In [1]:
```
import torch
from d2l import torch as d2l
```

In [2]:
```
X = torch.tensor([[1.0, 2.0, 3.0], [4.0, 5.0, 6.0]])
X.sum(0, keepdims=True), X.sum(1, keepdims=True)
```

Out[2]:
```
(tensor([[5., 7., 9.]]),
 tensor([[ 6.],
         [15.]]))
```

In [3]:
```
def softmax(X):
    X_exp = torch.exp(X)
    partition = X_exp.sum(1, keepdims=True)
    return X_exp / partition
```

In [4]:
```
X = torch.rand((2, 5))
X_prob = softmax(X)
X_prob, X_prob.sum(1)
```

Out[4]:
```
(tensor([[0.2128, 0.1564, 0.1422, 0.1940, 0.2945],
         [0.1785, 0.2052, 0.1561, 0.2170, 0.2432]]),
 tensor([1., 1.]))
```

In [5]:
```
class SoftmaxRegressionScratch(d2l.Classifier):
    def __init__(self, num_inputs, num_outputs, lr, sigma=0.01):
        super().__init__()
        self.save_hyperparameters()
        self.W = torch.normal(0, sigma, size=(num_inputs, num_outputs),
                              requires_grad=True)
        self.b = torch.zeros(num_outputs, requires_grad=True)

    def parameters(self):
        return [self.W, self.b]
```

In [6]:
```
@d2l.add_to_class(SoftmaxRegressionScratch)
def forward(self, X):
    X = X.reshape((-1, self.W.shape[0]))
    return softmax(torch.matmul(X, self.W) + self.b)
```

In [7]:
```
y = torch.tensor([0, 2])
y_hat = torch.tensor([[0.1, 0.3, 0.6], [0.3, 0.2, 0.5]])
y_hat[[0, 1], y]
```

Out[7]:
```
tensor([0.1000, 0.5000])
```

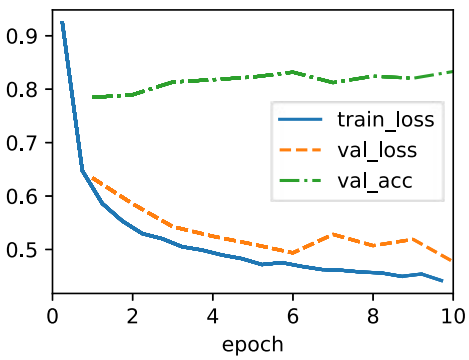
In [8]:
```
def cross_entropy(y_hat, y):
    return -torch.log(y_hat[list(range(len(y_hat))), y]).mean()

cross_entropy(y_hat, y)
```

Out[8]:
```
tensor(1.4979)
```

In [9]:
```
@d2l.add_to_class(SoftmaxRegressionScratch)
def loss(self, y_hat, y):
    return cross_entropy(y_hat, y)
```

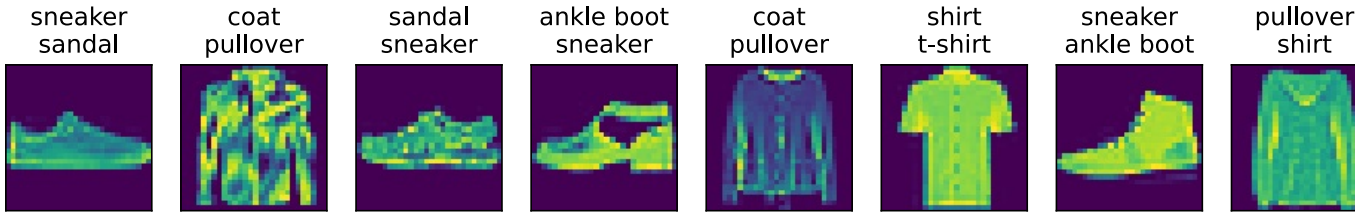
In [10]:
```
data = d2l.FashionMNIST(batch_size=256)
model = SoftmaxRegressionScratch(num_inputs=784, num_outputs=10, lr=0.1)
trainer = d2l.Trainer(max_epochs=10)
trainer.fit(model, data)
```



In [11]:
```
X, y = next(iter(data.val_dataloader()))
preds = model(X).argmax(axis=1)
preds.shape
```

Out[11]: torch.Size([256])

In [12]:
```
wrong = preds.type(y.dtype) != y
X, y, preds = X[wrong], y[wrong], preds[wrong]
labels = [a+'\n'+b for a, b in zip(
    data.text_labels(y), data.text_labels(preds))]
data.visualize([X, y], labels=labels)
```



Discussion: In this section, I implemented the softmax function and went through the process of actually training a model using the FashionMNIST dataset. Much of it involved reapplying concepts we had previously covered, with only slight changes in format, so it wasn't too difficult to understand. Watching the graph as the loss values steadily decreased during training was interesting. The part where incorrect predictions were visualized was something I hadn't seen before, but it was fascinating to note that there weren't any completely absurd mistakes, like confusing shoes for a top. The errors made were more in line with mistakes a tired person might make, which made me pay close attention to this aspect.

In [ ]: