In [0]:

```
# if you keras is not using tensorflow as backend set "KERAS_BACKEND=tensorflow" use this command
from keras.utils import np_utils
from keras.datasets import mnist
import seaborn as sns
from keras.initializers import RandomNormal
```

In [0]:

```
import warnings
plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = [10, 5]
warnings.filterwarnings("ignore", category=FutureWarning)
%config InlineBackend.figure_format = 'retina'
```

In [0]:

```
%matplotlib notebook
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
import time
# https://gist.github.com/greydanus/f6eee59eaf1d90fcb3b534a25362cea4
# https://stackoverflow.com/a/14434334
# this function is used to update the plots for each epoch and error
def plt_dynamic(x, vy, ty, ax, colors=['b']):
    ax.plot(x, vy, 'b', label="Validation Loss")
    ax.plot(x, ty, 'r', label="Train Loss")
    plt.legend()
    plt.grid()
    fig.canvas.draw()
```

In [0]:

```
# the data, shuffled and split between train and test sets
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

In [23]:

```
print("Number of training examples :", X_train.shape[0], "and each image is of shape (%d, %d)"%(X_
train.shape[1], X_train.shape[2]))
print("Number of training examples :", X_test.shape[0], "and each image is of shape (%d,
%d)"%(X_test.shape[1], X_test.shape[2]))
```

```
Number of training examples : 60000 and each image is of shape (28, 28)
Number of training examples : 10000 and each image is of shape (28, 28)
```

In [0]:

```
# if you observe the input shape its 3 dimensional vector
# for each image we have a (28*28) vector
# we will convert the (28*28) vector into single dimensional vector of 1 * 784

X_train = X_train.reshape(X_train.shape[0], X_train.shape[1]*X_train.shape[2])
X_test = X_test.reshape(X_test.shape[0], X_test.shape[1]*X_test.shape[2])
```

In [25]:

```
# after converting the input images from 3d to 2d vectors

print("Number of training examples :", X_train.shape[0], "and each image is of shape
(%d)"%(X_train.shape[1]))
print("Number of training examples :", X_test.shape[0], "and each image is of shape (%d)"%(X_test.
shape[1]))
```

```
Number of training examples : 60000 and each image is of shape (784)
```

```
Number of training examples : 10000 and each image is of shape (784)
```

In [0]:

```python
#if we observe the above matrix each cell is having a value between 0-255
# before we move to apply machine learning algorithms lets try to normalize the data
# X => (X - Xmin)/(Xmax-Xmin) = X/255

X_train = X_train/255
X_test = X_test/255
```

In [27]:

```python
# example data point after normlizing
print(X_train[0])
```

```
[0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.01176471 0.07058824 0.07058824 0.07058824
 0.49411765 0.53333333 0.68627451 0.10196078 0.65098039 1.
 0.96862745 0.49803922 0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.11764706 0.14117647 0.36862745 0.60392157
 0.66666667 0.99215686 0.99215686 0.99215686 0.99215686 0.99215686
 0.88235294 0.6745098  0.99215686 0.94901961 0.76470588 0.25098039
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.19215686
 0.93333333 0.99215686 0.99215686 0.99215686 0.99215686 0.99215686
 0.99215686 0.99215686 0.99215686 0.98431373 0.36470588 0.32156863
 0.32156863 0.21960784 0.15294118 0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.07058824 0.85882353 0.99215686
 0.99215686 0.99215686 0.99215686 0.99215686 0.77647059 0.71372549
 0.96862745 0.94509804 0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.31372549 0.61176471 0.41960784 0.99215686
 0.99215686 0.80392157 0.04313725 0.         0.16862745 0.60392157
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.05490196 0.00392157 0.60392157 0.99215686 0.35294118
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.54509804 0.99215686 0.74509804 0.00784314 0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.04313725
 0.74509804 0.99215686 0.2745098  0.         0.         0.
```

```
 0.74509804 0.99215686 0.2745098  0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.1372549  0.94509804
 0.88235294 0.62745098 0.42352941 0.00392157 0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.31764706 0.94117647 0.99215686
 0.99215686 0.46666667 0.09803922 0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.17647059 0.72941176 0.99215686 0.99215686
 0.58823529 0.10588235 0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.0627451  0.36470588 0.98823529 0.99215686 0.73333333
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.97647059 0.99215686 0.97647059 0.25098039 0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.18039216 0.50980392 0.71764706 0.99215686
 0.99215686 0.81176471 0.00784314 0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.15294118 0.58039216
 0.89803922 0.99215686 0.99215686 0.99215686 0.98039216 0.71372549
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.09411765 0.44705882 0.86666667 0.99215686 0.99215686 0.99215686
 0.99215686 0.78823529 0.30588235 0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.09019608 0.25882353 0.83529412 0.99215686
 0.99215686 0.99215686 0.99215686 0.77647059 0.31764706 0.00784314
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.07058824 0.67058824
 0.85882353 0.99215686 0.99215686 0.99215686 0.99215686 0.76470588
 0.31372549 0.03529412 0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.21568627 0.6745098  0.88627451 0.99215686 0.99215686 0.99215686
 0.99215686 0.95686275 0.52156863 0.04313725 0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.53333333 0.99215686
 0.99215686 0.99215686 0.83137255 0.52941176 0.51764706 0.0627451
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.        ]
```

In [28]:

```
# here we are having a class number for each image
```

```
# here we are having a class number for each image
print("Class label of first image :", y_train[0])

# lets convert this into a 10 dimensional vector
# ex: consider an image is 5 convert it into 5 => [0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
# this conversion needed for MLPs

Y_train = np_utils.to_categorical(y_train, 10)
Y_test = np_utils.to_categorical(y_test, 10)

print("After converting the output into a vector : ",Y_train[0])
```

```
Class label of first image : 5
After converting the output into a vector :  [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
```

In [0]:

```
from keras.models import Sequential
from keras.layers import Dense, Activation
from datetime import datetime
```

In [0]:

```
# some model parameters

output_dim = 10
input_dim = X_train.shape[1]

batch_size = 128
nb_epoch = 50
```

In [0]:

```
# start building a model
model = Sequential()

# The model needs to know what input shape it should expect.
# For this reason, the first layer in a Sequential model
# (and only the first, because following layers can do automatic shape inference)
# needs to receive information about its input shape.
# you can use input_shape and input_dim to pass the shape of input

# output_dim represent the number of nodes need in that layer
# here we have 10 nodes

model.add(Dense(output_dim, input_dim=input_dim, activation='softmax'))
```

In [40]:

```
# Before training a model, you need to configure the learning process, which is done via the compi
le method

# It receives three arguments:
# An optimizer. This could be the string identifier of an existing optimizer ,
https://keras.io/optimizers/
# A loss function. This is the objective that the model will try to minimize.,
https://keras.io/losses/
# A list of metrics. For any classification problem you will want to set this to metrics=['accurac
y'].  https://keras.io/metrics/


start = datetime.now()



# Note: when using the categorical_crossentropy loss, your targets should be in categorical format

# (e.g. if you have 10 classes, the target for each sample should be a 10-dimensional vector that
is all-zeros except
# for a 1 at the index corresponding to the class of the sample).

# that is why we converted out labels into vectors
```

```python
model.compile(optimizer='sgd', loss='categorical_crossentropy', metrics=['accuracy'])

# Keras models are trained on Numpy arrays of input data and labels.
# For training a model, you will typically use the  fit function

# fit(self, x=None, y=None, batch_size=None, epochs=1, verbose=1, callbacks=None,
validation_split=0.0,
# validation_data=None, shuffle=True, class_weight=None, sample_weight=None, initial_epoch=0, step
s_per_epoch=None,
# validation_steps=None)

# fit() function Trains the model for a fixed number of epochs (iterations on a dataset).

# it returns A History object. Its History.history attribute is a record of training loss values a
nd
# metrics values at successive epochs, as well as validation loss values and validation metrics va
lues (if applicable).

# https://github.com/openai/baselines/issues/20

history = model.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, validation
_data=(X_test, Y_test))


print('Time taken to run this cell :', datetime.now() - start)
```

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/50
60000/60000 [==============================] - 2s 36us/step - loss: 1.2946 - acc: 0.6954 -
val_loss: 0.8164 - val_acc: 0.8371
Epoch 2/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.7196 - acc: 0.8420 -
val_loss: 0.6096 - val_acc: 0.8636
Epoch 3/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.5891 - acc: 0.8599 -
val_loss: 0.5266 - val_acc: 0.8759
Epoch 4/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.5266 - acc: 0.8692 -
val_loss: 0.4807 - val_acc: 0.8837
Epoch 5/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.4887 - acc: 0.8754 -
val_loss: 0.4506 - val_acc: 0.8879
Epoch 6/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.4625 - acc: 0.8801 -
val_loss: 0.4291 - val_acc: 0.8905
Epoch 7/50
60000/60000 [==============================] - 2s 36us/step - loss: 0.4433 - acc: 0.8837 -
val_loss: 0.4131 - val_acc: 0.8930
Epoch 8/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.4283 - acc: 0.8869 -
val_loss: 0.4000 - val_acc: 0.8954
Epoch 9/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.4162 - acc: 0.8892 -
val_loss: 0.3899 - val_acc: 0.8974
Epoch 10/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.4061 - acc: 0.8911 -
val_loss: 0.3810 - val_acc: 0.8995
Epoch 11/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3977 - acc: 0.8930 -
val_loss: 0.3737 - val_acc: 0.9008
Epoch 12/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3904 - acc: 0.8945 -
val_loss: 0.3673 - val_acc: 0.9024
Epoch 13/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3840 - acc: 0.8958 -
val_loss: 0.3620 - val_acc: 0.9032
Epoch 14/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3784 - acc: 0.8974 -
val_loss: 0.3567 - val_acc: 0.9046
Epoch 15/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3733 - acc: 0.8982 -
val_loss: 0.3525 - val_acc: 0.9048
Epoch 16/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3688 - acc: 0.8994 -
val_loss: 0.3486 - val_acc: 0.9058
Epoch 17/50
```

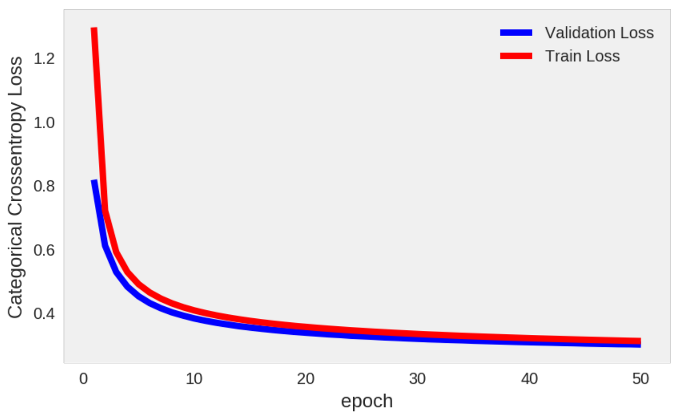
```
60000/60000 [==============================] - 2s 33us/step - loss: 0.3647 - acc: 0.9005 -
val_loss: 0.3450 - val_acc: 0.9061
Epoch 18/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3610 - acc: 0.9015 -
val_loss: 0.3417 - val_acc: 0.9063
Epoch 19/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3575 - acc: 0.9019 -
val_loss: 0.3388 - val_acc: 0.9077
Epoch 20/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3545 - acc: 0.9025 -
val_loss: 0.3361 - val_acc: 0.9081
Epoch 21/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3515 - acc: 0.9033 -
val_loss: 0.3337 - val_acc: 0.9093
Epoch 22/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3488 - acc: 0.9039 -
val_loss: 0.3312 - val_acc: 0.9097
Epoch 23/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3462 - acc: 0.9048 -
val_loss: 0.3292 - val_acc: 0.9090
Epoch 24/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3439 - acc: 0.9050 -
val_loss: 0.3268 - val_acc: 0.9101
Epoch 25/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3417 - acc: 0.9058 -
val_loss: 0.3251 - val_acc: 0.9110
Epoch 26/50
60000/60000 [==============================] - 2s 32us/step - loss: 0.3395 - acc: 0.9064 -
val_loss: 0.3235 - val_acc: 0.9106
Epoch 27/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3376 - acc: 0.9071 -
val_loss: 0.3215 - val_acc: 0.9113
Epoch 28/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3357 - acc: 0.9075 -
val_loss: 0.3202 - val_acc: 0.9117
Epoch 29/50
60000/60000 [==============================] - 2s 37us/step - loss: 0.3340 - acc: 0.9080 -
val_loss: 0.3185 - val_acc: 0.9118
Epoch 30/50
60000/60000 [==============================] - 2s 37us/step - loss: 0.3323 - acc: 0.9084 -
val_loss: 0.3170 - val_acc: 0.9125
Epoch 31/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3307 - acc: 0.9087 -
val_loss: 0.3156 - val_acc: 0.9129
Epoch 32/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3292 - acc: 0.9090 -
val_loss: 0.3146 - val_acc: 0.9125
Epoch 33/50
60000/60000 [==============================] - 2s 32us/step - loss: 0.3277 - acc: 0.9093 -
val_loss: 0.3134 - val_acc: 0.9141
Epoch 34/50
60000/60000 [==============================] - 2s 32us/step - loss: 0.3263 - acc: 0.9099 -
val_loss: 0.3124 - val_acc: 0.9131
Epoch 35/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3250 - acc: 0.9104 -
val_loss: 0.3110 - val_acc: 0.9145
Epoch 36/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3237 - acc: 0.9101 -
val_loss: 0.3101 - val_acc: 0.9148
Epoch 37/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3225 - acc: 0.9105 -
val_loss: 0.3091 - val_acc: 0.9145
Epoch 38/50
60000/60000 [==============================] - 2s 32us/step - loss: 0.3213 - acc: 0.9110 -
val_loss: 0.3082 - val_acc: 0.9151
Epoch 39/50
60000/60000 [==============================] - 2s 33us/step - loss: 0.3202 - acc: 0.9112 -
val_loss: 0.3072 - val_acc: 0.9149
Epoch 40/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3191 - acc: 0.9114 -
val_loss: 0.3064 - val_acc: 0.9161
Epoch 41/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3180 - acc: 0.9118 -
val_loss: 0.3057 - val_acc: 0.9161
Epoch 42/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3171 - acc: 0.9118 -
val_loss: 0.3050 - val_acc: 0.9159
```

```
Epoch 43/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3161 - acc: 0.9123 -
val_loss: 0.3040 - val_acc: 0.9157
Epoch 44/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3151 - acc: 0.9127 -
val_loss: 0.3033 - val_acc: 0.9156
Epoch 45/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3142 - acc: 0.9131 -
val_loss: 0.3024 - val_acc: 0.9155
Epoch 46/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3133 - acc: 0.9132 -
val_loss: 0.3018 - val_acc: 0.9166
Epoch 47/50
60000/60000 [==============================] - 2s 34us/step - loss: 0.3125 - acc: 0.9134 -
val_loss: 0.3011 - val_acc: 0.9164
Epoch 48/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3117 - acc: 0.9137 -
val_loss: 0.3003 - val_acc: 0.9163
Epoch 49/50
60000/60000 [==============================] - 2s 36us/step - loss: 0.3108 - acc: 0.9141 -
val_loss: 0.3000 - val_acc: 0.9166
Epoch 50/50
60000/60000 [==============================] - 2s 35us/step - loss: 0.3101 - acc: 0.9142 -
val_loss: 0.2991 - val_acc: 0.9164
Time taken to run this cell : 0:01:41.836958
```

In [41]:

```python
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
Test score: 0.2991245568871498
Test accuracy: 0.9164
```



In [0]:

```
# some model parameters

output_dim = 10
input_dim = X_train.shape[1]

batch_size = 128
nb_epoch = 10
```

