Result

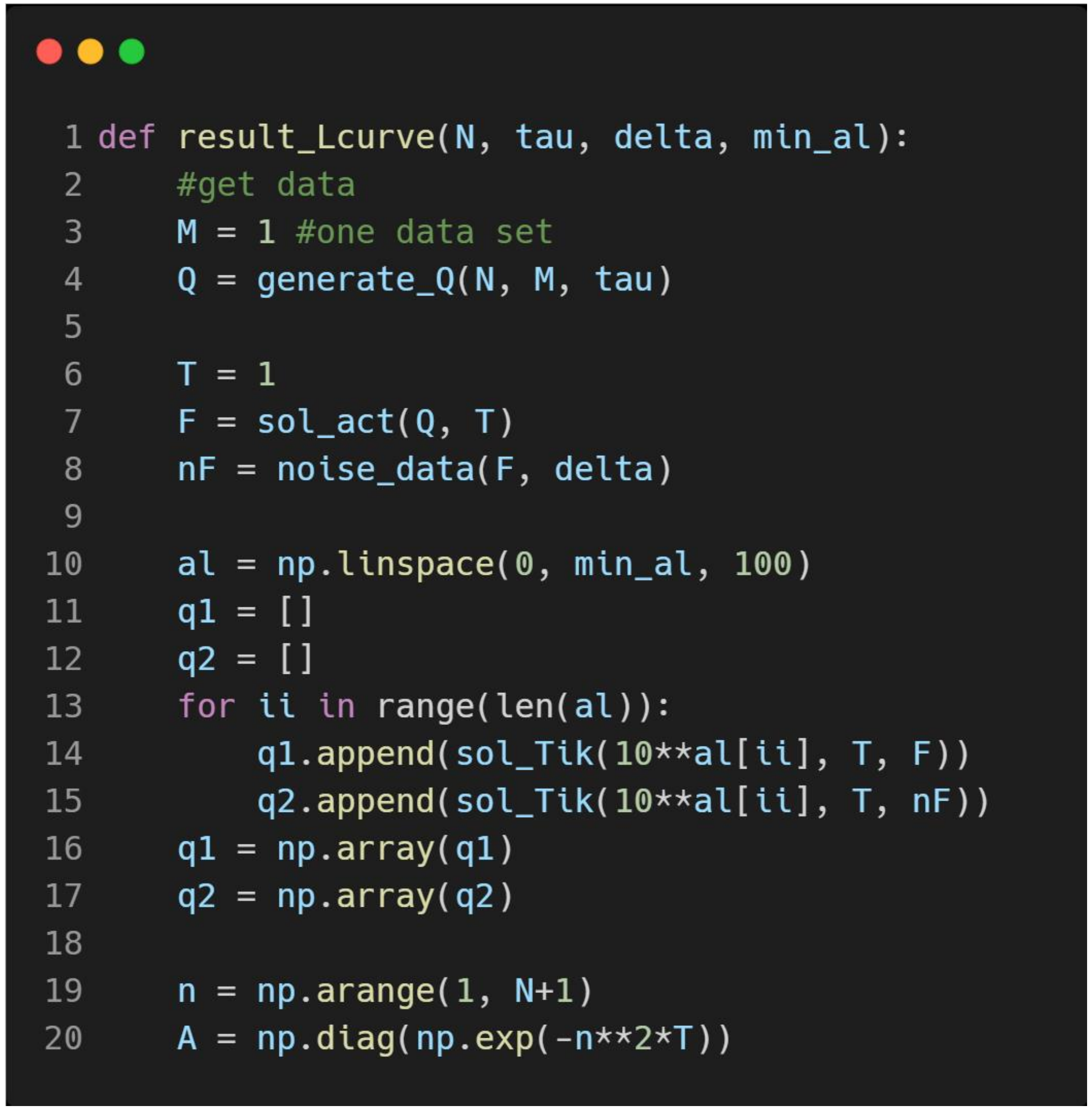
김민준

2021.11.12

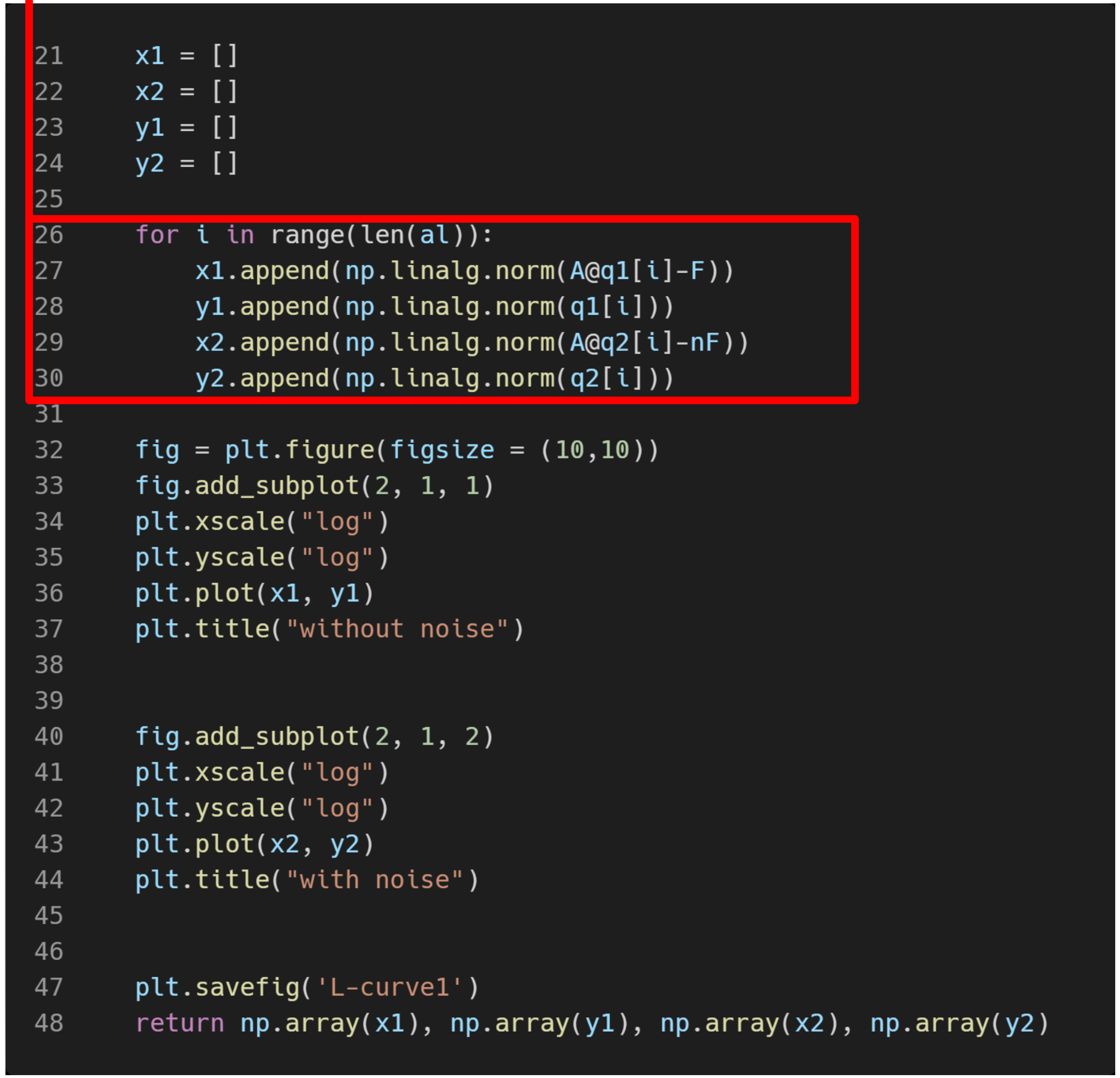
CODE

Without noise : $(\|Aq - f\|, \|q\|) = (x_1, y_1)$

With noise :$(\|A\hat{q} - nf\|, \|\hat{q}\|) = (x_2, y_2)$

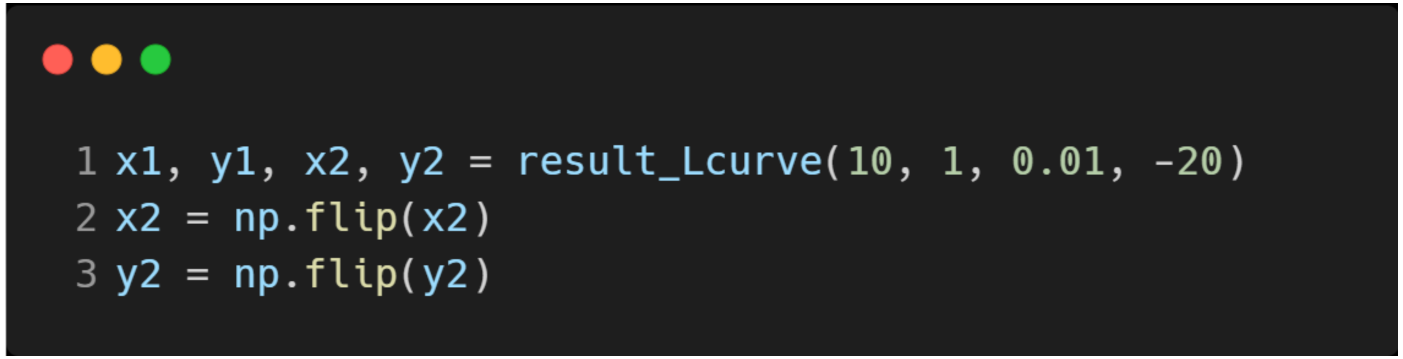


```python
def result_Lcurve(N, tau, delta, min_al):
    #get data
    M = 1 #one data set
    Q = generate_Q(N, M, tau)

    T = 1
    F = sol_act(Q, T)
    nF = noise_data(F, delta)

    al = np.linspace(0, min_al, 100)
    q1 = []
    q2 = []
    for ii in range(len(al)):
        q1.append(sol_Tik(10**al[ii], T, F))
        q2.append(sol_Tik(10**al[ii], T, nF))
    q1 = np.array(q1)
    q2 = np.array(q2)

    n = np.arange(1, N+1)
    A = np.diag(np.exp(-n**2*T))
```

