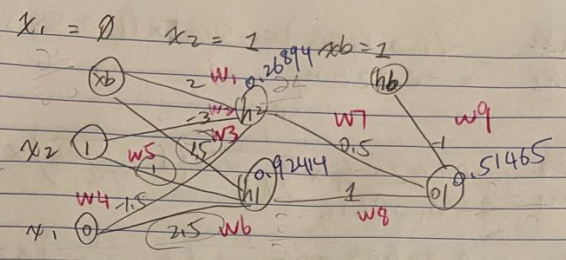


a). Forward pass computation

h_1 $(0, 1, 1) \cdot (2.5, 1, 1.5)$
$= 2.5(0) + (-1.5)(1) + (1)(1)$
$= 0 - 1.5 + 1 = -0.5$, sigmoid ≈ 0.92414

h_2 $(0, 1, 1) \cdot (-1.5, -3, 2)$
$= -1.5(0) + (-3)(1) + 2(1)$
$= -3 + 2 = -1$, sigmoid$(-1) \approx 0.26894$

o_0 sigmoid $((-1)(1) + (0.5)(0.26894) + (1)(0.92414)$
≈ 0.51465

b). $\frac{\partial L}{\partial h_1} = -(1 - 0.51465) \times 0.51465(1 + 0.51465) \times 0.92414$
$\frac{\partial h_1}{\partial h_1} = -0.11204$

$$\frac{\partial h_1}{\partial x_1} = 0.11204 \cdot \frac{1}{1+e^{-2.5}} \cdot \left(1 - \frac{1}{1+e^{-2.5}}\right) \cdot 2.5$$
$$\frac{\partial h_1}{\partial x_1} = 0.01964$$

$$\frac{\partial h_1}{\partial x_2} = 0.11204 \cdot \frac{1}{1+e^{-2.5}} \cdot \left(1 - \frac{1}{1+e^{-2.5}}\right) \cdot 1 = -0.00785$$

$$\frac{\partial L}{\partial h_2} = -(1-0.51465) \times 0.51465 \times (1-0.51465) \times 0.26894$$

$$\frac{\partial L}{\partial h_2} = -0.32604$$

$$\frac{\partial h_2}{\partial x_1} = 0.32604 \times \frac{1}{1+e^{-(-1)}}\left(1-\frac{1}{1+e^{-(-1)}}\right) \times -1.5$$

$$\frac{\partial h_2}{\partial x_1} = -0.09616$$

$$\frac{\partial h_2}{\partial x_2} = 0.32604 \times \frac{1}{1+e^{1}}\left(1-\frac{1}{1+e^{1}}\right) \times -3$$

$$\frac{\partial h_2}{\partial x_2} = -0.19231$$

c). $w_9 = -1 - (0.1)\cdot(-1) = -0.9$

$w_8 = 1 - (0.1)\cdot(-0.11204) = 1.011204$

$w_7 = 0.5 - (0.1)\cdot(-0.32604) = 0.532604$

$w_6 = 2.5 - (0.1)\cdot(0.01964) = 2.498036$

$w_5 = 1 - (0.1)\cdot(-0.00785) = 1.000785$

$w_4 = -1.5 - (0.1)\cdot(-0.09616) = -1.490384$

$w_3 = 1.5 - (0.1)\cdot(1.5) = 1.35$

$w_2 = -3 - (0.1)\cdot(-0.19231) = -2.980769$

$w_1 = 2 - (0.1)\cdot(2) = 1.8$

Elaina-7-lab-neural-net-music-classification

August 2, 2021

1 Assignment: Neural Networks for Music Classification

TODO: Edit this cell to fill in your NYU Net ID and your name:

- **Net ID**: yh4310
- **Name**: Elaina Huang

In this assignment, we will look at an audio classification problem. Given a sample of music, we want to determine which instrument (e.g. trumpet, violin, piano) is playing.

This assignment is closely based on one by Sundeep Rangan, from his IntroML GitHub repo.

[ ]:
```
import tensorflow as tf
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

1.1 Audio Feature Extraction with Librosa

The key to audio classification is to extract the correct features. The `librosa` package in python has a rich set of methods for extracting the features of audio samples commonly used in machine learning tasks, such as speech recognition and sound classification.

[ ]:
```
import librosa
import librosa.display
import librosa.feature
```

In this lab, we will use a set of music samples from the website:
http://theremin.music.uiowa.edu
This website has a great set of samples for audio processing.

We will use the `wget` command to retrieve one file to our Google Colab storage area. (We can run `wget` and many other basic Linux commands in Colab by prefixing them with a `!` or `%`.)

[ ]:
```
!wget "http://theremin.music.uiowa.edu/sound files/MIS/Woodwinds/
↪sopranosaxophone/SopSax.Vib.pp.C6Eb6.aiff"
```

```
--2021-08-02 01:16:06--  http://theremin.music.uiowa.edu/sound%20files/MIS/Woodw
inds/sopranosaxophone/SopSax.Vib.pp.C6Eb6.aiff
Resolving theremin.music.uiowa.edu (theremin.music.uiowa.edu)...
```

```
128.255.102.154, 2620:0:e50:680c::73
Connecting to theremin.music.uiowa.edu
(theremin.music.uiowa.edu)|128.255.102.154|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1418242 (1.4M) [audio/aiff]
Saving to: SopSax.Vib.pp.C6Eb6.aiff.2

SopSax.Vib.pp.C6Eb6 100%[===================>]   1.35M  3.68MB/s    in 0.4s

2021-08-02 01:16:06 (3.68 MB/s) - SopSax.Vib.pp.C6Eb6.aiff.2 saved
[1418242/1418242]
```

Now, if you click on the small folder icon on the far left of the Colab interface, you can see the files in your Colab storage. You should see the "SopSax.Vib.pp.C6Eb6.aiff" file appear there.

In order to listen to this file, we'l first convert it into the `wav` format. Again, we'l use a magic command to run a basic command-line utility: `ffmpeg`, a powerful tool for working with audio and video files.

[ ]:
```
aiff_file = 'SopSax.Vib.pp.C6Eb6.aiff'
wav_file = 'SopSax.Vib.pp.C6Eb6.wav'

!ffmpeg -y -i $aiff_file $wav_file
```

```
ffmpeg version 3.4.8-0ubuntu0.2 Copyright (c) 2000-2020 the FFmpeg developers
  built with gcc 7 (Ubuntu 7.5.0-3ubuntu1~18.04)
  configuration: --prefix=/usr --extra-version=0ubuntu0.2 --toolchain=hardened
--libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu
--enable-gpl --disable-stripping --enable-avresample --enable-avisynth --enable-
gnutls --enable-ladspa --enable-libass --enable-libbluray --enable-libbs2b
--enable-libcaca --enable-libcdio --enable-libflite --enable-libfontconfig
--enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm
--enable-libmp3lame --enable-libmysofa --enable-libopenjpeg --enable-libopenmpt
--enable-libopus --enable-libpulse --enable-librubberband --enable-librsvg
--enable-libshine --enable-libsnappy --enable-libsoxr --enable-libspeex
--enable-libssh --enable-libtheora --enable-libtwolame --enable-libvorbis
--enable-libvpx --enable-libwavpack --enable-libwebp --enable-libx265 --enable-
libxml2 --enable-libxvid --enable-libzmq --enable-libzvbi --enable-omx --enable-
openal --enable-opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm
--enable-libiec61883 --enable-chromaprint --enable-frei0r --enable-libopencv
--enable-libx264 --enable-shared
  libavutil      55. 78.100 / 55. 78.100
  libavcodec     57.107.100 / 57.107.100
  libavformat    57. 83.100 / 57. 83.100
  libavdevice    57. 10.100 / 57. 10.100
  libavfilter     6.107.100 /  6.107.100
  libavresample   3.  7.  0 /  3.  7.  0
  libswscale      4.  8.100 /  4.  8.100
  libswresample   2.  9.100 /  2.  9.100
```

```
  libpostproc    54.  7.100 / 54.  7.100
Guessed Channel Layout for Input Stream #0.0 : mono
Input #0, aiff, from 'SopSax.Vib.pp.C6Eb6.aiff':
  Duration: 00:00:16.07, start: 0.000000, bitrate: 705 kb/s
    Stream #0:0: Audio: pcm_s16be, 44100 Hz, mono, s16, 705 kb/s
Stream mapping:
  Stream #0:0 -> #0:0 (pcm_s16be (native) -> pcm_s16le (native))
Press [q] to stop, [?] for help
Output #0, wav, to 'SopSax.Vib.pp.C6Eb6.wav':
  Metadata:
    ISFT            : Lavf57.83.100
    Stream #0:0: Audio: pcm_s16le ([1][0][0][0] / 0x0001), 44100 Hz, mono, s16,
705 kb/s
    Metadata:
      encoder         : Lavc57.107.100 pcm_s16le
size=    1385kB time=00:00:16.07 bitrate= 705.6kbits/s speed=2.05e+03x
video:0kB audio:1384kB subtitle:0kB other streams:0kB global headers:0kB muxing
overhead: 0.005502%
```

Now, we can play the file directly from Colab. If you press the button, you will hear a soprano saxaphone (with vibrato) playing four notes (C, C#, D, Eb).

[ ]:
```
import IPython.display as ipd
ipd.Audio(wav_file)
```

[ ]:
```
<IPython.lib.display.Audio object>
```

Next, use `librosa` command `librosa.load` to read the audio file with filename `audio_file` and get the samples `y` and sample rate `sr`.

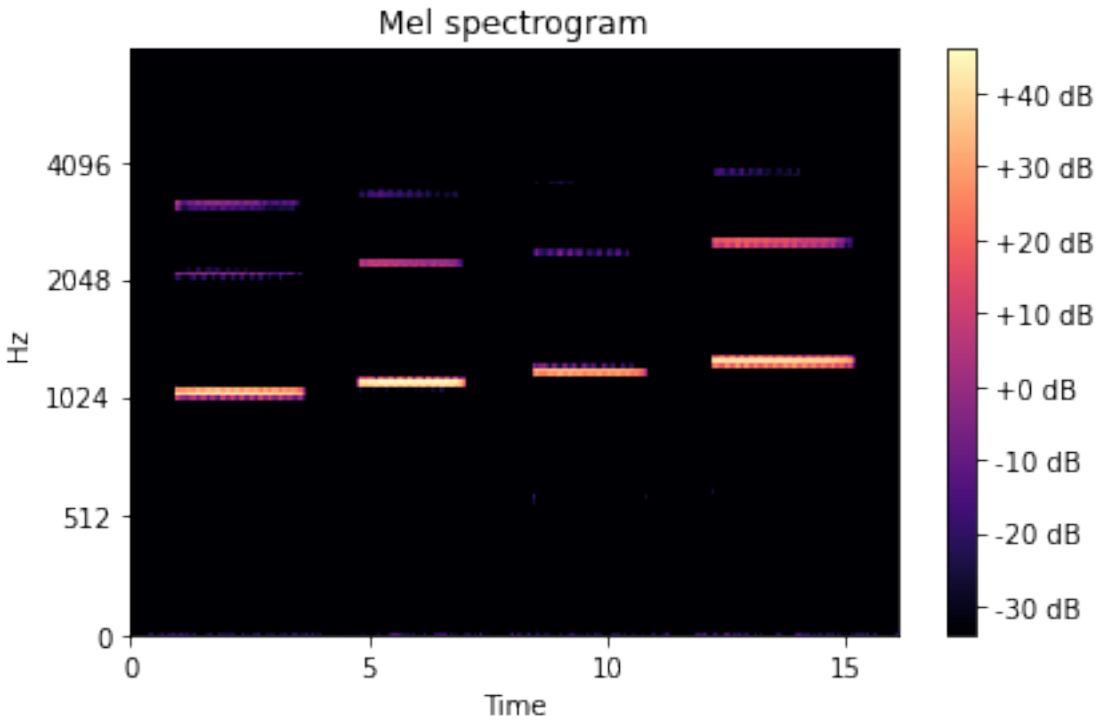
[ ]:
```
y, sr = librosa.load(aiff_file)
```

Feature engineering from audio files is an entire subject in its own right. A commonly used set of features are called the Mel Frequency Cepstral Coefficients (MFCCs). These are derived from the so-called mel spectrogram, which is something like a regular spectrogram, but the power and frequency are represented in log scale, which more naturally aligns with human perceptual processing.

You can run the code below to display the mel spectrogram from the audio sample.

You can easily see the four notes played in the audio track. You also see the 'harmonics' of each notes, which are other tones at integer multiples of the fundamental frequency of each note.

[ ]:
```
S = librosa.feature.melspectrogram(y=y, sr=sr, n_mels=128, fmax=8000)
librosa.display.specshow(librosa.amplitude_to_db(S),
                         y_axis='mel', fmax=8000, x_axis='time')
plt.colorbar(format='%+2.0f dB')
plt.title('Mel spectrogram')
plt.tight_layout()
```



1.2 Downloading the Data

Using the MFCC features described above, Prof. Juan Bello at NYU Steinhardt and his former PhD student Eric Humphrey have created a complete data set that can used for instrument classification. Essentially, they collected a number of data files from the website above. For each audio file, the segmented the track into notes and then extracted 120 MFCCs for each note. The goal is to recognize the instrument from the 120 MFCCs. The process of feature extraction is quite involved. So, we will just use their processed data.

To retrieve their data, visit

https://github.com/marl/dl4mir-tutorial/blob/master/README.md

and note the password listed on that page. Click on the link for "Instrument Dataset", enter the password, click on `instrument_dataset` to open the folder, and download the four files there. (You can "direct download" straight from this site, you don't need a Dropbox account.)

Then, upload the files to your Google Colab storage: click on the folder icon on the left to see your storage, if it isn't already open, and then click on "Upload".

Wait until *all* uploads have completed and the orange "circles" indicating uploads in progress are *gone*. (The training data especially will take some time to upload.)

Then, load the files with:

[ ]:
```
Xtr = np.load('uiowa_train_data.npy')
ytr = np.load('uiowa_train_labels.npy')
Xts = np.load('uiowa_test_data.npy')
yts = np.load('uiowa_test_labels.npy')
```

Examine the data you have just loaded in:

- How many training samples are there?
- How many test samples are there?
- What is the number of features for each sample?
- How many classes (i.e. instruments) are there?

Write some code to find these values and print them.

[ ]:
```
# TODO 1 Print basic details of the data
print(Xtr.shape) #66247 training samples, 120 features
print(ytr.shape) #66247 training samples
print(Xts.shape) #14904 testing samples, 120 features
print(yts.shape) #14904 testing samples
```

