In [1]:

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

In [2]:

```
fashion_mnist = tf.keras.datasets.fashion_mnist
(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-l
abels-idx1-ubyte.gz
29515/29515 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/train-i
mages-idx3-ubyte.gz
26421880/26421880 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-la
bels-idx1-ubyte.gz
5148/5148 [==============================] - 0s 0us/step
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/t10k-im
ages-idx3-ubyte.gz
4422102/4422102 [==============================] - 0s 0us/step
```

In [3]:

```
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat', 'Sandal', 'Shirt',
'Sneaker', 'Bag', 'Ankle boot']
```

In [4]:

```
train_images.shape
```

Out[4]:

```
(60000, 28, 28)
```

In [5]:

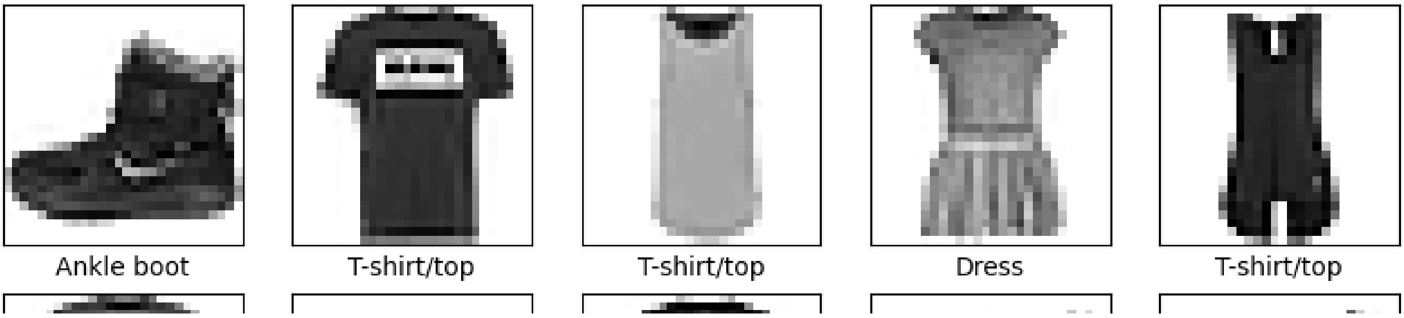
```
train_images = train_images / 255.0
test_images = test_images / 255.0
```

In [6]:

```
# Verify data is in correct format
plt.figure(figsize = (10, 10))
for i in range(25):
  plt.subplot(5, 5, i + 1)
  plt.xticks([])
  plt.yticks([])
  plt.grid(False)
  plt.imshow(train_images[i], cmap = plt.cm.binary)
  plt.xlabel(class_names[train_labels[i]])
plt.show
```

Out[6]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



Pullover
Sneaker
Pullover
Sandal
Sandal
T-shirt/top
Ankle boot
Sandal
Sandal
Sneaker
Ankle boot
Trouser
T-shirt/top
Shirt
Coat
Dress
Trouser
Coat
Bag
Coat

In [7]:

```
from keras.models import Sequential
from keras.layers import Dense

model = Sequential()
model = tf.keras.Sequential([
    tf.keras.layers.Flatten(input_shape = (28, 28)),
    tf.keras.layers.Dense(128, activation = 'relu'),
    tf.keras.layers.Dense(10)
])

model.compile(optimizer = 'adam', loss = tf.keras.losses.SparseCategoricalCrossentropy(f
rom_logits = True), metrics = ['accuracy'])
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 flatten (Flatten)           (None, 784)               0         
                                                                 
 dense (Dense)               (None, 128)               100480    
                                                                 
 dense_1 (Dense)             (None, 10)                1290      
                                                                 
=================================================================
Total params: 101,770
Trainable params: 101,770
Non-trainable params: 0
_________________________________________________________________
```

In [8]:

```
model.fit(train_images, train_labels, epochs = 10)
```

```
Epoch 1/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.5014 - accuracy: 0.824
9
Epoch 2/10
1875/1875 [==============================] - 8s 5ms/step - loss: 0.3798 - accuracy: 0.863
5
Epoch 3/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.3409 - accuracy: 0.876
1
Epoch 4/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.3150 - accuracy: 0.884
0
Epoch 5/10
1875/1875 [==============================] - 8s 4ms/step - loss: 0.2947 - accuracy: 0.892
0
Epoch 6/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2790 - accuracy: 0.896
3
Epoch 7/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2676 - accuracy: 0.900
4
Epoch 8/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2555 - accuracy: 0.905
0
Epoch 9/10
1875/1875 [==============================] - 7s 4ms/step - loss: 0.2473 - accuracy: 0.908
1
Epoch 10/10
1875/1875 [==============================] - 6s 3ms/step - loss: 0.2402 - accuracy: 0.910
6
```

Out[8]:

```
<keras.callbacks.History at 0x7f9b451b2d10>
```

In [9]:

```
test_loss, test_acc = model.evaluate(test_images, test_labels, verbose = 2)
print("\n Test accuracy = ", test_acc)
```

```
313/313 - 1s - loss: 0.3345 - accuracy: 0.8803 - 1s/epoch - 5ms/step

 Test accuracy =  0.8802999854087830
```

In [10]:

```
probability_model = tf.keras.Sequential([model, tf.keras.layers.Softmax()])
```

In [11]:

```
predictions = probability_model.predict(test_images)
```

```
313/313 [==============================] - 1s 2ms/step
```

In [12]:

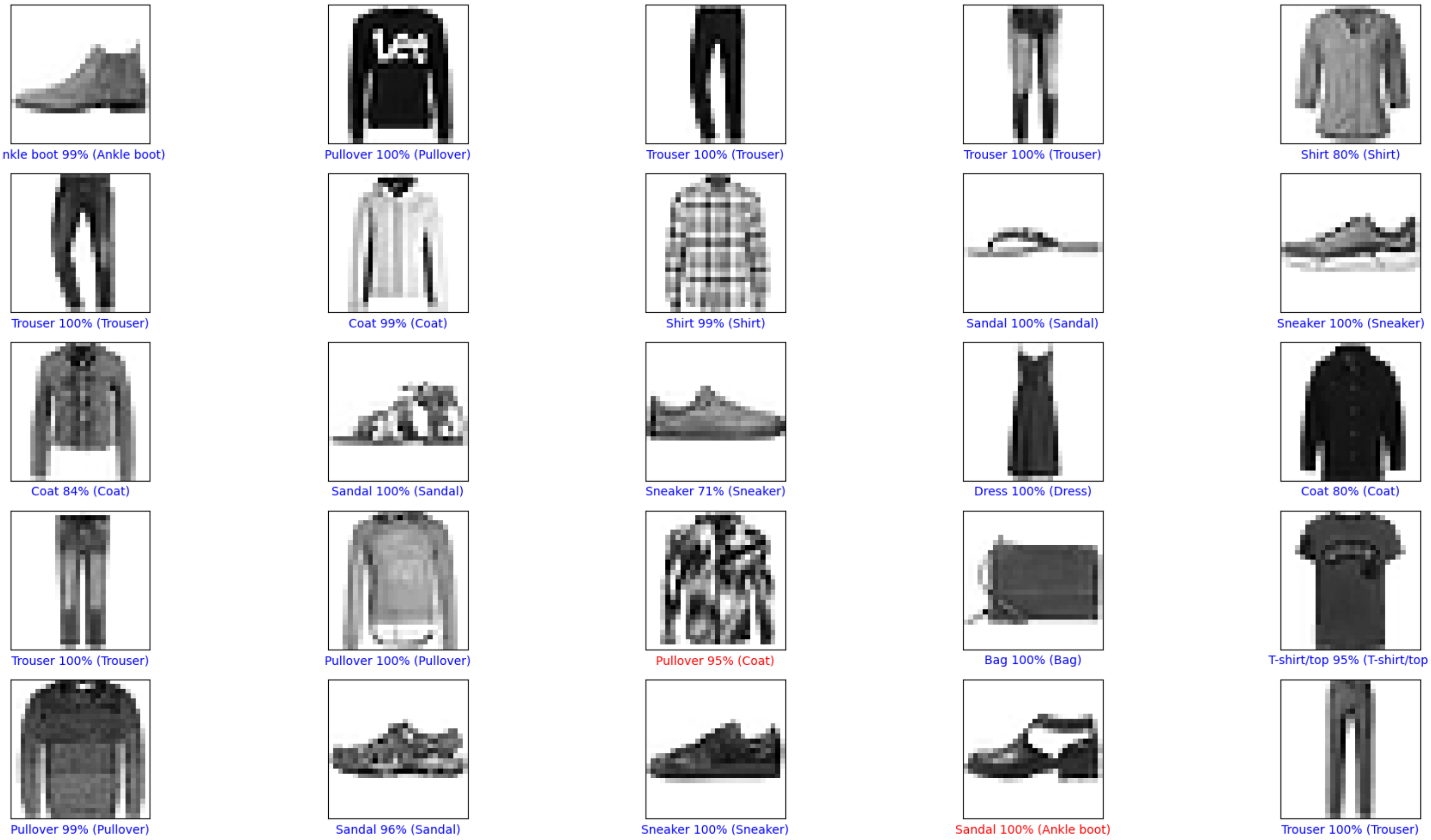
```
def plot_image(i, predictions_array, true_label, img):
  true_label, img = true_label[i], img[i]
  plt.grid(False)
  plt.xticks([])
  plt.yticks([])

  plt.imshow(img, cmap=plt.cm.binary)

  predicted_label = np.argmax(predictions_array)
  if predicted_label == true_label:
    color = 'blue'
  else:
    color = 'red'
  plt.xlabel("{} {:2.0f}% ({})".format(class_names[predicted_label],
                                100*np.max(predictions_array),
                                class_names[true_label]),
                                color=color)
```

In [14]:

```
rows = 5
cols = 5
total_images = rows * cols
plt.figure(figsize = (2*2*cols, 2*rows))
for i in range(total_images):
  plt.subplot(rows, cols, i + 1)
  plot_image(i, predictions[i], test_labels, test_images)
plt.tight_layout()
plt.show()
```


nkle boot 99% (Ankle boot)
Pullover 100% (Pullover)
Trouser 100% (Trouser)
Trouser 100% (Trouser)
Shirt 80% (Shirt)
Trouser 100% (Trouser)
Coat 99% (Coat)
Shirt 99% (Shirt)
Sandal 100% (Sandal)
Sneaker 100% (Sneaker)
Coat 84% (Coat)
Sandal 100% (Sandal)
Sneaker 71% (Sneaker)
Dress 100% (Dress)
Coat 80% (Coat)
Trouser 100% (Trouser)
Pullover 100% (Pullover)
Pullover 95% (Coat)
Bag 100% (Bag)
T-shirt/top 95% (T-shirt/top
Pullover 99% (Pullover)
Sandal 96% (Sandal)
Sneaker 100% (Sneaker)
Sandal 100% (Ankle boot)
Trouser 100% (Trouser)