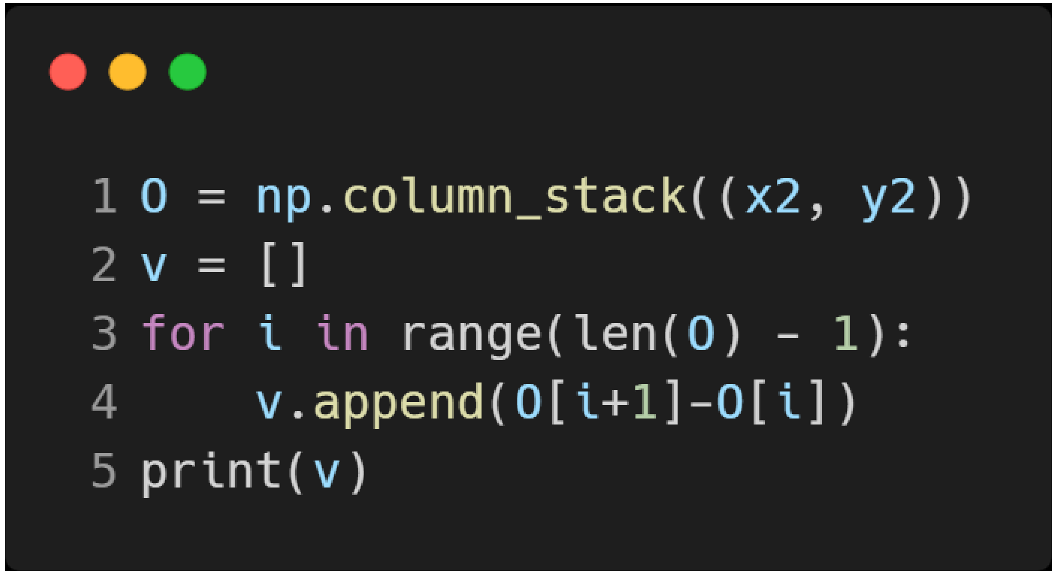


```python
    x1 = []
    x2 = []
    y1 = []
    y2 = []

    for i in range(len(al)):
        x1.append(np.linalg.norm(A@q1[i]-F))
        y1.append(np.linalg.norm(q1[i]))
        x2.append(np.linalg.norm(A@q2[i]-nF))
        y2.append(np.linalg.norm(q2[i]))

    fig = plt.figure(figsize = (10,10))
    fig.add_subplot(2, 1, 1)
    plt.xscale("log")
    plt.yscale("log")
    plt.plot(x1, y1)
    plt.title("without noise")


    fig.add_subplot(2, 1, 2)
    plt.xscale("log")
    plt.yscale("log")
    plt.plot(x2, y2)
    plt.title("with noise")


    plt.savefig('L-curve1')
    return np.array(x1), np.array(y1), np.array(x2), np.array(y2)
```

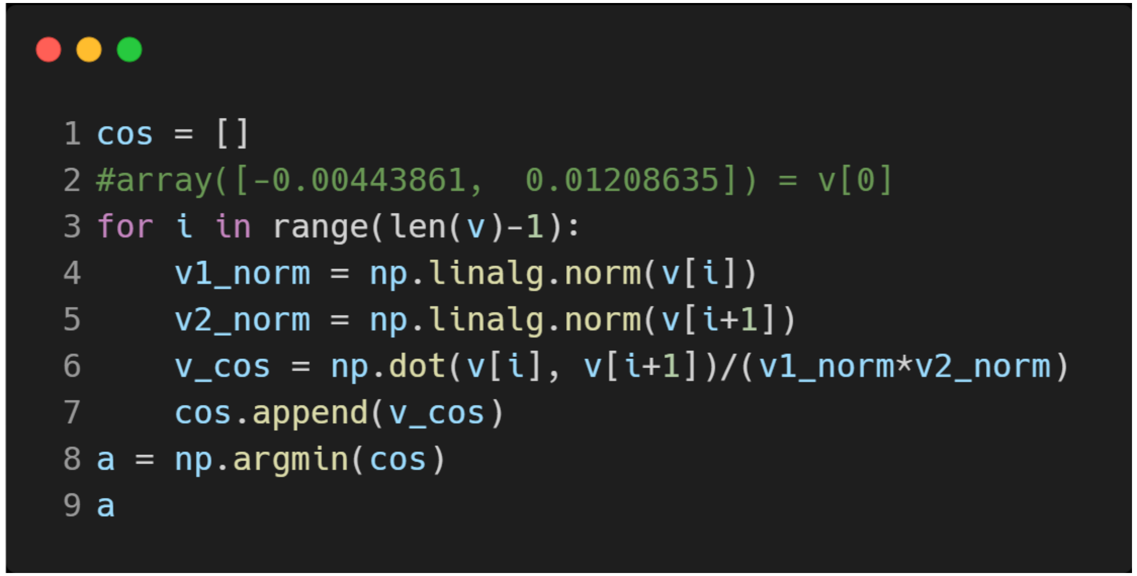
CODE



```
x1, y1, x2, y2 = result_Lcurve(10, 1, 0.01, -20)
x2 = np.flip(x2)
y2 = np.flip(y2)
```



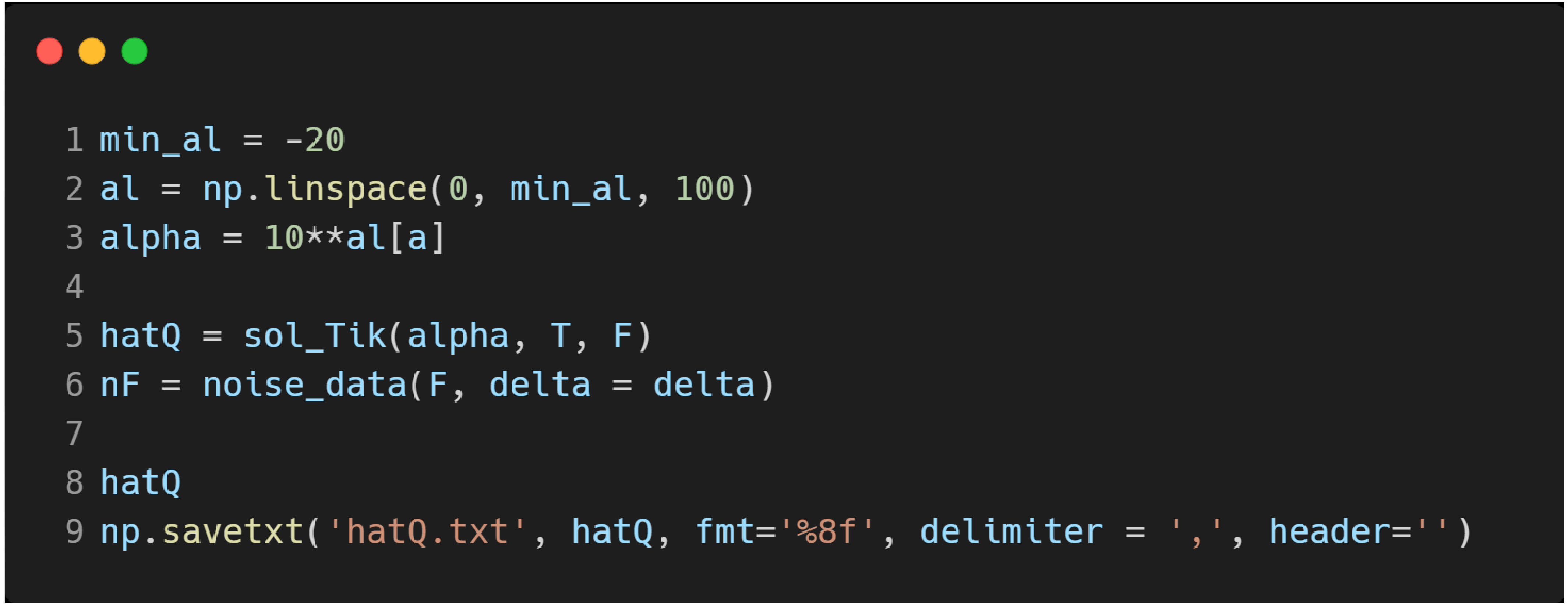
```
O = np.column_stack((x2, y2))
v = []
for i in range(len(O) - 1):
    v.append(O[i+1]-O[i])
print(v)
```



```
cos = []
#array([-0.00443861,  0.01208635]) = v[0]
for i in range(len(v)-1):
    v1_norm = np.linalg.norm(v[i])
    v2_norm = np.linalg.norm(v[i+1])
    v_cos = np.dot(v[i], v[i+1])/(v1_norm*v2_norm)
    cos.append(v_cos)
a = np.argmin(cos)
a
```

$$\cos\theta = \frac{v_1 \cdot v_2}{|v_1||v_2|}, a = \arg\min(\cos\_arr)$$

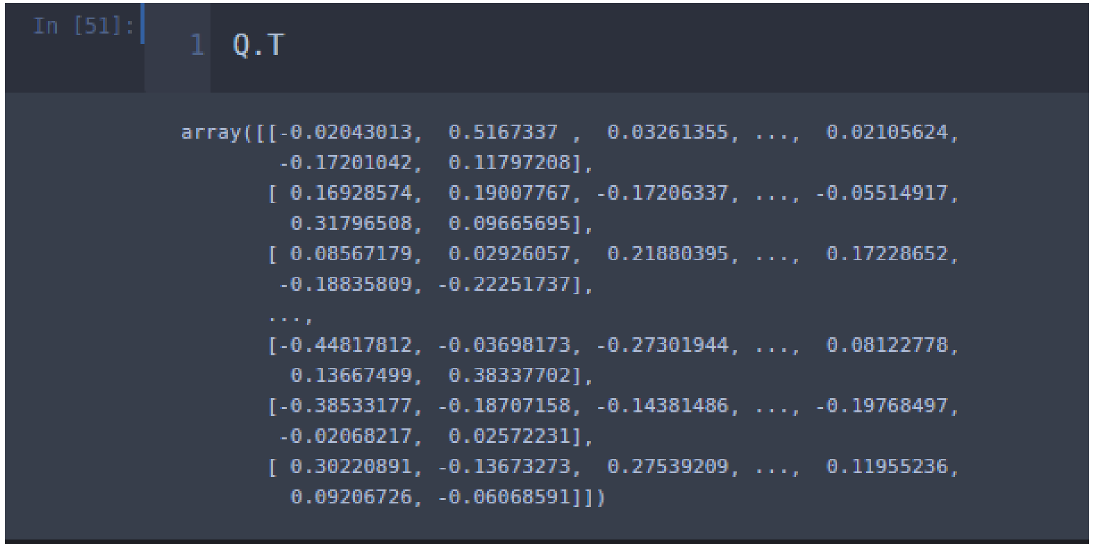
CODE



```python
min_al = -20
al = np.linspace(0, min_al, 100)
alpha = 10**al[a]

hatQ = sol_Tik(alpha, T, F)
nF = noise_data(F, delta = delta)

hatQ
np.savetxt('hatQ.txt', hatQ, fmt='%8f', delimiter = ',', header='')
```