MLP + sigmoid + Softmax

In [44]:

```
# Multilayer perceptron
start = datetime.now()


# Multilayer perceptron

model_sigmoid = Sequential()
model_sigmoid.add(Dense(700, activation='sigmoid', input_shape=(input_dim,)))
model_sigmoid.add(Dense(600, activation='sigmoid'))
model_sigmoid.add(Dense(500, activation='sigmoid'))
model_sigmoid.add(Dense(400, activation='sigmoid'))
model_sigmoid.add(Dense(300, activation='sigmoid'))
model_sigmoid.add(Dense(200, activation='sigmoid'))
model_sigmoid.add(Dense(100, activation='sigmoid'))
model_sigmoid.add(Dense(output_dim, activation='softmax'))

model_sigmoid.summary()

model_sigmoid.compile(optimizer='sgd', loss='categorical_crossentropy', metrics=['accuracy'])

history = model_sigmoid.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))



print('Time taken to run this cell :', datetime.now() - start)
```

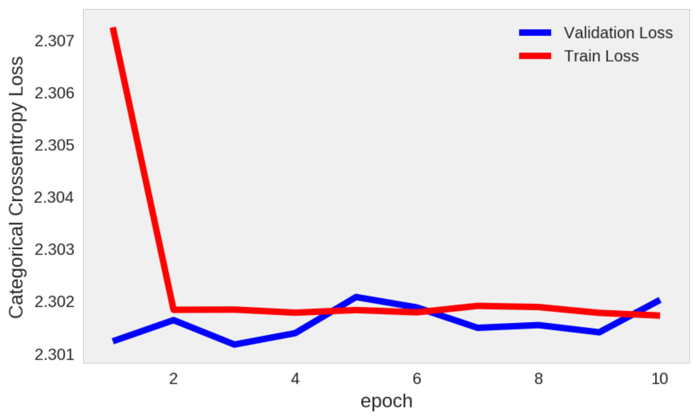
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_5 (Dense)              (None, 700)               549500    
_________________________________________________________________
dense_6 (Dense)              (None, 600)               420600    
_________________________________________________________________
dense_7 (Dense)              (None, 500)               300500    
_________________________________________________________________
dense_8 (Dense)              (None, 400)               200400    
_________________________________________________________________
dense_9 (Dense)              (None, 300)               120300    
_________________________________________________________________
dense_10 (Dense)             (None, 200)               60200     
_________________________________________________________________
dense_11 (Dense)             (None, 100)               20100     
_________________________________________________________________
dense_12 (Dense)             (None, 10)                1010      
=================================================================
Total params: 1,672,610
Trainable params: 1,672,610
Non-trainable params: 0
_________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 5s 83us/step - loss: 2.3072 - acc: 0.1096 -
val_loss: 2.3012 - val_acc: 0.1135
Epoch 2/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3018 - acc: 0.1116 -
val_loss: 2.3016 - val_acc: 0.1135
Epoch 3/10
60000/60000 [==============================] - 4s 72us/step - loss: 2.3018 - acc: 0.1112 -
val_loss: 2.3012 - val_acc: 0.1135
```

```
Epoch 4/10
60000/60000 [==============================] - 4s 71us/step - loss: 2.3018 - acc: 0.1118 -
val_loss: 2.3014 - val_acc: 0.1135
Epoch 5/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3018 - acc: 0.1111 -
val_loss: 2.3021 - val_acc: 0.1135
Epoch 6/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3018 - acc: 0.1112 -
val_loss: 2.3019 - val_acc: 0.1135
Epoch 7/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3019 - acc: 0.1115 -
val_loss: 2.3015 - val_acc: 0.1135
Epoch 8/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3019 - acc: 0.1114 -
val_loss: 2.3015 - val_acc: 0.1135
Epoch 9/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3018 - acc: 0.1120 -
val_loss: 2.3014 - val_acc: 0.1135
Epoch 10/10
60000/60000 [==============================] - 4s 70us/step - loss: 2.3017 - acc: 0.1115 -
val_loss: 2.3020 - val_acc: 0.1135
Time taken to run this cell : 0:00:43.554584
```

In [45]:

```
score = model_sigmoid.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
Test score: 2.3020178695678712
Test accuracy: 0.1135
```



In [47]:

```
w_after = model_sigmoid.get_weights()
```

```python
h1_w = w_after[0].flatten().reshape(-1,1)
h2_w = w_after[2].flatten().reshape(-1,1)
out_w = w_after[4].flatten().reshape(-1,1)
print('weights :', w_after)
print('flattened:', out_w)
```

```
weights : [array([[-0.03175347, -0.02756435, -0.04442617, ..., -0.01751344,
         -0.02968387, -0.001544  ],
        [-0.04045071,  0.02116323, -0.02986418, ...,  0.00762647,
          0.00490525,  0.03204468],
        [ 0.03250112,  0.03804421,  0.04024108, ...,  0.03825025,
         -0.04479065, -0.00907486],
        ...,
        [-0.04954398,  0.01768996, -0.00568111, ...,  0.02882903,
          0.02857331, -0.03528655],
        [ 0.03453752, -0.02654835, -0.03478636, ..., -0.00620486,
         -0.01380543,  0.05020246],
        [-0.00890188,  0.04409403, -0.05805328, ...,  0.04521473,
         -0.00757775, -0.02316989]], dtype=float32), array([-2.17641087e-07,  8.19320974e-07, -2.494
95827e-07, -4.85557393e-07,
       -2.98457138e-07, -9.21600588e-07,  3.53969540e-07, -2.64251963e-08,
       -3.19018909e-07,  7.11022096e-07,  7.97982764e-08,  4.81347115e-07,
       -4.88186970e-07,  7.64569052e-07, -7.97204734e-07,  6.84598263e-07,
       -2.63221438e-07, -6.74300679e-07, -3.24098067e-07, -2.26711720e-07,
       -4.00414194e-07, -2.46963232e-08, -8.22987204e-07,  5.83579777e-07,
        4.35540784e-07,  5.12530178e-08,  2.05356582e-06,  3.91766832e-08,
       -2.85046127e-07,  7.85806833e-07,  3.62405558e-07, -1.79440264e-07,
       -6.55798942e-07,  4.08137254e-07, -5.86730891e-07, -7.02665659e-08,
       -6.82166956e-07,  4.28555182e-07,  7.18086639e-08,  3.08899018e-07,
        8.39706956e-07, -5.61348315e-07,  2.45273100e-07, -3.51070639e-07,
        3.07521333e-07, -7.23736093e-08, -7.89851981e-07, -2.49364302e-06,
       -5.47498871e-07, -3.21609235e-07,  4.99339706e-07,  5.51602852e-07,
       -3.47191289e-08,  2.59115524e-07, -6.33231956e-09,  3.54964591e-08,
       -2.14909619e-07,  6.99158250e-07,  2.15920096e-07,  1.20448988e-06,
       -4.83773874e-08,  5.63969536e-07,  7.07607512e-07, -6.52622987e-08,
       -1.12382656e-06,  1.28390795e-07,  1.64685758e-07,  1.47062588e-06,
       -4.82681628e-08,  1.04802086e-06, -1.03230275e-06, -3.23340169e-06,
       -1.25924737e-06, -9.34158038e-07,  2.03102729e-07, -1.29038057e-07,
       -1.44745670e-07, -4.82981250e-07,  8.89319210e-07, -1.46098307e-06,
        1.82211716e-06, -3.55614560e-07,  2.46122499e-06, -7.61315562e-08,
       -5.73793955e-07,  7.92709841e-07,  3.17876754e-07,  9.73674332e-07,
       -4.42820436e-07,  2.48142896e-07,  6.92387857e-07, -1.26912573e-06,
       -7.62292700e-08, -1.28567393e-07, -2.81374298e-07,  9.78080152e-07,
       -3.35985050e-07, -3.50810922e-07,  1.72258510e-07,  1.41635439e-06,
        1.00289430e-08,  3.36745103e-07,  3.14572731e-07, -1.44463263e-06,
       -2.12035616e-08,  1.18555818e-07,  3.01198355e-07, -3.45779358e-06,
       -7.57400755e-07,  3.93166147e-07, -4.96229518e-07, -1.04642432e-07,
       -2.05545902e-07,  6.03818137e-07, -7.26414328e-07,  7.56280485e-07,
       -4.23516838e-07,  1.56186888e-06, -2.54919229e-07,  9.47177625e-08,
        2.47979301e-07, -1.04211585e-06,  1.35857647e-07,  9.03069008e-07,
       -1.26706212e-07,  8.91763374e-09, -7.93416973e-07, -9.95346340e-07,
       -1.03039542e-06, -7.78971696e-07, -2.92058445e-07,  2.77420258e-06,
       -1.88192104e-07, -4.75857178e-07,  3.11915542e-08, -2.39393648e-06,
        2.46878585e-07,  7.40363078e-07, -1.67345308e-06, -1.88032104e-07,
       -4.26068510e-07, -1.47886983e-08, -2.14975486e-07, -3.20066079e-07,
       -2.29718353e-07,  7.61183117e-07,  7.13505585e-07, -1.39595443e-07,
        1.41290047e-06,  5.90401044e-07,  1.23639643e-06,  1.89339445e-07,
       -1.02916715e-06, -4.77595904e-07, -7.41852887e-07, -4.86990189e-07,
        7.75617650e-07, -1.13873978e-06, -1.03235595e-06,  4.62201548e-08,
       -2.67446666e-07, -6.23734422e-07,  2.59036312e-07,  3.45481453e-07,
       -6.26108658e-08,  8.39466026e-08, -7.94830726e-07,  5.26222266e-07,
       -3.23467702e-07,  2.50885222e-07,  1.45055458e-06,  2.48520621e-07,
       -3.61615577e-07,  1.25922293e-06,  1.50850397e-07,  2.15457561e-07,
       -2.40814074e-07, -6.16559305e-07, -2.01805904e-07,  3.42056211e-07,
        2.08105931e-07,  1.03537104e-06,  2.41719050e-07,  5.72868601e-07,
        1.04466596e-06,  9.96709218e-07, -5.97432575e-07,  8.42021393e-07,
       -1.46368006e-06, -8.91609830e-07, -1.93996854e-07,  1.77133529e-07,
       -1.08176494e-07, -3.78810932e-06, -1.65055508e-06,  3.29595139e-07,
        9.33475803e-07, -6.12985104e-07, -6.15731210e-07, -2.46240631e-07,
        9.14968496e-07, -1.50848649e-07,  3.39884923e-06, -9.44925659e-07,
        1.89213551e-07,  4.81102290e-07,  1.21572953e-07,  4.68995125e-07,
       -2.31257459e-07, -1.08391653e-07,  1.54952303e-07, -6.77538537e-07,
        1.48902032e-07,  7.02908665e-07, -2.90118834e-07,  7.95225290e-08,
        9.59310356e-08,  1.45753845e-07, -4.37720331e-07, -1.43949717e-06,
        5.02828641e-07, -7.14800805e-08, -2.50518895e-07, -2.51316749e-07,
        2.43256949e-07, -2.66587313e-06, -7.06725416e-07, -8.88450558e-09,
```

```
 1.10033795e-06, -7.65316599e-07, -7.19849025e-09,  6.27819873e-07,
-5.09992162e-07, -3.92993257e-07, -2.34019737e-08,  4.24035505e-07,
-5.14521446e-07, -1.00970304e-07, -2.99396788e-07, -7.57251826e-08,
-2.85104278e-08,  1.44820618e-07,  2.23488883e-07,  6.16750924e-07,
-6.97196327e-08,  2.47672716e-07,  2.66698891e-07, -1.92918264e-07,
 3.19499350e-07, -5.11075200e-07, -9.15218266e-08, -8.15282704e-07,
-5.64207767e-07,  1.26527326e-07, -2.57314468e-07, -9.82474830e-07,
 1.67666613e-07, -5.20875574e-07, -2.03514310e-06,  5.24965458e-08,
 1.94170161e-06,  3.98985577e-07,  5.22663299e-07,  7.15280137e-08,
-6.43808278e-08, -1.52789440e-07,  1.15933879e-07, -1.53998371e-07,
-3.65402116e-07, -1.81914530e-07,  1.12471275e-06,  3.88040377e-07,
-4.90891239e-09,  4.04775733e-07,  2.08794131e-06,  2.71118779e-07,
 2.15738908e-07,  2.51751004e-07, -4.64157978e-07,  1.08089750e-07,
-6.29096007e-07,  5.19064997e-07,  5.10820115e-08,  1.00315674e-07,
 3.41330889e-07,  1.06829441e-06, -5.16121865e-08,  1.12288308e-06,
-6.59227979e-08, -4.49088958e-07, -3.37754130e-07,  3.87297462e-07,
-1.79359699e-06, -5.02372473e-07,  4.13320606e-07, -3.72896210e-07,
-8.00430143e-07,  1.08052473e-06, -1.30565383e-07, -1.42073276e-07,
 2.00068158e-07,  3.39812857e-07,  1.20843902e-06,  4.63162024e-07,
-6.13911709e-07, -1.60140723e-06, -1.15400702e-07, -2.31226815e-09,
-1.15280173e-07,  3.83344883e-07, -8.43698018e-08, -5.59832529e-07,
 6.21519689e-07,  1.65797376e-09,  8.75394278e-07, -2.38663134e-08,
 6.75903635e-08,  6.50219704e-07, -1.36984397e-06, -6.48724949e-07,
-6.36945430e-08,  2.19497537e-07,  3.53998786e-07, -5.00697830e-08,
 1.37136311e-07,  2.60672067e-07, -3.42229612e-07, -1.96133428e-07,
-5.58749775e-07,  6.05690957e-07,  3.82041435e-07, -1.61321350e-06,
-1.44092368e-07, -3.46972456e-07,  1.48215634e-07, -2.22352682e-07,
 1.69918252e-07,  1.09508580e-08,  4.53610625e-08, -2.69278189e-07,
 7.80340827e-07,  1.05560025e-06,  1.52496838e-07,  1.00699958e-06,
-3.39277264e-08,  6.38723634e-07, -4.74307797e-07,  4.32160334e-07,
 7.45129739e-07,  1.00184070e-06,  8.46050796e-07, -1.12524469e-06,
 5.86087992e-07, -7.04486411e-07,  8.97938008e-08, -2.81681622e-07,
 1.03752413e-07, -2.49130522e-07, -1.30876174e-07,  9.24696451e-07,
 4.01181381e-07, -4.46631390e-07,  4.42909652e-07,  2.46201523e-07,
 6.23071514e-07, -1.20783034e-06,  5.81520908e-07, -4.97832559e-07,
-4.68540520e-07,  4.38490360e-07, -4.15684184e-07, -7.15873341e-07,
 4.60191927e-07, -6.06834618e-08,  6.13364477e-07, -2.08577740e-06,
-1.10976384e-06,  1.04025480e-06, -1.85497981e-07,  1.08493191e-06,
 2.95126998e-08,  5.19321532e-07, -1.96268715e-07,  7.09126411e-08,
 5.65272025e-08,  8.01190822e-07, -1.20064229e-07,  1.08574693e-06,
 9.66303126e-09,  5.49246920e-07,  4.61686199e-07, -7.18753085e-07,
-6.72707870e-07, -2.17210868e-07, -3.37786275e-07, -4.63623140e-09,
-9.35568778e-07, -2.06350904e-07, -7.75446438e-07, -3.13365234e-07,
 4.92858192e-07, -4.86604279e-07,  2.80935183e-07, -2.69689082e-07,
 1.59703347e-07,  8.86527914e-07, -4.34648200e-07, -2.46378164e-07,
-2.41391490e-07,  1.70191498e-07, -2.49731073e-07, -2.56903974e-07,
-2.38303343e-07, -1.21299388e-06, -8.59098606e-08,  8.10423671e-07,
 1.24875078e-06,  1.38575717e-06, -7.31054627e-07,  3.54462415e-08,
-6.98298948e-08, -1.05969811e-07, -7.33299316e-07, -7.85268170e-08,
 1.33795311e-07, -7.06127594e-07, -6.15378212e-07, -1.48580028e-07,
-3.30446994e-07,  1.06599896e-06,  2.68136915e-07,  1.10618970e-07,
 2.04144897e-07,  2.46535222e-07,  2.75906416e-07, -5.14871317e-07,
 3.70261859e-07,  1.27154931e-06,  1.59996702e-07,  1.29999535e-07,
-1.64992642e-07, -3.26252547e-07, -9.74681370e-07, -6.99796772e-07,
 6.01504155e-07,  5.69734709e-07, -4.92636218e-07,  7.49508047e-08,
 1.20530554e-08,  1.13589022e-06,  3.52152142e-07,  4.81825424e-08,
 2.01856558e-08, -4.46108913e-07, -6.02664045e-07, -1.64869914e-06,
 5.31303961e-08, -1.18461969e-06, -6.93803315e-07,  5.97454118e-07,
 2.49236990e-07, -4.74883848e-07,  1.52577712e-07, -5.23438246e-07,
-5.93196830e-07, -5.22349694e-07,  4.52853499e-07,  9.95629762e-07,
-2.49039882e-08,  2.51776413e-07,  8.35408343e-07,  2.96035580e-07,
 2.46095823e-07,  1.80679265e-07,  2.66173448e-08, -1.43146352e-07,
 1.26606164e-06, -1.07618598e-06, -2.09893855e-07, -1.57441559e-06,
-4.79530456e-07, -1.59446699e-07,  1.94881551e-07,  3.05047507e-07,
-2.90317445e-07, -7.84981580e-07,  1.25012366e-06,  3.88331109e-08,
-4.07973772e-07,  1.27190802e-07, -6.52089156e-08,  5.93092651e-08,
-2.41063560e-07, -1.64828817e-10,  6.15617751e-07, -9.19496756e-07,
-6.11090698e-08,  2.86117398e-07,  9.24443441e-07, -3.19025702e-07,
 1.22316965e-06,  4.58818548e-08,  1.25884185e-06,  2.51959506e-07,
 3.10094083e-06, -6.70128273e-08,  2.34242592e-07,  7.09505571e-07,
-7.17916251e-08,  2.30791173e-07,  3.39955903e-07,  3.83149512e-07,
 4.41916256e-07, -1.23257777e-07, -3.96744838e-07,  4.09533385e-09,
 3.52948814e-06, -1.70390095e-07,  4.19148591e-07, -2.79515916e-07,
-5.14703800e-07,  2.33945244e-07,  6.98649387e-07, -5.40826591e-07,
-4.35144294e-08,  3.73304545e-07, -1.57791092e-07, -1.46188341e-07,
-1.22802908e-06,  9.54388497e-07, -1.30292250e-07, -1.10435124e-07,
-2.36167310e-07,  8.49689670e-08, -2.09203620e-07,  1.16429874e-06,
```

```
        3.66783695e-07, -1.03373952e-06, -1.14887484e-06, -2.38595447e-07,
        1.59059698e-06,  3.33960173e-07,  3.47535490e-07,  5.05387561e-07,
        6.06881144e-07,  2.65091700e-07,  9.22063350e-07, -9.39289066e-07,
        9.99322765e-07,  1.22728261e-06,  6.42421810e-07,  1.91025620e-07,
       -8.82101059e-10, -3.99408520e-07,  6.48664923e-07,  9.88950788e-08,
        9.46867971e-08, -8.51259870e-07,  2.99344219e-06, -1.10084000e-06,
       -6.63315916e-07, -6.12273368e-07, -1.10503059e-07, -6.09850019e-07,
        4.81474046e-07,  7.83496091e-07, -1.77561776e-06,  4.16074553e-09,
       -3.12757493e-07, -1.43013801e-06, -2.63058809e-07,  3.41485205e-08,
       -3.72871796e-07, -8.15693340e-07, -2.77201366e-07,  1.77902663e-07,
        1.08023823e-06,  1.82885131e-07,  6.94629762e-07,  4.31415572e-07,
       -1.39960014e-06, -6.06259675e-07, -3.91008911e-07, -9.14206382e-08,
       -4.39274743e-07, -9.30343560e-07, -6.21133609e-07,  7.77667424e-07,
       -1.00972407e-07, -2.91157249e-07,  4.40005323e-07, -2.44975212e-07,
        6.96931068e-07,  2.16035843e-07,  9.29197370e-07,  6.08789321e-07,
        6.38129336e-07, -2.32888766e-07,  9.02653241e-08, -4.41865780e-07,
       -4.22674105e-07,  1.31892406e-07,  3.16316658e-07,  2.40265710e-08,
        1.86612692e-06,  2.14074475e-07, -1.93187887e-07, -2.72487569e-07,
       -2.29138024e-07, -2.69164730e-07,  1.76819796e-07, -3.40653159e-08,
       -4.69268429e-07, -2.95920836e-06, -8.97999826e-08, -9.89817082e-08,
       -2.10956017e-07,  8.59277179e-07, -1.34198956e-07, -8.98011692e-08,
       -7.19307252e-07,  5.84365978e-07,  7.42328837e-07,  6.70375357e-07,
       -5.84304303e-07,  9.27681299e-07,  1.31942249e-06,  3.67161078e-07,
        9.03780233e-07,  4.00069553e-07,  2.83176178e-07, -4.36939274e-07,
       -4.29051880e-07, -4.96808241e-07,  1.61222545e-07, -8.62193644e-07,
        7.74845148e-07,  1.66989594e-07,  4.61718372e-07,  2.50117296e-07,
       -8.22013817e-07, -6.10627104e-09, -3.03702024e-08, -7.02149308e-08,
       -8.49980779e-07, -6.00281737e-07,  4.76783754e-07,  2.82814341e-08,
        2.75300920e-07,  5.91778644e-07,  3.43375916e-07, -5.80343794e-07,
        4.12613133e-07,  1.90447088e-06, -4.32730701e-07, -2.51627398e-07,
        9.85729571e-07, -7.75817682e-07,  5.75242382e-07,  2.54374175e-08,
       -3.43580018e-06,  2.09033601e-07, -4.80332972e-07, -3.66525910e-07,
        3.54582255e-07, -3.95503804e-07,  2.49003534e-07, -2.71226259e-08,
       -1.17510911e-06,  6.65467155e-07, -3.73634805e-07,  1.40661143e-06,
        3.96926964e-07, -6.22480911e-07,  1.02449974e-06, -1.02725312e-06,
        3.56358612e-07,  6.58738486e-07, -5.80485278e-07, -3.60302465e-07,
        1.55682741e-07, -3.79755107e-07,  1.30990756e-07, -1.42679637e-08,
        9.31429724e-07, -3.46381555e-08, -3.50133973e-07, -6.49975618e-07,
       -1.42721291e-07, -5.93312564e-07, -1.08194229e-06,  3.41208647e-07,
       -4.30032770e-07,  6.28806944e-08,  1.36053723e-06,  3.05011412e-07,
        8.65795926e-07,  1.55740895e-06,  2.23141342e-07, -2.04002674e-07],
      dtype=float32), array([[-0.04979942,  0.00436301,  0.00298063, ...,  0.00335499,
        -0.05312689,  0.0240337 ],
       [-0.03560917,  0.00714466, -0.02483935, ..., -0.01306534,
         0.04147837,  0.06001421],
       [ 0.00644081, -0.04169319,  0.05315369, ..., -0.02536394,
         0.05035724,  0.03970043],
       ...,
       [ 0.06079875,  0.06355181,  0.06730648, ..., -0.01199376,
         0.02592999, -0.05360856],
       [-0.0218608 ,  0.04614245,  0.05374676, ...,  0.02346343,
        -0.01394023,  0.01899254],
       [-0.02354216, -0.05235848, -0.00650346, ...,  0.01258566,
        -0.01145427,  0.05614954]], dtype=float32), array([ 2.4429908e-06, -1.9063329e-07, -9.21013
91e-07,  1.1411603e-06,
       -6.0490794e-07, -2.4217616e-08,  8.8316904e-07,  7.2075176e-07,
       -3.8419171e-06,  1.9401875e-06,  5.0593786e-07,  9.5360258e-07,
       -2.9409034e-07, -1.2472584e-06,  3.8485402e-07, -1.5005274e-07,
       -1.0943774e-06,  1.4173482e-06,  2.4939443e-06, -1.4599857e-06,
       -6.8630015e-08,  1.2852350e-06,  6.9727912e-07, -6.0998946e-07,
        5.6055310e-08,  1.6302158e-06,  6.3802281e-07, -8.5660133e-07,
       -3.6653756e-07,  9.8823614e-07, -6.0638712e-07, -1.0913607e-06,
        8.0407318e-07,  8.4699707e-07,  1.6573232e-06,  2.3678754e-07,
        4.5138959e-06, -4.9677141e-07, -1.4373107e-06,  4.8359929e-08,
       -3.8584594e-06,  2.5383551e-06,  1.7757571e-07,  1.0257889e-06,
       -7.8353889e-07,  4.9337717e-07, -9.8412428e-08, -1.3303501e-06,
       -1.1446380e-06,  1.9343456e-06,  5.4957292e-07, -3.8918083e-07,
        1.3007950e-06,  2.4488082e-07, -2.7356228e-08,  6.1502044e-07,
       -1.9521385e-06, -4.9673457e-07, -4.5831307e-08,  1.1468939e-06,
        1.5231403e-06,  2.3049120e-07, -9.7840973e-07, -1.8973833e-06,
       -2.2356348e-06, -1.3522806e-07,  9.4177466e-08, -6.2946140e-07,
        2.7921723e-07,  6.3384351e-07,  9.2754107e-07,  4.3017604e-07,
        4.6078137e-07,  1.1691316e-06, -3.1917509e-07,  1.5835889e-06,
        9.9187741e-07,  2.5631393e-06, -7.0655449e-07,  2.4961278e-06,
       -1.7458485e-06, -1.6815178e-06,  5.4632312e-07, -1.7655061e-06,
        2.8363314e-07,  4.3698316e-07,  1.3489622e-06, -3.3152025e-06,
        7.0380122e-07,  7.9526092e-07, -2.1714043e-06,  1.5021928e-07,
```

```
 1.6947562e-06,  1.0100671e-06, -1.8358870e-07, -1.7330512e-06,
 1.0977750e-06, -1.3266373e-07,  8.7620754e-07, -3.4970978e-07,
-1.1642940e-06,  1.2901167e-07, -5.4664298e-08,  5.1084140e-07,
 7.5575946e-07,  5.2532886e-07, -7.8933982e-07,  2.5631205e-06,
-4.4068616e-07,  8.1251488e-07, -3.8090109e-06,  1.6529474e-06,
 3.0617952e-07, -8.2452090e-07, -8.9738802e-07,  1.4124190e-06,
 5.1681968e-07,  1.1241777e-07, -1.0462924e-06,  2.8820987e-06,
-1.4450113e-06, -1.5385856e-07,  7.0830976e-07,  6.2745323e-07,
-9.5378016e-07,  5.4784755e-07, -2.4837098e-07,  1.5390043e-06,
-1.5025361e-06, -4.4607643e-07, -1.5244757e-06,  2.4739613e-06,
-1.3441373e-06, -4.6735491e-07, -3.5996645e-06,  9.8232590e-07,
-1.5334585e-06,  1.3652691e-06,  3.6545529e-07,  4.6035069e-07,
 2.6441083e-07, -1.7040904e-07, -4.5166030e-07, -1.4450885e-06,
 1.0662686e-06, -3.2056130e-06,  8.4318748e-07, -2.0787968e-06,
-2.2056563e-06, -8.9484570e-07, -7.4737994e-07, -3.7223407e-07,
 3.0604787e-07,  1.9169590e-07,  1.1882817e-06,  1.1220613e-06,
-4.0907554e-07, -1.9801748e-06, -3.6822571e-06, -1.2567060e-06,
 7.6450272e-07, -5.1557316e-07,  4.9644964e-06,  3.9079922e-07,
 1.1551899e-08,  8.8064627e-07,  1.1643632e-06,  1.1566804e-06,
-5.7929920e-08,  1.0393065e-06, -1.3535263e-06, -1.0760315e-06,
 1.1593145e-06,  3.0932122e-06, -8.3196915e-07, -4.6360762e-07,
 1.7430557e-07, -1.1532466e-06,  8.0266830e-07, -1.2953775e-06,
-7.8254863e-07,  4.7403492e-07,  2.7557564e-07,  2.2283379e-07,
-4.7568298e-07, -9.8157852e-07,  1.9923291e-06,  3.2397397e-06,
 5.3781044e-08,  6.5574511e-07,  1.6614330e-06, -3.0918002e-07,
-6.8050150e-07, -3.3031724e-06,  3.9488506e-07, -1.0025618e-06,
 8.1712011e-07,  1.0164695e-06, -4.2564420e-08, -2.2316468e-07,
 1.2757911e-06, -3.1782020e-07, -1.8816729e-07, -4.0361553e-07,
-2.1454329e-07, -1.7876997e-06, -5.5175133e-07,  2.5856542e-07,
-9.8206715e-07, -1.2185430e-07, -6.8607886e-07, -1.7860383e-07,
 3.7695744e-07,  3.7301600e-08,  5.7792192e-07,  9.6781218e-07,
 1.1554437e-06,  2.6261705e-06, -7.0663128e-07, -4.4370623e-07,
 2.8193544e-06, -2.0314556e-07, -3.4589630e-07,  5.9719679e-07,
-7.4235237e-07,  1.7031847e-06, -3.7560889e-07, -8.2421366e-08,
-8.4311239e-07, -2.0768643e-07, -8.6337968e-07, -2.0778889e-06,
-1.6743120e-06,  3.3635840e-06, -5.5807982e-07, -5.0312809e-07,
 2.0510649e-06, -9.0026396e-07, -8.9666241e-07,  2.6368292e-07,
 1.4346497e-06, -1.5694452e-06, -7.7323847e-07, -8.4032189e-07,
 2.4319593e-06,  4.3556179e-07, -1.3377098e-06, -2.3271364e-06,
-8.9456285e-07,  2.9462117e-07, -1.2772022e-06, -2.0249581e-06,
-2.0471375e-06, -7.6207709e-07,  4.6097847e-07,  1.0497573e-06,
 4.7277800e-07, -5.1249322e-07, -2.1688468e-06,  2.3426097e-07,
-6.7934701e-07, -7.9547436e-07,  1.1180462e-06, -3.7954663e-07,
-4.2162020e-07,  6.7562530e-07, -7.1760076e-07,  1.5390892e-08,
 4.6307651e-09,  2.0368485e-07, -1.4331860e-06, -1.3786686e-06,
-2.5284342e-06, -9.0592113e-07, -1.4014785e-06, -2.0131395e-06,
-7.6906980e-07,  7.4736397e-07,  6.4984016e-07, -8.6962089e-08,
-5.1601960e-06,  1.6503839e-07,  8.2170533e-07, -2.4674828e-06,
 9.1357570e-08,  1.1241603e-06,  5.2551525e-07, -7.5563764e-07,
-7.4068373e-07, -3.6773679e-07, -3.6673967e-07, -6.5058435e-07,
-2.3412606e-06,  2.4056843e-07,  7.0778327e-07,  1.3901791e-06,
 1.3178627e-06, -4.4779281e-06, -4.7489146e-07,  1.8033136e-06,
-1.3158473e-07,  2.4215106e-07,  1.7259245e-07,  1.9001901e-07,
 2.8116769e-06, -4.2049197e-07, -2.3318195e-07, -3.4786675e-07,
-1.3282950e-06,  1.9503538e-07,  7.3187738e-07, -1.6755374e-07,
 2.0260818e-06, -2.2896513e-06, -9.4511449e-07,  1.4487009e-07,
 4.7739327e-06,  1.1032379e-06,  1.2762994e-06,  1.8838649e-06,
 1.2822295e-07, -1.2806696e-06,  1.3857540e-07,  3.4866712e-06,
 3.2189462e-06, -1.7058185e-06, -4.8687718e-07, -6.4557446e-07,
 4.7367718e-07, -9.6545011e-07, -8.6823277e-08, -6.1244657e-07,
-1.1692937e-06,  1.1977182e-06,  6.1014487e-07, -2.0983905e-06,
 2.3249734e-06, -9.3769629e-08, -5.0816095e-07,  2.9237415e-07,
 3.0179478e-07,  6.6264744e-07,  1.2512727e-06, -7.0907540e-06,
 2.6590104e-07, -1.2883393e-06,  3.1351215e-06, -1.3980632e-06,
 5.9080435e-07, -5.0462961e-07,  6.9173149e-07, -4.3324030e-06,
 1.9745993e-07,  1.8513138e-06, -1.8875475e-07, -1.3279644e-06,
-2.2378448e-07, -2.3392512e-07,  9.5300271e-07, -5.0323433e-07,
-4.7245262e-08,  2.6775862e-07,  7.5876693e-07, -2.0769360e-08,
-7.1014057e-07, -3.9749611e-07,  8.7892232e-07, -1.4200609e-08,
-7.5822646e-07, -2.9200526e-06,  9.9025908e-07,  6.1361516e-07,
-1.0987474e-06, -9.0708431e-07, -6.1000299e-07, -5.9354261e-07,
-7.1406669e-07, -5.9200966e-07, -1.1275853e-06, -3.1277116e-06,
 1.9584074e-06,  1.3738864e-06, -1.9612683e-06,  2.6272184e-07,
-1.6644166e-06,  1.4215509e-06,  1.6773980e-06, -2.7457449e-07,
-1.0966780e-06, -1.0370272e-06, -1.9156420e-07, -2.0275004e-07,
 1.0921439e-06,  5.1455363e-08,  1.5942355e-06, -2.4238295e-06,
 2.4654085e-07, -4.2398989e-07, -1.0241440e-08,  9.6251176e-07,
```

```
        2.0979239e-06, -2.4751495e-08, -1.4939591e-06, -4.4436649e-07,
       -9.9018200e-07,  3.4375742e-06,  1.5368121e-06,  2.4223266e-06,
       -3.7400710e-06, -1.5991681e-06, -1.9138602e-06,  4.0612210e-07,
       -1.9795891e-06, -1.5347789e-06, -1.2878576e-06,  2.7026335e-08,
       -1.8728111e-06, -6.7570045e-07,  2.2471312e-07, -6.0191661e-07,
       -5.7885740e-07,  4.2563238e-07, -6.1395087e-07,  2.0728144e-06,
       -3.3458086e-06, -8.6501632e-07,  9.4360900e-07, -8.2565705e-08,
       -8.4791338e-07, -1.2293364e-07, -1.7304999e-06, -1.1000853e-07,
       -4.6341205e-07,  1.5732246e-06,  1.6367258e-07,  1.1959621e-06,
       -1.1980553e-07, -1.3323496e-06, -7.1203044e-07,  8.4091835e-07,
        2.8207299e-07, -1.1071019e-07,  1.4358054e-07,  4.4571271e-07,
        5.3701001e-07, -1.4935380e-07,  7.1589102e-07,  1.7932624e-07,
        1.6013994e-06,  3.0043439e-06,  2.9648632e-08,  2.6706499e-07,
       -1.3539939e-06, -8.9018289e-07, -2.6999542e-06,  4.2532932e-07,
       -2.0728880e-06,  5.3075598e-07, -2.0725609e-07, -3.9012090e-07,
        3.0478117e-07, -1.7759949e-06, -1.2273239e-07,  1.6773292e-06,
        2.9515672e-07, -6.0425970e-07,  8.7881489e-07,  4.2492729e-07,
        1.4341583e-06,  4.5859310e-06, -1.4797339e-06,  1.2501791e-06,
       -1.5812038e-06, -9.6041254e-07, -6.2077390e-08,  6.4760286e-07,
       -2.2696236e-06,  7.5270691e-06, -7.8965053e-07, -1.2298964e-06,
       -3.8296994e-06, -5.9909109e-07,  4.8790474e-07, -1.2000884e-06,
        3.6828071e-06, -1.1637169e-06,  4.6841342e-07,  2.1587259e-06,
       -1.0092203e-06,  4.0888850e-07,  9.5252312e-07, -1.2693790e-06,
        1.1578883e-06,  7.2781404e-07, -7.1294164e-07,  8.3881190e-07,
        1.2970787e-06,  1.6676038e-06, -1.1981311e-06, -1.2617713e-06,
        1.1030000e-07,  9.5781616e-07,  2.7952052e-07, -1.9803799e-07,
       -7.5962879e-07, -2.3042949e-06, -3.5158046e-06,  2.1192648e-06,
        2.3063077e-08,  1.1082215e-06,  2.3296776e-07,  6.2374869e-07,
       -1.4018821e-06,  5.6218653e-07,  5.4310664e-08,  4.9246330e-07,
        3.2082157e-06,  1.9624563e-06, -3.0965907e-07, -1.5154104e-06,
       -4.5566946e-09, -2.1743119e-06, -3.7694306e-07,  6.6964998e-07,
        7.4714529e-07, -1.5605341e-06,  3.4356955e-07,  2.1862197e-06,
       -3.2514535e-07,  1.2974859e-06, -4.9922977e-07,  2.4903695e-06,
       -4.4552226e-07, -2.5586751e-06,  1.5682859e-07,  1.0908978e-07,
       -1.8448271e-06, -3.6825492e-07,  4.0931997e-07, -6.4367526e-07,
       -4.8635223e-07,  4.8509116e-07, -8.7852777e-07, -3.0266262e-06,
       -1.3254373e-07, -3.0485293e-07,  4.4566167e-07, -4.1144881e-06,
       -1.9253009e-06, -2.0730896e-07,  3.4405622e-07,  1.9012240e-06,
        1.6844937e-06, -7.0059670e-08, -2.2037736e-06, -2.6010872e-07,
        1.1263195e-06, -3.6880618e-07, -6.0097960e-07,  2.2646125e-06,
       -9.0411578e-07, -5.8753807e-07, -1.8020868e-06, -3.0957821e-07,
        1.8183810e-06, -1.3175379e-06, -6.0977777e-07,  1.1626468e-06,
       -1.6122162e-06,  1.7759062e-07,  6.1751700e-07,  6.9236432e-07,
        5.4391126e-07, -1.0141420e-06,  4.3058841e-07, -2.4448814e-06,
        2.6729793e-08, -5.4220499e-07,  2.2507423e-08,  1.6651799e-06,
        2.0406296e-06, -6.5464019e-07, -1.2382228e-06,  1.0670372e-07,
       -6.6122715e-07, -6.8869575e-08, -1.1862993e-06, -9.1186126e-07,
       -3.5676703e-07, -1.3710106e-06, -8.0633868e-07, -1.8091794e-06,
        2.0669194e-07,  2.8919729e-07,  2.4718480e-07, -3.8672260e-06,
       -9.7086868e-08,  4.7274770e-07,  6.8758186e-07,  1.2245094e-07],
      dtype=float32), array([[ 0.00424895, -0.00761683,  0.03843338, ...,  0.00315931,
         0.01617977, -0.00771289],
       [-0.06642503, -0.04334348,  0.06223907, ..., -0.02383602,
        -0.05399771,  0.06699533],
       [-0.01999485,  0.05062384,  0.04312722, ...,  0.0355491 ,
        -0.02184569,  0.0435383 ],
       ...,
       [ 0.0100348 , -0.06562182,  0.06689391, ..., -0.03051082,
        -0.00689487,  0.05071403],
       [-0.042885  , -0.00857915, -0.01636489, ..., -0.04719684,
        -0.00657317, -0.06966235],
       [-0.05729357, -0.01291644, -0.04748272, ..., -0.00642902,
         0.01188784, -0.06732762]], dtype=float32), array([-3.5873927e-07,  3.2332343e-06,  4.75632
85e-07, -4.8057146e-08,
       -1.7298172e-06, -3.6996851e-06, -2.7607884e-06, -4.3279115e-06,
        3.8595822e-06,  8.8186425e-07, -1.5927840e-06, -1.3724247e-06,
        3.1822970e-06,  1.7857985e-06,  1.1216146e-06, -2.5327714e-07,
       -1.7685646e-06, -2.2751194e-06,  4.6579294e-06, -9.6345627e-07,
       -2.8861398e-06,  1.8680181e-06,  2.8856161e-06,  1.8214259e-06,
        4.4018698e-06,  3.8779358e-06,  2.8314573e-06,  2.6031671e-06,
       -3.8727258e-06,  1.2498302e-07, -6.5282507e-06, -1.5683274e-06,
       -9.2854344e-07, -1.5601838e-06,  3.0826568e-06,  7.9094821e-07,
        4.2855941e-06,  3.3201588e-06,  2.0399020e-06, -1.1141555e-06,
       -4.0806285e-07, -4.4941462e-06, -1.1998189e-06, -1.6587238e-06,
       -2.4246160e-06, -6.4908468e-06, -4.0141535e-07, -6.1543824e-06,
        2.0277673e-07, -7.0927604e-07, -1.9642025e-07, -1.1766880e-06,
       -5.5494606e-06, -9.3423864e-07,  3.9807383e-06,  1.0150811e-06,
```

```
 5.5494000e-06,  9.5425804e-07,  5.9807585e-06,  1.0150811e-06,
 9.3405811e-07, -2.9199309e-06,  3.6559509e-06, -7.8357889e-06,
-8.1661638e-06,  1.5128990e-06, -2.1000579e-07, -1.5131252e-06,
-1.3493652e-06, -7.6814895e-06,  6.5619874e-06,  2.3047289e-06,
-5.5723393e-07, -1.3995445e-07,  1.8956712e-06,  8.7511143e-07,
-2.5281263e-06,  9.7061065e-06,  4.1951253e-06,  2.8832480e-06,
 4.5252650e-06, -5.3600561e-06, -1.7354236e-06, -3.3180406e-08,
 2.8879997e-06,  4.3068326e-06,  2.0136997e-06,  8.2189808e-07,
-1.4970146e-06,  3.0811259e-07,  2.1619962e-06,  1.4545561e-06,
-3.2302692e-06, -4.3583914e-06,  1.7065526e-06, -1.0403148e-06,
 4.1557219e-06, -3.9424418e-07, -3.3192935e-06,  2.7897504e-06,
 4.2265424e-06, -1.4592302e-06,  5.4884009e-07,  2.7280211e-07,
-4.1976118e-06,  3.3020272e-06,  1.8023524e-06,  3.9590564e-06,
 5.3874186e-07, -2.1061439e-06,  3.7292047e-08,  1.1840588e-06,
 5.6824939e-07,  1.1301529e-06, -5.4852981e-06, -6.5207100e-06,
-9.3123026e-06,  4.8181141e-07, -2.1193225e-06, -1.3217877e-06,
-3.9179431e-06, -1.0567924e-05,  2.5150734e-06, -2.2606241e-06,
 5.0590597e-07, -2.8158515e-06, -3.1187915e-06,  4.6909522e-06,
-1.3221899e-06, -1.0929691e-07, -3.4285349e-06,  5.0967642e-06,
 9.2073861e-07,  2.8642228e-06, -3.1505833e-06,  2.5498812e-06,
-1.0881053e-06,  6.0752673e-06, -6.8547070e-06, -6.6260095e-06,
 2.3146260e-06,  8.9587957e-07,  2.9230969e-06, -4.5179431e-06,
-1.6157487e-06, -4.7272733e-06,  1.7705097e-06,  1.4145976e-06,
-1.3519490e-06, -2.7457593e-06, -3.7832419e-07, -2.9493590e-06,
-3.6052768e-06, -1.8886595e-06, -7.4035540e-07,  5.3979898e-06,
-7.4299555e-06, -3.7586337e-06, -2.9667194e-07,  2.0635473e-06,
-3.5204541e-06,  3.9281731e-06,  5.8721494e-06, -2.1448443e-06,
 4.4006247e-06,  4.9457308e-06, -6.0379029e-06, -2.3318689e-06,
 1.4582766e-06, -1.7852537e-06,  2.3740292e-06,  1.8276901e-06,
 4.1698845e-06, -1.3902354e-06,  2.3780278e-06,  8.8748271e-07,
 5.4969983e-06, -3.2701317e-07,  1.8730593e-06,  1.1005679e-06,
 2.7856463e-06, -1.2368807e-06, -1.3106423e-06, -1.6839957e-06,
 1.9048664e-06,  3.0820786e-06,  2.7596400e-06, -1.0575769e-06,
-2.4020026e-06,  3.9677443e-06, -6.0829086e-07,  7.3118122e-06,
-9.3790368e-07, -2.0826403e-06, -2.2905308e-06,  3.2409007e-06,
 8.8183873e-07, -1.5087275e-06,  1.1597034e-06, -2.7292781e-06,
 9.6918893e-07, -1.8397022e-06,  1.1119676e-07, -1.0536346e-06,
-2.6191872e-06, -3.2452413e-06, -4.2294073e-06, -6.4672931e-06,
 5.0366757e-06,  3.6274891e-06, -1.0617358e-07, -1.1295208e-07,
 2.8901408e-07, -8.5891634e-07, -6.1815963e-06,  2.8494885e-06,
-5.4711440e-06,  2.0820175e-07,  7.5684295e-07,  1.4088949e-06,
 1.9994793e-06, -2.0766258e-06,  2.4075950e-07,  2.0525031e-06,
-3.4697634e-06, -1.4060474e-06,  7.1603490e-06, -1.9066011e-09,
 1.9467097e-06, -6.2273843e-06, -7.4089112e-06,  2.0381940e-06,
 3.4265729e-06,  5.0483854e-06, -8.6399888e-07, -3.3449246e-06,
-5.4433724e-07,  3.6476119e-06,  3.7216437e-06, -1.3951759e-06,
 9.0477798e-08, -7.2806661e-06,  5.9695299e-06, -2.8086351e-06,
 1.6287305e-06, -1.3543847e-06,  2.8348182e-07, -6.1641600e-07,
-2.3822379e-06, -4.0769341e-06, -1.7764644e-06, -1.2246312e-06,
 1.5006691e-06,  1.5442166e-07, -1.3723643e-06,  2.6945979e-06,
-4.1164672e-06,  8.3566147e-06, -2.7729209e-06,  2.8890465e-06,
-5.9008039e-06,  5.6132541e-07, -3.0067702e-07, -8.1078082e-07,
-4.1453627e-06,  8.4539106e-06, -1.4663965e-06,  5.9599411e-06,
-2.3085715e-06, -3.4493539e-06,  3.4868540e-06,  8.4980711e-06,
 2.6570440e-06,  4.5067272e-06, -1.4750132e-06, -1.0208732e-07,
-4.1447884e-06, -3.6537267e-06, -4.5718552e-06,  1.4820096e-06,
-2.0226291e-06,  1.9932947e-06, -7.8304868e-07, -2.1170317e-06,
-1.8769504e-06, -2.9475127e-06, -3.4268946e-06,  1.1633459e-06,
 2.4123110e-06,  2.5834092e-06, -3.8423805e-06, -2.5160914e-06,
 1.4158450e-06,  1.3741586e-06, -1.1560564e-06,  2.6900634e-06,
 1.6516261e-06,  1.7882278e-06, -9.9733222e-07, -4.1835706e-06,
-1.7760719e-06, -4.6550649e-06,  2.2697072e-06,  8.5225711e-06,
-1.5218995e-06,  5.6582157e-06, -8.4550429e-06, -1.6063875e-07,
 5.3831009e-06,  3.0101428e-06,  2.5606104e-07,  1.4971151e-06,
 2.5352085e-06,  1.6104641e-06, -7.6400793e-06, -3.2448499e-06,
 1.5944945e-06,  1.7579197e-06,  2.3697833e-06,  6.5501736e-06,
-4.8021366e-07,  4.4413914e-06,  5.4992311e-06, -6.2200143e-06,
-2.8428767e-06,  5.2226233e-06,  4.9622313e-06, -6.1812000e-07,
 5.6079148e-06,  4.2331353e-06,  6.9659029e-07, -2.9991527e-06,
-1.1871936e-06, -1.9612171e-06,  2.2414301e-06, -2.4508101e-06,
-2.1002569e-07, -2.7493677e-06,  3.0977092e-06, -1.8177236e-06,
-2.5791687e-07,  2.9859718e-06,  1.5184293e-06,  2.2893776e-06,
-1.0901128e-07, -5.3655885e-06, -1.2555555e-06,  2.8904849e-06,
-5.4313437e-06, -1.1993816e-08, -1.2916745e-06, -1.1608782e-06,
 2.6176831e-06, -2.3347529e-06,  2.4909372e-07, -5.7798015e-06,
-1.8739821e-06,  1.1083540e-06, -3.0899450e-06,  4.8360021e-06,
 6.5901536e-06,  1.4310504e-06, -1.8898230e-07, -4.7398066e-06,
-4.4385420e-06,  1.4952459e-06,  1.8139873e-06, -6.5889358e-06,
```

