0.3345 -
acc: 0.8782
Epoch 4/5
60000/60000 [==============================] - 2s 30us/sample - loss: 0.3117 -
acc: 0.8851
Epoch 5/5
60000/60000 [==============================] - 2s 31us/sample - loss: 0.2916 -
acc: 0.8920
```

```
[17]: <tensorflow.python.keras.callbacks.History at 0x16a9a9ca608>
```

```
[18]: #Evaluating Our Model
```

```
[19]: test_loss, test_acc = model.evaluate(test_images, test_labels)
      print(f"Model Accuracy: {test_acc * 100}%")
```

```
10000/10000 [==============================] - 0s 17us/sample - loss: 0.3446 -
acc: 0.8744
Model Accuracy: 87.44000196456909%
```

```
[20]: # Predictions
```

[21]:
```
predictions = model.predict(test_images)
predictions[1]
```

[21]:
```
array([7.3403758e-06, 1.2584223e-07, 9.9378633e-01, 1.1884340e-07,
       4.1631521e-03, 9.2803152e-12, 2.0428218e-03, 3.2963191e-15,
       8.7890584e-08, 1.7815925e-14], dtype=float32)
```

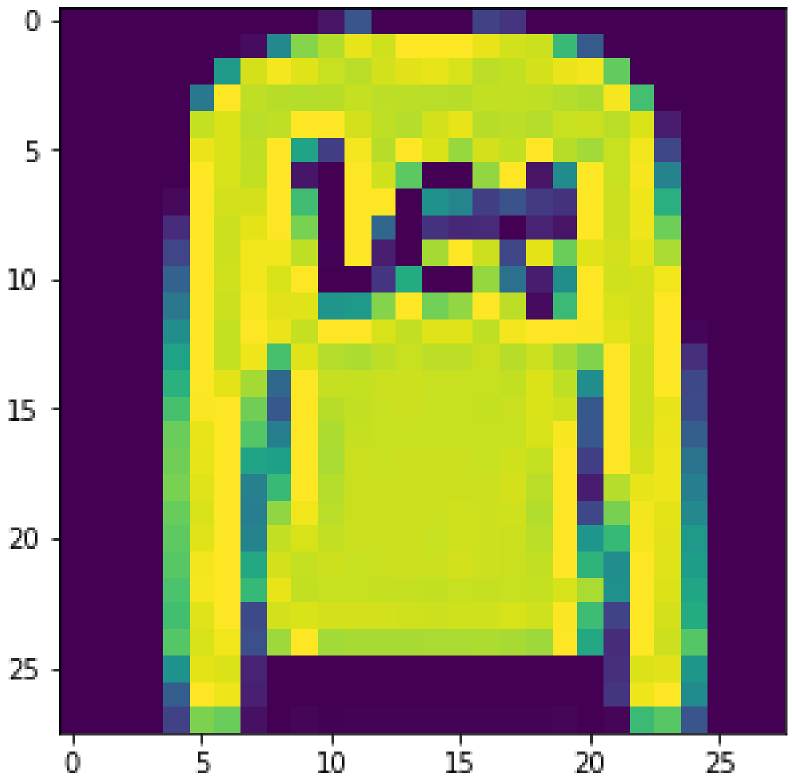
[22]:
```
np.argmax(predictions[1])
```

[22]:
```
2
```

[23]:
```
#verify predictions
test_labels[1]
```

[23]:
```
2
```

[24]:
```
plt.figure(figsize=(10,5))
plt.imshow(test_images[1])
plt.colormaps()
plt.show()
```



[ ]:
```
```

```
[ ]:
```