Q, hatQ : N = 12

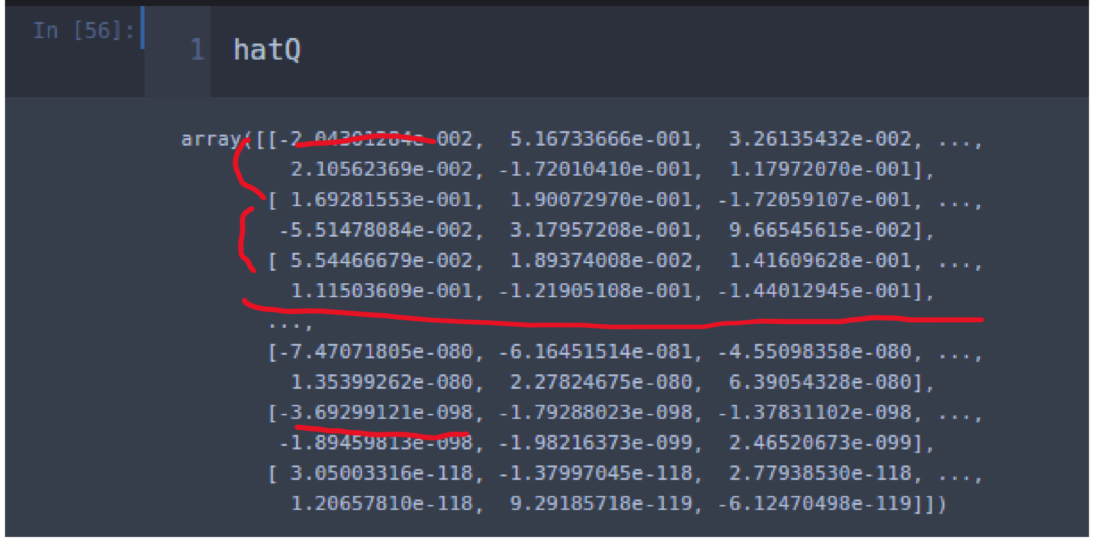
Q shape = (12, 100000)



```
In [51]:  Q.T
```

```
array([[-0.02043013,  0.5167337 ,  0.03261355, ...,  0.02105624,
        -0.17201042,  0.11797208],
       [ 0.16928574,  0.19007767, -0.17206337, ..., -0.05514917,
         0.31796508,  0.09665695],
       [ 0.08567179,  0.02926057,  0.21880395, ...,  0.17228652,
        -0.18835809, -0.22251737],
       ...,
       [-0.44817812, -0.03698173, -0.27301944, ...,  0.08122778,
         0.13667499,  0.38337702],
       [-0.38533177, -0.18707158, -0.14381486, ..., -0.19768497,
        -0.02068217,  0.02572231],
       [ 0.30220891, -0.13673273,  0.27539209, ...,  0.11955236,
         0.09206726, -0.06068591]])
```

Hat Q shape = (12, 100000)



```
In [56]:  hatQ
```

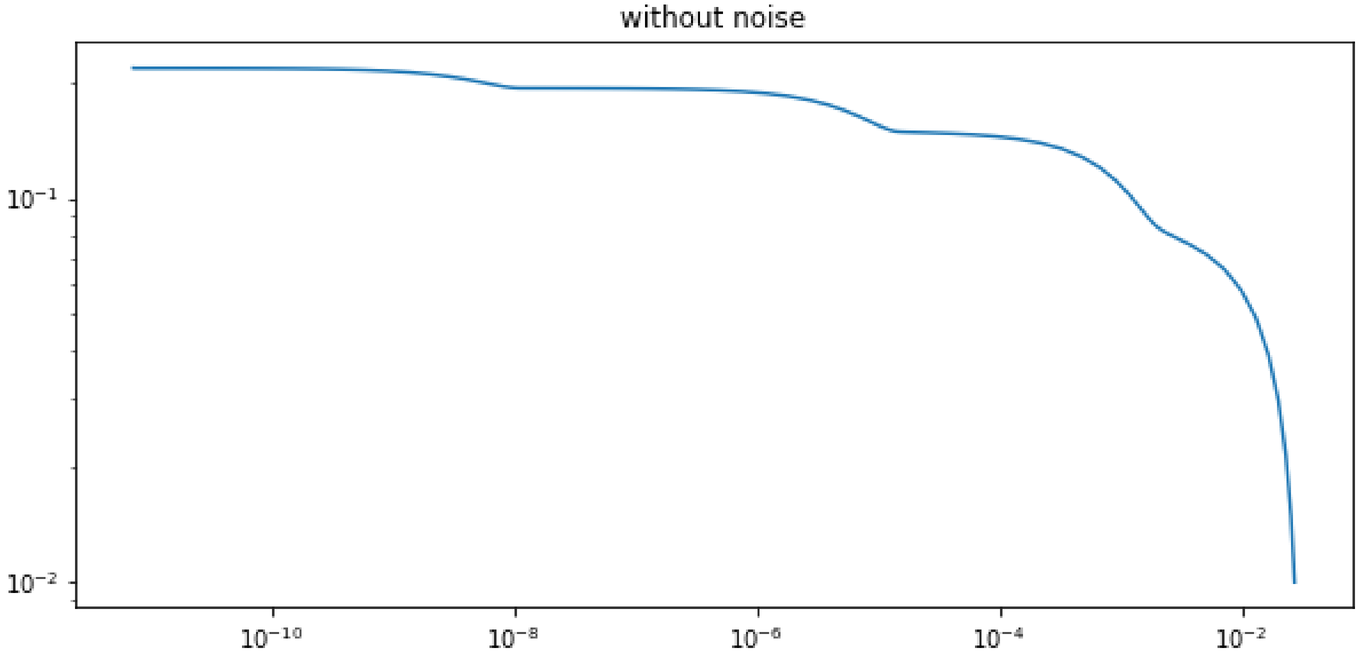
```
array([[-2.04301204e-002,  5.16733666e-001,  3.26135432e-002, ...,
         2.10562369e-002, -1.72010410e-001,  1.17972070e-001],
       [ 1.69281553e-001,  1.90072970e-001, -1.72059107e-001, ...,
        -5.51478084e-002,  3.17957208e-001,  9.66545615e-002],
       [ 5.54466679e-002,  1.89374008e-002,  1.41609628e-001, ...,
         1.11503609e-001, -1.21905108e-001, -1.44012945e-001],
       ...,
       [-7.47071805e-080, -6.16451514e-081, -4.55098358e-080, ...,
         1.35399262e-080,  2.27824675e-080,  6.39054328e-080],
       [-3.69299121e-098, -1.79288023e-098, -1.37831102e-098, ...,
        -1.89459813e-098, -1.98216373e-099,  2.46520673e-099],
       [ 3.05003316e-118, -1.37997045e-118,  2.77938530e-118, ...,
         1.20657810e-118,  9.29185718e-119, -6.12470498e-119]])
```