```
       -4.4385420e-06,  1.4952459e-06,  1.8159875e-06, -6.3889358e-06,
        2.6050623e-07,  2.0850507e-06, -6.1935380e-06, -1.3595917e-06,
        8.7242797e-06, -4.0946429e-06, -5.8874468e-07, -2.2388563e-06,
        1.6803290e-06,  2.1508592e-06,  2.3783432e-06,  8.1198812e-07,
        2.5527263e-06,  7.2469021e-07, -3.8720186e-06,  2.5122517e-06,
       -7.5982564e-07, -2.0353807e-06,  2.2216630e-06, -3.4147542e-06,
       -1.0158096e-06,  1.9488248e-06,  2.7003688e-07,  6.2140612e-06,
        6.7913511e-06,  6.0174403e-07, -4.6227041e-07,  2.8899869e-06,
        2.5898555e-06,  1.7079824e-06,  1.7861379e-07, -4.1709063e-06,
       -8.3459975e-07, -4.0074588e-06,  4.7532617e-06,  5.2531759e-06,
        1.3729172e-07, -4.7385815e-06,  2.3911732e-06, -1.8136527e-06,
        1.1949312e-06, -1.3428549e-06,  1.3015253e-06,  4.0138757e-06,
        2.6604000e-06, -3.5242263e-06,  3.1490924e-06,  1.0298592e-06,
        3.2619587e-06,  1.0361983e-06,  4.9738755e-06,  2.9099253e-06,
       -5.4986053e-06,  4.2938582e-06, -7.8824866e-07, -8.6905590e-07,
        3.1778698e-06, -5.7030675e-06,  5.1729279e-07, -5.8585938e-06,
       -2.0658852e-06, -3.8102705e-06, -3.5241248e-06, -3.5794915e-06,
        2.5784025e-06,  2.8665417e-07,  3.1805835e-06, -2.3437215e-06,
       -1.7071774e-06,  5.6234553e-06,  3.8773192e-07,  2.3865905e-06,
       -5.5240057e-06,  4.0385789e-06, -6.2746301e-07,  8.9202251e-07,
        2.2462533e-08, -5.1718803e-06, -3.2905971e-06, -3.4943077e-06,
        1.5120413e-06,  4.3409404e-06,  3.1862416e-07,  2.0304991e-07,
       -9.3602523e-07,  1.7098315e-06,  7.1043614e-06, -1.6801955e-06,
        4.4193798e-06,  1.2744516e-06, -2.8683128e-06, -1.9843578e-06,
        1.9078007e-06, -5.7499610e-06, -4.8904981e-06, -3.1543337e-07,
       -8.8688591e-07, -1.3083255e-07,  1.3541260e-06,  3.1806508e-06,
        2.1888316e-06,  8.7243534e-06, -7.6814494e-08, -8.2138229e-07,
       -1.8566463e-06,  3.5005312e-06,  3.6913418e-06, -4.9676355e-06,
       -2.8511026e-06,  4.6458795e-06, -5.4590864e-06, -4.0674154e-06,
       -1.6728502e-06,  2.3402025e-08,  1.0479060e-06, -4.3085479e-06,
        2.4816427e-06,  6.2964506e-07,  4.3246368e-06, -6.3592993e-06,
        8.4344365e-06, -7.2181197e-06,  5.7574807e-06, -3.4723064e-06,
       -1.8074043e-06, -6.7428310e-08, -4.9190368e-07, -4.1171852e-06,
        2.1596645e-06, -1.0877897e-05, -2.2237364e-06, -7.6530332e-06,
       -1.6175753e-06,  1.9875652e-06,  1.5123937e-06, -4.1673961e-06],
      dtype=float32), array([[-0.08138584, -0.05236163, -0.06691989, ...,  0.07465757,
        -0.007274  ,  0.04541391],
       [ 0.02055095,  0.06968019, -0.08164417, ...,  0.00268274,
         0.04377765,  0.04611763],
       [ 0.02568058,  0.0461651 ,  0.07598647, ..., -0.00102526,
         0.00310636,  0.01739639],
       ...,
       [-0.005032  , -0.03355887,  0.03254119, ..., -0.05007885,
         0.02192311, -0.06921124],
       [-0.0064716 , -0.06902616, -0.06498689, ..., -0.04756   ,
         0.00639575, -0.05341933],
       [ 0.01705906, -0.07996665,  0.00018313, ..., -0.06433164,
        -0.06991205,  0.00617962]], dtype=float32), array([ 6.67004042e-06, -1.85766021e-05, -1.060
05473e-06, -8.16321005e-07,
        3.08819726e-05,  2.37709501e-05, -6.11619362e-06, -1.63167133e-05,
        2.63726565e-06,  2.70784913e-05, -3.50638629e-07,  1.91356812e-05,
        1.23151131e-05, -2.46753166e-06,  1.50930582e-05,  1.49145717e-05,
        1.43799562e-05, -5.07560117e-06,  1.85953413e-05,  3.56243686e-06,
       -1.08183367e-05,  1.28987849e-05, -2.19631102e-06,  1.53944029e-06,
        1.11061572e-05, -4.00303497e-06, -1.55049293e-05,  1.85778845e-05,
       -9.26562734e-06, -4.79534219e-06,  4.82319911e-06, -1.04833532e-06,
        1.56850401e-05, -1.77690945e-05, -4.56714815e-06, -1.47758556e-06,
       -1.07455332e-06,  2.62456870e-05,  3.85015801e-06,  7.70591669e-06,
       -8.96839538e-06, -1.18575899e-05, -4.08837786e-06,  1.06316074e-05,
       -1.52646116e-05,  3.59303340e-06, -4.72852298e-06, -2.43104441e-05,
        2.49089844e-05, -4.63930373e-05, -3.44321143e-06, -1.60800046e-05,
       -4.83839631e-06, -6.77744129e-06,  2.30485475e-05, -2.79999711e-07,
        2.29327179e-05, -2.49007426e-05,  2.17761340e-06,  1.58021394e-05,
        2.25022777e-05,  2.63057314e-06,  1.14841296e-05,  2.44554467e-05,
       -5.50618961e-06,  8.99292718e-06,  3.66555541e-05, -1.78382106e-05,
        3.58316015e-06,  2.68060394e-05,  5.68432552e-06,  1.47460469e-06,
        7.80207847e-06,  1.58389521e-05,  1.55030773e-06,  1.90080518e-05,
        1.29715181e-05,  2.58900054e-05, -2.80756331e-06,  4.08052256e-05,
        3.25311908e-06, -1.12274347e-06,  2.75155026e-06,  1.22124766e-05,
       -2.43705908e-05, -2.07291723e-05,  2.58518912e-06,  5.97213557e-06,
        1.86665548e-05,  1.42068075e-06, -6.43102339e-06,  5.43432043e-06,
       -1.59080791e-05, -6.96358302e-06, -8.99601400e-06, -2.77972322e-06,
       -2.72633542e-06, -1.67937633e-05, -9.35077151e-06,  2.62874983e-05,
        9.79927245e-06,  1.30733097e-05,  8.12411508e-06, -1.82684453e-05,
        9.52292430e-06, -1.05836875e-06, -1.17346553e-05, -1.04011879e-05,
        2.73072351e-06,  2.98922478e-05, -8.88447266e-06, -4.02884007e-06,
       -3.84635896e-05, -2.70133660e-06,  3.14248318e-05, -3.08342464e-06,
        5.92538618e-06,  1.41050496e-05,  9.31995964e-06,  2.05893730e-06,
```

```
 5.92538618e-06, -1.41050496e-05, -9.31995964e-06,  2.05893730e-06,
 4.34332924e-06, -2.91173592e-06, -5.24707730e-06, -5.06052174e-06,
-3.81764357e-06,  1.63923032e-05,  7.55667088e-06, -1.18832149e-05,
-1.03065977e-05,  9.11634106e-06,  9.45452030e-06,  1.09191546e-06,
 2.20201764e-05,  9.04200351e-06,  1.87001187e-05, -1.22250067e-05,
 9.64134961e-06,  2.72844754e-05,  2.85342285e-05, -7.54848952e-06,
-1.06521102e-05, -1.66652135e-05,  2.52407663e-05,  1.21956527e-05,
 3.09442567e-06,  1.81599135e-05, -8.96107576e-06, -4.88789083e-05,
-1.41929586e-05, -2.18071054e-05, -9.42766178e-07, -1.58887869e-05,
-1.42143117e-05,  8.25194002e-06, -2.25864142e-06,  2.08872389e-05,
 2.64153987e-05, -2.65183007e-05,  3.48669619e-06,  1.53320452e-05,
 8.48228774e-06,  2.11772494e-05,  2.22288290e-05, -5.65292885e-06,
-3.98038856e-06, -9.25314180e-06, -3.88834223e-06, -7.79846141e-06,
-2.58714658e-06, -4.49474328e-06,  1.95997018e-05,  9.94069342e-06,
-2.39704896e-05, -1.98231482e-05,  1.45950105e-06, -7.76978050e-06,
-1.60884690e-06,  2.14109186e-05, -2.79554843e-06, -1.40087432e-05,
 8.50617198e-06,  9.79489869e-07,  6.45057253e-06,  3.98617522e-06,
-5.31182059e-06, -3.71373676e-06, -1.10584533e-05,  1.47498622e-05,
 1.17385889e-06, -6.01777310e-06, -2.81527318e-05, -9.60355919e-07,
-8.27771055e-06, -3.99046894e-06, -5.82340908e-06,  5.63777394e-06,
-1.27647008e-05,  5.08297126e-06,  1.23678103e-06,  4.51988126e-06,
 3.47020750e-06, -8.66046230e-06, -3.57359590e-06, -4.09804716e-06,
 7.43227793e-06,  2.27806549e-06, -6.96347297e-06, -1.44327314e-05,
-6.54875021e-06, -1.09011344e-05,  3.69538520e-06,  1.73653225e-05,
 5.21157563e-06, -1.48672243e-05,  4.15836803e-06,  8.32194837e-06,
 1.54708468e-05, -1.83986867e-06, -7.70438874e-06, -6.76240063e-07,
-3.31366232e-06,  3.61325601e-06, -1.32541927e-05, -1.64141529e-05,
-1.48340587e-05, -2.77911681e-06, -1.16619558e-05,  3.34558831e-06,
-6.86871044e-06, -1.84229502e-05,  7.14005819e-06,  5.13013401e-06,
-1.23089858e-05, -5.47390471e-07, -1.35879336e-05, -1.34919210e-05,
 7.23664061e-06, -1.73324697e-05,  1.26767982e-05,  1.12287023e-06,
 7.83918858e-06, -8.07830111e-06, -1.26717750e-05,  1.69832765e-05,
-2.62555786e-05,  1.90574228e-05, -1.72687960e-05,  2.40947338e-06,
-2.13718668e-05, -8.90110550e-06,  9.23310381e-06, -6.64217032e-06,
 2.19965132e-05,  3.03771549e-06, -1.65337588e-05,  1.48624747e-06,
 1.09584753e-05, -2.78705866e-05, -7.87238059e-06,  7.63244952e-06,
 1.61633088e-05,  2.30485148e-06, -4.75611205e-06, -1.44000605e-05,
 5.50555751e-06, -7.90763806e-06, -7.40022097e-06, -1.67847029e-05,
-4.56264706e-06,  1.71197235e-06,  1.00265629e-06,  7.73041302e-06,
 9.36571155e-07, -8.14972907e-07, -1.26268401e-06,  2.27751298e-05,
-1.20112927e-05,  6.46126045e-06, -5.72142062e-06,  2.66127718e-05,
-1.23896115e-07, -1.62168599e-05, -1.53412529e-05, -1.83985176e-05,
 1.45551421e-05, -7.97884786e-06,  9.12705673e-06, -1.39694394e-05,
 8.09401809e-06,  1.21156481e-05, -7.75307399e-06,  7.26202245e-08,
 2.22098133e-05,  3.08312519e-06,  1.16607926e-05, -1.53479095e-06,
-3.35695972e-06, -3.08487215e-05,  7.41826170e-06, -1.22860192e-05,
-1.77362381e-05,  1.69014766e-05,  8.53016081e-06, -1.12977284e-06,
-7.31021601e-07, -1.21619714e-05, -1.30895096e-05,  1.19356546e-05,
-9.12466112e-06, -1.40229986e-05,  4.27010434e-07,  1.40845814e-05,
-1.98371508e-05, -1.55469606e-05, -6.62339744e-07, -3.88027092e-06,
 2.94044821e-06,  5.82291932e-06,  3.07209893e-06,  2.49547829e-05,
-1.20085897e-05, -2.90055777e-06, -1.13763290e-05,  7.87235513e-06,
 7.07154641e-06, -6.35329252e-06, -2.40750651e-05,  1.52883786e-05,
-1.48168474e-05,  9.30585975e-06,  3.34766696e-06, -1.26656744e-06,
 2.33557003e-05, -2.36347020e-07, -5.63806452e-06,  5.23651124e-07,
 1.63574714e-05,  1.28901793e-05, -1.51095028e-05,  8.84954989e-06,
-1.19180422e-05, -2.28352542e-06,  2.98212653e-05, -1.05766394e-05,
-5.35046320e-06,  8.05496802e-06,  3.88909712e-06,  1.08733720e-05,
-5.68881705e-06,  6.91973923e-07,  8.12792405e-06,  4.58421755e-06,
 4.18862919e-06, -1.05900535e-05, -3.54416238e-06, -5.00391616e-06,
 1.38358282e-05, -3.97844560e-05, -2.34352592e-05, -9.10662675e-06,
 6.40380767e-06, -6.92135836e-06,  4.34433741e-06,  4.96062312e-05,
 1.40580448e-06, -1.67609996e-05,  2.66791321e-05, -3.17078957e-05,
-1.09957102e-06,  9.95202754e-07,  2.32834464e-06, -3.09204484e-06,
-1.29216996e-05, -9.34010404e-06,  6.66959431e-06, -1.68147726e-05,
-3.04819650e-07, -4.69922315e-06, -2.77108120e-05, -3.69401205e-05,
-6.82938116e-06,  1.32322648e-05, -1.07452115e-05, -1.10463327e-06,
 3.09895768e-06,  2.12250452e-05, -6.18839113e-06,  1.17394275e-05,
 1.39445001e-05,  5.43078340e-06, -1.40965003e-05,  1.34456141e-05,
-1.46462617e-06, -2.83377562e-06, -8.10291658e-06, -4.35809761e-06,
 1.36879635e-05,  1.68500355e-05, -3.15903344e-05,  1.19043680e-05],
dtype=float32), array([[-1.00493180e-02, -2.85987221e-02,  2.64115296e-02, ...,
  -7.67545030e-02, -3.00296545e-02, -4.42466624e-02],
 [-3.49550657e-02, -4.39688079e-02,  3.24793672e-03, ...,
  -4.03811364e-03,  4.08451706e-02,  8.79716873e-02],
 [ 2.88664959e-02,  2.97997212e-05,  5.34106158e-02, ...,
   7.99163282e-02, -5.34251556e-02,  4.73011062e-02],
```

```
        ...,
        [-6.27458692e-02,  5.25595099e-02,  2.80637946e-02, ...,
         -7.49521842e-03, -5.35107069e-02, -1.51451295e-02],
        [ 8.29473510e-02,  3.02935485e-02, -2.60349121e-02, ...,
          2.29422916e-02,  5.90905547e-02,  2.77061928e-02],
        [ 3.20805684e-02,  7.29308426e-02,  2.86203460e-03, ...,
         -1.10407788e-02, -4.86491844e-02,  7.29204789e-02]], dtype=float32), array([-6.00513267e-
05,  1.06096530e-04,  1.18272319e-05,  3.82130274e-05,
        -2.43724389e-05, -4.90945822e-05,  2.59004155e-05,  6.83706457e-05,
        -8.43944508e-05,  1.07051783e-04, -5.92918477e-05,  3.85824314e-05,
         3.07914597e-05, -9.76179290e-05, -2.96863200e-05,  9.16697754e-05,
        -7.94024436e-06, -2.36555952e-05,  3.55956763e-05,  1.47173250e-05,
        -1.44408521e-04, -1.30466251e-05,  8.51267760e-05, -6.93994662e-05,
         1.96843303e-05,  2.72093785e-05, -3.05525828e-05, -1.92385178e-05,
        -5.82878856e-05, -1.71681695e-05,  1.30871282e-04,  4.97697438e-05,
         9.25232089e-05,  8.52868179e-05,  3.97319927e-05, -4.29177526e-05,
         3.18862949e-05, -5.23116341e-06, -1.62519627e-05,  1.14400907e-04,
         2.04225162e-05,  2.45112406e-05, -1.01445170e-04, -4.11684377e-05,
        -1.72613454e-05,  5.11792714e-05, -6.55986423e-06,  1.32807530e-04,
         3.35200129e-05, -2.45905335e-06, -6.68616995e-05, -5.13235573e-05,
        -1.81699738e-06,  1.49459913e-04,  8.44258975e-05,  2.06753703e-05,
         1.13669659e-04, -3.20729691e-06,  6.78249708e-05, -1.02930993e-04,
         9.73465649e-05, -4.22881167e-05,  4.58988434e-05,  2.07545545e-05,
         1.72803884e-05, -8.93465185e-05, -2.14639120e-04, -5.68472715e-05,
         8.76139893e-05, -4.20776923e-05, -6.84847691e-05, -1.21747398e-05,
         7.50352774e-06, -9.88676329e-05, -7.66983721e-05, -1.09345747e-04,
         1.30976014e-05, -3.24839721e-06,  1.92284460e-05,  6.48656569e-05,
        -3.00447591e-05,  1.70359650e-04,  6.34763128e-05,  1.28608635e-05,
         1.61922580e-05, -2.19367530e-06,  2.92930235e-05, -1.46065149e-05,
         6.60475853e-05,  4.14418137e-05,  5.35470936e-05, -9.92129135e-05,
         9.37439290e-06,  1.24524187e-04,  1.98242124e-05, -4.54304864e-05,
         1.28672027e-05, -9.20453167e-05, -3.84454288e-05,  5.28636301e-05,
         2.13717300e-04, -3.48792310e-05,  4.20013203e-05,  1.91563504e-05,
         2.82123310e-05, -3.45354492e-05, -4.58961513e-05,  8.63515015e-05,
        -9.74660816e-06, -4.70178929e-05,  9.78982862e-05,  1.96851615e-05,
         1.01185124e-05,  3.71482784e-05,  9.20870662e-05, -7.87202735e-05,
        -2.09571608e-05,  9.53511972e-06, -2.19030117e-05, -9.04957460e-08,
        -1.47117971e-04, -5.11075814e-05, -1.87868245e-05,  5.20668436e-05,
         7.92117953e-06,  1.66254380e-04, -1.81663381e-05, -6.05283967e-05,
        -1.61045871e-04, -3.94800882e-05,  6.65261177e-05,  7.64390570e-05,
        -4.50918378e-05,  4.09269196e-05, -9.96044255e-05, -3.93711161e-05,
         4.70557934e-05,  8.34764724e-05, -5.94550256e-05, -5.96010577e-05,
        -2.94939236e-05,  4.44860634e-05, -6.48137066e-05, -6.62655948e-05,
        -3.76286007e-05,  9.17670513e-06, -4.23824386e-05,  1.11743168e-04,
         1.55932663e-04, -3.07072769e-05,  1.42615249e-07, -3.62538267e-05,
        -2.92075238e-05,  1.69721257e-04, -7.37645751e-05,  5.84719019e-05,
         4.83297044e-05, -2.36788637e-05,  5.82378198e-05, -5.98463812e-05,
        -6.85764462e-05, -6.10161660e-05, -8.89757866e-05,  5.19966989e-06,
         4.40228978e-05, -1.92739353e-05,  5.69097028e-05,  1.35165072e-04,
        -4.28404837e-06,  4.31525659e-05,  2.23221668e-05, -1.72073815e-05,
        -3.96384939e-06, -7.19016753e-05, -3.93426999e-05,  5.40472938e-05,
        -6.13501979e-05, -1.47311348e-05, -4.51109117e-05, -3.89598099e-05,
        -4.64878249e-05, -1.15552764e-04, -1.79454655e-05,  8.64304675e-05,
         7.14782946e-05, -8.51702644e-05, -1.18071810e-04, -9.18534806e-06,
        -3.14818644e-05,  8.58000567e-06,  6.44105530e-05, -3.63238505e-05,
         4.45904989e-05, -4.50782827e-05,  1.45573344e-04,  6.01560205e-05,
         5.03294141e-05,  4.31276858e-05, -6.50108359e-06,  7.19135351e-05,
        -6.51696027e-05,  1.83457130e-04, -1.18532917e-04, -2.95435602e-05,
        -1.06268446e-04, -2.48136221e-05,  1.18497381e-04,  1.28234082e-04,
        -6.53117313e-05,  7.47804661e-05, -3.06423026e-05, -5.48839998e-05,
        -3.67561042e-05, -7.11805405e-05, -1.81163141e-05,  9.35895441e-05,
        -4.82080650e-05,  8.63240166e-06,  5.78946238e-05,  4.56257912e-05,
        -1.74270390e-05,  1.16757210e-05,  5.09411002e-05, -6.85476471e-06,
         1.72525746e-04, -2.62610065e-05,  3.95032148e-05,  9.89068212e-07,
         6.56957636e-05,  2.35543084e-05, -8.34838138e-05, -8.37018015e-05,
        -1.10782188e-04, -2.63468446e-05, -3.48760223e-05,  1.80452807e-05,
         4.83959520e-05,  1.42508234e-05, -2.84884827e-05,  1.22414145e-04,
        -1.07015214e-04, -1.16505587e-04, -4.51014384e-05,  3.24349130e-05,
        -5.34265200e-06,  7.03753540e-05,  7.87507815e-05,  3.32608688e-05,
         6.32699521e-05,  1.16191459e-05, -3.89182896e-05, -6.83757098e-05,
         6.03975641e-05,  4.17690098e-05,  9.84209546e-05, -7.47846207e-06,
         1.67409653e-05,  4.97624642e-05,  1.39242584e-05, -5.37615633e-05,
         1.26057157e-05, -6.60599289e-06, -2.01888543e-05, -1.17630407e-04,
        -7.08840089e-05,  5.20271642e-05, -1.12383328e-04, -3.63595937e-05,
         1.19850687e-04,  6.86201602e-05, -1.93495125e-05,  8.79492654e-05,
         3.05344838e-05,  3.60303966e-05, -8.64309332e-05, -3.08141789e-05,
         2.27921955e-05, -8.37811458e-05, -6.35430697e-05,  8.37379412e-06,
```

```
        1.46274133e-05, -1.09080618e-04, -4.65838893e-05,  1.31855064e-04,
        2.72177622e-05,  3.97313452e-05,  3.05611575e-05,  1.37121879e-05,
       -2.76146257e-05,  7.59366594e-05, -2.79267497e-05,  7.22164841e-05,
        1.53985220e-05,  1.10198300e-04,  9.91817360e-05,  3.10185060e-05,
       -5.44558588e-06, -1.69514133e-05, -1.19107361e-04, -1.62836459e-05],
      dtype=float32), array([[ 0.09970868, -0.01742327,  0.00389498, ...,  0.01114637,
         0.0013447 ,  0.09837274],
       [-0.02905613,  0.10961567,  0.06854772, ...,  0.05608926,
        -0.08272354,  0.04448966],
       [ 0.01238412, -0.06597862,  0.06317218, ..., -0.02949279,
        -0.04547245, -0.08729645],
       ...,
       [-0.07009579, -0.06147412,  0.10798448, ...,  0.07897439,
        -0.02051538,  0.07661559],
       [-0.04043257, -0.00226437, -0.08582428, ...,  0.05477638,
         0.03499624, -0.09294565],
       [ 0.10411858, -0.06615025, -0.0133122 , ..., -0.01117307,
        -0.07776159,  0.02750229]], dtype=float32), array([-1.95710054e-05,  6.81542500e-04, -4.987
05194e-05, -4.75767214e-04,
       -3.84739076e-04,  3.09045281e-04, -7.32349930e-04, -4.95263026e-04,
       -2.51913298e-04,  2.44277151e-04,  1.59072282e-04,  1.53703251e-04,
        2.97081308e-04,  1.14187351e-05,  5.92870172e-04, -2.10850980e-04,
       -6.88592670e-04,  1.11093805e-05,  4.06182051e-04,  9.92579735e-05,
       -8.82734836e-04, -1.04584004e-04, -3.74651921e-04, -9.89677792e-05,
       -6.40990678e-04,  2.68382252e-07, -3.22088221e-04,  2.29083293e-04,
        3.25698667e-04,  3.28114373e-04, -1.14499971e-05,  5.06683311e-04,
       -4.28008498e-04,  5.63179259e-04,  3.48364920e-05,  6.31143412e-05,
       -5.20268048e-04,  1.85621990e-04,  4.37270755e-05, -1.42272824e-04,
        2.85165239e-04,  1.60633877e-04, -3.23707063e-04,  2.99345993e-04,
        2.11368970e-04, -2.43034694e-04, -2.66737363e-04,  2.04134180e-04,
        2.72090227e-04, -1.88753897e-04, -4.58980765e-04,  1.98833339e-04,
        2.77663727e-04, -6.93600799e-04,  3.83079459e-05,  7.68208629e-05,
       -2.67044990e-04,  2.72476609e-04,  4.77661997e-05,  2.77501444e-04,
       -2.13008519e-04,  1.20211291e-04,  3.88545741e-05,  6.38578649e-06,
       -1.31966168e-04, -5.39335924e-05,  6.42942250e-05,  2.63789989e-04,
       -2.70175573e-04, -3.74964933e-04,  3.02515051e-04, -7.02792167e-05,
        1.14498769e-04,  1.36285627e-04, -6.34162279e-05, -2.92669079e-04,
       -2.28716919e-04,  4.21740930e-04,  1.49305240e-04,  1.71408275e-04,
       -3.33251752e-04, -1.11255576e-04, -3.39837337e-04,  1.97811765e-04,
        3.94601579e-04, -2.18385292e-04,  1.76679503e-04,  4.90761246e-04,
       -1.79819152e-04, -1.30993908e-06, -8.55776714e-04, -7.33095454e-04,
        2.87877629e-04,  4.07263171e-04, -2.15078617e-04,  2.40080262e-04,
       -3.23185464e-04,  8.56443250e-04, -6.51958981e-04,  2.48194556e-04,
       -3.44898144e-05,  4.29941021e-04,  1.73780005e-04,  3.08703557e-06,
        4.16379917e-04, -2.64507486e-04, -3.22436215e-04, -2.45797943e-04,
       -1.72925371e-04, -5.24781412e-04,  5.42985799e-04,  4.51348715e-05,
       -6.13197772e-05,  4.57165479e-05, -3.12211370e-04,  4.03779501e-04,
       -1.14364993e-04, -6.12170843e-05, -1.19750082e-04,  3.06220085e-04,
       -3.22885084e-04,  1.04022074e-04, -3.18369974e-04,  3.66642917e-06,
       -4.58251481e-04, -4.24146769e-04,  4.15363465e-04,  2.04766708e-04,
        2.75318598e-04, -8.30491481e-04,  4.06123509e-05, -2.78707914e-04,
        1.53323577e-04, -3.61078361e-04,  3.67643719e-04, -2.15983237e-04,
       -6.98859105e-04, -4.78713118e-05, -5.44981740e-04, -8.22150978e-05,
        1.10210385e-04,  2.05809687e-04,  2.84417853e-04, -6.78921642e-04,
       -2.13467749e-04,  2.88017181e-04, -7.53596774e-04,  1.71749823e-04,
       -5.73426660e-05, -1.23348043e-04, -4.76737914e-05,  3.37817910e-04,
       -4.62973228e-04,  1.09898821e-04, -2.86375027e-04,  1.66896658e-04,
        5.42613561e-04,  3.08224087e-04, -1.64900193e-04, -1.21377407e-05,
        5.56690538e-05,  2.21949958e-05,  2.32986335e-04,  1.12526650e-04,
        2.32262450e-04,  2.02843963e-04, -7.64556753e-05, -5.25883006e-05,
       -6.64491265e-04, -9.25150161e-05, -6.79835386e-04, -9.32841504e-05,
        2.94014346e-04, -6.55360200e-05,  5.77764076e-05,  4.80429240e-04,
       -1.97476416e-04,  1.74666566e-04,  1.95875953e-04,  1.43013429e-04,
        4.24066704e-04, -4.85575729e-04,  5.32385777e-04,  5.41296962e-04,
        9.55712167e-05, -6.55716343e-04, -2.65139515e-05,  1.51133951e-04,
       -4.68667917e-04,  1.57401810e-04,  3.95491341e-04,  2.54753919e-04,
       -9.17285390e-04,  3.06273403e-04,  1.23272766e-04, -3.71501781e-04,
        2.24789164e-05, -2.02517709e-04, -8.04732044e-05, -2.26334538e-04],
      dtype=float32), array([[ 0.03194631, -0.01612179, -0.0510719 , ..., -0.04801689,
        -0.06340221, -0.13630413],
       [-0.01783801,  0.12953423, -0.11690785, ...,  0.09339511,
        -0.04863   ,  0.05645644],
       [ 0.08256371,  0.11416705,  0.09843397, ..., -0.04241565,
        -0.10996096, -0.14053416],
       ...,
       [ 0.02919956, -0.02632162, -0.07468853, ..., -0.08364539,
        -0.11853302,  0.07096918],
```

```
       [ 0.00910728, -0.03398846,  0.01471517, ...,  0.0976157 ,
        -0.0180706 , -0.12873241],
       [-0.0085458 ,  0.00715151,  0.05734326, ...,  0.1065176 ,
         0.09564799,  0.11364484]], dtype=float32), array([-2.9021241e-03, -7.9873216e-04, -1.56739
54e-04,  1.5926970e-03,
       -2.0034916e-03, -2.1328016e-03, -1.7259317e-03, -2.0406048e-03,
        2.5879024e-03, -4.2544940e-04, -3.8149878e-03, -2.0566960e-03,
       -5.6393543e-04, -1.1197424e-03, -8.0310198e-04, -1.6792754e-03,
        2.5243089e-03, -1.3499642e-03, -1.9359750e-03, -1.3984415e-03,
       -6.4701150e-04, -2.0627424e-04, -8.7896059e-04,  3.5441066e-03,
       -1.2151644e-03,  1.1444509e-03, -1.0913184e-04,  2.2052624e-04,
        1.4784675e-03, -2.2321062e-03,  1.2035335e-03, -2.9555955e-03,
       -1.2663999e-03,  1.7629190e-03, -1.3871387e-03,  8.8472373e-04,
       -4.6836426e-03, -1.8749706e-03,  2.3031982e-03, -2.6915818e-03,
        5.9403572e-04, -5.8247894e-04,  1.0795697e-03, -1.2867956e-03,
        2.9506667e-03, -2.9609846e-03,  7.0078659e-04,  1.0887861e-03,
        1.4138005e-03, -7.7982864e-04, -6.7102548e-04, -3.8002008e-03,
        1.1451925e-03, -4.6755958e-06, -3.6207261e-04, -5.4756732e-04,
       -3.1556657e-03,  4.5303786e-03,  2.1408414e-03,  1.0343613e-03,
        1.5972840e-03,  2.4091667e-03, -7.4558594e-04, -2.8530429e-03,
       -6.4648644e-05, -2.6706327e-03, -2.4095513e-03, -2.0120146e-03,
       -2.3365696e-03,  5.0812837e-04, -5.4417071e-03, -1.3605627e-03,
       -1.9093349e-03, -1.4585406e-03, -1.7475257e-04,  9.7086735e-04,
        9.9464867e-04,  1.4738750e-03, -1.5516771e-03, -1.0642965e-03,
       -1.6797688e-03,  1.3243475e-03,  1.3052794e-03,  6.2179915e-04,
        6.9734984e-04,  2.2081637e-03, -2.3817420e-03,  6.3592626e-04,
        1.7377867e-03, -2.9411823e-03,  1.5851510e-04, -4.0244460e-03,
        1.4768695e-03,  3.4499643e-03, -1.1811586e-03,  1.4210605e-03,
       -1.5753959e-03, -1.1991388e-03,  1.0764861e-04, -1.1314156e-03],
      dtype=float32), array([[-5.34017645e-02, -2.26741716e-01, -2.23494768e-01,
        -7.89509565e-02,  2.62275450e-02,  4.17720824e-02,
        -1.96065471e-01, -2.12752208e-01,  1.73028946e-01,
        -3.39529961e-02],
       [-1.38275558e-02, -2.27949306e-01,  8.23683292e-03,
        -5.27356379e-02,  7.33089298e-02,  9.84571353e-02,
        -1.93316285e-02,  2.26297099e-02,  1.78144425e-02,
         5.51111698e-02],
       [ 7.12968931e-02, -2.31592536e-01, -7.10785612e-02,
        -9.99636948e-02, -2.11766213e-01,  2.97894347e-02,
        -1.30386785e-01,  1.31650075e-01, -5.10667264e-02,
         1.82597682e-01],
       [-3.52548882e-02, -1.20417349e-01, -2.25315198e-01,
         2.05342639e-02, -1.94985196e-01,  2.14736655e-01,
         8.95948112e-02, -1.69333905e-01, -9.75956172e-02,
         1.17736213e-01],
       [ 1.26021549e-01,  1.42738342e-01,  1.78484559e-01,
        -1.84220895e-01,  1.63613826e-01, -4.57966235e-03,
        -5.39229400e-02,  6.94330335e-02,  1.40602468e-02,
         4.83498834e-02],
       [ 2.73371004e-02, -9.69242379e-02, -1.89901993e-01,
         9.62813422e-02, -1.80765867e-01, -1.93649411e-01,
         1.06423482e-01, -4.75589819e-02,  1.63127393e-01,
        -2.14439392e-01],
       [ 1.33675084e-01,  3.30295600e-02, -1.60049070e-02,
        -1.69817716e-01,  4.59303968e-02, -2.24876657e-01,
         1.13806196e-01,  9.51468106e-03, -6.33578897e-02,
         5.91128580e-02],
       [ 1.88009426e-01, -1.76046476e-01, -1.60784632e-01,
        -1.97953656e-01,  1.87035978e-01, -1.33508695e-02,
        -1.19354739e-03, -4.62946147e-02, -7.59117454e-02,
         1.64675429e-01],
       [ 2.92962044e-02, -2.01625526e-01,  6.94927499e-02,
         2.42174745e-01,  9.35682952e-02, -2.12786458e-02,
         1.05534039e-01,  1.46366179e-01, -2.40897998e-01,
        -1.20173171e-01],
       [ 2.04914615e-01,  2.84384992e-02,  3.83456834e-02,
         2.23667145e-01, -8.74156654e-02, -1.05839454e-01,
        -2.47474555e-02,  3.44648808e-02,  1.47850066e-01,
         5.52700870e-02],
       [-7.99806714e-02,  1.75701261e-01, -6.51910976e-02,
         1.17625715e-02,  1.85159147e-01, -1.20912656e-01,
         1.15514942e-01,  1.69679925e-01,  1.88545614e-01,
        -2.17646554e-01],
       [ 6.10861257e-02, -5.59334308e-02,  1.10635765e-01,
        -3.63436118e-02, -5.10826595e-02, -8.10551399e-04,
         2.33120322e-01, -5.01949191e-02,  1.05526708e-01,
        -1.58371821e-01],
```

```
[-1.65159330e-01, -2.24356279e-01, -3.87099087e-02,
 -9.59560797e-02,  1.49150848e-01, -5.00646718e-02,
  2.05826029e-01,  1.42550021e-01,  3.38289440e-02,
  2.31801569e-01],
[-7.71086067e-02,  1.36016414e-01, -2.17108548e-01,
 -6.06942363e-02,  1.03793822e-01, -1.88257501e-01,
 -1.11673865e-02, -1.75666481e-01,  1.30694523e-01,
  2.07297668e-01],
[ 8.03958923e-02,  8.10526162e-02, -1.27804816e-01,
 -1.73062250e-01,  1.01630799e-01,  2.20812067e-01,
  2.23853886e-01, -7.73628503e-02, -1.12372473e-01,
  8.12276527e-02],
[-1.97075844e-01, -1.38362929e-01, -5.13930293e-03,
 -1.20140882e-02,  1.55171841e-01,  8.52898601e-03,
 -2.05948561e-01, -1.68879181e-01,  2.19011113e-01,
  1.08947447e-02],
[ 9.62677151e-02,  1.42802820e-01,  6.66754544e-02,
  2.15185314e-01, -8.80078748e-02, -1.73305944e-01,
 -7.49644414e-02, -1.03496365e-01, -1.86249450e-01,
 -1.12527758e-01],
[ 4.94063124e-02,  3.77022363e-02, -1.98025495e-01,
 -1.42127410e-01, -2.38510489e-01,  5.79930507e-02,
  1.73947752e-01, -2.24633977e-01, -1.24819810e-02,
 -7.88368136e-02],
[ 9.50933695e-02,  1.11697577e-02, -9.64603946e-02,
 -2.83219162e-02,  9.54458639e-02, -9.89839956e-02,
  8.94240141e-02, -2.13654377e-02, -1.68843959e-02,
 -1.31605238e-01],
[-1.91862166e-01, -8.60055313e-02,  9.54969823e-02,
 -2.15904623e-01,  1.50061801e-01,  9.25973579e-02,
  6.11042231e-02,  1.86423242e-01, -6.85194284e-02,
 -1.18036002e-01],
[-3.98069508e-02, -1.82411477e-01, -3.96715775e-02,
  1.92657501e-01,  1.84224278e-01,  1.05612375e-01,
  2.58900616e-02,  2.09794179e-01,  1.63058400e-01,
  8.80397111e-02],
[-6.32127151e-02,  9.13961753e-02, -2.19921991e-01,
  1.68602206e-02,  6.00702465e-02,  9.11030173e-02,
 -4.90346588e-02,  7.13857338e-02,  1.48775140e-02,
  1.99684605e-01],
[ 1.61045983e-01,  7.64914975e-02,  2.17735752e-01,
 -1.96409330e-01, -5.23812231e-03,  1.93982925e-02,
 -6.75037652e-02, -4.82698567e-02,  8.32927320e-03,
  1.82781890e-01],
[-1.56215832e-01, -5.57055362e-02,  1.77267954e-01,
 -3.42889987e-02, -1.89836264e-01, -1.30660176e-01,
  1.06377617e-01, -1.40439257e-01, -2.50107348e-01,
  5.63748814e-02],
[ 5.49078286e-02, -9.04259086e-02, -9.50075984e-02,
  2.12017819e-01, -8.15038905e-02, -2.26367667e-01,
 -7.34329000e-02,  6.74636662e-02,  1.14031069e-01,
 -2.01068386e-01],
[-2.55562477e-02, -1.09277599e-01,  2.26660326e-01,
  1.70736149e-01, -1.63321152e-01,  1.76224530e-01,
 -3.76281887e-02, -5.69967590e-02,  2.18528166e-01,
  2.20557824e-01],
[ 1.03209250e-01, -1.66814312e-01,  2.25023314e-01,
  1.22280121e-01,  1.42545357e-01, -1.72247037e-01,
 -1.51568413e-01,  1.69722233e-02,  9.62246954e-02,
  1.18411042e-01],
[ 9.44919735e-02, -1.66425690e-01, -3.05171087e-02,
  1.36544034e-01,  1.87561274e-01,  6.61263391e-02,
  1.34905040e-01,  1.92174241e-01, -7.13600591e-02,
 -3.21189029e-04],
[-4.87501696e-02, -1.35116309e-01,  6.28358126e-02,
  2.13047728e-01, -2.12695785e-02,  7.73145333e-02,
 -1.21175162e-01,  6.58139363e-02,  8.27267617e-02,
  1.68663472e-01],
[-1.04328595e-01, -3.84166278e-02, -2.22922593e-01,
  3.99566293e-02,  2.12121844e-01, -1.92769065e-01,
 -5.21827862e-02,  1.46559194e-01,  1.61411405e-01,
  8.04901868e-02],
[-2.34885812e-01, -2.15980764e-02, -4.93654683e-02,
 -3.07899974e-02,  2.06468642e-01,  9.58374366e-02,
  7.72357881e-02, -6.63410872e-02, -2.24591941e-01,
 -1.46906674e-01],
[ 1.09972857e-01,  1.68094531e-01,  9.96075049e-02,
```

```
 -2.07103133e-01, -1.93068553e-02,  1.24383077e-01,
  1.35422349e-01,  5.45966402e-02,  1.58863395e-01,
  5.57911508e-02],
[-4.95529622e-02,  8.95425677e-02, -1.12175249e-01,
  1.65820912e-01,  1.35120988e-01,  2.16212094e-01,
 -3.10998932e-02,  1.20243216e-02,  1.60505354e-01,
 -9.46474448e-02],
[ 1.96714073e-01, -9.09511521e-02,  1.71644986e-01,
  1.38300285e-01, -6.63746372e-02, -1.44102633e-01,
 -8.07579085e-02, -7.05071166e-02, -6.39744326e-02,
  1.37666419e-01],
[-5.43679530e-03,  1.75802261e-01, -1.15463875e-01,
  2.71775518e-02,  1.32652298e-01,  6.17693774e-02,
 -2.70676333e-03,  3.69315185e-02, -2.75781136e-02,
 -2.22280815e-01],
[ 1.42455474e-01, -7.21127987e-02, -2.06213593e-01,
  2.27922112e-01,  1.16825446e-01, -6.98003918e-02,
  1.10393606e-01, -1.35085247e-02, -1.41003486e-02,
  2.00220942e-01],
[ 2.13570371e-01,  3.22955451e-03, -2.08604261e-01,
 -1.06653459e-01,  2.06151590e-01, -1.68150276e-01,
 -1.87812895e-01,  1.97247565e-01,  1.52452692e-01,
  7.52656832e-02],
[ 1.95506036e-01,  8.70906487e-02, -5.92012070e-02,
 -7.77603462e-02, -1.31338865e-01,  1.49605662e-01,
 -2.15379864e-01,  5.93905002e-02, -1.11457463e-02,
 -1.43911645e-01],
[-1.99813053e-01,  1.84277683e-01,  2.04511806e-01,
  1.38253793e-01,  1.07329682e-01, -2.24378109e-01,
 -2.68373284e-02, -1.70781091e-01, -1.09711066e-01,
 -1.06171422e-01],
[ 1.73109360e-02,  1.11310914e-01,  8.68395269e-02,
 -2.11635500e-01, -5.19989096e-02, -2.03615874e-01,
 -6.34242073e-02,  5.35697676e-02,  1.55933335e-01,
  1.56810924e-01],
[ 1.50265828e-01,  2.13106439e-01,  1.23691641e-01,
  6.94332132e-03, -1.02406643e-01,  6.68158829e-02,
 -9.50706005e-03,  5.10978624e-02, -1.22616872e-01,
 -1.30131003e-02],
[-1.73093498e-01, -8.68725125e-03, -1.52012289e-01,
  1.12689786e-01,  8.44116509e-02,  1.28638834e-01,
 -7.91365206e-02, -1.41218662e-01,  7.71101713e-02,
 -1.31708875e-01],
[ 1.08609416e-01, -2.32196990e-02,  1.72720805e-01,
  1.99896351e-01,  4.55737226e-02, -7.32964277e-02,
 -6.86229020e-02, -2.18649298e-01,  3.34879868e-02,
  1.97849050e-02],
[-1.49370983e-01, -8.23966712e-02, -1.13144577e-01,
 -6.51552454e-02,  3.26713920e-02, -8.14128295e-02,
 -6.61259592e-02,  1.20602384e-01, -3.49425897e-02,
 -1.11823305e-01],
[-2.21457139e-01,  8.13023522e-02,  1.38750821e-01,
 -7.62894303e-02,  3.75862047e-02,  9.57111269e-02,
  1.21211611e-01, -1.68490142e-01, -2.01258868e-01,
  1.07169650e-01],
[ 7.74193630e-02,  9.48814824e-02,  2.06918225e-01,
 -1.29487798e-01,  1.84687480e-01,  2.26711091e-02,
  2.31822416e-01,  1.03012010e-01,  1.83405504e-01,
  3.10105253e-02],
[ 8.00558031e-02,  9.80037302e-02,  1.99011803e-01,
  1.11844651e-02,  7.58475885e-02,  6.29556179e-02,
 -2.34268084e-02,  1.20722666e-01, -3.61747928e-02,
  1.85789585e-01],
[-1.97455347e-01,  9.83820781e-02,  2.99313571e-03,
  1.76614657e-01, -8.75632688e-02,  2.32679442e-01,
  2.19833791e-01,  2.00434908e-01,  1.15506038e-01,
  9.18351933e-02],
[ 1.90987587e-01, -1.50741577e-01, -6.00936487e-02,
  1.79342672e-01, -2.09725406e-02,  2.63782758e-02,
 -1.84848636e-01,  7.83446953e-02, -5.57762310e-02,
  1.99451208e-01],
[ 2.08156928e-01,  1.43193841e-01, -6.63073361e-02,
  6.69204816e-02,  5.22173420e-02,  1.18647151e-01,
  5.74623831e-02,  6.54630288e-02, -1.39234532e-02,
  5.65417064e-03],
[-1.37403002e-02,  7.36260712e-02,  1.51085883e-01,
 -1.71323374e-01, -8.52411017e-02,  1.66760013e-01,
```

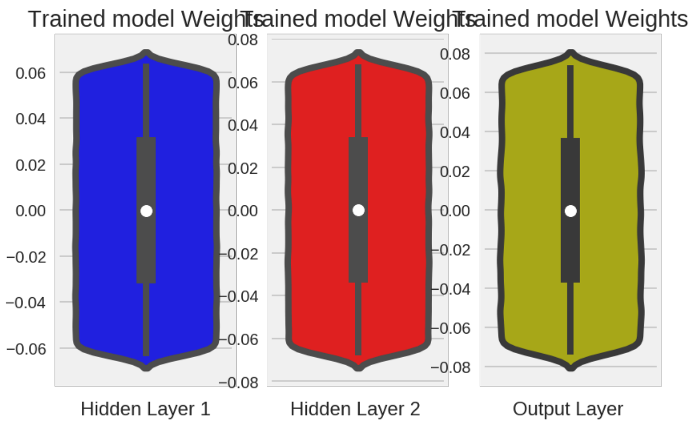
```
 -1.55413806e-01,  8.76471847e-02,  3.00887693e-02,
  1.23252869e-01],
[ 3.72854024e-02, -1.36840940e-01,  3.47216912e-02,
 -2.11355448e-01, -1.02120005e-01,  5.85890040e-02,
 -1.65725723e-01, -1.48267224e-01,  2.03327879e-01,
 -1.12745576e-01],
[-1.63858458e-01, -1.39515385e-01,  1.14847034e-01,
 -5.09497756e-03,  1.84576243e-01, -9.82522685e-03,
 -1.67098656e-01,  2.18916640e-01, -8.24485719e-02,
  1.63828820e-01],
[ 7.33054895e-03,  1.93137527e-01,  7.76774529e-03,
 -1.93218783e-01, -1.70882121e-01, -1.73317101e-02,
  1.21500082e-01,  8.87569934e-02, -1.94725409e-01,
 -9.28934216e-02],
[ 1.94073752e-01,  3.22247334e-02, -1.34416357e-01,
  1.38067544e-01, -2.00832233e-01,  1.05900489e-01,
 -1.21898845e-01, -1.59143284e-01,  1.83439907e-02,
 -1.37844235e-01],
[-2.75398996e-02, -7.47350650e-03,  1.61154196e-01,
 -1.01244599e-01, -5.43020405e-02, -1.78013876e-01,
 -8.90516266e-02, -7.82444105e-02, -1.37962447e-03,
 -1.11006135e-02],
[ 1.93567634e-01, -1.02017030e-01,  8.01750049e-02,
 -1.41152188e-01,  4.16703671e-02,  1.05722181e-01,
 -8.56285822e-03,  1.06313519e-01,  1.55670434e-01,
  7.97395781e-02],
[-1.79633901e-01,  6.19485006e-02,  9.58212316e-02,
  1.63526833e-01,  5.54828309e-02,  2.19114915e-01,
  1.06328182e-01, -9.44539458e-02, -2.35075042e-01,
  1.27239719e-01],
[-1.29668713e-01, -3.99054261e-03,  7.56797493e-02,
  7.32327178e-02, -2.81935837e-02,  1.28618211e-01,
  6.44155368e-02,  1.73782587e-01, -2.44670630e-01,
 -1.90086931e-01],
[-6.89747781e-02, -3.45480368e-02, -1.16595291e-01,
  1.95623547e-01, -3.19558084e-02, -2.13607758e-01,
  2.10997447e-01, -2.04676971e-01,  9.83275399e-02,
  1.82778701e-01],
[ 1.59631759e-01,  9.98657346e-02, -1.21866368e-01,
  1.29091278e-01,  1.96866781e-01,  1.00930460e-01,
  2.33011514e-01, -4.69362997e-02, -1.46886736e-01,
  1.87363639e-01],
[-8.96509364e-02, -2.19269082e-01,  2.27665484e-01,
 -5.44124097e-02,  1.13135250e-02,  1.11978032e-01,
  1.56652272e-01, -2.21906498e-01, -2.23413587e-01,
  8.44104681e-03],
[-1.50867581e-01, -1.05424270e-01, -2.19464377e-01,
  1.35977045e-01,  2.13329330e-01, -2.66478937e-02,
 -2.19280377e-01, -1.64828807e-01,  4.33615129e-03,
 -1.90556601e-01],
[-1.96624398e-01,  1.04142189e-01, -1.16013810e-01,
 -1.56039104e-01, -3.54049243e-02, -3.49651203e-02,
  8.96516722e-03,  1.19228050e-01,  1.98984399e-01,
 -1.50742635e-01],
[ 4.17317003e-02,  5.09741455e-02, -1.49660587e-01,
  1.57215461e-01, -9.87041891e-02, -2.11864978e-01,
  2.39342041e-02,  4.57957797e-02,  9.26626381e-03,
  1.53222093e-02],
[ 5.05196564e-02, -5.00003882e-02,  5.32541983e-02,
 -1.26919836e-01, -1.26776427e-01, -3.77197452e-02,
  7.34182149e-02, -5.60886879e-03,  1.69007480e-01,
 -1.32983714e-01],
[ 5.31279743e-02,  1.67456433e-01,  1.62191093e-01,
 -1.20113894e-01,  1.07594505e-01,  2.03772590e-01,
  1.51919931e-01,  9.31045264e-02,  1.78999737e-01,
 -5.38816601e-02],
[ 8.52167234e-02,  1.65149391e-01, -5.68589615e-03,
 -1.13942571e-01,  8.59099701e-02,  4.53307619e-03,
 -8.82947296e-02, -1.86732322e-01,  1.06273167e-01,
  8.34852383e-02],
[-2.32059747e-01,  1.39607675e-02,  5.91135323e-02,
 -1.69907302e-01,  5.54422848e-03,  2.17900410e-01,
 -2.37531131e-04, -1.15664445e-01,  1.83271408e-01,
 -8.91774893e-02],
[ 5.81536628e-02, -9.17328745e-02,  2.37501860e-01,
  1.29425466e-01, -1.73836231e-01,  6.48890212e-02,
  2.52768379e-02,  1.10269323e-01,  3.80763523e-02,
```