```
(66247, 120)
(66247,)
(14904, 120)
(14904,)
```

[ ]:
```
unique, counts = np.unique(ytr, return_counts=True)
dict(zip(unique, counts)) # 19 clases range from 0 - 9
```

[ ]:
```
{0: 6737,
 1: 2189,
 2: 4167,
 3: 8930,
 4: 4185,
 5: 8885,
 6: 12250,
 7: 2505,
 8: 2056,
 9: 14343}
```

Then, standardize the training and test data, `Xtr` and `Xts`, by removing the mean of each feature and scaling to unit variance.

You can do this manually, or using `sklearn`'s StandardScaler. (For an example showing how to use a `StandardScaler`, you can refer to the notebook on regularization.)

Standardizing the input data can make the gradient descent work better, by making the loss function "easier" to descend.

[ ]:
```
# TODO 2 Standardize the training and test data
```

[ ]:
```
from sklearn.preprocessing import StandardScaler
```

[ ]:
```
sXtr = StandardScaler().fit(Xtr)
sXts = StandardScaler().fit(Xts)
sedXtr = sXtr.transform(Xtr)
sedXts = sXtr.transform(Xts)
```

[ ]:
```
sedXtr
```

[ ]:
```
array([[-0.31827517, -0.32885595, -0.29351324, ..., -0.36022673,
        -0.25311227, -0.41772567],
```

```
        [-0.31018965, -0.2727317 , -0.2771818 , ..., -0.17807889,
          0.22098027, -0.04118938],
        [-0.27996699, -0.30067912, -0.26579675, ..., -0.22750957,
          0.43279931, -0.00953712],
        ...,
        [-0.70541453, -0.70723946, -0.70942862, ...,  3.33693492,
          4.20324336,  3.70233809],
        [-0.71897237, -0.71054126, -0.71014551, ...,  3.61543866,
          6.39878693,  3.31839917],
        [-0.72390302, -0.7118741 , -0.70965863, ...,  1.07922697,
          1.84041084,  1.42883157]])
```

[ ]:
```
sedXts
```

[ ]:
```
array([[-0.21750643, -0.21681385, -0.22310685, ..., -0.68837059,
        -0.80603516, -0.72767992],
       [-0.23309243, -0.23444571, -0.23947694, ..., -0.73434555,
        -0.71418557, -0.791695  ],
       [-0.16197075, -0.20414146, -0.22111234, ..., -0.66818614,
        -0.80244538, -0.68702929],
       ...,
       [-0.66945232, -0.66272087, -0.66150228, ..., -0.74286231,
        -0.74274835, -0.81456784],
       [-0.66689074, -0.66199011, -0.66236822, ..., -0.59806106,
        -0.64959124, -0.54035891],
       [-0.66550011, -0.66405765, -0.66227782, ..., -0.55623913,
        -0.55268092, -0.63606318]])
```

1.3 Building a Neural Network Classifier

Following the example in the demos you have seen, clear the keras session. Then, create a neural network `model` with:

- `nh=256` hidden units in a single dense hidden layer
- `sigmoid` activation
- select the input and output shapes, and output activation, according to the problem requirements
- print the model summary

[ ]:
```
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.layers import Dense, Activation
from tensorflow.keras import optimizers
import tensorflow.keras.backend as K
```

[ ]:
```
K.clear_session()
```

[ ]:
```
# TODO 3 construct the model
# model =  ...
# model.add( ...
```

```
# make sure to print the model summary
```

[ ]:
```
nin = 120  # dimension of input data
nh = 256  # number of hidden units
nout = 10  # number of outputs = 1 since this is binary
model = Sequential()
model.add(Dense(units=nh, input_shape=(nin,), activation='sigmoid',␣
 ↪name='hidden'))
model.add(Dense(units=nout, activation='sigmoid', name='output'))
```

[ ]:
```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
hidden (Dense)               (None, 256)               30976
_________________________________________________________________
output (Dense)               (None, 10)                2570
=================================================================
Total params: 33,546
Trainable params: 33,546
Non-trainable params: 0
_________________________________________________________________
```

Create an optimizer and compile the model. Select the appropriate loss function for this multi-class classification problem, and use an accuracy metric. For the optimizer, use the Adam optimizer with a learning rate of 0.001

[ ]:
```
# TODO 4
# opt = ...
# model.compile(...)
```

[ ]:
```
from tensorflow.keras import optimizers

opt = optimizers.Adam(learning_rate=0.01)
model.compile(optimizer=opt,
              loss='SparseCategoricalCrossentropy',
              metrics=['accuracy'])
```

Fit the model for 10 epochs using the scaled data for both the training and validation. Use the `validation_data` option to pass the *test* data. Use a batch size of 128. Your final accuracy should be >99%.

[ ]:
```
# TODO 5
hist = model.fit(sedXtr, ytr, epochs=10, batch_size=128,␣
 ↪validation_data=(sedXts,yts))
```

```
Epoch 1/10
518/518 [==============================] - 2s 4ms/step - loss: 0.1159 -
```

```
accuracy: 0.9649 - val_loss: 0.0401 - val_accuracy: 0.9873
Epoch 2/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0292 -
accuracy: 0.9905 - val_loss: 0.0684 - val_accuracy: 0.9719
Epoch 3/10
518/518 [==============================] - 2s 3ms/step - loss: 0.0221 -
accuracy: 0.9928 - val_loss: 0.0298 - val_accuracy: 0.9883
Epoch 4/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0162 -
accuracy: 0.9946 - val_loss: 0.0319 - val_accuracy: 0.9887
Epoch 5/10
518/518 [==============================] - 2s 3ms/step - loss: 0.0131 -
accuracy: 0.9958 - val_loss: 0.0428 - val_accuracy: 0.9852
Epoch 6/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0137 -
accuracy: 0.9955 - val_loss: 0.0698 - val_accuracy: 0.9777
Epoch 7/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0136 -
accuracy: 0.9954 - val_loss: 0.1998 - val_accuracy: 0.9419
Epoch 8/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0119 -
accuracy: 0.9960 - val_loss: 0.0266 - val_accuracy: 0.9913
Epoch 9/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0112 -
accuracy: 0.9964 - val_loss: 0.0451 - val_accuracy: 0.9862
Epoch 10/10
518/518 [==============================] - 2s 4ms/step - loss: 0.0138 -
accuracy: 0.9955 - val_loss: 0.0248 - val_accuracy: 0.9924
```

[ ]:
```
layer_hid = model.get_layer('hidden')
Wh, bh = layer_hid.get_weights()
print('Wh=')
print(Wh)
print('bh=')
print(bh)

layer_out = model.get_layer('output')
Wo, bo = layer_out.get_weights()
print('Wo=')
print(Wo)
print('bo=')
print(bo)
```

```
Wh=
[[-1.0357347  -1.288332    0.26060858 ...  0.44607073 -0.2388519
   0.89173204]
 [-1.9841896  -1.1273352   0.35888445 ...  0.6772284   0.1878151
```

```
   0.97033626]
 [-2.243175   -1.0619316   0.2198993  ...  0.68432826  0.39029834
   0.9071025 ]
 ...
 [ 0.5681561  -0.24930915 -0.38704965 ... -0.09193856  0.2649032
   0.6449734 ]
 [ 0.5067169  -0.26378724  0.46758956 ... -0.19509844 -0.06371458
   0.3396835 ]
 [ 0.83916914 -0.24262987  0.38685656 ... -0.3607718  -0.21308637
   0.37131703]]
bh=
[-1.7596115  -0.08492861  0.15221629 -0.21685426 -1.2319691  -0.88725215
 -0.5227871   0.17183185 -1.1838698  -0.3013433   0.53660494  0.17818618
  0.04679026  0.8654858  -0.04050119 -0.7232094  -0.03272694  0.30670503
 -1.2389654  -1.1851254  -0.36678976  0.46816745 -1.0467991  -0.06012592
  0.24032684 -0.72078556 -0.8260454  -1.0249029  -0.79410416 -0.37095472
 -0.09212591 -0.55634433  0.06060988  0.4694435  -0.41151142  0.7551771
  0.26216823 -0.52598137 -0.714271   -0.55842984 -0.6264155   0.8158096
 -0.07365468 -0.02667901 -0.18209712 -0.84102154 -0.20690697  0.34167135
  0.42607346 -1.2307453   1.3535889  -0.42203656 -1.2110945   0.02282099
 -0.38234743  0.53132975  1.5414295  -1.4167897   0.34984067  0.17686428
  0.4914292  -0.8149264   0.38840845 -0.910645   -0.45358327 -0.92157656
  1.4892316   1.9278079  -0.2727216   1.5776145   0.24959521 -1.212435
 -0.6649028   0.3350745  -0.12930086  1.3467954  -1.2873743  -0.8778744
  0.0481641   0.6329966  -0.5412258   0.60287577 -0.6952399  -0.8797395
  0.5883572  -0.18504362  0.95148426 -0.77677256  0.25933334 -1.2664529
 -1.3858135  -1.4711524  -0.86436     0.86901575  0.00678577 -0.7754064
  0.35579127  0.46240705 -0.14357436  0.5432021  -0.45955464  0.09947857
 -0.6040458  -1.5546392   0.41362515 -0.1336553   0.15326945 -0.40464038
  0.09384907 -1.1234016  -0.14842656  0.43991187 -1.2125529  -1.0113354
 -1.0277265   0.15439898 -0.30521205 -0.2693156  -0.33074245 -0.96944046
 -0.7121394   0.21464399  0.6177835  -0.24721022 -0.57662076  0.2797125
 -0.26752326 -0.09159581  0.05187046  0.14804426  0.09472799 -0.44631892
 -0.8285191  -0.94084346 -0.06837861 -0.14368287 -1.8438268  -0.28678495
  0.42246804  0.57034963 -0.21802685 -0.67085016 -0.61451304 -0.30713707
 -0.97488153  0.09406853 -0.088985    1.0236152   0.3785611  -0.3889793
  0.30640215  0.6081981   0.08023655 -0.06319301 -1.0280299   0.1874598
 -0.54906905 -1.7011296  -1.5484954  -0.81040955 -0.02296739 -0.43861443
 -0.31718278  0.53174865  0.08775523 -0.17599252 -0.98187184  0.10860491
 -0.80870396 -0.37846562  1.5010173  -0.1735553  -0.725549    0.22947301
 -0.22593658  0.4485167   0.22130856 -0.8865778  -0.38595206 -0.08077076
 -0.78780526  0.22693491 -0.41978672  0.04761651 -0.37782148 -0.23955576
  0.41164526 -0.4491597  -0.29091147 -0.8068753  -0.9311858   0.5866984
  0.6876991  -0.52446985 -0.9240306   0.30788636 -0.40822285 -0.9820488
 -1.0284501  -0.49402523 -0.08578175 -1.2966938  -0.564863   -0.4942856
 -0.3811124  -0.4548744  -0.5265873  -0.64132327 -1.1826363  -0.7609367
 -0.18824682  0.62824696  0.45280084 -0.4790501  -0.55433524 -0.6007936
  0.2820123   0.33980674  0.12267814  0.61811614  0.66677475 -0.5106261
```

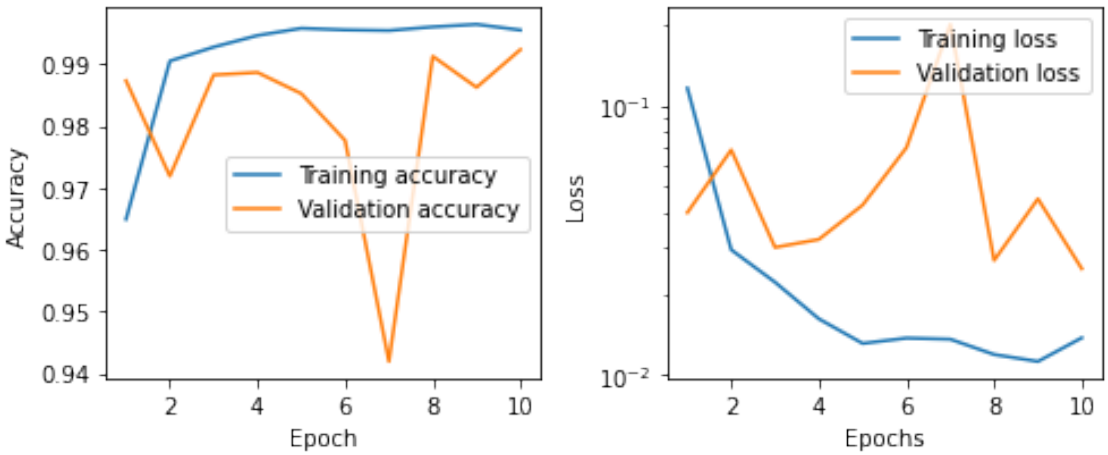
```
 -1.1502196   0.07825414 -0.38304994  0.63874984 -1.2326524   0.22343968
 -0.8614427  -0.8087146  -0.5299849   0.5233409  -0.14494528 -0.26180777
 -1.2489101   0.82824135 -0.5620146  -0.60060734 -0.28256044 -1.2867603
 -1.08644    -0.2560121  -0.8641446  -0.48940682  1.3379982  -0.6266881
  0.7635394   0.44981718 -0.49861887 -0.50217366  0.2407402  -0.8119059
  0.45333824  0.09741175  0.37146282 -1.6534733 ]
Wo=
[[-0.95994383 -0.2951179  -0.0564741  ... -0.01335652  0.30383164
   0.07108483]
 [ 0.14035158 -0.76995134  0.4359499  ... -0.17286243 -0.54954916
  -0.07828793]
 [-0.22436078  0.35794336 -0.83017147 ...  0.22320193  0.32632357
   0.4279773 ]
 ...
 [-0.26360443  0.27576268 -1.0762918  ...  0.31758732  0.092834
   0.10360134]
 [-0.32399324  0.33788443 -1.875721   ... -0.1007391  -0.99300057
   0.17744914]
 [-0.23568925 -1.1435298  -0.01502115 ... -1.6165428   0.12906754
   0.08558007]]
bo=
[ 0.00484908  0.05791347  0.09361214 -0.00021645 -0.14475901 -0.0192975
 -0.01161742  0.03807561  0.07494733 -0.07414016]
```

Plot the training and validation accuracy saved in `hist.history` dictionary, on the same plot. This gives one accuracy value per epoch. You should see that the validation accuracy saturates around 99%. After that it may "bounce around" a little due to the noise in the stochastic mini-batch gradient descent.

Make sure to label each axis, and each series (training vs. validation).

[ ]:
```python
# TODO 6A
import seaborn as sns