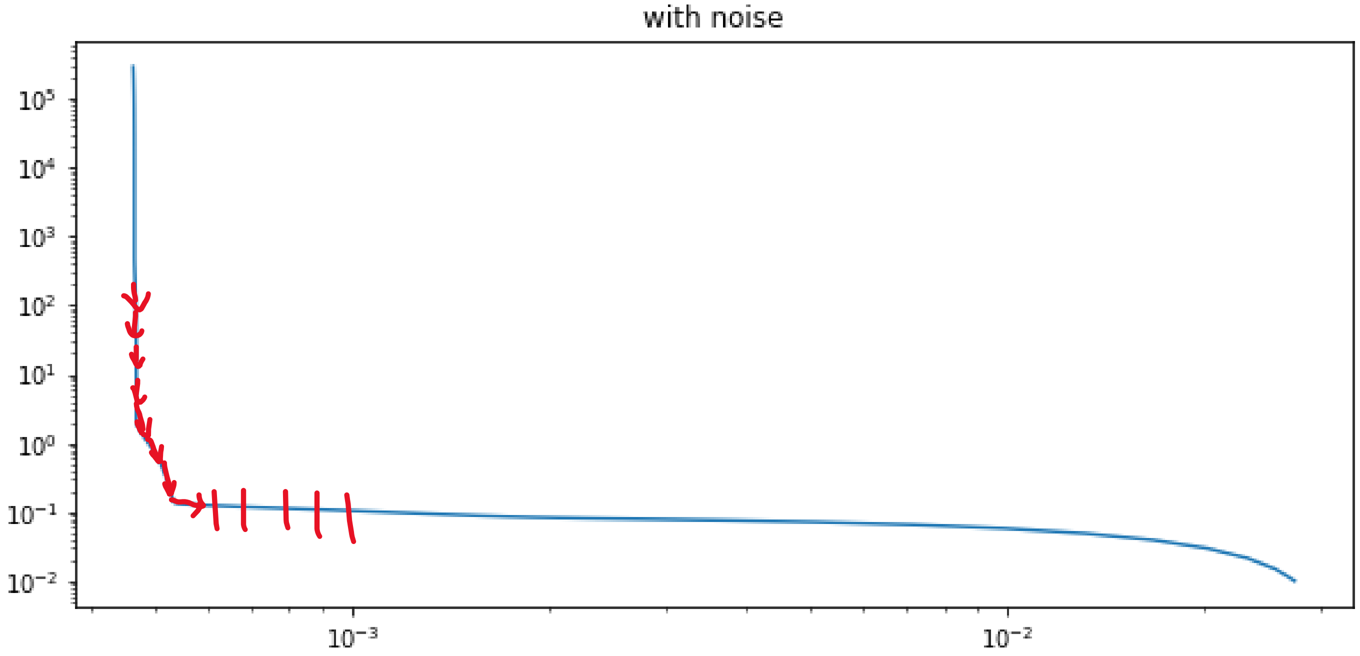
#1

- Data

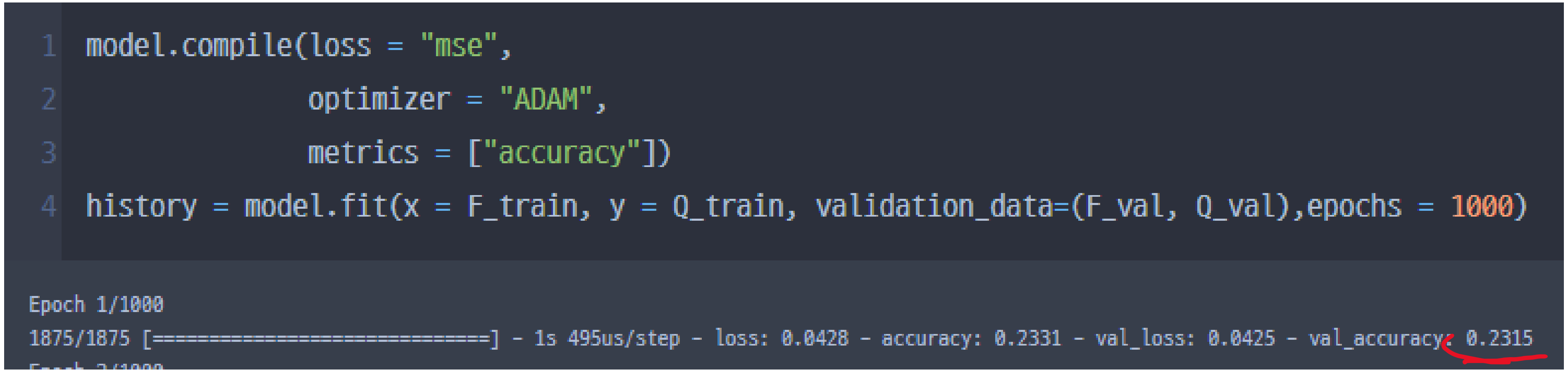
T = 1, delta = 0.01, tau = 1

N = 10, M = 100000





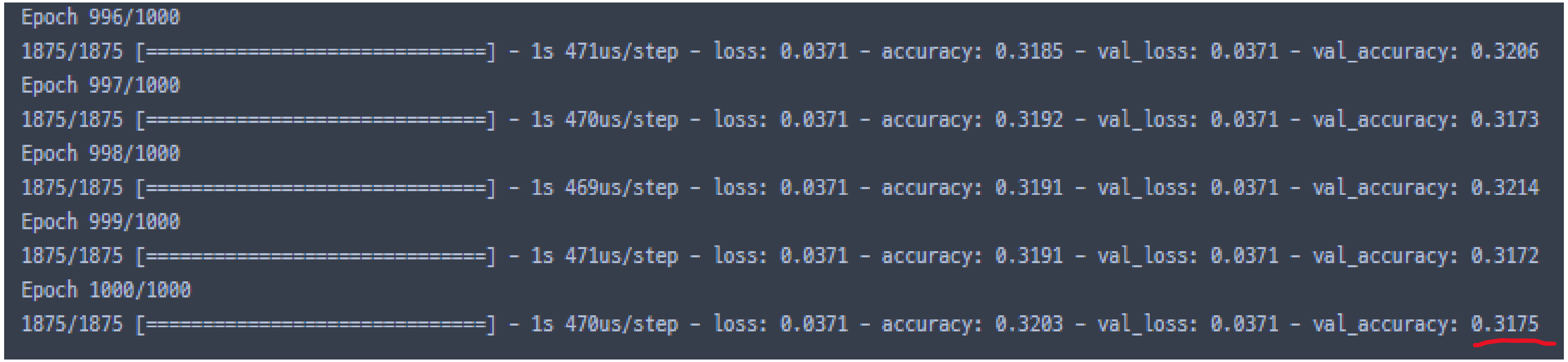
N = 10, M = 100000 Train(Q, F)



```
model.compile(loss = "mse",
              optimizer = "ADAM",
              metrics = ["accuracy"])
history = model.fit(x = F_train, y = Q_train, validation_data=(F_val, Q_val),epochs = 1000)
```

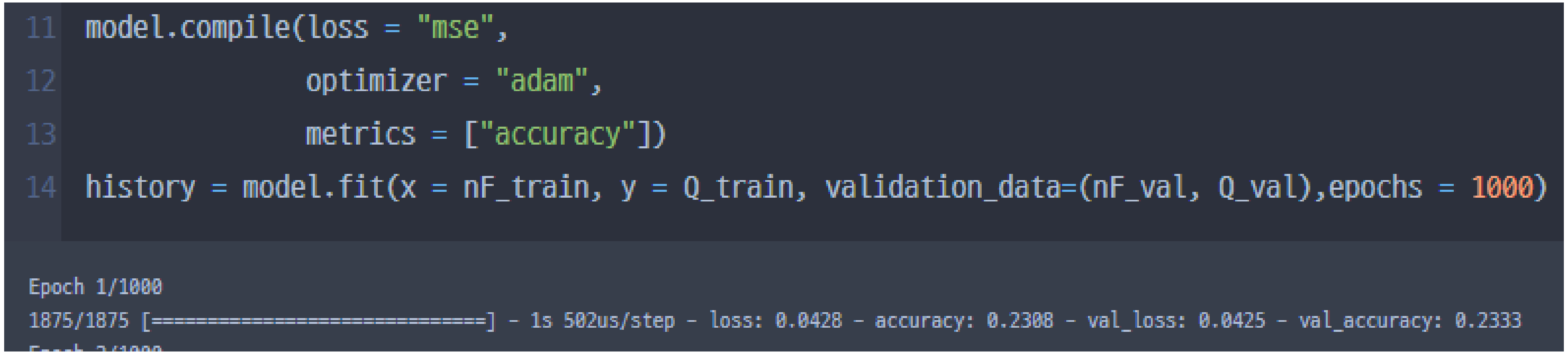
```
Epoch 1/1000
1875/1875 [==============================] - 1s 495us/step - loss: 0.0428 - accuracy: 0.2331 - val_loss: 0.0425 - val_accuracy: 0.2315
Epoch 2/1000
```

⋮



```
Epoch 996/1000
1875/1875 [==============================] - 1s 471us/step - loss: 0.0371 - accuracy: 0.3185 - val_loss: 0.0371 - val_accuracy: 0.3206
Epoch 997/1000
1875/1875 [==============================] - 1s 470us/step - loss: 0.0371 - accuracy: 0.3192 - val_loss: 0.0371 - val_accuracy: 0.3173
Epoch 998/1000
1875/1875 [==============================] - 1s 469us/step - loss: 0.0371 - accuracy: 0.3191 - val_loss: 0.0371 - val_accuracy: 0.3214
Epoch 999/1000
1875/1875 [==============================] - 1s 471us/step - loss: 0.0371 - accuracy: 0.3191 - val_loss: 0.0371 - val_accuracy: 0.3172
Epoch 1000/1000
1875/1875 [==============================] - 1s 470us/step - loss: 0.0371 - accuracy: 0.3203 - val_loss: 0.0371 - val_accuracy: 0.3175
```

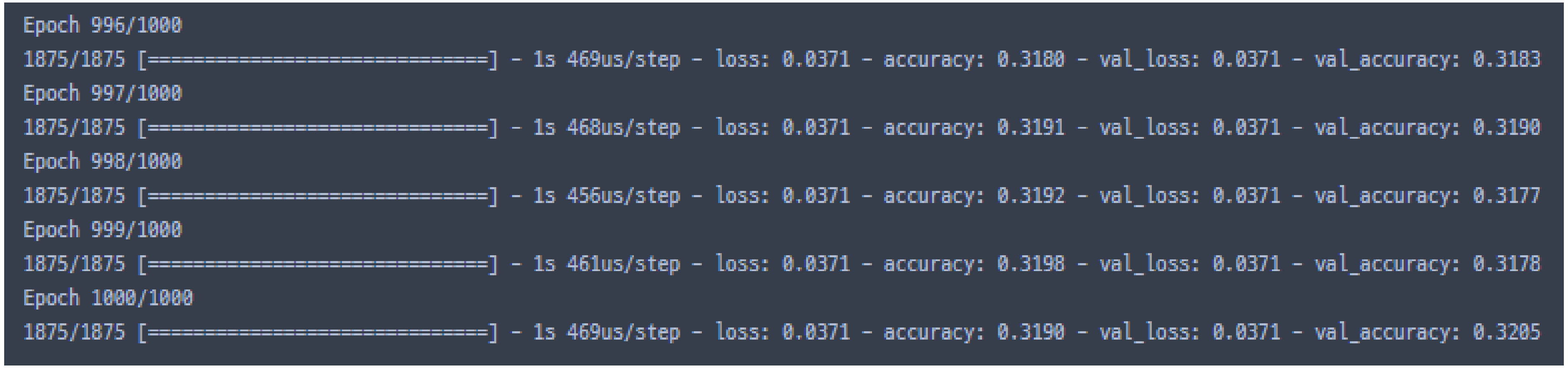
N = 10, M = 100000 Train(Q, nF)



```
model.compile(loss = "mse",
              optimizer = "adam",
              metrics = ["accuracy"])
history = model.fit(x = nF_train, y = Q_train, validation_data=(nF_val, Q_val),epochs = 1000)
```

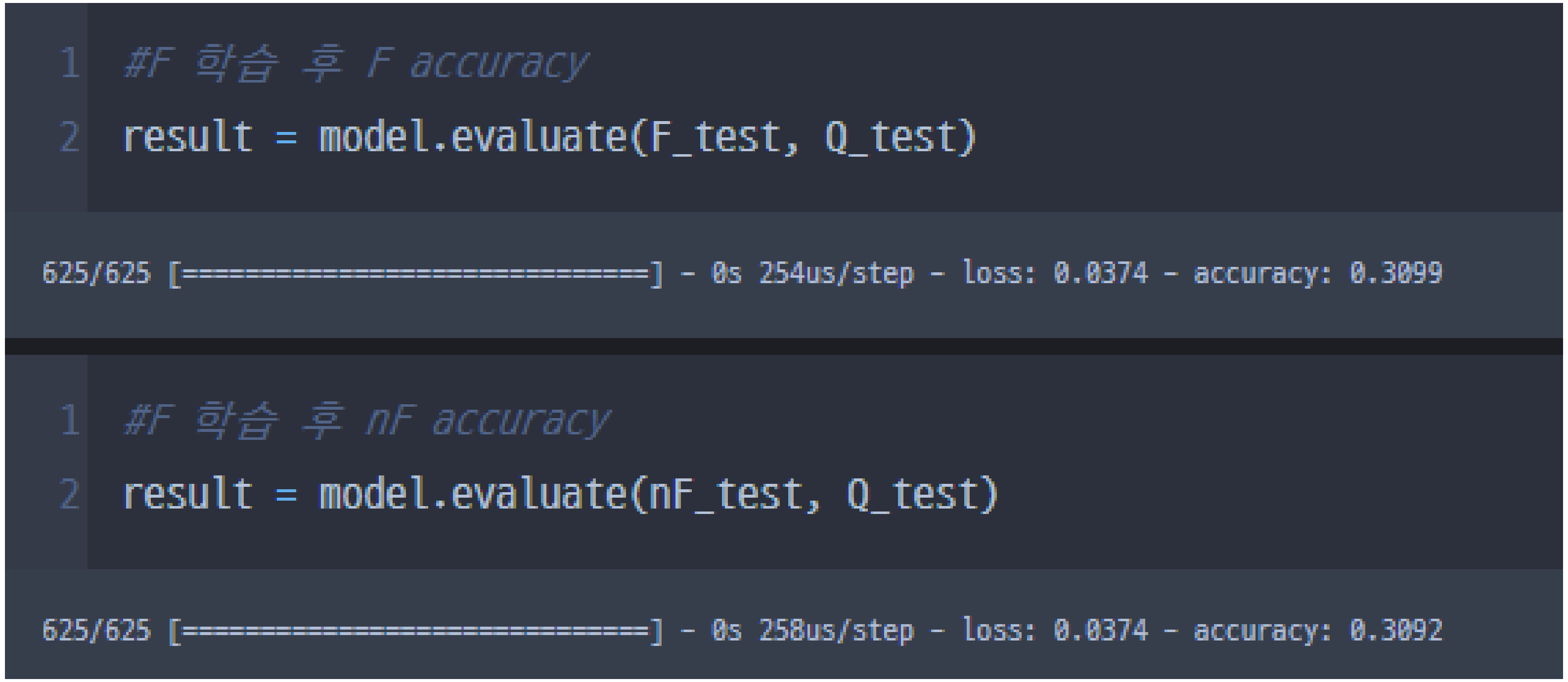
```
Epoch 1/1000
1875/1875 [==============================] - 1s 502us/step - loss: 0.0428 - accuracy: 0.2308 - val_loss: 0.0425 - val_accuracy: 0.2333
Epoch 2/1000
```

⋮



```
Epoch 996/1000
1875/1875 [==============================] - 1s 469us/step - loss: 0.0371 - accuracy: 0.3180 - val_loss: 0.0371 - val_accuracy: 0.3183
Epoch 997/1000
1875/1875 [==============================] - 1s 468us/step - loss: 0.0371 - accuracy: 0.3191 - val_loss: 0.0371 - val_accuracy: 0.3190
Epoch 998/1000
1875/1875 [==============================] - 1s 456us/step - loss: 0.0371 - accuracy: 0.3192 - val_loss: 0.0371 - val_accuracy: 0.3177
Epoch 999/1000
1875/1875 [==============================] - 1s 461us/step - loss: 0.0371 - accuracy: 0.3198 - val_loss: 0.0371 - val_accuracy: 0.3178
Epoch 1000/1000
1875/1875 [==============================] - 1s 469us/step - loss: 0.0371 - accuracy: 0.3190 - val_loss: 0.0371 - val_accuracy: 0.3205
```

N = 10, M = 100000 Test data accuracy-Q

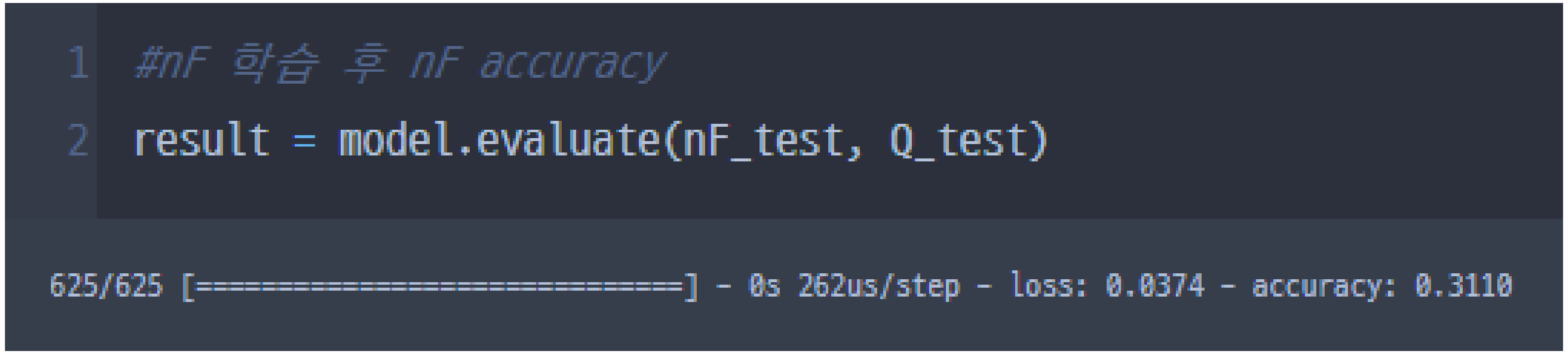


```
#F 학습 후 F accuracy
result = model.evaluate(F_test, Q_test)
```

```
625/625 [==============================] - 0s 254us/step - loss: 0.0374 - accuracy: 0.3099
```

```
#F 학습 후 nF accuracy
result = model.evaluate(nF_test, Q_test)
```

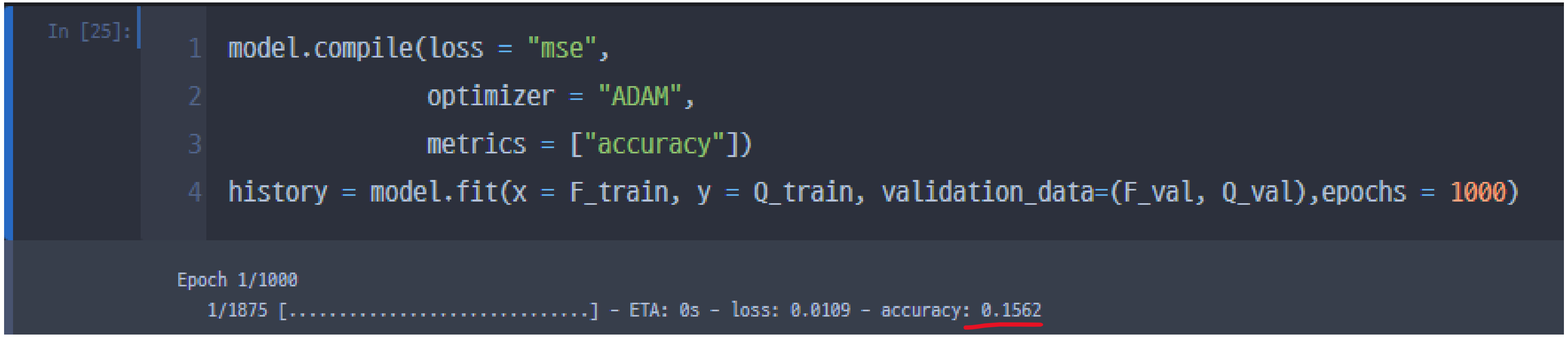
```
625/625 [==============================] - 0s 258us/step - loss: 0.0374 - accuracy: 0.3092
```



```
#nF 학습 후 nF accuracy
result = model.evaluate(nF_test, Q_test)
```

```
625/625 [==============================] - 0s 262us/step - loss: 0.0374 - accuracy: 0.3110
```

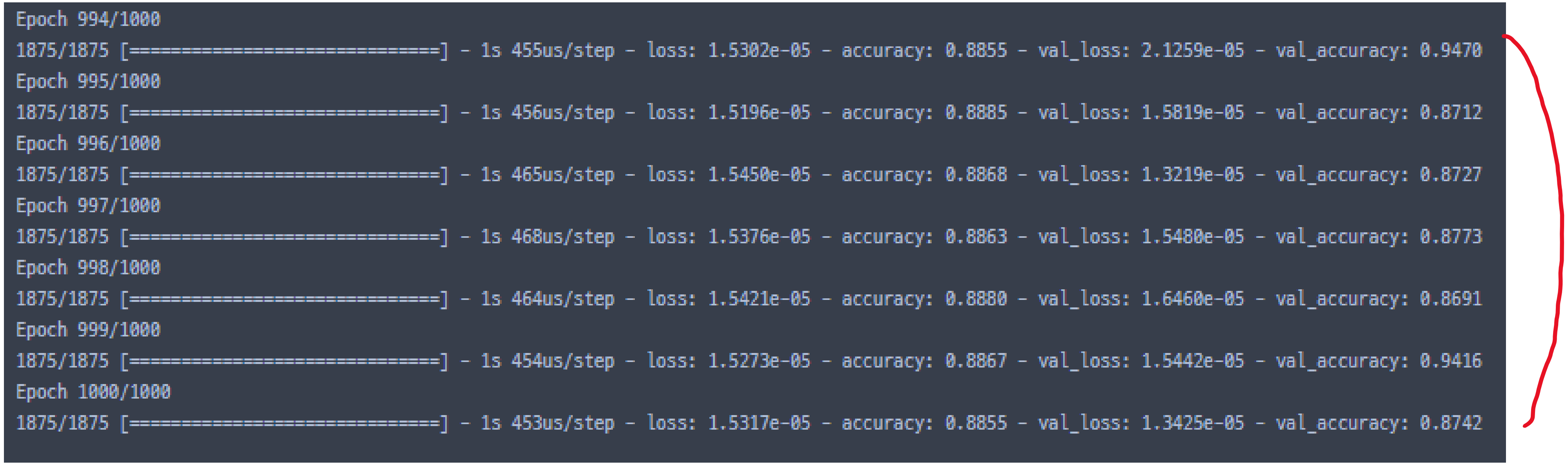
N = 10, M = 100000 Train(hatQ, F)



```
In [25]:
model.compile(loss = "mse",
              optimizer = "ADAM",
              metrics = ["accuracy"])
history = model.fit(x = F_train, y = Q_train, validation_data=(F_val, Q_val),epochs = 1000)
```

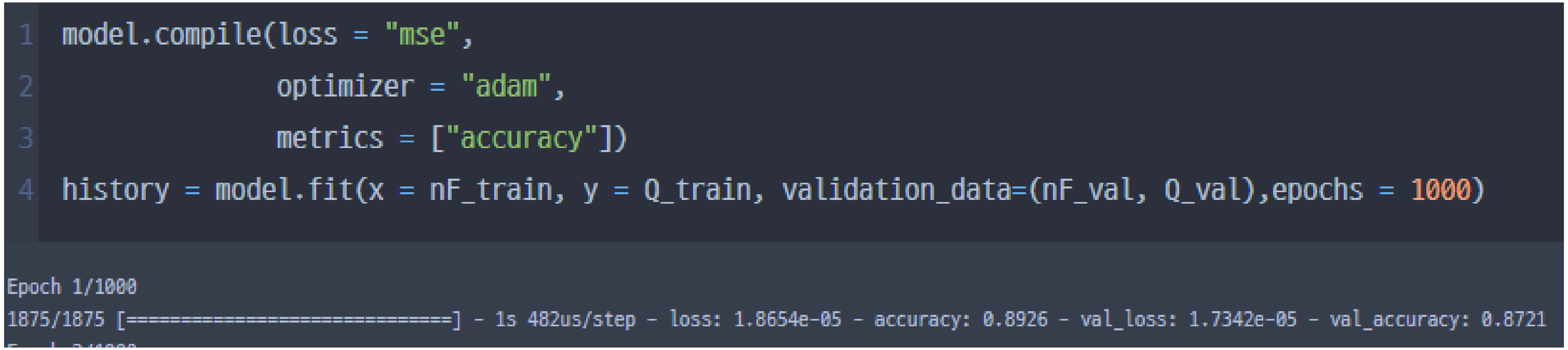
```
Epoch 1/1000
   1/1875 [..............................] - ETA: 0s - loss: 0.0109 - accuracy: 0.1562
```

