In [1]:

```
from tensorflow.keras.datasets import fashion_mnist
```

In [2]:

```
import numpy as np
import tensorflow as tf

from tensorflow.keras.utils import to_categorical

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

from tensorflow.keras.optimizers import Adam

import matplotlib.pyplot as plt
```

In [3]:

```
(x_train_fashion, t_train_fashion), (x_test_fashion, t_test_fashion) = fashion_mnist.load_data()

print("x_train_fashion : ", np.shape(x_train_fashion))
print("t_train_fashion : ", np.shape(t_train_fashion))
print("x_test_fashion : ", np.shape(x_test_fashion))
print("t_test_fashion : ", np.shape(t_test_fashion))
```

```
x_train_fashion :  (60000, 28, 28)
t_train_fashion :  (60000,)
x_test_fashion :  (10000, 28, 28)
t_test_fashion :  (10000,)
```

In [4]:

```
label_names = ["T-shirt/top", "Trousser", "Pullover", "Dress", "Coat", "Sandal", "Shirt", "Sneaker", "Bag", "Ankel boot"]

plt.figure(figsize = (10, 10))

for i in range(25):
    plt.subplot(5, 5, i + 1)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(x_train_fashion[i], cmap = 'gray')
    plt.xlabel(label_names[t_train_fashion[i]])

plt.show()
```

In [5]:

```
x_train_fashion = x_train_fashion.astype("float32") / 255.0
x_test_fashion = x_test_fashion.astype("float32") / 255.0

t_train_fashion = to_categorical(t_train_fashion)
t_test_fashion = to_categorical(t_test_fashion)

print("One-hot Vector 적용 후 t_train shape : ", t_train_fashion.shape)
print("One-hot Vector 적용 후 t_test shape : ", t_test_fashion)
```

```
One-hot Vector 적용 후 t_train shape :  (60000, 10)
One-hot Vector 적용 후 t_test shape :  [[0. 0. 0. ... 0. 0. 1.]
 [0. 0. 1. ... 0. 0. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 ...
 [0. 0. 0. ... 0. 1. 0.]
 [0. 1. 0. ... 0. 0. 0.]
 [0. 0. 0. ... 0. 0. 0.]]
```

In [6]:

```
width = 28
height = 28
channel = 1

model = Sequential(name = 'Fashion_MNIST_CNN')

model.add(Conv2D(filters = 32, kernel_size = (3, 3), padding='same', activation='relu', input_shape = (width, height, channe

model.add(MaxPooling2D(pool_size=(2, 2)))


model.add(Conv2D(filters = 64, kernel_size = (3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters = 128, kernel_size = (3, 3), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())

model.add(Dense(10, activation='softmax'))
model.compile(loss = 'categorical_crossentropy', optimizer = Adam(learning_rate = 0.001), metrics = ['accuracy'])
model.summary()
```

```
Model: "Fashion_MNIST_CNN"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 28, 28, 32)        320

 max_pooling2d (MaxPooling2D  (None, 14, 14, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 14, 14, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 7, 7, 64)         0
 2D)

 conv2d_2 (Conv2D)           (None, 7, 7, 128)         73856

 max_pooling2d_2 (MaxPooling  (None, 3, 3, 128)        0
 2D)

 flatten (Flatten)           (None, 1152)              0

 dense (Dense)               (None, 10)                11530

=================================================================
Total params: 104,202
Trainable params: 104,202
Non-trainable params: 0
_________________________________________________________________
```

In [7]:

```
model.fit(x_train_fashion, t_train_fashion, epochs = 10, batch_size = 16)
```

```
Epoch 1/10
3750/3750 [==============================] - 79s 21ms/step - loss: 0.3965 - accuracy: 0.8560
Epoch 2/10
3750/3750 [==============================] - 79s 21ms/step - loss: 0.2658 - accuracy: 0.9028
Epoch 3/10
3750/3750 [==============================] - 80s 21ms/step - loss: 0.2242 - accuracy: 0.9176
Epoch 4/10
3750/3750 [==============================] - 83s 22ms/step - loss: 0.1960 - accuracy: 0.9280
Epoch 5/10
3750/3750 [==============================] - 81s 22ms/step - loss: 0.1718 - accuracy: 0.9365
Epoch 6/10
3750/3750 [==============================] - 83s 22ms/step - loss: 0.1525 - accuracy: 0.9440
Epoch 7/10
3750/3750 [==============================] - 85s 23ms/step - loss: 0.1336 - accuracy: 0.9510
Epoch 8/10
3750/3750 [==============================] - 92s 25ms/step - loss: 0.1192 - accuracy: 0.9565
Epoch 9/10
3750/3750 [==============================] - 102s 27ms/step - loss: 0.1049 - accuracy: 0.9613
Epoch 10/10
3750/3750 [==============================] - 90s 24ms/step - loss: 0.0915 - accuracy: 0.9650
```

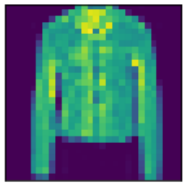
Out[7]:

```
<keras.callbacks.History at 0x2d468204e50>
```

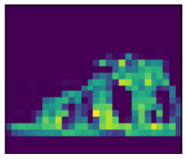
In [9]:

```
28
 28
= 1

abel_names = ["T-shirt/top", "Trousser", "Pullover", "Dress", "Coat", "Sandal", "Shirt", "Sneaker", "Bag", "Ankel boot"]

 range(10, 20):
igure(figsize = (2, 2))

ut = model.predict(x_test_fashion[i].reshape(1, width, height, channel))

ticks([])
yticks([])
mshow(x_test_fashion[i].reshape(width, height, channel))

("예측 : " + fashion_label_names[np.argmax(output)] + '/ 정답 : ' + fashion_label_names[np.argmax(t_test_fashion[i])])
show()
```

```
1/1 [==============================] - 0s 21ms/step
예측 : Coat/ 정답 : Coat
```



```
1/1 [==============================] - 0s 19ms/step
예측 : Sandal/ 정답 : Sandal
```



In [10]:

```
loss, accuracy = model.evaluate(x_test_fashion, t_test_fashion, verbose = 1)

print("test loss : ", round(loss, 6))
print("test accuracy : ", round(accuracy * 100, 3), "%")
```

```
313/313 [==============================] - 4s 11ms/step - loss: 0.3043 - accuracy: 0.9156
test loss :  0.304308
test accuracy :  91.56 %
```