plt.figure(figsize=(7,3))

plt.subplot(1,2,1)
train_acc = hist.history['accuracy'];
val_acc = hist.history['val_accuracy'];

nepochs = len(train_acc);
sns.lineplot(x=np.arange(1,nepochs+1), y=train_acc, label='Training accuracy');
sns.lineplot(x=np.arange(1,nepochs+1), y=val_acc, label='Validation accuracy');
plt.xlabel('Epoch');
plt.ylabel('Accuracy');

plt.subplot(1,2,2)
train_loss = hist.history['loss']
val_loss = hist.history['val_loss']
```

```
sns.lineplot(x=np.arange(1,nepochs+1), y=train_loss, label='Training loss');
sns.lineplot(x=np.arange(1,nepochs+1), y=val_loss, label='Validation loss');
plt.yscale('log')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.tight_layout()
```

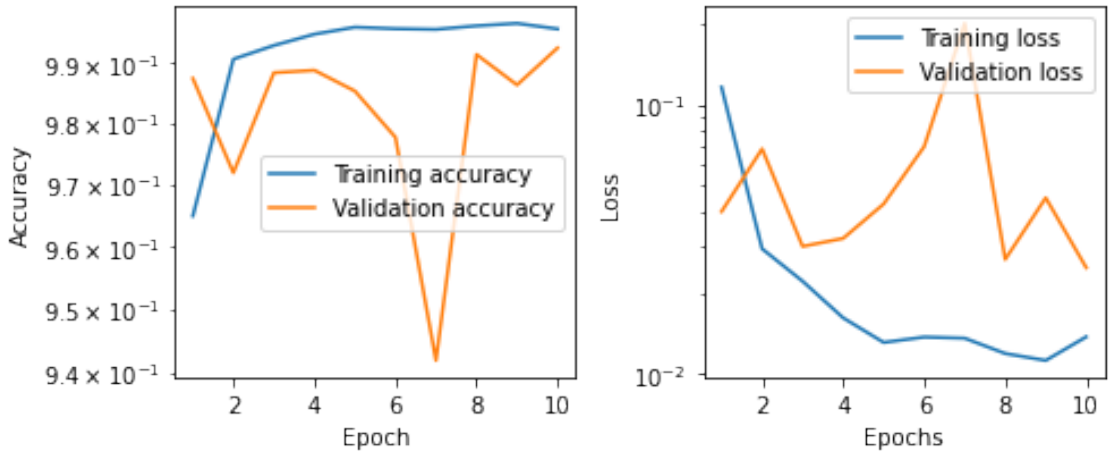


Plot the training and validation loss values saved in the `hist.history` dictionary, on the same plot. You should see that the training loss is steadily decreasing. Use the `semilogy` plot so that the y-axis is log scale.

Make sure to label each axis, and each series (training vs. validation).

[ ]:
```
# TODO 6B
plt.figure(figsize=(7,3))

plt.subplot(1,2,1)
train_acc = hist.history['accuracy'];
val_acc = hist.history['val_accuracy'];

nepochs = len(train_acc);

plt.semilogy(np.arange(1,nepochs+1), train_acc , label='Training accuracy')
plt.semilogy(np.arange(1,nepochs+1), val_acc , label='Validation accuracy')
plt.legend()

plt.xlabel('Epoch');
plt.ylabel('Accuracy');

plt.subplot(1,2,2)
train_loss = hist.history['loss']
val_loss = hist.history['val_loss']
```

```
plt.semilogy(np.arange(1,nepochs+1), train_loss, label='Training loss' )
plt.semilogy(np.arange(1,nepochs+1), val_loss, label='Validation loss' )
plt.legend()

plt.yscale('log')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.tight_layout()
```



1.4 Varying the Learning Rate

One challenge in training neural networks is the selection of the learning rate. Repeat your model preparation and fitting code, but try four learning rates as shown in the vector `rates`. In each iteration of the loop:

- clear the session with `K.clear_session()` to free up memory from models that are no longer in scope
- construct the network
- select the optimizer. Use the Adam optimizer with the learning rate specific to this iteration
- train the model for 20 epochs
- save the history of training and validation accuracy and loss for this model

[ ]:
```
# TODO 7
rates = [0.1, 0.01,0.001,0.0001]
graph_name = ['hist1', 'hist2', 'hist3', 'hist4']
idx = 0

for r in rates:
  K.clear_session()

  nin = 120  # dimension of input data
```

```
nh = 256  # number of hidden units
nout = 10  # number of outputs = 1 since this is binary
model = Sequential()
model.add(Dense(units=nh, input_shape=(nin,), activation='sigmoid',␣
↪name='hidden'))
model.add(Dense(units=nout, activation='sigmoid', name='output'))

opt = optimizers.Adam(learning_rate=r) #choose optimizer
model.compile(optimizer=opt,
              loss='SparseCategoricalCrossentropy',
              metrics=['accuracy'])

graph_name[idx] = model.fit(sedXtr, ytr, epochs=20, batch_size=128,␣
↪validation_data=(sedXts,yts)) #fit the model with 20 epochs
print(graph_name[idx])
idx += 1
```

```
Epoch 1/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2802 -
accuracy: 0.9408 - val_loss: 0.2795 - val_accuracy: 0.9329
Epoch 2/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2166 -
accuracy: 0.9545 - val_loss: 0.3743 - val_accuracy: 0.9361
Epoch 3/20
518/518 [==============================] - 2s 3ms/step - loss: 0.1840 -
accuracy: 0.9648 - val_loss: 0.4052 - val_accuracy: 0.9345
Epoch 4/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2248 -
accuracy: 0.9630 - val_loss: 0.3451 - val_accuracy: 0.9558
Epoch 5/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1896 -
accuracy: 0.9722 - val_loss: 0.3549 - val_accuracy: 0.9585
Epoch 6/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2134 -
accuracy: 0.9677 - val_loss: 0.2988 - val_accuracy: 0.9587
Epoch 7/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2227 -
accuracy: 0.9706 - val_loss: 0.5441 - val_accuracy: 0.9597
Epoch 8/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2245 -
accuracy: 0.9740 - val_loss: 0.4811 - val_accuracy: 0.9469
Epoch 9/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2192 -
accuracy: 0.9755 - val_loss: 0.7720 - val_accuracy: 0.8851
Epoch 10/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2126 -
accuracy: 0.9738 - val_loss: 0.4838 - val_accuracy: 0.9621
```

```
Epoch 11/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1803 -
accuracy: 0.9804 - val_loss: 0.3295 - val_accuracy: 0.9682
Epoch 12/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1866 -
accuracy: 0.9793 - val_loss: 0.6714 - val_accuracy: 0.9534
Epoch 13/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1907 -
accuracy: 0.9807 - val_loss: 0.8081 - val_accuracy: 0.9532
Epoch 14/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2046 -
accuracy: 0.9802 - val_loss: 0.6472 - val_accuracy: 0.9489
Epoch 15/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2488 -
accuracy: 0.9775 - val_loss: 0.6648 - val_accuracy: 0.9607
Epoch 16/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2213 -
accuracy: 0.9804 - val_loss: 0.4641 - val_accuracy: 0.9700
Epoch 17/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2451 -
accuracy: 0.9788 - val_loss: 0.6512 - val_accuracy: 0.9614
Epoch 18/20
518/518 [==============================] - 2s 3ms/step - loss: 0.2383 -
accuracy: 0.9805 - val_loss: 1.0682 - val_accuracy: 0.9401
Epoch 19/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2062 -
accuracy: 0.9823 - val_loss: 0.4503 - val_accuracy: 0.9699
Epoch 20/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1923 -
accuracy: 0.9826 - val_loss: 0.7991 - val_accuracy: 0.9563
<tensorflow.python.keras.callbacks.History object at 0x7ff7f99dc450>
Epoch 1/20
518/518 [==============================] - 3s 5ms/step - loss: 0.1238 -
accuracy: 0.9623 - val_loss: 0.0422 - val_accuracy: 0.9862
Epoch 2/20
518/518 [==============================] - 2s 3ms/step - loss: 0.0293 -
accuracy: 0.9904 - val_loss: 0.0453 - val_accuracy: 0.9832
Epoch 3/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0195 -
accuracy: 0.9940 - val_loss: 0.0319 - val_accuracy: 0.9885
Epoch 4/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0198 -
accuracy: 0.9932 - val_loss: 0.0570 - val_accuracy: 0.9799
Epoch 5/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0133 -
accuracy: 0.9956 - val_loss: 0.0439 - val_accuracy: 0.9862
Epoch 6/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0148 -
```