⋮



```
Epoch 994/1000
1875/1875 [==============================] - 1s 455us/step - loss: 1.5302e-05 - accuracy: 0.8855 - val_loss: 2.1259e-05 - val_accuracy: 0.9470
Epoch 995/1000
1875/1875 [==============================] - 1s 456us/step - loss: 1.5196e-05 - accuracy: 0.8885 - val_loss: 1.5819e-05 - val_accuracy: 0.8712
Epoch 996/1000
1875/1875 [==============================] - 1s 465us/step - loss: 1.5450e-05 - accuracy: 0.8868 - val_loss: 1.3219e-05 - val_accuracy: 0.8727
Epoch 997/1000
1875/1875 [==============================] - 1s 468us/step - loss: 1.5376e-05 - accuracy: 0.8863 - val_loss: 1.5480e-05 - val_accuracy: 0.8773
Epoch 998/1000
1875/1875 [==============================] - 1s 464us/step - loss: 1.5421e-05 - accuracy: 0.8880 - val_loss: 1.6460e-05 - val_accuracy: 0.8691
Epoch 999/1000
1875/1875 [==============================] - 1s 454us/step - loss: 1.5273e-05 - accuracy: 0.8867 - val_loss: 1.5442e-05 - val_accuracy: 0.9416
Epoch 1000/1000
1875/1875 [==============================] - 1s 453us/step - loss: 1.5317e-05 - accuracy: 0.8855 - val_loss: 1.3425e-05 - val_accuracy: 0.8742
```

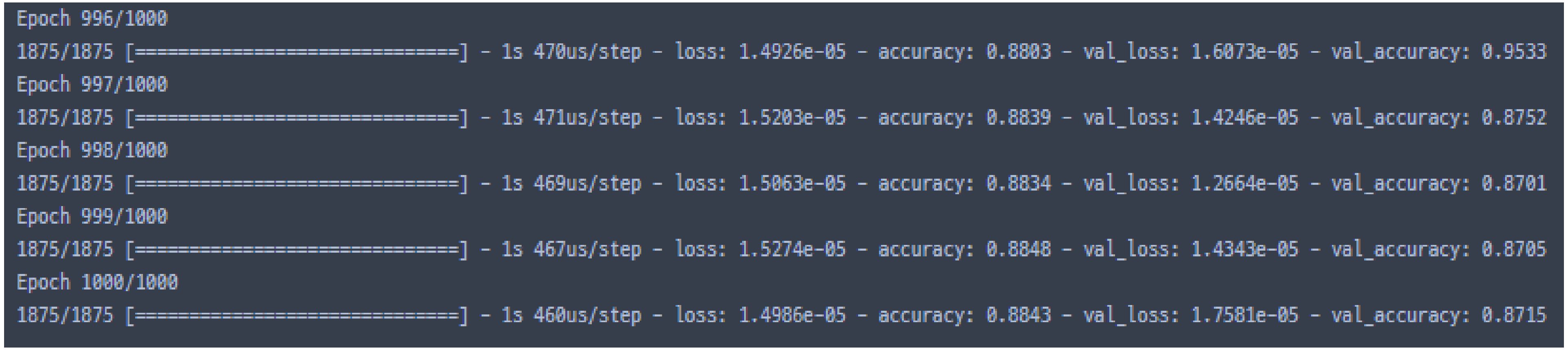
N = 10, M = 100000 Train(hatQ, nF)



```
model.compile(loss = "mse",
              optimizer = "adam",
              metrics = ["accuracy"])
history = model.fit(x = nF_train, y = Q_train, validation_data=(nF_val, Q_val),epochs = 1000)
```

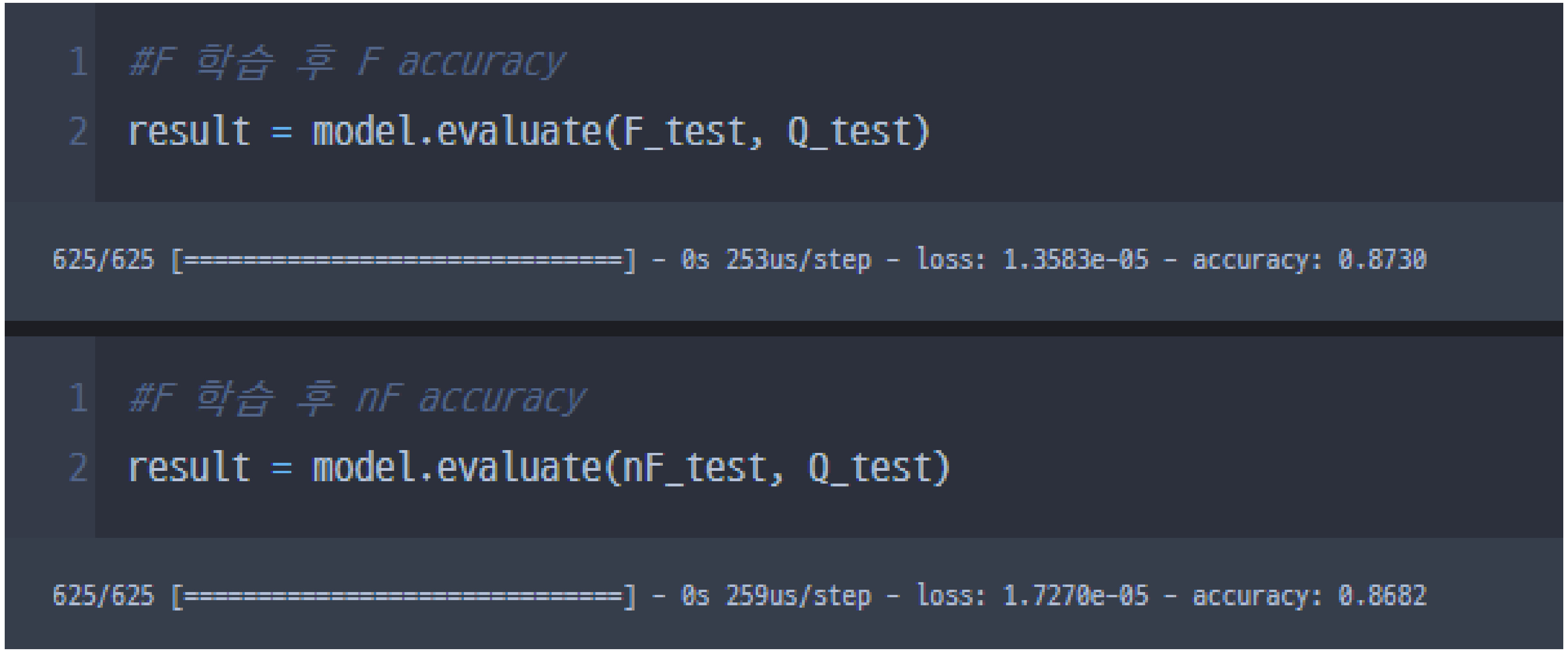
```
Epoch 1/1000
1875/1875 [==============================] - 1s 482us/step - loss: 1.8654e-05 - accuracy: 0.8926 - val_loss: 1.7342e-05 - val_accuracy: 0.8721
```

⋮



```
Epoch 996/1000
1875/1875 [==============================] - 1s 470us/step - loss: 1.4926e-05 - accuracy: 0.8803 - val_loss: 1.6073e-05 - val_accuracy: 0.9533
Epoch 997/1000
1875/1875 [==============================] - 1s 471us/step - loss: 1.5203e-05 - accuracy: 0.8839 - val_loss: 1.4246e-05 - val_accuracy: 0.8752
Epoch 998/1000
1875/1875 [==============================] - 1s 469us/step - loss: 1.5063e-05 - accuracy: 0.8834 - val_loss: 1.2664e-05 - val_accuracy: 0.8701
Epoch 999/1000
1875/1875 [==============================] - 1s 467us/step - loss: 1.5274e-05 - accuracy: 0.8848 - val_loss: 1.4343e-05 - val_accuracy: 0.8705
Epoch 1000/1000
1875/1875 [==============================] - 1s 460us/step - loss: 1.4986e-05 - accuracy: 0.8843 - val_loss: 1.7581e-05 - val_accuracy: 0.8715
```

N = 10, M = 100000 Test data accuracy - hatQ

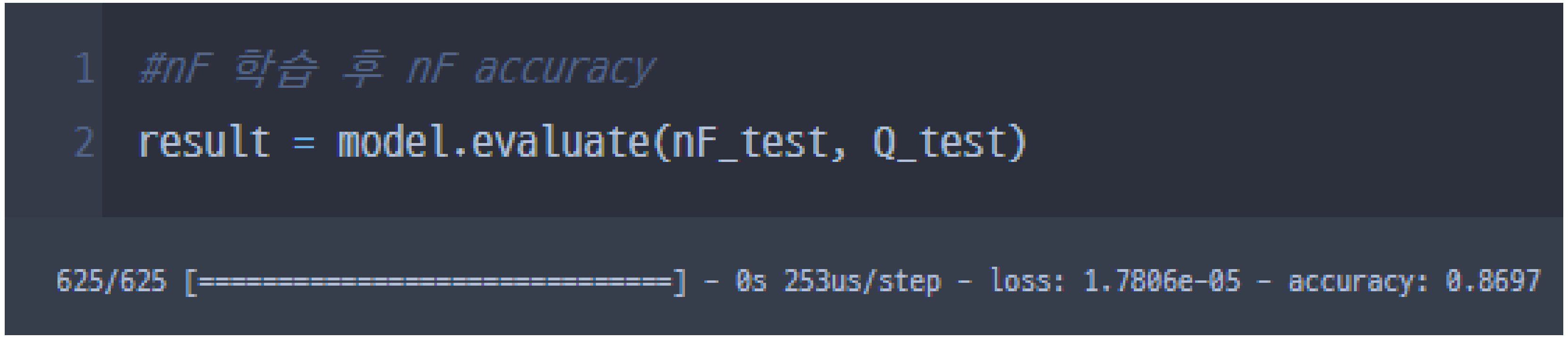


```
#F 학습 후 F accuracy
result = model.evaluate(F_test, Q_test)
```

```
625/625 [==============================] - 0s 253us/step - loss: 1.3583e-05 - accuracy: 0.8730
```

```
#F 학습 후 nF accuracy
result = model.evaluate(nF_test, Q_test)
```

```
625/625 [==============================] - 0s 259us/step - loss: 1.7270e-05 - accuracy: 0.8682
```



```
#nF 학습 후 nF accuracy
result = model.evaluate(nF_test, Q_test)
```

