Fine Tuning

In [1]:
```python
%matplotlib inline
import d2l
from mxnet import gluon, init, nd
from mxnet.gluon import data as gdata, loss as gloss, model_zoo
from mxnet.gluon import utils as gutils
import os
import zipfile
```

Hot Dog Recognition Dataset

Download the Data Set

In [2]:
```python
data_dir = '../data'
base_url = 'https://apache-mxnet.s3-accelerate.amazonaws.com/'
fname = gutils.download(
    base_url + 'gluon/dataset/hotdog.zip',
    path=data_dir, sha1_hash='fba480ffa8aa7e0febbb511d181409f899b9baa5')
with zipfile.ZipFile(fname, 'r') as z:
    z.extractall(data_dir)
```

Read the Dataset

In [3]:
```
train_imgs = gdata.vision.ImageFolderDataset(
    os.path.join(data_dir, 'hotdog/train'))
test_imgs = gdata.vision.ImageFolderDataset(
    os.path.join(data_dir, 'hotdog/test'))

hotdogs = [train_imgs[i][0] for i in range(8)]
not_hotdogs = [train_imgs[-i - 1][0] for i in range(8)]
d2l.show_images(hotdogs + not_hotdogs, 2, 8, scale=1.4);
```

Image Augmentation

```
In [5]:  # We specify the mean and variance of the three RGB channels to normalize the imag
         e channel.
         normalize = gdata.vision.transforms.Normalize(
             [0.485, 0.456, 0.406], [0.229, 0.224, 0.225])

         train_augs = gdata.vision.transforms.Compose([
             gdata.vision.transforms.RandomResizedCrop(224),
             gdata.vision.transforms.RandomFlipLeftRight(),
             gdata.vision.transforms.ToTensor(),
             normalize])

         test_augs = gdata.vision.transforms.Compose([
             gdata.vision.transforms.Resize(256),
             gdata.vision.transforms.CenterCrop(224),
             gdata.vision.transforms.ToTensor(),
             normalize])
```

Model

Download Pre-trained Models

In [6]:
```
pretrained_net = model_zoo.vision.resnet18_v2(pretrained=True)
pretrained_net.output
```

Initialize Model for Fine Tuning

In [8]:
```
finetune_net = model_zoo.vision.resnet18_v2(classes=2)
finetune_net.features = pretrained_net.features
finetune_net.output.initialize(init.Xavier())
# The model parameters in output will be updated using a learning rate ten times g
reater.
finetune_net.output.collect_params().setattr('lr_mult', 10)
```

Fine Tune

```
In [9]:  def train_fine_tuning(net, learning_rate, batch_size=128, num_epochs=5):
             train_iter = gdata.DataLoader(
                 train_imgs.transform_first(train_augs), batch_size, shuffle=True)
             test_iter = gdata.DataLoader(
                 test_imgs.transform_first(test_augs), batch_size)
             ctx = d2l.try_all_gpus()
             net.collect_params().reset_ctx(ctx)
             net.hybridize()
             loss = gloss.SoftmaxCrossEntropyLoss()
             trainer = gluon.Trainer(net.collect_params(), 'sgd', {
                 'learning_rate': learning_rate, 'wd': 0.001})
             d2l.train(train_iter, test_iter, net, loss, trainer, ctx, num_epochs)
```

Train

In [10]:
```
train_fine_tuning(finetune_net, 0.01)
```

```
training on [gpu(0), gpu(1)]
epoch 1, loss 3.3067, train acc 0.690, test acc 0.921, time 16.9 sec
epoch 2, loss 0.3924, train acc 0.902, test acc 0.919, time 12.3 sec
epoch 3, loss 0.2863, train acc 0.919, test acc 0.911, time 12.4 sec
epoch 4, loss 0.4881, train acc 0.877, test acc 0.936, time 12.4 sec
epoch 5, loss 0.2394, train acc 0.923, test acc 0.870, time 12.5 sec
```

Train from Scratch

In [11]:
```
scratch_net = model_zoo.vision.resnet18_v2(classes=2)
scratch_net.initialize(init=init.Xavier())
train_fine_tuning(scratch_net, 0.1)
```

```
training on [gpu(0), gpu(1)]
epoch 1, loss 0.6512, train acc 0.708, test acc 0.795, time 12.4 sec
epoch 2, loss 0.3935, train acc 0.830, test acc 0.781, time 12.4 sec
epoch 3, loss 0.4056, train acc 0.820, test acc 0.828, time 12.5 sec
epoch 4, loss 0.3863, train acc 0.831, test acc 0.781, time 12.5 sec
epoch 5, loss 0.3592, train acc 0.842, test acc 0.830, time 12.4 sec
```