```
accuracy: 0.9951 - val_loss: 0.0792 - val_accuracy: 0.9722
Epoch 7/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0105 -
accuracy: 0.9967 - val_loss: 0.0501 - val_accuracy: 0.9813
Epoch 8/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0110 -
accuracy: 0.9964 - val_loss: 0.0328 - val_accuracy: 0.9895
Epoch 9/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0113 -
accuracy: 0.9962 - val_loss: 0.0500 - val_accuracy: 0.9838
Epoch 10/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0096 -
accuracy: 0.9968 - val_loss: 0.0630 - val_accuracy: 0.9805
Epoch 11/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0104 -
accuracy: 0.9968 - val_loss: 0.0464 - val_accuracy: 0.9888
Epoch 12/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0093 -
accuracy: 0.9973 - val_loss: 0.0792 - val_accuracy: 0.9796
Epoch 13/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0067 -
accuracy: 0.9979 - val_loss: 0.1032 - val_accuracy: 0.9758
Epoch 14/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0105 -
accuracy: 0.9968 - val_loss: 0.1050 - val_accuracy: 0.9822
Epoch 15/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0070 -
accuracy: 0.9979 - val_loss: 0.0882 - val_accuracy: 0.9839
Epoch 16/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0076 -
accuracy: 0.9976 - val_loss: 0.0541 - val_accuracy: 0.9871
Epoch 17/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0109 -
accuracy: 0.9967 - val_loss: 0.0658 - val_accuracy: 0.9852
Epoch 18/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0056 -
accuracy: 0.9983 - val_loss: 0.0594 - val_accuracy: 0.9887
Epoch 19/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0046 -
accuracy: 0.9986 - val_loss: 0.0577 - val_accuracy: 0.9866
Epoch 20/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0067 -
accuracy: 0.9979 - val_loss: 0.0617 - val_accuracy: 0.9877
<tensorflow.python.keras.callbacks.History object at 0x7ff7e5100150>
Epoch 1/20
518/518 [==============================] - 2s 4ms/step - loss: 0.3977 -
accuracy: 0.8905 - val_loss: 0.2179 - val_accuracy: 0.9421
Epoch 2/20
```

```
518/518 [==============================] - 2s 4ms/step - loss: 0.1177 -
accuracy: 0.9715 - val_loss: 0.1135 - val_accuracy: 0.9672
Epoch 3/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0711 -
accuracy: 0.9829 - val_loss: 0.0758 - val_accuracy: 0.9816
Epoch 4/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0500 -
accuracy: 0.9878 - val_loss: 0.0599 - val_accuracy: 0.9845
Epoch 5/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0387 -
accuracy: 0.9898 - val_loss: 0.0471 - val_accuracy: 0.9868
Epoch 6/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0305 -
accuracy: 0.9922 - val_loss: 0.0360 - val_accuracy: 0.9901
Epoch 7/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0254 -
accuracy: 0.9934 - val_loss: 0.0380 - val_accuracy: 0.9893
Epoch 8/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0211 -
accuracy: 0.9946 - val_loss: 0.0420 - val_accuracy: 0.9868
Epoch 9/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0183 -
accuracy: 0.9951 - val_loss: 0.0302 - val_accuracy: 0.9919
Epoch 10/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0156 -
accuracy: 0.9961 - val_loss: 0.0309 - val_accuracy: 0.9900
Epoch 11/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0141 -
accuracy: 0.9965 - val_loss: 0.0262 - val_accuracy: 0.9916
Epoch 12/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0124 -
accuracy: 0.9968 - val_loss: 0.0313 - val_accuracy: 0.9893
Epoch 13/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0112 -
accuracy: 0.9972 - val_loss: 0.0227 - val_accuracy: 0.9935
Epoch 14/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0101 -
accuracy: 0.9975 - val_loss: 0.0218 - val_accuracy: 0.9929
Epoch 15/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0091 -
accuracy: 0.9977 - val_loss: 0.0273 - val_accuracy: 0.9899
Epoch 16/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0085 -
accuracy: 0.9979 - val_loss: 0.0209 - val_accuracy: 0.9925
Epoch 17/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0076 -
accuracy: 0.9982 - val_loss: 0.0234 - val_accuracy: 0.9917
Epoch 18/20
```

```
518/518 [==============================] - 2s 4ms/step - loss: 0.0069 -
accuracy: 0.9984 - val_loss: 0.0231 - val_accuracy: 0.9918
Epoch 19/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0067 -
accuracy: 0.9983 - val_loss: 0.0219 - val_accuracy: 0.9925
Epoch 20/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0069 -
accuracy: 0.9982 - val_loss: 0.0243 - val_accuracy: 0.9915
<tensorflow.python.keras.callbacks.History object at 0x7ff7f9990690>
Epoch 1/20
518/518 [==============================] - 2s 4ms/step - loss: 1.2483 -
accuracy: 0.6057 - val_loss: 0.9682 - val_accuracy: 0.6250
Epoch 2/20
518/518 [==============================] - 2s 4ms/step - loss: 0.6501 -
accuracy: 0.8108 - val_loss: 0.6733 - val_accuracy: 0.7863
Epoch 3/20
518/518 [==============================] - 2s 3ms/step - loss: 0.4562 -
accuracy: 0.8903 - val_loss: 0.5180 - val_accuracy: 0.8440
Epoch 4/20
518/518 [==============================] - 2s 4ms/step - loss: 0.3545 -
accuracy: 0.9193 - val_loss: 0.4244 - val_accuracy: 0.8760
Epoch 5/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2901 -
accuracy: 0.9351 - val_loss: 0.3488 - val_accuracy: 0.9053
Epoch 6/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2445 -
accuracy: 0.9452 - val_loss: 0.3029 - val_accuracy: 0.9159
Epoch 7/20
518/518 [==============================] - 2s 4ms/step - loss: 0.2099 -
accuracy: 0.9529 - val_loss: 0.2588 - val_accuracy: 0.9275
Epoch 8/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1825 -
accuracy: 0.9586 - val_loss: 0.2243 - val_accuracy: 0.9385
Epoch 9/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1601 -
accuracy: 0.9631 - val_loss: 0.1969 - val_accuracy: 0.9459
Epoch 10/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1416 -
accuracy: 0.9668 - val_loss: 0.1712 - val_accuracy: 0.9543
Epoch 11/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1262 -
accuracy: 0.9705 - val_loss: 0.1565 - val_accuracy: 0.9559
Epoch 12/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1133 -
accuracy: 0.9732 - val_loss: 0.1448 - val_accuracy: 0.9585
Epoch 13/20
518/518 [==============================] - 2s 4ms/step - loss: 0.1024 -
accuracy: 0.9759 - val_loss: 0.1346 - val_accuracy: 0.9587
```

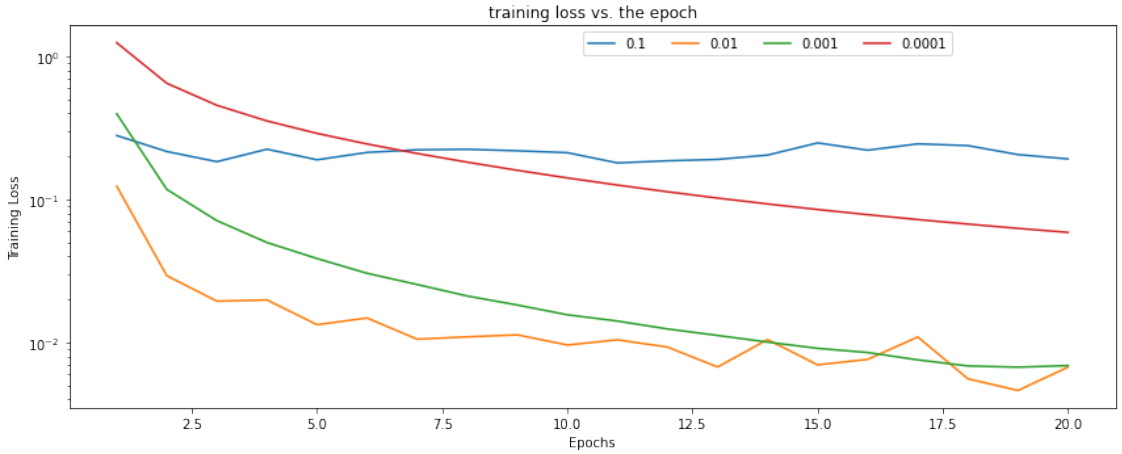
```
Epoch 14/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0931 -
accuracy: 0.9781 - val_loss: 0.1144 - val_accuracy: 0.9695
Epoch 15/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0851 -
accuracy: 0.9800 - val_loss: 0.1019 - val_accuracy: 0.9749
Epoch 16/20
518/518 [==============================] - 2s 3ms/step - loss: 0.0783 -
accuracy: 0.9817 - val_loss: 0.0954 - val_accuracy: 0.9758
Epoch 17/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0724 -
accuracy: 0.9833 - val_loss: 0.0901 - val_accuracy: 0.9760
Epoch 18/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0673 -
accuracy: 0.9842 - val_loss: 0.0832 - val_accuracy: 0.9797
Epoch 19/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0628 -
accuracy: 0.9852 - val_loss: 0.0767 - val_accuracy: 0.9813
Epoch 20/20
518/518 [==============================] - 2s 4ms/step - loss: 0.0589 -
accuracy: 0.9860 - val_loss: 0.0750 - val_accuracy: 0.9805
<tensorflow.python.keras.callbacks.History object at 0x7ff7f262cdd0>
```

Plot the training loss vs. the epoch number for all of the learning rates on one graph (use `semilogy` again). You should see that the lower learning rates are more stable, but converge slower, while with a learning rate that is too high, the gradient descent may fail to move towards weights that decrease the loss function.

Make sure to label each axis, and each series (training and validation).

Comment on the results.

[ ]:
```python
# TODO 8
plt.figure(figsize=(12,5))

train_loss1 = graph_name[0].history['loss']
train_loss2 = graph_name[1].history['loss']
train_loss3 = graph_name[2].history['loss']
train_loss4 = graph_name[3].history['loss']

plt.semilogy(np.arange(1,21), train_loss1, label = '0.1' )
plt.semilogy(np.arange(1,21), train_loss2, label = '0.01')
plt.semilogy(np.arange(1,21), train_loss3, label = '0.001')
plt.semilogy(np.arange(1,21), train_loss4, label = '0.0001')
plt.legend(bbox_to_anchor =(0.85, 1), ncol = 4)

plt.xlabel('Epochs')
plt.ylabel('Training Loss')
plt.title('training loss vs. the epoch')
plt.tight_layout()
```



Comments:

Rate 0.1 doesn't seem to show a trend that it will converge even with more trainings. The line flunctuates but seems to move around a straight horizontal line.

Rate 0.01 is very bumpy as well, but it does a better job than the rate 0.1. It's falling toward the minima, however the end appears to overshoot a little bit. Overall the trend shows it will converge if provide more trainings.

Rate 0.001 this lines is a lot smoother compare to the above two lines. It falls towards bottom quickly with a little sharp drop when approaching 2.5 epochs. The trend shows is currenly the closest line to converge soon.

Rate 0.0001 it very smooth. The smoothiest line among all the lines in this plot. The line, however, moving very slowly. It's still clearly to see a trend to go downwards but in a much slower rate. It is likely to converge if more training is provided.

1.5 Hidden layer size

The size of the hidden layer controls the network's ability to learn complicated feature representations.

Repeat your model preparation and fitting code, but loop over a range of hidden layer sizes: try all powers of 2 from 4 to 1024. In each iteration of the loop:

- clear the session with `K.clear_session()` to free up memory from models that are no longer in scope
- construct the network with the hidden layer size specific to this iteration.
- select the optimizer. Use the Adam optimizer with a 0.001 learning rate.
- train the model for **40** epochs
- save the history of training and validation accuracy and losses for this model

[205]:
```
# TODO 9
diff_hid_layer = []
str_name = []
exp = 2
nh = 0
```

```
while nh < 1024:
  K.clear_session()

  nin = 120  # dimension of input data
  nh = 2 ** exp # number of hidden units
  nout = 10  # number of outputs = 1 since this is binary
  model = Sequential()
  model.add(Dense(units=nh, input_shape=(nin,), activation='sigmoid',␣
 ↪name='hidden'))
  model.add(Dense(units=nout, activation='sigmoid', name='output'))

  opt = optimizers.Adam(learning_rate=0.001) #choose optimizer & learning rate
  model.compile(optimizer=opt,
                loss='SparseCategoricalCrossentropy',
                metrics=['accuracy'])
  model_name = "HiddenLayer - "+str(nh)
  str_name.append(model_name)
  model_name = model.fit(sedXtr, ytr, epochs=40, batch_size=128,␣
 ↪validation_data=(sedXts,yts)) #fit the model with 40 epochs
  diff_hid_layer.append(model_name)
  print(nh, exp)
  exp += 1
```

```
Epoch 1/40
518/518 [==============================] - 1s 2ms/step - loss: 1.7548 -
accuracy: 0.4254 - val_loss: 1.4893 - val_accuracy: 0.5070
Epoch 2/40
518/518 [==============================] - 1s 2ms/step - loss: 1.3076 -
accuracy: 0.5437 - val_loss: 1.1882 - val_accuracy: 0.6102
Epoch 3/40
518/518 [==============================] - 1s 2ms/step - loss: 1.0632 -
accuracy: 0.6447 - val_loss: 0.9992 - val_accuracy: 0.6618
Epoch 4/40
518/518 [==============================] - 1s 2ms/step - loss: 0.9044 -
accuracy: 0.6718 - val_loss: 0.8704 - val_accuracy: 0.6845
Epoch 5/40
518/518 [==============================] - 1s 2ms/step - loss: 0.7913 -
accuracy: 0.7005 - val_loss: 0.7709 - val_accuracy: 0.7071
Epoch 6/40
518/518 [==============================] - 1s 2ms/step - loss: 0.7069 -
accuracy: 0.7202 - val_loss: 0.6964 - val_accuracy: 0.7301
Epoch 7/40
518/518 [==============================] - 1s 2ms/step - loss: 0.6425 -
accuracy: 0.7402 - val_loss: 0.6429 - val_accuracy: 0.7443
Epoch 8/40
```