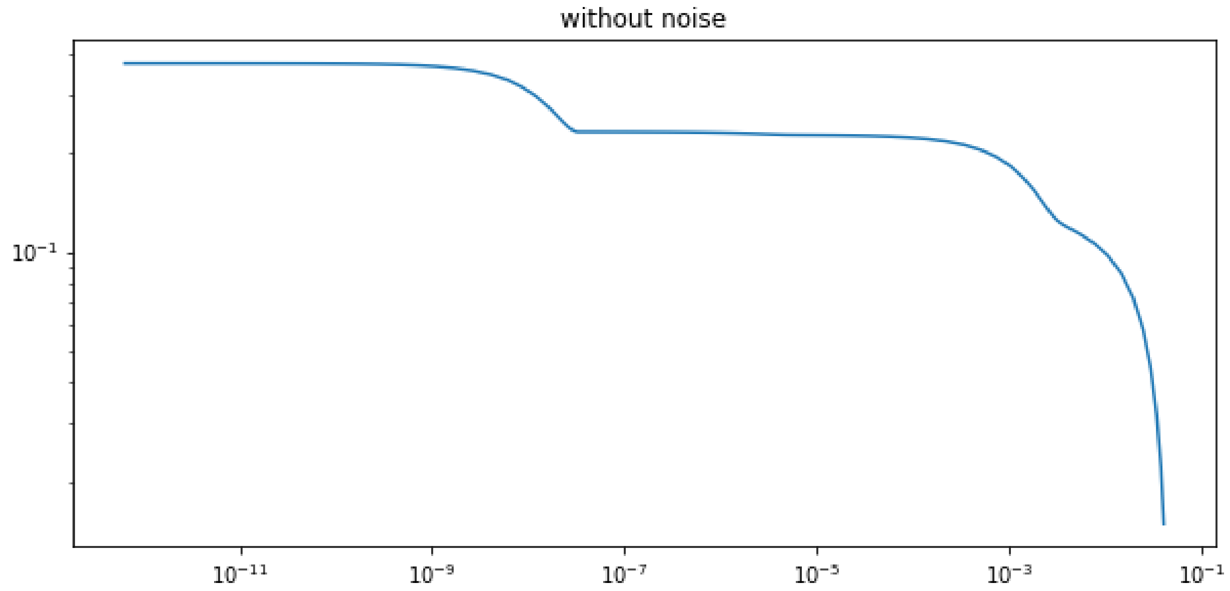
```
625/625 [==============================] - 0s 253us/step - loss: 1.7806e-05 - accuracy: 0.8697
```

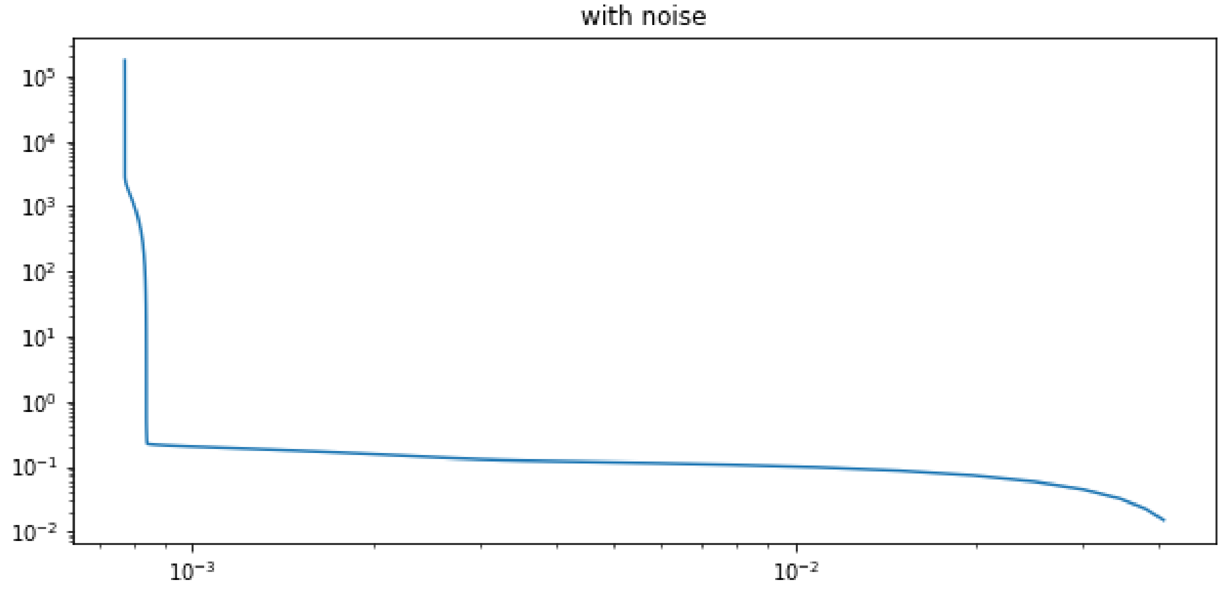
#2

- Data

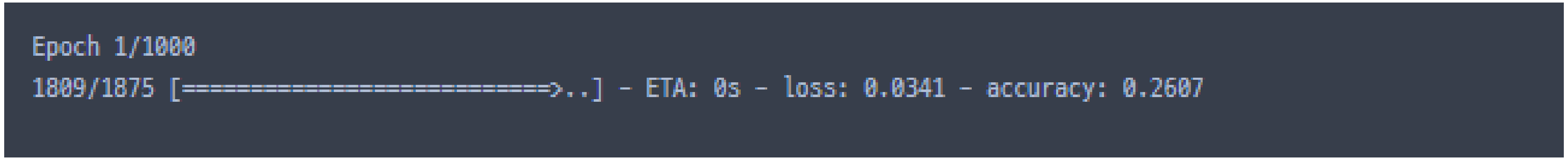
T = 1, delta = 0.01, tau = 1

N = 12, M = 100000



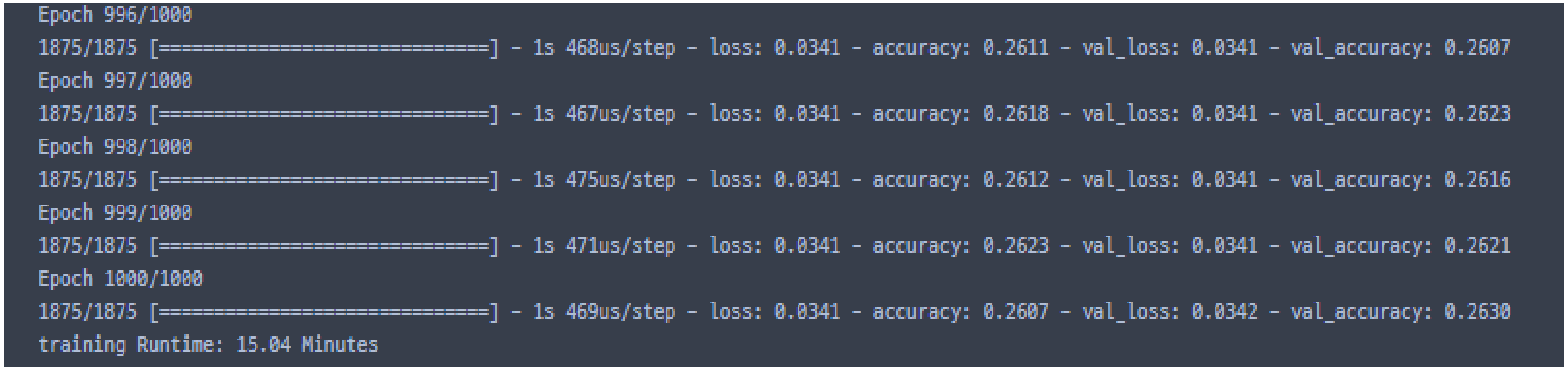


N = 12, M = 100000 Train(Q, F)



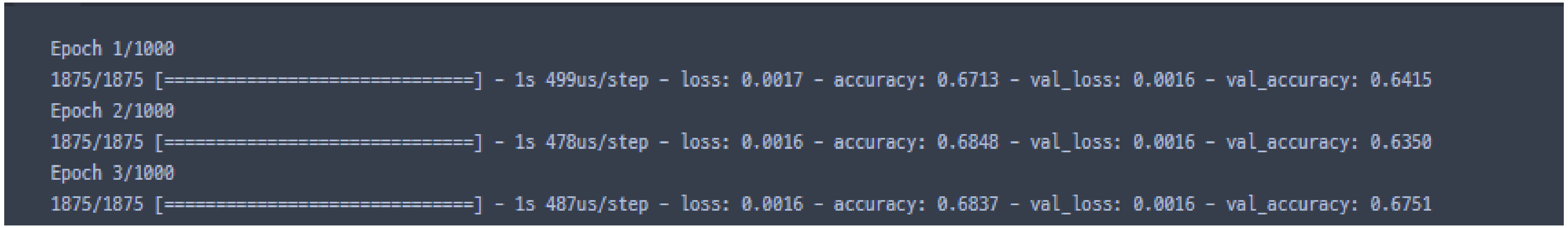
```
Epoch 1/1000
1809/1875 [===========================>..] - ETA: 0s - loss: 0.0341 - accuracy: 0.2607
```

