In [14]:
```python
import tensorflow as tf
```

In [22]:
```python
fashion_mnist = tf.keras.datasets.fashion_mnist.load_data()
(X_train_full,y_train_full), (X_test, y_test) = fashion_mnist
X_train, y_train = X_train_full[:-5000], y_train_full[:-5000]
X_valid, y_valid = X_train_full[-5000:], y_train_full[-5000:]
```

In [23]:
```python
X_train.shape
```

Out[23]:
```
(55000, 28, 28)
```

In [24]:
```python
y_train.shape
```

Out[24]:
```
(55000,)
```

In [25]:
```python
X_train, X_valid, X_test = X_train/255., X_valid / 255., X_test / 255.
```

In [28]:
```python
class_names = ["T_shirt\top", "Trouser", "Pullover", "Dress", "coat", "Sandal", "shirt", "Sneakers", "bags", "A
class_names[y_train[1]]
```

Out[28]:
```
'T_shirt\top'
```

In [32]:
```python
tf.random.set_seed(42)
model = tf.keras.Sequential()
model.add(tf.keras.layers.Input(shape=[28,28]))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(300, activation='relu'))
model.add(tf.keras.layers.Dense(100, activation='relu'))
model.add(tf.keras.layers.Dense(10, activation='softmax'))
```

In [33]:
```python
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 784)               0

 dense (Dense)               (None, 300)               235500

 dense_1 (Dense)             (None, 100)               30100

 dense_2 (Dense)             (None, 10)                1010

=================================================================
Total params: 266,610
Trainable params: 266,610
Non-trainable params: 0
_________________________________________________________________
```

In [34]:
```python
model.layers
```

Out[34]:
```
[<keras.layers.reshaping.flatten.Flatten at 0x27441d040d0>,
 <keras.layers.core.dense.Dense at 0x274481cdd00>,
 <keras.layers.core.dense.Dense at 0x274481cdfa0>,
 <keras.layers.core.dense.Dense at 0x274481cdd90>]
```

In [37]:
```python
hidden_2 = model.layers[2]
hidden_2.name
```

Out[37]:
```
'dense_1'
```

In [44]:
```python
weights, biases = hidden_1.get_weights()
weights
```

Out[44]:
```
array([[ 0.06354225,  0.01114564, -0.03783986, ...,  0.06744486,
         0.07393704,  0.03122401],
       [ 0.04999159,  0.06953265,  0.05095394, ...,  0.05454569,
         0.07067709,  0.0530611 ],
       [ 0.0180122 , -0.02287985,  0.05376284, ..., -0.03066178,
         0.0235892 , -0.01028987],
       ...,
       [-0.06614577,  0.06801577,  0.01924263, ..., -0.06020033,
        -0.03502942,  0.00944853],
       [-0.01033042,  0.05214235, -0.04772285, ...,  0.04146505,
        -0.07438264,  0.00270782],
       [ 0.06299587, -0.01260961,  0.03533407, ...,  0.01683035,
         0.00940549,  0.05805051]], dtype=float32)
```

In [45]:
```
weights.shape
```

Out[45]:
```
(784, 300)
```

In [46]:
```
biases.shape
```

Out[46]:
```
(300,)
```

In [47]:
```
model.compile(loss="sparse_categorical_crossentropy",
              optimizer = "sgd",
              metrics=["accuracy"])
```

In [49]:
```
history = model.fit(X_train, y_train, epochs = 30,
                    validation_data = (X_valid,y_valid))
```

```
Epoch 1/30
1719/1719 [==============================] - 5s 2ms/step - loss: 0.7236 - accuracy: 0.7624 - val_loss: 0.5019 -
val_accuracy: 0.8304
Epoch 2/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.4841 - accuracy: 0.8329 - val_loss: 0.4612 -
val_accuracy: 0.8326
Epoch 3/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.4373 - accuracy: 0.8470 - val_loss: 0.4245 -
val_accuracy: 0.8500
Epoch 4/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.4119 - accuracy: 0.8559 - val_loss: 0.3968 -
val_accuracy: 0.8592
Epoch 5/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3909 - accuracy: 0.8630 - val_loss: 0.3938 -
val_accuracy: 0.8584
Epoch 6/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3747 - accuracy: 0.8683 - val_loss: 0.3935 -
val_accuracy: 0.8620
Epoch 7/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3626 - accuracy: 0.8713 - val_loss: 0.3744 -
val_accuracy: 0.8652
Epoch 8/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3513 - accuracy: 0.8750 - val_loss: 0.3668 -
val_accuracy: 0.8662
Epoch 9/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3411 - accuracy: 0.8784 - val_loss: 0.3483 -
val_accuracy: 0.8744
Epoch 10/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3312 - accuracy: 0.8815 - val_loss: 0.3486 -
val_accuracy: 0.8742
Epoch 11/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3228 - accuracy: 0.8848 - val_loss: 0.3702 -
val_accuracy: 0.8646
Epoch 12/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3148 - accuracy: 0.8868 - val_loss: 0.3491 -
val_accuracy: 0.8746
Epoch 13/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3080 - accuracy: 0.8902 - val_loss: 0.3312 -
val_accuracy: 0.8800
Epoch 14/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.3012 - accuracy: 0.8911 - val_loss: 0.3439 -
val_accuracy: 0.8774
Epoch 15/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2952 - accuracy: 0.8938 - val_loss: 0.3371 -
val_accuracy: 0.8782
Epoch 16/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2887 - accuracy: 0.8963 - val_loss: 0.3277 -
val_accuracy: 0.8786
Epoch 17/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2836 - accuracy: 0.8981 - val_loss: 0.3361 -
val_accuracy: 0.8750
Epoch 18/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2777 - accuracy: 0.9002 - val_loss: 0.3303 -
val_accuracy: 0.8822
Epoch 19/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2731 - accuracy: 0.9011 - val_loss: 0.3506 -
val_accuracy: 0.8674
Epoch 20/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2679 - accuracy: 0.9043 - val_loss: 0.3223 -
val_accuracy: 0.8800
Epoch 21/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2631 - accuracy: 0.9049 - val_loss: 0.3163 -
val_accuracy: 0.8850
Epoch 22/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2581 - accuracy: 0.9065 - val_loss: 0.3153 -
val_accuracy: 0.8840
Epoch 23/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2544 - accuracy: 0.9090 - val_loss: 0.3411 -
```

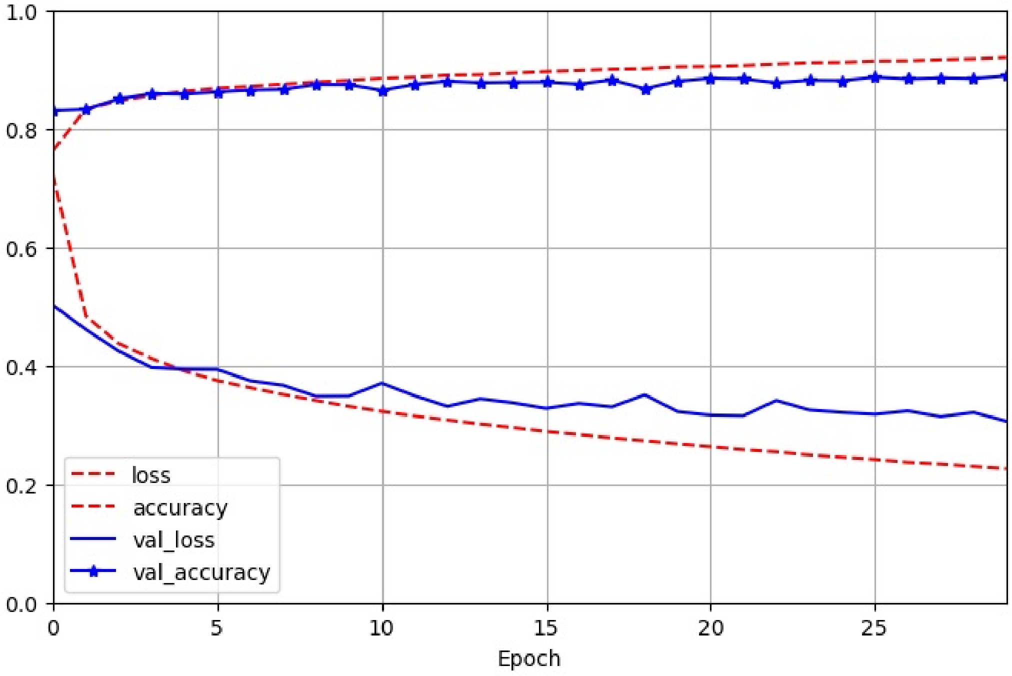
```
val_accuracy: 0.8772
Epoch 24/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2491 - accuracy: 0.9112 - val_loss: 0.3252 -
val_accuracy: 0.8816
Epoch 25/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2452 - accuracy: 0.9118 - val_loss: 0.3210 -
val_accuracy: 0.8810
Epoch 26/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2415 - accuracy: 0.9139 - val_loss: 0.3179 -
val_accuracy: 0.8868
Epoch 27/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2366 - accuracy: 0.9144 - val_loss: 0.3236 -
val_accuracy: 0.8838
Epoch 28/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2337 - accuracy: 0.9164 - val_loss: 0.3138 -
val_accuracy: 0.8854
Epoch 29/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2298 - accuracy: 0.9177 - val_loss: 0.3210 -
val_accuracy: 0.8844
Epoch 30/30
1719/1719 [==============================] - 4s 2ms/step - loss: 0.2260 - accuracy: 0.9201 - val_loss: 0.3059 -
val_accuracy: 0.8892
```

In [50]:

```
import matplotlib.pyplot as plt
import pandas as pd
```

In [51]:

```
pd.DataFrame(history.history).plot(
    figsize=(8,5), xlim=[0,29], ylim=[0, 1], grid=True, xlabel = "Epoch",
    style=["r--", "r--", "b-", "b-*"])
plt.show()
```



In [52]:

```
model.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 0s 1ms/step - loss: 0.3250 - accuracy: 0.8840
```

Out[52]:

```
[0.3250047266483307, 0.8840000033378601]
```

In [55]:

```
X_new = X_test[:3]
y_proba = model.predict(X_new)
y_proba.round(2)
```

```
1/1 [==============================] - 0s 21ms/step
```

Out[55]:

```
array([[0.  , 0.  , 0.  , 0.  , 0.  , 0.01, 0.  , 0.02, 0.  , 0.97],
       [0.  , 0.  , 1.  , 0.  , 0.  , 0.  , 0.  , 0.  , 0.  , 0.  ],
       [0.  , 1.  , 0.  , 0.  , 0.  , 0.  , 0.  , 0.  , 0.  , 0.  ]],
      dtype=float32)
```

In [57]:

```
import numpy as np
y_pred = y_proba.argmax(axis=-1)
y_pred
```

Out[57]:

```
array([9, 2, 1], dtype=int64)
```

In [59]:

```
np.array(class_names)[y_pred]
```

Out[59]:

```
array(['Ankle_boot', 'Pullover', 'Trouser'], dtype='<U10')
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js