```
518/518 [==============================] - 1s 2ms/step - loss: 0.5929 -
accuracy: 0.7468 - val_loss: 0.6027 - val_accuracy: 0.7512
Epoch 9/40
518/518 [==============================] - 1s 2ms/step - loss: 0.5530 -
accuracy: 0.7608 - val_loss: 0.5659 - val_accuracy: 0.7613
Epoch 10/40
518/518 [==============================] - 1s 2ms/step - loss: 0.5177 -
accuracy: 0.7990 - val_loss: 0.5433 - val_accuracy: 0.7616
Epoch 11/40
518/518 [==============================] - 1s 2ms/step - loss: 0.4790 -
accuracy: 0.8537 - val_loss: 0.5118 - val_accuracy: 0.7958
Epoch 12/40
518/518 [==============================] - 1s 2ms/step - loss: 0.4132 -
accuracy: 0.9070 - val_loss: 0.4448 - val_accuracy: 0.8600
Epoch 13/40
518/518 [==============================] - 1s 2ms/step - loss: 0.3329 -
accuracy: 0.9407 - val_loss: 0.3895 - val_accuracy: 0.8797
Epoch 14/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2874 -
accuracy: 0.9431 - val_loss: 0.3518 - val_accuracy: 0.8906
Epoch 15/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2577 -
accuracy: 0.9445 - val_loss: 0.3375 - val_accuracy: 0.8873
Epoch 16/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2369 -
accuracy: 0.9454 - val_loss: 0.3142 - val_accuracy: 0.8963
Epoch 17/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2206 -
accuracy: 0.9463 - val_loss: 0.2964 - val_accuracy: 0.9002
Epoch 18/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2079 -
accuracy: 0.9475 - val_loss: 0.2863 - val_accuracy: 0.9032
Epoch 19/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1977 -
accuracy: 0.9491 - val_loss: 0.2768 - val_accuracy: 0.9041
Epoch 20/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1894 -
accuracy: 0.9495 - val_loss: 0.2795 - val_accuracy: 0.8995
Epoch 21/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1822 -
accuracy: 0.9504 - val_loss: 0.2725 - val_accuracy: 0.9013
Epoch 22/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1761 -
accuracy: 0.9512 - val_loss: 0.2740 - val_accuracy: 0.8997
Epoch 23/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1715 -
accuracy: 0.9513 - val_loss: 0.2613 - val_accuracy: 0.9050
Epoch 24/40
```

```
518/518 [==============================] - 1s 2ms/step - loss: 0.1669 -
accuracy: 0.9527 - val_loss: 0.2536 - val_accuracy: 0.9071
Epoch 25/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1630 -
accuracy: 0.9533 - val_loss: 0.2522 - val_accuracy: 0.9063
Epoch 26/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1596 -
accuracy: 0.9534 - val_loss: 0.2552 - val_accuracy: 0.9048
Epoch 27/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1566 -
accuracy: 0.9545 - val_loss: 0.2556 - val_accuracy: 0.9043
Epoch 28/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1538 -
accuracy: 0.9546 - val_loss: 0.2509 - val_accuracy: 0.9063
Epoch 29/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1514 -
accuracy: 0.9550 - val_loss: 0.2577 - val_accuracy: 0.9032
Epoch 30/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1490 -
accuracy: 0.9557 - val_loss: 0.2500 - val_accuracy: 0.9061
Epoch 31/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1470 -
accuracy: 0.9555 - val_loss: 0.2598 - val_accuracy: 0.9018
Epoch 32/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1447 -
accuracy: 0.9565 - val_loss: 0.2488 - val_accuracy: 0.9081
Epoch 33/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1425 -
accuracy: 0.9568 - val_loss: 0.2505 - val_accuracy: 0.9057
Epoch 34/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1405 -
accuracy: 0.9572 - val_loss: 0.2717 - val_accuracy: 0.8967
Epoch 35/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1390 -
accuracy: 0.9574 - val_loss: 0.2512 - val_accuracy: 0.9049
Epoch 36/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1370 -
accuracy: 0.9576 - val_loss: 0.2482 - val_accuracy: 0.9071
Epoch 37/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1351 -
accuracy: 0.9581 - val_loss: 0.2647 - val_accuracy: 0.8979
Epoch 38/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1339 -
accuracy: 0.9586 - val_loss: 0.2588 - val_accuracy: 0.9001
Epoch 39/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1322 -
accuracy: 0.9587 - val_loss: 0.2511 - val_accuracy: 0.9056
Epoch 40/40
```

```
518/518 [==============================] - 1s 2ms/step - loss: 0.1307 -
accuracy: 0.9589 - val_loss: 0.2489 - val_accuracy: 0.9052
4 2
Epoch 1/40
518/518 [==============================] - 1s 2ms/step - loss: 1.4918 -
accuracy: 0.6011 - val_loss: 1.2077 - val_accuracy: 0.6201
Epoch 2/40
518/518 [==============================] - 1s 2ms/step - loss: 0.8586 -
accuracy: 0.7460 - val_loss: 0.7990 - val_accuracy: 0.7387
Epoch 3/40
518/518 [==============================] - 1s 2ms/step - loss: 0.5836 -
accuracy: 0.8403 - val_loss: 0.5944 - val_accuracy: 0.8247
Epoch 4/40
518/518 [==============================] - 1s 2ms/step - loss: 0.4404 -
accuracy: 0.8991 - val_loss: 0.4651 - val_accuracy: 0.8631
Epoch 5/40
518/518 [==============================] - 1s 2ms/step - loss: 0.3530 -
accuracy: 0.9208 - val_loss: 0.3820 - val_accuracy: 0.8875
Epoch 6/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2922 -
accuracy: 0.9395 - val_loss: 0.3313 - val_accuracy: 0.9018
Epoch 7/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2466 -
accuracy: 0.9491 - val_loss: 0.2835 - val_accuracy: 0.9194
Epoch 8/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2121 -
accuracy: 0.9549 - val_loss: 0.2513 - val_accuracy: 0.9254
Epoch 9/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1859 -
accuracy: 0.9584 - val_loss: 0.2242 - val_accuracy: 0.9333
Epoch 10/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1652 -
accuracy: 0.9610 - val_loss: 0.2089 - val_accuracy: 0.9312
Epoch 11/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1484 -
accuracy: 0.9643 - val_loss: 0.1903 - val_accuracy: 0.9388
Epoch 12/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1345 -
accuracy: 0.9667 - val_loss: 0.1790 - val_accuracy: 0.9416
Epoch 13/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1226 -
accuracy: 0.9689 - val_loss: 0.1763 - val_accuracy: 0.9391
Epoch 14/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1125 -
accuracy: 0.9711 - val_loss: 0.1559 - val_accuracy: 0.9485
Epoch 15/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1039 -
accuracy: 0.9729 - val_loss: 0.1457 - val_accuracy: 0.9518
```

```
Epoch 16/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0962 -
accuracy: 0.9747 - val_loss: 0.1449 - val_accuracy: 0.9480
Epoch 17/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0897 -
accuracy: 0.9763 - val_loss: 0.1306 - val_accuracy: 0.9552
Epoch 18/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0838 -
accuracy: 0.9775 - val_loss: 0.1248 - val_accuracy: 0.9569
Epoch 19/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0782 -
accuracy: 0.9795 - val_loss: 0.1193 - val_accuracy: 0.9588
Epoch 20/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0732 -
accuracy: 0.9810 - val_loss: 0.1112 - val_accuracy: 0.9632
Epoch 21/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0686 -
accuracy: 0.9823 - val_loss: 0.1156 - val_accuracy: 0.9592
Epoch 22/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0646 -
accuracy: 0.9833 - val_loss: 0.1069 - val_accuracy: 0.9641
Epoch 23/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0611 -
accuracy: 0.9844 - val_loss: 0.1068 - val_accuracy: 0.9615
Epoch 24/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0581 -
accuracy: 0.9849 - val_loss: 0.0990 - val_accuracy: 0.9661
Epoch 25/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0555 -
accuracy: 0.9855 - val_loss: 0.0995 - val_accuracy: 0.9654
Epoch 26/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0532 -
accuracy: 0.9864 - val_loss: 0.1046 - val_accuracy: 0.9630
Epoch 27/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0512 -
accuracy: 0.9866 - val_loss: 0.0953 - val_accuracy: 0.9663
Epoch 28/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0492 -
accuracy: 0.9872 - val_loss: 0.0971 - val_accuracy: 0.9652
Epoch 29/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0475 -
accuracy: 0.9872 - val_loss: 0.0833 - val_accuracy: 0.9729
Epoch 30/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0460 -
accuracy: 0.9877 - val_loss: 0.0895 - val_accuracy: 0.9688
Epoch 31/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0444 -
accuracy: 0.9884 - val_loss: 0.0860 - val_accuracy: 0.9700
```

```
Epoch 32/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0432 -
accuracy: 0.9885 - val_loss: 0.0863 - val_accuracy: 0.9692
Epoch 33/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0418 -
accuracy: 0.9887 - val_loss: 0.0849 - val_accuracy: 0.9703
Epoch 34/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0407 -
accuracy: 0.9890 - val_loss: 0.0872 - val_accuracy: 0.9686
Epoch 35/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0397 -
accuracy: 0.9893 - val_loss: 0.0849 - val_accuracy: 0.9693
Epoch 36/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0385 -
accuracy: 0.9895 - val_loss: 0.1020 - val_accuracy: 0.9620
Epoch 37/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0376 -
accuracy: 0.9899 - val_loss: 0.0818 - val_accuracy: 0.9713
Epoch 38/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0367 -
accuracy: 0.9901 - val_loss: 0.0805 - val_accuracy: 0.9712
Epoch 39/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0357 -
accuracy: 0.9902 - val_loss: 0.0779 - val_accuracy: 0.9724
Epoch 40/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0349 -
accuracy: 0.9904 - val_loss: 0.0810 - val_accuracy: 0.9707
8 3
Epoch 1/40
518/518 [==============================] - 1s 2ms/step - loss: 1.1122 -
accuracy: 0.6923 - val_loss: 0.7741 - val_accuracy: 0.7709
Epoch 2/40
518/518 [==============================] - 1s 2ms/step - loss: 0.4905 -
accuracy: 0.8775 - val_loss: 0.4690 - val_accuracy: 0.8638
Epoch 3/40
518/518 [==============================] - 1s 2ms/step - loss: 0.3136 -
accuracy: 0.9278 - val_loss: 0.3284 - val_accuracy: 0.9225
Epoch 4/40
518/518 [==============================] - 1s 2ms/step - loss: 0.2303 -
accuracy: 0.9510 - val_loss: 0.2546 - val_accuracy: 0.9414
Epoch 5/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1777 -
accuracy: 0.9640 - val_loss: 0.2052 - val_accuracy: 0.9504
Epoch 6/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1407 -
accuracy: 0.9714 - val_loss: 0.1677 - val_accuracy: 0.9605
Epoch 7/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1147 -
```

```
accuracy: 0.9759 - val_loss: 0.1402 - val_accuracy: 0.9656
Epoch 8/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0959 -
accuracy: 0.9797 - val_loss: 0.1202 - val_accuracy: 0.9717
Epoch 9/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0819 -
accuracy: 0.9822 - val_loss: 0.1081 - val_accuracy: 0.9718
Epoch 10/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0711 -
accuracy: 0.9843 - val_loss: 0.0955 - val_accuracy: 0.9748
Epoch 11/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0627 -
accuracy: 0.9857 - val_loss: 0.0867 - val_accuracy: 0.9774
Epoch 12/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0557 -
accuracy: 0.9875 - val_loss: 0.0765 - val_accuracy: 0.9799
Epoch 13/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0498 -
accuracy: 0.9884 - val_loss: 0.0728 - val_accuracy: 0.9804
Epoch 14/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0448 -
accuracy: 0.9895 - val_loss: 0.0685 - val_accuracy: 0.9805
Epoch 15/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0406 -
accuracy: 0.9903 - val_loss: 0.0639 - val_accuracy: 0.9824
Epoch 16/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0374 -
accuracy: 0.9912 - val_loss: 0.0605 - val_accuracy: 0.9828
Epoch 17/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0345 -
accuracy: 0.9917 - val_loss: 0.0570 - val_accuracy: 0.9842
Epoch 18/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0319 -
accuracy: 0.9924 - val_loss: 0.0538 - val_accuracy: 0.9840
Epoch 19/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0298 -
accuracy: 0.9931 - val_loss: 0.0516 - val_accuracy: 0.9851
Epoch 20/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0280 -
accuracy: 0.9933 - val_loss: 0.0466 - val_accuracy: 0.9861
Epoch 21/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0264 -
accuracy: 0.9936 - val_loss: 0.0476 - val_accuracy: 0.9854
Epoch 22/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0249 -
accuracy: 0.9941 - val_loss: 0.0470 - val_accuracy: 0.9860
Epoch 23/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0236 -
```