⋮



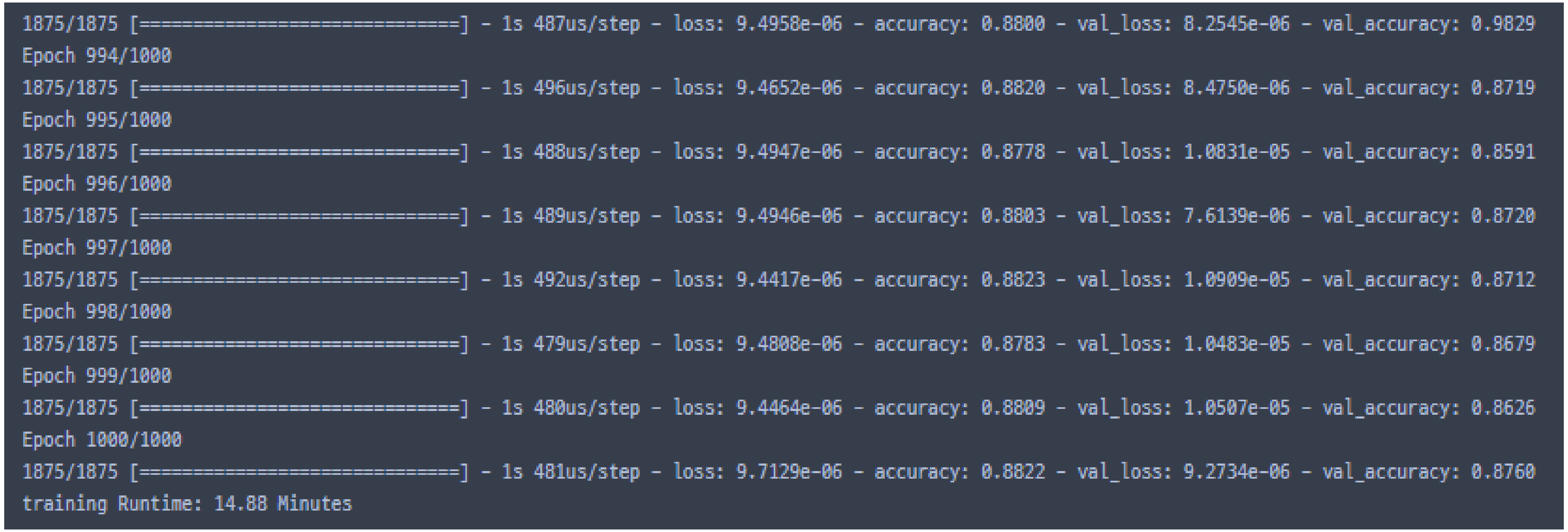
```
Epoch 996/1000
1875/1875 [==============================] - 1s 468us/step - loss: 0.0341 - accuracy: 0.2611 - val_loss: 0.0341 - val_accuracy: 0.2607
Epoch 997/1000
1875/1875 [==============================] - 1s 467us/step - loss: 0.0341 - accuracy: 0.2618 - val_loss: 0.0341 - val_accuracy: 0.2623
Epoch 998/1000
1875/1875 [==============================] - 1s 475us/step - loss: 0.0341 - accuracy: 0.2612 - val_loss: 0.0341 - val_accuracy: 0.2616
Epoch 999/1000
1875/1875 [==============================] - 1s 471us/step - loss: 0.0341 - accuracy: 0.2623 - val_loss: 0.0341 - val_accuracy: 0.2621
Epoch 1000/1000
1875/1875 [==============================] - 1s 469us/step - loss: 0.0341 - accuracy: 0.2607 - val_loss: 0.0342 - val_accuracy: 0.2630
training Runtime: 15.04 Minutes
```

N = 12, M = 100000 Train(hatQ, F)



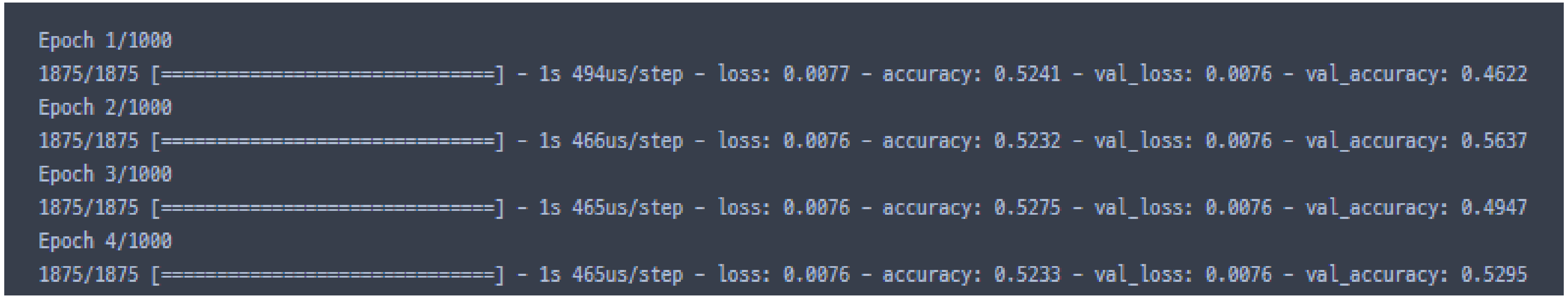
```
Epoch 1/1000
1875/1875 [==============================] - 1s 499us/step - loss: 0.0017 - accuracy: 0.6713 - val_loss: 0.0016 - val_accuracy: 0.6415
Epoch 2/1000
1875/1875 [==============================] - 1s 478us/step - loss: 0.0016 - accuracy: 0.6848 - val_loss: 0.0016 - val_accuracy: 0.6350
Epoch 3/1000
1875/1875 [==============================] - 1s 487us/step - loss: 0.0016 - accuracy: 0.6837 - val_loss: 0.0016 - val_accuracy: 0.6751
```

⋮



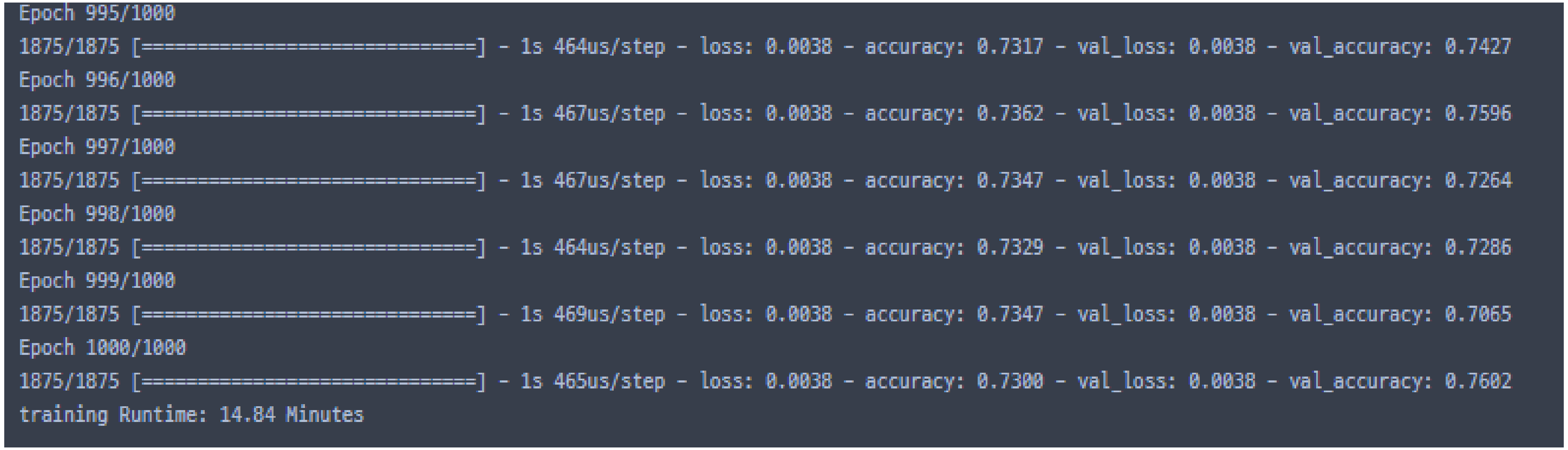
```
1875/1875 [==============================] - 1s 487us/step - loss: 9.4958e-06 - accuracy: 0.8800 - val_loss: 8.2545e-06 - val_accuracy: 0.9829
Epoch 994/1000
1875/1875 [==============================] - 1s 496us/step - loss: 9.4652e-06 - accuracy: 0.8820 - val_loss: 8.4750e-06 - val_accuracy: 0.8719
Epoch 995/1000
1875/1875 [==============================] - 1s 488us/step - loss: 9.4947e-06 - accuracy: 0.8778 - val_loss: 1.0831e-05 - val_accuracy: 0.8591
Epoch 996/1000
1875/1875 [==============================] - 1s 489us/step - loss: 9.4946e-06 - accuracy: 0.8803 - val_loss: 7.6139e-06 - val_accuracy: 0.8720
Epoch 997/1000
1875/1875 [==============================] - 1s 492us/step - loss: 9.4417e-06 - accuracy: 0.8823 - val_loss: 1.0909e-05 - val_accuracy: 0.8712
Epoch 998/1000
1875/1875 [==============================] - 1s 479us/step - loss: 9.4808e-06 - accuracy: 0.8783 - val_loss: 1.0483e-05 - val_accuracy: 0.8679
Epoch 999/1000
1875/1875 [==============================] - 1s 480us/step - loss: 9.4464e-06 - accuracy: 0.8809 - val_loss: 1.0507e-05 - val_accuracy: 0.8626
Epoch 1000/1000
1875/1875 [==============================] - 1s 481us/step - loss: 9.7129e-06 - accuracy: 0.8822 - val_loss: 9.2734e-06 - val_accuracy: 0.8760
training Runtime: 14.88 Minutes
```

N = 12, M = 100000 Train(hatQ, nF)



```
Epoch 1/1000
1875/1875 [==============================] - 1s 494us/step - loss: 0.0077 - accuracy: 0.5241 - val_loss: 0.0076 - val_accuracy: 0.4622
Epoch 2/1000
1875/1875 [==============================] - 1s 466us/step - loss: 0.0076 - accuracy: 0.5232 - val_loss: 0.0076 - val_accuracy: 0.5637
Epoch 3/1000
1875/1875 [==============================] - 1s 465us/step - loss: 0.0076 - accuracy: 0.5275 - val_loss: 0.0076 - val_accuracy: 0.4947
Epoch 4/1000
1875/1875 [==============================] - 1s 465us/step - loss: 0.0076 - accuracy: 0.5233 - val_loss: 0.0076 - val_accuracy: 0.5295
```

⋮



```
Epoch 995/1000
1875/1875 [==============================] - 1s 464us/step - loss: 0.0038 - accuracy: 0.7317 - val_loss: 0.0038 - val_accuracy: 0.7427
Epoch 996/1000
1875/1875 [==============================] - 1s 467us/step - loss: 0.0038 - accuracy: 0.7362 - val_loss: 0.0038 - val_accuracy: 0.7596
Epoch 997/1000
1875/1875 [==============================] - 1s 467us/step - loss: 0.0038 - accuracy: 0.7347 - val_loss: 0.0038 - val_accuracy: 0.7264
Epoch 998/1000
1875/1875 [==============================] - 1s 464us/step - loss: 0.0038 - accuracy: 0.7329 - val_loss: 0.0038 - val_accuracy: 0.7286
Epoch 999/1000
1875/1875 [==============================] - 1s 469us/step - loss: 0.0038 - accuracy: 0.7347 - val_loss: 0.0038 - val_accuracy: 0.7065
Epoch 1000/1000
1875/1875 [==============================] - 1s 465us/step - loss: 0.0038 - accuracy: 0.7300 - val_loss: 0.0038 - val_accuracy: 0.7602
training Runtime: 14.84 Minutes
```