```
accuracy: 0.9943 - val_loss: 0.0456 - val_accuracy: 0.9860
Epoch 24/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0225 -
accuracy: 0.9944 - val_loss: 0.0431 - val_accuracy: 0.9875
Epoch 25/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0215 -
accuracy: 0.9949 - val_loss: 0.0438 - val_accuracy: 0.9862
Epoch 26/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0205 -
accuracy: 0.9948 - val_loss: 0.0425 - val_accuracy: 0.9870
Epoch 27/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0196 -
accuracy: 0.9953 - val_loss: 0.0412 - val_accuracy: 0.9874
Epoch 28/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0188 -
accuracy: 0.9955 - val_loss: 0.0418 - val_accuracy: 0.9866
Epoch 29/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0180 -
accuracy: 0.9956 - val_loss: 0.0387 - val_accuracy: 0.9877
Epoch 30/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0174 -
accuracy: 0.9959 - val_loss: 0.0403 - val_accuracy: 0.9875
Epoch 31/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0168 -
accuracy: 0.9959 - val_loss: 0.0380 - val_accuracy: 0.9879
Epoch 32/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0163 -
accuracy: 0.9960 - val_loss: 0.0384 - val_accuracy: 0.9876
Epoch 33/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0156 -
accuracy: 0.9962 - val_loss: 0.0387 - val_accuracy: 0.9879
Epoch 34/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0151 -
accuracy: 0.9964 - val_loss: 0.0404 - val_accuracy: 0.9867
Epoch 35/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0148 -
accuracy: 0.9965 - val_loss: 0.0395 - val_accuracy: 0.9865
Epoch 36/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0143 -
accuracy: 0.9967 - val_loss: 0.0379 - val_accuracy: 0.9870
Epoch 37/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0138 -
accuracy: 0.9965 - val_loss: 0.0371 - val_accuracy: 0.9877
Epoch 38/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0135 -
accuracy: 0.9967 - val_loss: 0.0361 - val_accuracy: 0.9881
Epoch 39/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0130 -
```

```
accuracy: 0.9970 - val_loss: 0.0362 - val_accuracy: 0.9880
Epoch 40/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0130 -
accuracy: 0.9968 - val_loss: 0.0356 - val_accuracy: 0.9887
16 4
Epoch 1/40
518/518 [==============================] - 2s 2ms/step - loss: 0.8563 -
accuracy: 0.7574 - val_loss: 0.5407 - val_accuracy: 0.8329
Epoch 2/40
518/518 [==============================] - 1s 2ms/step - loss: 0.3123 -
accuracy: 0.9335 - val_loss: 0.2996 - val_accuracy: 0.9240
Epoch 3/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1887 -
accuracy: 0.9607 - val_loss: 0.2042 - val_accuracy: 0.9471
Epoch 4/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1319 -
accuracy: 0.9713 - val_loss: 0.1562 - val_accuracy: 0.9548
Epoch 5/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0994 -
accuracy: 0.9775 - val_loss: 0.1158 - val_accuracy: 0.9712
Epoch 6/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0788 -
accuracy: 0.9824 - val_loss: 0.0927 - val_accuracy: 0.9764
Epoch 7/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0646 -
accuracy: 0.9857 - val_loss: 0.0762 - val_accuracy: 0.9814
Epoch 8/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0544 -
accuracy: 0.9877 - val_loss: 0.0647 - val_accuracy: 0.9848
Epoch 9/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0466 -
accuracy: 0.9892 - val_loss: 0.0598 - val_accuracy: 0.9848
Epoch 10/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0406 -
accuracy: 0.9907 - val_loss: 0.0545 - val_accuracy: 0.9862
Epoch 11/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0357 -
accuracy: 0.9919 - val_loss: 0.0478 - val_accuracy: 0.9875
Epoch 12/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0317 -
accuracy: 0.9927 - val_loss: 0.0447 - val_accuracy: 0.9879
Epoch 13/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0283 -
accuracy: 0.9934 - val_loss: 0.0424 - val_accuracy: 0.9870
Epoch 14/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0258 -
accuracy: 0.9941 - val_loss: 0.0414 - val_accuracy: 0.9886
Epoch 15/40
```

```
518/518 [==============================] - 1s 2ms/step - loss: 0.0234 -
accuracy: 0.9946 - val_loss: 0.0401 - val_accuracy: 0.9875
Epoch 16/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0216 -
accuracy: 0.9947 - val_loss: 0.0362 - val_accuracy: 0.9886
Epoch 17/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0197 -
accuracy: 0.9952 - val_loss: 0.0342 - val_accuracy: 0.9890
Epoch 18/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0183 -
accuracy: 0.9956 - val_loss: 0.0343 - val_accuracy: 0.9885
Epoch 19/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0170 -
accuracy: 0.9960 - val_loss: 0.0352 - val_accuracy: 0.9881
Epoch 20/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0159 -
accuracy: 0.9963 - val_loss: 0.0351 - val_accuracy: 0.9877
Epoch 21/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0148 -
accuracy: 0.9966 - val_loss: 0.0356 - val_accuracy: 0.9873
Epoch 22/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0140 -
accuracy: 0.9965 - val_loss: 0.0336 - val_accuracy: 0.9879
Epoch 23/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0131 -
accuracy: 0.9970 - val_loss: 0.0356 - val_accuracy: 0.9866
Epoch 24/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0125 -
accuracy: 0.9972 - val_loss: 0.0329 - val_accuracy: 0.9873
Epoch 25/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0119 -
accuracy: 0.9973 - val_loss: 0.0360 - val_accuracy: 0.9866
Epoch 26/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0113 -
accuracy: 0.9974 - val_loss: 0.0329 - val_accuracy: 0.9874
Epoch 27/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0108 -
accuracy: 0.9976 - val_loss: 0.0339 - val_accuracy: 0.9870
Epoch 28/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0102 -
accuracy: 0.9976 - val_loss: 0.0339 - val_accuracy: 0.9864
Epoch 29/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0096 -
accuracy: 0.9977 - val_loss: 0.0341 - val_accuracy: 0.9860
Epoch 30/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0093 -
accuracy: 0.9978 - val_loss: 0.0343 - val_accuracy: 0.9866
Epoch 31/40
```

```
518/518 [==============================] - 1s 2ms/step - loss: 0.0088 -
accuracy: 0.9980 - val_loss: 0.0339 - val_accuracy: 0.9858
Epoch 32/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0086 -
accuracy: 0.9980 - val_loss: 0.0342 - val_accuracy: 0.9860
Epoch 33/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0081 -
accuracy: 0.9981 - val_loss: 0.0355 - val_accuracy: 0.9857
Epoch 34/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0077 -
accuracy: 0.9983 - val_loss: 0.0337 - val_accuracy: 0.9865
Epoch 35/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0075 -
accuracy: 0.9984 - val_loss: 0.0362 - val_accuracy: 0.9856
Epoch 36/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0072 -
accuracy: 0.9984 - val_loss: 0.0356 - val_accuracy: 0.9860
Epoch 37/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0068 -
accuracy: 0.9986 - val_loss: 0.0326 - val_accuracy: 0.9871
Epoch 38/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0067 -
accuracy: 0.9984 - val_loss: 0.0338 - val_accuracy: 0.9866
Epoch 39/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0065 -
accuracy: 0.9985 - val_loss: 0.0370 - val_accuracy: 0.9859
Epoch 40/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0061 -
accuracy: 0.9988 - val_loss: 0.0349 - val_accuracy: 0.9860
32 5
Epoch 1/40
518/518 [==============================] - 2s 3ms/step - loss: 0.6531 -
accuracy: 0.8117 - val_loss: 0.3966 - val_accuracy: 0.8899
Epoch 2/40
518/518 [==============================] - 1s 3ms/step - loss: 0.2244 -
accuracy: 0.9509 - val_loss: 0.2207 - val_accuracy: 0.9372
Epoch 3/40
518/518 [==============================] - 1s 2ms/step - loss: 0.1318 -
accuracy: 0.9706 - val_loss: 0.1410 - val_accuracy: 0.9612
Epoch 4/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0896 -
accuracy: 0.9803 - val_loss: 0.1045 - val_accuracy: 0.9720
Epoch 5/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0665 -
accuracy: 0.9854 - val_loss: 0.0762 - val_accuracy: 0.9811
Epoch 6/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0523 -
accuracy: 0.9881 - val_loss: 0.0644 - val_accuracy: 0.9842
```

```
Epoch 7/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0429 -
accuracy: 0.9901 - val_loss: 0.0576 - val_accuracy: 0.9844
Epoch 8/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0359 -
accuracy: 0.9918 - val_loss: 0.0511 - val_accuracy: 0.9858
Epoch 9/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0304 -
accuracy: 0.9930 - val_loss: 0.0421 - val_accuracy: 0.9882
Epoch 10/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0264 -
accuracy: 0.9937 - val_loss: 0.0387 - val_accuracy: 0.9890
Epoch 11/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0233 -
accuracy: 0.9947 - val_loss: 0.0343 - val_accuracy: 0.9895
Epoch 12/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0205 -
accuracy: 0.9954 - val_loss: 0.0321 - val_accuracy: 0.9900
Epoch 13/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0184 -
accuracy: 0.9958 - val_loss: 0.0308 - val_accuracy: 0.9911
Epoch 14/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0164 -
accuracy: 0.9961 - val_loss: 0.0287 - val_accuracy: 0.9915
Epoch 15/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0148 -
accuracy: 0.9966 - val_loss: 0.0268 - val_accuracy: 0.9919
Epoch 16/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0133 -
accuracy: 0.9968 - val_loss: 0.0249 - val_accuracy: 0.9920
Epoch 17/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0122 -
accuracy: 0.9972 - val_loss: 0.0231 - val_accuracy: 0.9928
Epoch 18/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0112 -
accuracy: 0.9974 - val_loss: 0.0242 - val_accuracy: 0.9919
Epoch 19/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0104 -
accuracy: 0.9975 - val_loss: 0.0255 - val_accuracy: 0.9916
Epoch 20/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0096 -
accuracy: 0.9976 - val_loss: 0.0248 - val_accuracy: 0.9914
Epoch 21/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0089 -
accuracy: 0.9979 - val_loss: 0.0233 - val_accuracy: 0.9924
Epoch 22/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0084 -
accuracy: 0.9978 - val_loss: 0.0241 - val_accuracy: 0.9923
```

```
Epoch 23/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0077 -
accuracy: 0.9982 - val_loss: 0.0236 - val_accuracy: 0.9916
Epoch 24/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0072 -
accuracy: 0.9983 - val_loss: 0.0228 - val_accuracy: 0.9919
Epoch 25/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0067 -
accuracy: 0.9984 - val_loss: 0.0218 - val_accuracy: 0.9922
Epoch 26/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0064 -
accuracy: 0.9985 - val_loss: 0.0210 - val_accuracy: 0.9927
Epoch 27/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0062 -
accuracy: 0.9985 - val_loss: 0.0217 - val_accuracy: 0.9919
Epoch 28/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0056 -
accuracy: 0.9988 - val_loss: 0.0203 - val_accuracy: 0.9932
Epoch 29/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0054 -
accuracy: 0.9987 - val_loss: 0.0207 - val_accuracy: 0.9932
Epoch 30/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0051 -
accuracy: 0.9987 - val_loss: 0.0216 - val_accuracy: 0.9930
Epoch 31/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0048 -
accuracy: 0.9989 - val_loss: 0.0240 - val_accuracy: 0.9913
Epoch 32/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0047 -
accuracy: 0.9988 - val_loss: 0.0198 - val_accuracy: 0.9928
Epoch 33/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0045 -
accuracy: 0.9989 - val_loss: 0.0219 - val_accuracy: 0.9920
Epoch 34/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0041 -
accuracy: 0.9991 - val_loss: 0.0215 - val_accuracy: 0.9915
Epoch 35/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0040 -
accuracy: 0.9991 - val_loss: 0.0228 - val_accuracy: 0.9914
Epoch 36/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0037 -
accuracy: 0.9992 - val_loss: 0.0213 - val_accuracy: 0.9923
Epoch 37/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0037 -
accuracy: 0.9991 - val_loss: 0.0197 - val_accuracy: 0.9928
Epoch 38/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0035 -
accuracy: 0.9992 - val_loss: 0.0236 - val_accuracy: 0.9912
```

```
Epoch 39/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0035 -
accuracy: 0.9992 - val_loss: 0.0213 - val_accuracy: 0.9925
Epoch 40/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0033 -
accuracy: 0.9992 - val_loss: 0.0201 - val_accuracy: 0.9929
64 6
Epoch 1/40
518/518 [==============================] - 2s 3ms/step - loss: 0.5014 -
accuracy: 0.8622 - val_loss: 0.2810 - val_accuracy: 0.9259
Epoch 2/40
518/518 [==============================] - 1s 3ms/step - loss: 0.1557 -
accuracy: 0.9640 - val_loss: 0.1403 - val_accuracy: 0.9675
Epoch 3/40
518/518 [==============================] - 2s 3ms/step - loss: 0.0924 -
accuracy: 0.9783 - val_loss: 0.0951 - val_accuracy: 0.9760
Epoch 4/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0644 -
accuracy: 0.9851 - val_loss: 0.0680 - val_accuracy: 0.9854
Epoch 5/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0490 -
accuracy: 0.9885 - val_loss: 0.0646 - val_accuracy: 0.9822
Epoch 6/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0393 -
accuracy: 0.9905 - val_loss: 0.0521 - val_accuracy: 0.9866
Epoch 7/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0322 -
accuracy: 0.9924 - val_loss: 0.0536 - val_accuracy: 0.9835
Epoch 8/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0272 -
accuracy: 0.9932 - val_loss: 0.0387 - val_accuracy: 0.9885
Epoch 9/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0231 -
accuracy: 0.9942 - val_loss: 0.0344 - val_accuracy: 0.9903
Epoch 10/40
518/518 [==============================] - 2s 3ms/step - loss: 0.0199 -
accuracy: 0.9952 - val_loss: 0.0343 - val_accuracy: 0.9900
Epoch 11/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0175 -
accuracy: 0.9956 - val_loss: 0.0270 - val_accuracy: 0.9922
Epoch 12/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0155 -
accuracy: 0.9961 - val_loss: 0.0269 - val_accuracy: 0.9920
Epoch 13/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0141 -
accuracy: 0.9965 - val_loss: 0.0306 - val_accuracy: 0.9893
Epoch 14/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0126 -
```

```
accuracy: 0.9970 - val_loss: 0.0257 - val_accuracy: 0.9914
Epoch 15/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0114 -
accuracy: 0.9973 - val_loss: 0.0245 - val_accuracy: 0.9918
Epoch 16/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0102 -
accuracy: 0.9976 - val_loss: 0.0295 - val_accuracy: 0.9898
Epoch 17/40
518/518 [==============================] - 2s 3ms/step - loss: 0.0093 -
accuracy: 0.9977 - val_loss: 0.0273 - val_accuracy: 0.9907
Epoch 18/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0089 -
accuracy: 0.9979 - val_loss: 0.0259 - val_accuracy: 0.9909
Epoch 19/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0080 -
accuracy: 0.9982 - val_loss: 0.0266 - val_accuracy: 0.9907
Epoch 20/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0074 -
accuracy: 0.9982 - val_loss: 0.0247 - val_accuracy: 0.9911
Epoch 21/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0071 -
accuracy: 0.9983 - val_loss: 0.0256 - val_accuracy: 0.9918
Epoch 22/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0064 -
accuracy: 0.9984 - val_loss: 0.0251 - val_accuracy: 0.9918
Epoch 23/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0064 -
accuracy: 0.9984 - val_loss: 0.0295 - val_accuracy: 0.9913
Epoch 24/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0056 -
accuracy: 0.9987 - val_loss: 0.0280 - val_accuracy: 0.9911
Epoch 25/40
518/518 [==============================] - 1s 2ms/step - loss: 0.0052 -
accuracy: 0.9988 - val_loss: 0.0268 - val_accuracy: 0.9915
Epoch 26/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0052 -
accuracy: 0.9987 - val_loss: 0.0299 - val_accuracy: 0.9911
Epoch 27/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0048 -
accuracy: 0.9988 - val_loss: 0.0346 - val_accuracy: 0.9888
Epoch 28/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0044 -
accuracy: 0.9990 - val_loss: 0.0285 - val_accuracy: 0.9920
Epoch 29/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0043 -
accuracy: 0.9989 - val_loss: 0.0307 - val_accuracy: 0.9918
Epoch 30/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0040 -
```

```
accuracy: 0.9989 - val_loss: 0.0355 - val_accuracy: 0.9897
Epoch 31/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0039 -
accuracy: 0.9989 - val_loss: 0.0337 - val_accuracy: 0.9908
Epoch 32/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0035 -
accuracy: 0.9992 - val_loss: 0.0408 - val_accuracy: 0.9881
Epoch 33/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0035 -
accuracy: 0.9992 - val_loss: 0.0400 - val_accuracy: 0.9892
Epoch 34/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0034 -
accuracy: 0.9993 - val_loss: 0.0363 - val_accuracy: 0.9913
Epoch 35/40
518/518 [==============================] - 2s 3ms/step - loss: 0.0033 -
accuracy: 0.9992 - val_loss: 0.0386 - val_accuracy: 0.9910
Epoch 36/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0029 -
accuracy: 0.9994 - val_loss: 0.0378 - val_accuracy: 0.9901
Epoch 37/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0027 -
accuracy: 0.9994 - val_loss: 0.0428 - val_accuracy: 0.9902
Epoch 38/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0028 -
accuracy: 0.9993 - val_loss: 0.0414 - val_accuracy: 0.9908
Epoch 39/40
518/518 [==============================] - 1s 3ms/step - loss: 0.0026 -
accuracy: 0.9994 - val_loss: 0.0403 - val_accuracy: 0.9913
Epoch 40/40
518/518 [==============================] - 2s 3ms/step - loss: 0.0025 -
accuracy: 0.9995 - val_loss: 0.0399 - val_accuracy: 0.9915
128 7
Epoch 1/40
518/518 [==============================] - 2s 4ms/step - loss: 0.4024 -
accuracy: 0.8906 - val_loss: 0.2185 - val_accuracy: 0.9342
Epoch 2/40
518/518 [==============================] - 2s 4ms/step - loss: 0.1184 -
accuracy: 0.9718 - val_loss: 0.1140 - val_accuracy: 0.9714
Epoch 3/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0699 -
accuracy: 0.9837 - val_loss: 0.0708 - val_accuracy: 0.9839
Epoch 4/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0496 -
accuracy: 0.9881 - val_loss: 0.0615 - val_accuracy: 0.9840
Epoch 5/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0379 -
accuracy: 0.9905 - val_loss: 0.0465 - val_accuracy: 0.9871
Epoch 6/40
```

```
518/518 [==============================] - 2s 4ms/step - loss: 0.0306 -
accuracy: 0.9920 - val_loss: 0.0399 - val_accuracy: 0.9895
Epoch 7/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0251 -
accuracy: 0.9936 - val_loss: 0.0347 - val_accuracy: 0.9911
Epoch 8/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0212 -
accuracy: 0.9947 - val_loss: 0.0301 - val_accuracy: 0.9912
Epoch 9/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0183 -
accuracy: 0.9953 - val_loss: 0.0296 - val_accuracy: 0.9919
Epoch 10/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0158 -
accuracy: 0.9960 - val_loss: 0.0285 - val_accuracy: 0.9914
Epoch 11/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0137 -
accuracy: 0.9966 - val_loss: 0.0256 - val_accuracy: 0.9921
Epoch 12/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0123 -
accuracy: 0.9968 - val_loss: 0.0235 - val_accuracy: 0.9918
Epoch 13/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0113 -
accuracy: 0.9970 - val_loss: 0.0251 - val_accuracy: 0.9921
Epoch 14/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0100 -
accuracy: 0.9975 - val_loss: 0.0216 - val_accuracy: 0.9926
Epoch 15/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0090 -
accuracy: 0.9978 - val_loss: 0.0219 - val_accuracy: 0.9921
Epoch 16/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0082 -
accuracy: 0.9979 - val_loss: 0.0289 - val_accuracy: 0.9888
Epoch 17/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0076 -
accuracy: 0.9982 - val_loss: 0.0281 - val_accuracy: 0.9900
Epoch 18/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0068 -
accuracy: 0.9982 - val_loss: 0.0261 - val_accuracy: 0.9901
Epoch 19/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0064 -
accuracy: 0.9984 - val_loss: 0.0218 - val_accuracy: 0.9918
Epoch 20/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0060 -
accuracy: 0.9985 - val_loss: 0.0246 - val_accuracy: 0.9909
Epoch 21/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0057 -
accuracy: 0.9985 - val_loss: 0.0202 - val_accuracy: 0.9925
Epoch 22/40
```

```
518/518 [==============================] - 2s 4ms/step - loss: 0.0056 -
accuracy: 0.9984 - val_loss: 0.0279 - val_accuracy: 0.9909
Epoch 23/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0051 -
accuracy: 0.9986 - val_loss: 0.0236 - val_accuracy: 0.9915
Epoch 24/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0047 -
accuracy: 0.9987 - val_loss: 0.0231 - val_accuracy: 0.9917
Epoch 25/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0042 -
accuracy: 0.9989 - val_loss: 0.0285 - val_accuracy: 0.9909
Epoch 26/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0042 -
accuracy: 0.9990 - val_loss: 0.0309 - val_accuracy: 0.9900
Epoch 27/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0040 -
accuracy: 0.9989 - val_loss: 0.0225 - val_accuracy: 0.9930
Epoch 28/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0037 -
accuracy: 0.9991 - val_loss: 0.0273 - val_accuracy: 0.9915
Epoch 29/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0036 -
accuracy: 0.9991 - val_loss: 0.0236 - val_accuracy: 0.9928
Epoch 30/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0034 -
accuracy: 0.9992 - val_loss: 0.0255 - val_accuracy: 0.9925
Epoch 31/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0030 -
accuracy: 0.9993 - val_loss: 0.0261 - val_accuracy: 0.9920
Epoch 32/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0032 -
accuracy: 0.9992 - val_loss: 0.0283 - val_accuracy: 0.9917
Epoch 33/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0029 -
accuracy: 0.9993 - val_loss: 0.0288 - val_accuracy: 0.9921
Epoch 34/40
518/518 [==============================] - 2s 3ms/step - loss: 0.0025 -
accuracy: 0.9995 - val_loss: 0.0312 - val_accuracy: 0.9912
Epoch 35/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0028 -
accuracy: 0.9992 - val_loss: 0.0347 - val_accuracy: 0.9913
Epoch 36/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0025 -
accuracy: 0.9994 - val_loss: 0.0293 - val_accuracy: 0.9919
Epoch 37/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0025 -
accuracy: 0.9993 - val_loss: 0.0341 - val_accuracy: 0.9907
Epoch 38/40
```

```
518/518 [==============================] - 2s 4ms/step - loss: 0.0026 -
accuracy: 0.9992 - val_loss: 0.0321 - val_accuracy: 0.9919
Epoch 39/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0022 -
accuracy: 0.9995 - val_loss: 0.0348 - val_accuracy: 0.9913
Epoch 40/40
518/518 [==============================] - 2s 4ms/step - loss: 0.0022 -
accuracy: 0.9995 - val_loss: 0.0329 - val_accuracy: 0.9915
256 8
Epoch 1/40
518/518 [==============================] - 3s 5ms/step - loss: 0.3452 -
accuracy: 0.9035 - val_loss: 0.1839 - val_accuracy: 0.9440
Epoch 2/40
518/518 [==============================] - 3s 5ms/step - loss: 0.1008 -
accuracy: 0.9757 - val_loss: 0.1111 - val_accuracy: 0.9636
Epoch 3/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0602 -
accuracy: 0.9856 - val_loss: 0.0699 - val_accuracy: 0.9783
Epoch 4/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0426 -
accuracy: 0.9891 - val_loss: 0.0457 - val_accuracy: 0.9872
Epoch 5/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0336 -
accuracy: 0.9907 - val_loss: 0.0418 - val_accuracy: 0.9887
Epoch 6/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0258 -
accuracy: 0.9933 - val_loss: 0.0339 - val_accuracy: 0.9901
Epoch 7/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0217 -
accuracy: 0.9941 - val_loss: 0.0317 - val_accuracy: 0.9899
Epoch 8/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0182 -
accuracy: 0.9952 - val_loss: 0.0321 - val_accuracy: 0.9899
Epoch 9/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0158 -
accuracy: 0.9957 - val_loss: 0.0256 - val_accuracy: 0.9924
Epoch 10/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0135 -
accuracy: 0.9965 - val_loss: 0.0241 - val_accuracy: 0.9924
Epoch 11/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0125 -
accuracy: 0.9966 - val_loss: 0.0216 - val_accuracy: 0.9936
Epoch 12/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0112 -
accuracy: 0.9968 - val_loss: 0.0240 - val_accuracy: 0.9917
Epoch 13/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0098 -
accuracy: 0.9975 - val_loss: 0.0271 - val_accuracy: 0.9898
```

```
Epoch 14/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0087 -
accuracy: 0.9976 - val_loss: 0.0242 - val_accuracy: 0.9917
Epoch 15/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0085 -
accuracy: 0.9976 - val_loss: 0.0252 - val_accuracy: 0.9905
Epoch 16/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0072 -
accuracy: 0.9980 - val_loss: 0.0244 - val_accuracy: 0.9905
Epoch 17/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0072 -
accuracy: 0.9979 - val_loss: 0.0385 - val_accuracy: 0.9855
Epoch 18/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0062 -
accuracy: 0.9983 - val_loss: 0.0262 - val_accuracy: 0.9905
Epoch 19/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0060 -
accuracy: 0.9984 - val_loss: 0.0237 - val_accuracy: 0.9908
Epoch 20/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0058 -
accuracy: 0.9983 - val_loss: 0.0258 - val_accuracy: 0.9911
Epoch 21/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0052 -
accuracy: 0.9984 - val_loss: 0.0245 - val_accuracy: 0.9907
Epoch 22/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0045 -
accuracy: 0.9989 - val_loss: 0.0254 - val_accuracy: 0.9909
Epoch 23/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0045 -
accuracy: 0.9988 - val_loss: 0.0313 - val_accuracy: 0.9891
Epoch 24/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0042 -
accuracy: 0.9988 - val_loss: 0.0264 - val_accuracy: 0.9909
Epoch 25/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0042 -
accuracy: 0.9987 - val_loss: 0.0229 - val_accuracy: 0.9928
Epoch 26/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0042 -
accuracy: 0.9989 - val_loss: 0.0349 - val_accuracy: 0.9891
Epoch 27/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0036 -
accuracy: 0.9990 - val_loss: 0.0294 - val_accuracy: 0.9911
Epoch 28/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0033 -
accuracy: 0.9991 - val_loss: 0.0277 - val_accuracy: 0.9916
Epoch 29/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0038 -
accuracy: 0.9988 - val_loss: 0.0344 - val_accuracy: 0.9899
```

```
Epoch 30/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0029 -
accuracy: 0.9992 - val_loss: 0.0338 - val_accuracy: 0.9909
Epoch 31/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0030 -
accuracy: 0.9991 - val_loss: 0.0321 - val_accuracy: 0.9920
Epoch 32/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0027 -
accuracy: 0.9992 - val_loss: 0.0444 - val_accuracy: 0.9880
Epoch 33/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0029 -
accuracy: 0.9990 - val_loss: 0.0312 - val_accuracy: 0.9909
Epoch 34/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0024 -
accuracy: 0.9994 - val_loss: 0.0351 - val_accuracy: 0.9911
Epoch 35/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0023 -
accuracy: 0.9994 - val_loss: 0.0390 - val_accuracy: 0.9901
Epoch 36/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0023 -
accuracy: 0.9994 - val_loss: 0.0427 - val_accuracy: 0.9899
Epoch 37/40
518/518 [==============================] - 3s 6ms/step - loss: 0.0028 -
accuracy: 0.9991 - val_loss: 0.0592 - val_accuracy: 0.9859
Epoch 38/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0020 -
accuracy: 0.9994 - val_loss: 0.0338 - val_accuracy: 0.9919
Epoch 39/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0024 -
accuracy: 0.9992 - val_loss: 0.0351 - val_accuracy: 0.9917
Epoch 40/40
518/518 [==============================] - 3s 5ms/step - loss: 0.0019 -
accuracy: 0.9994 - val_loss: 0.0414 - val_accuracy: 0.9905
512 9
Epoch 1/40
518/518 [==============================] - 4s 7ms/step - loss: 0.2772 -
accuracy: 0.9224 - val_loss: 0.1593 - val_accuracy: 0.9491
Epoch 2/40
518/518 [==============================] - 3s 7ms/step - loss: 0.0826 -
accuracy: 0.9788 - val_loss: 0.0742 - val_accuracy: 0.9846
Epoch 3/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0493 -
accuracy: 0.9871 - val_loss: 0.0575 - val_accuracy: 0.9822
Epoch 4/40
518/518 [==============================] - 3s 7ms/step - loss: 0.0362 -
accuracy: 0.9903 - val_loss: 0.0474 - val_accuracy: 0.9856
Epoch 5/40
518/518 [==============================] - 3s 7ms/step - loss: 0.0276 -
```

```
accuracy: 0.9925 - val_loss: 0.0376 - val_accuracy: 0.9890
Epoch 6/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0223 -
accuracy: 0.9935 - val_loss: 0.0370 - val_accuracy: 0.9882
Epoch 7/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0182 -
accuracy: 0.9950 - val_loss: 0.0291 - val_accuracy: 0.9906
Epoch 8/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0160 -
accuracy: 0.9953 - val_loss: 0.0364 - val_accuracy: 0.9868
Epoch 9/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0139 -
accuracy: 0.9961 - val_loss: 0.0266 - val_accuracy: 0.9907
Epoch 10/40
518/518 [==============================] - 3s 7ms/step - loss: 0.0125 -
accuracy: 0.9968 - val_loss: 0.0376 - val_accuracy: 0.9856
Epoch 11/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0110 -
accuracy: 0.9968 - val_loss: 0.0377 - val_accuracy: 0.9872
Epoch 12/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0104 -
accuracy: 0.9969 - val_loss: 0.0307 - val_accuracy: 0.9883
Epoch 13/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0095 -
accuracy: 0.9973 - val_loss: 0.0338 - val_accuracy: 0.9879
Epoch 14/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0092 -
accuracy: 0.9974 - val_loss: 0.0363 - val_accuracy: 0.9873
Epoch 15/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0075 -
accuracy: 0.9978 - val_loss: 0.0256 - val_accuracy: 0.9906
Epoch 16/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0070 -
accuracy: 0.9978 - val_loss: 0.0330 - val_accuracy: 0.9885
Epoch 17/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0063 -
accuracy: 0.9983 - val_loss: 0.0321 - val_accuracy: 0.9892
Epoch 18/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0063 -
accuracy: 0.9981 - val_loss: 0.0363 - val_accuracy: 0.9882
Epoch 19/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0068 -
accuracy: 0.9978 - val_loss: 0.0320 - val_accuracy: 0.9893
Epoch 20/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0053 -
accuracy: 0.9984 - val_loss: 0.0258 - val_accuracy: 0.9917
Epoch 21/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0050 -
```

```
accuracy: 0.9985 - val_loss: 0.0322 - val_accuracy: 0.9893
Epoch 22/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0058 -
accuracy: 0.9982 - val_loss: 0.0310 - val_accuracy: 0.9917
Epoch 23/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0042 -
accuracy: 0.9990 - val_loss: 0.0406 - val_accuracy: 0.9876
Epoch 24/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0049 -
accuracy: 0.9985 - val_loss: 0.0417 - val_accuracy: 0.9882
Epoch 25/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0041 -
accuracy: 0.9987 - val_loss: 0.0400 - val_accuracy: 0.9878
Epoch 26/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0041 -
accuracy: 0.9986 - val_loss: 0.0517 - val_accuracy: 0.9848
Epoch 27/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0037 -
accuracy: 0.9988 - val_loss: 0.0441 - val_accuracy: 0.9868
Epoch 28/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0036 -
accuracy: 0.9989 - val_loss: 0.0381 - val_accuracy: 0.9899
Epoch 29/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0038 -
accuracy: 0.9989 - val_loss: 0.0464 - val_accuracy: 0.9887
Epoch 30/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0054 -
accuracy: 0.9983 - val_loss: 0.0403 - val_accuracy: 0.9903
Epoch 31/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0026 -
accuracy: 0.9992 - val_loss: 0.0377 - val_accuracy: 0.9913
Epoch 32/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0035 -
accuracy: 0.9989 - val_loss: 0.0310 - val_accuracy: 0.9907
Epoch 33/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0035 -
accuracy: 0.9988 - val_loss: 0.0349 - val_accuracy: 0.9910
Epoch 34/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0026 -
accuracy: 0.9991 - val_loss: 0.0367 - val_accuracy: 0.9892
Epoch 35/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0036 -
accuracy: 0.9988 - val_loss: 0.0376 - val_accuracy: 0.9902
Epoch 36/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0025 -
accuracy: 0.9991 - val_loss: 0.0407 - val_accuracy: 0.9903
Epoch 37/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0024 -
```

```
accuracy: 0.9993 - val_loss: 0.0490 - val_accuracy: 0.9877
Epoch 38/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0036 -
accuracy: 0.9988 - val_loss: 0.0393 - val_accuracy: 0.9907
Epoch 39/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0021 -
accuracy: 0.9993 - val_loss: 0.0425 - val_accuracy: 0.9894
Epoch 40/40
518/518 [==============================] - 4s 7ms/step - loss: 0.0034 -
accuracy: 0.9989 - val_loss: 0.0495 - val_accuracy: 0.9879
1024 10
```

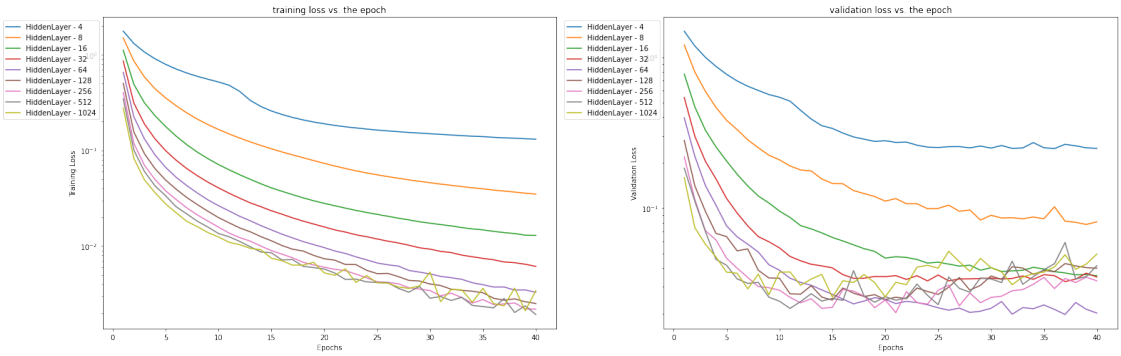
Plot the training loss vs. the epoch number for all of the hidden layer sizes on one graph (use `semilogy` again).

Then, plot the validation loss vs. the epoch number for all of the hidden layer sizes on a second graph (use `semilogy` again).

Make sure to label each axis, and each series.

Comment on the results.

[206]:
```
# TODO 10

plt.figure(figsize=(22,7))
idx = 0

plt.subplot(1,2,1)
plt.title('training loss vs. the epoch')
for i in diff_hid_layer:
  plt.semilogy(np.arange(1,41), i.history['loss'], label = str_name[idx])
  idx += 1
plt.xlabel('Epochs')
plt.ylabel('Training Loss')

plt.legend(bbox_to_anchor =(0, 1), ncol = 1)

idx = 0
plt.subplot(1,2,2)
plt.title('validation loss vs. the epoch')
for i in diff_hid_layer:
  plt.semilogy(np.arange(1,41), i.history['val_loss'], label = str_name[idx])
  idx += 1
plt.legend(bbox_to_anchor =(0, 1), ncol = 1)
plt.xlabel('Epochs')
plt.ylabel('Validation Loss')

plt.tight_layout()
```



Comments:

The overall trending is that, as the number of hidden layer goes up, the lines in both the training loss and validation loss tend to have a larger sharp drop as well.

Similarly in both plots, as the hidden layer gets bigger, the line gets bumpier, as the layer stays small, the line is more smooth.

The smallest layer (HiddenLayer - 4) is the smoothest is both plots.

However, the difference between these two plots are, when in the training loss, only the top 3 largest layers is obviously bumpy, all the previous lines are visually smooth. The lines in training appears that the most optimal line should be HiddenLayer - 64, because it converges fast enough without having overshoot situation, the other larger layers has an increased risk of overshooting. HiddenLayer - 1024 appear to have a very hard time to converge easily.

As in the validation chart, the performance is not doing very well. Most likely because the overfitting from the training or the noises is unpredictably high in the validation set. Except HiddenLayer - 4 and 8, all the other line now appear to rise again after a quick drop from 0 - 15 epochs, only 4 and 8 sppear to continue going downward.

Similarly, the lines get way more bumpier when the hidden layer numbers get larger. With the overshoot situation happen at its worst is the HiddenLayer - 1024, it doesn't seem that even more epochs is provided, it will ever converge.