```
        -1.23445205e-01],
       [ 1.59719169e-01,  6.02798015e-02, -6.54837990e-04,
        -2.07701445e-01,  1.89850628e-01, -9.07112285e-02,
        -2.23674685e-01, -7.89844766e-02,  1.86379999e-01,
        -1.70531690e-01],
       [-1.45055592e-01, -1.19173259e-01, -3.26154493e-02,
        -8.64537656e-02, -2.25976586e-01,  2.34121814e-01,
         1.55488299e-02, -5.52822985e-02,  1.43192187e-01,
        -7.06197172e-02],
       [ 5.59149310e-02,  8.24436992e-02,  2.11136520e-01,
        -5.42201959e-02, -1.73107982e-01, -2.62913518e-02,
         1.96384117e-01,  1.62362278e-01,  1.42635569e-01,
        -9.57568139e-02],
       [ 1.10225506e-01,  2.16709599e-01,  7.48195052e-02,
         9.09849163e-03,  6.17608726e-02, -2.26023514e-02,
         8.42185244e-02,  5.15433401e-02,  9.12305638e-02,
         1.08450465e-02],
       [ 1.94465414e-01,  1.36124760e-01, -3.30568068e-02,
         5.11773974e-02, -3.03890500e-02,  1.93449825e-01,
        -1.73712030e-01,  2.48951185e-02,  7.81934708e-02,
         2.12883145e-01],
       [ 1.57419086e-01,  1.92391321e-01, -2.11527273e-01,
         2.25246146e-01, -1.68393373e-01, -4.92003805e-04,
        -1.77235186e-01,  1.51380822e-01, -5.04184887e-02,
         5.70139773e-02],
       [-2.05899730e-01, -7.74102807e-02,  8.39846358e-02,
         2.32753139e-02, -2.15997174e-01,  1.20145991e-01,
        -1.48164749e-01, -2.11724769e-02,  5.20399259e-03,
        -4.15523835e-02],
       [ 6.17307350e-02,  9.14235339e-02,  2.37127215e-01,
         1.47684053e-01,  1.64933652e-01, -1.16649615e-02,
        -1.73216462e-01,  1.96742415e-01, -1.90782491e-02,
         2.03018174e-01],
       [-6.49842098e-02, -1.58559382e-01,  1.49389639e-01,
        -1.76752344e-01,  1.96529895e-01, -7.44444225e-03,
         2.21399710e-01, -6.68098107e-02, -1.84112750e-02,
        -7.63881058e-02],
       [ 2.06597298e-02,  8.56355056e-02,  1.33035138e-01,
        -3.82651351e-02, -6.97501078e-02,  1.60673842e-01,
         5.59273660e-02,  1.99145496e-01, -1.97175834e-02,
        -2.09477127e-01],
       [ 1.44274786e-01, -1.55999631e-01,  1.87646717e-01,
         1.32009685e-01, -1.79404542e-01, -2.28323609e-01,
         1.50635183e-01, -8.46268982e-02,  2.03099057e-01,
        -1.68427110e-01],
       [-1.14200741e-01, -1.19780049e-01,  1.13191837e-02,
         1.81100458e-01, -2.29950324e-02,  2.16628060e-01,
        -3.11034992e-02, -1.03695378e-01,  1.63326263e-01,
         2.43851647e-01],
       [ 1.79769307e-01, -2.07529098e-01, -1.74490303e-01,
         1.72000080e-01,  1.17399767e-01,  2.19439164e-01,
         1.94360629e-01, -3.11093610e-02, -2.32090041e-01,
        -9.36900675e-02],
       [ 1.73783168e-01,  1.82291478e-01, -7.99251124e-02,
         6.02710284e-02, -1.63491040e-01,  2.43921336e-02,
         2.72583850e-02,  8.38315487e-03, -2.04658344e-01,
        -2.03172982e-01],
       [-9.05466899e-02,  1.88681126e-01, -6.02999628e-02,
         1.01916447e-01, -6.13018908e-02, -2.91665755e-02,
         9.69050266e-03,  1.35072947e-01, -7.93587342e-02,
        -4.03158627e-02],
       [ 5.06908856e-02,  8.33238661e-02,  1.08718216e-01,
         1.15325123e-01,  1.10233903e-01, -1.85964093e-01,
         1.56882018e-01, -2.90827863e-02, -2.45782942e-01,
        -1.78921688e-02],
       [-2.16479093e-01, -1.83915034e-01, -9.66709554e-02,
        -8.76713991e-02, -9.14253518e-02, -1.81790888e-01,
        -2.32988670e-01,  2.10972205e-01,  1.69067502e-01,
         5.45356609e-02],
       [-1.41264424e-01, -9.82937515e-02,  6.11278228e-02,
         1.31071702e-01, -1.89836875e-01, -2.07336664e-01,
        -6.69605136e-02,  1.98829010e-01,  3.79522070e-02,
        -1.70948487e-02],
       [ 7.53171444e-02,  6.75166398e-02,  2.22250298e-01,
        -8.15955698e-02,  1.92713559e-01,  3.16592911e-03,
        -1.63548440e-02,  1.47444934e-01, -2.30998039e-01,
         1.88394830e-01],
```

```
       [ 1.52545735e-01,  8.53811204e-02, -1.58583865e-01,
        -4.44746837e-02, -1.66861460e-01,  1.30746767e-01,
        -2.29634732e-01, -2.02336431e-01,  1.68020397e-01,
        -1.76888496e-01],
       [ 6.90570176e-02, -1.01847164e-01, -9.57621261e-02,
         3.59967276e-02,  7.88589865e-02, -1.88733444e-01,
        -4.27982444e-03, -7.76536912e-02, -1.48945123e-01,
        -3.59744802e-02],
       [ 1.88101754e-01,  1.78803518e-01, -1.25421152e-01,
        -1.42293334e-01,  1.52814627e-01,  4.87945266e-02,
         3.67128756e-03,  9.64145735e-02,  1.21536262e-01,
        -1.86651293e-02],
       [-2.00441137e-01,  1.09475497e-02,  1.91591382e-01,
        -2.02872530e-02, -1.74344674e-01, -1.47463828e-01,
        -4.32011783e-02,  2.09937483e-01, -6.15053214e-02,
         8.33559930e-02],
       [ 1.49200042e-03,  4.39879224e-02,  1.95150927e-01,
         1.52588263e-01, -2.00027630e-01, -2.00106382e-01,
        -1.99284703e-02, -1.22515626e-01, -6.88749477e-02,
         2.25193396e-01],
       [-1.80103883e-01, -1.80622078e-02, -2.24683478e-01,
        -1.63751587e-01, -8.44132230e-02, -1.36427373e-01,
         2.18938962e-01, -2.12646388e-02, -6.88903108e-02,
        -1.59234762e-01],
       [ 1.96071371e-01,  2.30152249e-01,  1.06131919e-01,
         4.26700562e-02, -1.13926545e-01,  1.12161681e-01,
        -1.77948773e-01, -1.77769765e-01, -1.05614997e-01,
         2.98326649e-02],
       [-1.69525623e-01,  1.29407912e-01, -9.86331478e-02,
        -5.39815202e-02,  9.63462815e-02, -2.21721381e-01,
        -6.98978901e-02,  1.95514843e-01, -6.65930510e-02,
        -1.95353061e-01],
       [ 1.73759758e-02,  2.24277392e-01, -1.04652837e-01,
        -1.49774730e-01, -1.45905674e-01,  1.02964424e-01,
         5.02364784e-02,  1.43772975e-01, -1.91643015e-02,
         5.90604767e-02],
       [-2.18003556e-01,  1.61794927e-02, -2.22603589e-01,
        -1.15603797e-01,  2.04896227e-01, -2.20345691e-01,
         1.89761780e-02, -2.10448489e-01, -1.78432912e-01,
         3.57535854e-02],
       [-4.76144366e-02,  1.18110865e-01, -1.11381583e-01,
         2.87946202e-02, -1.69419169e-01,  1.37650996e-01,
         1.35546893e-01, -2.27601916e-01,  1.15548618e-01,
        -1.55999795e-01]], dtype=float32), array([-0.0140571 ,  0.00111038,  0.0127641 ,
0.0303174 , -0.00922577,
        0.00487374,  0.00020527, -0.00718112, -0.03716082,  0.01835428],
      dtype=float32)]
flattened: [[ 0.00424895]
 [-0.00761683]
 [ 0.03843338]
 ...
 [-0.00642902]
 [ 0.01188784]
 [-0.06732762]]
```

In [48]:

```
fig = plt.figure()
plt.title("Weight matrices after model trained")
plt.subplot(1, 3, 1)
plt.title("Trained model Weights")
ax = sns.violinplot(y=h1_w,color='b')
plt.xlabel('Hidden Layer 1')

plt.subplot(1, 3, 2)
plt.title("Trained model Weights")
ax = sns.violinplot(y=h2_w, color='r')
plt.xlabel('Hidden Layer 2 ')

plt.subplot(1, 3, 3)
plt.title("Trained model Weights")
ax = sns.violinplot(y=out_w,color='y')
plt.xlabel('Output Layer ')
plt.show()
```



In [49]:

```
model_sigmoid = Sequential()
model_sigmoid.add(Dense(700, activation='sigmoid', input_shape=(input_dim,)))
model_sigmoid.add(Dense(600, activation='sigmoid'))
model_sigmoid.add(Dense(500, activation='sigmoid'))
model_sigmoid.add(Dense(400, activation='sigmoid'))
model_sigmoid.add(Dense(300, activation='sigmoid'))
model_sigmoid.add(Dense(200, activation='sigmoid'))
model_sigmoid.add(Dense(100, activation='sigmoid'))
model_sigmoid.add(Dense(output_dim, activation='softmax'))

model_sigmoid.add(Dense(output_dim, activation='softmax'))

model_sigmoid.summary()

model_sigmoid.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model_sigmoid.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))
```

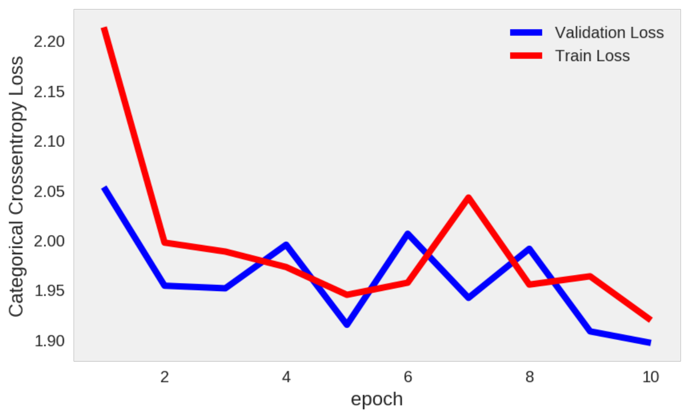
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_13 (Dense)             (None, 700)               549500    
_________________________________________________________________
dense_14 (Dense)             (None, 600)               420600    
_________________________________________________________________
dense_15 (Dense)             (None, 500)               300500    
_________________________________________________________________
dense_16 (Dense)             (None, 400)               200400    
_________________________________________________________________
dense_17 (Dense)             (None, 300)               120300    
_________________________________________________________________
dense_18 (Dense)             (None, 200)               60200     
_________________________________________________________________
dense_19 (Dense)             (None, 100)               20100     
_________________________________________________________________
dense_20 (Dense)             (None, 10)                1010      
_________________________________________________________________
dense_21 (Dense)             (None, 10)                110       
=================================================================
Total params: 1,672,720
Trainable params: 1,672,720
Non-trainable params: 0
_________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 7s 113us/step - loss: 2.2132 - acc: 0.1494 -
val_loss: 2.0529 - val_acc: 0.1914
Epoch 2/10
60000/60000 [==============================] - 6s 101us/step - loss: 1.9973 - acc: 0.2063 -
val_loss: 1.9541 - val_acc: 0.2090
Epoch 3/10
60000/60000 [==============================] - 6s 100us/step - loss: 1.9883 - acc: 0.2019 -
val_loss: 1.9515 - val_acc: 0.2095
Epoch 4/10
60000/60000 [==============================] - 6s 103us/step - loss: 1.9728 - acc: 0.2053 -
val_loss: 1.9952 - val_acc: 0.1994
Epoch 5/10
```

```
60000/60000 [==============================] - 6s 101us/step - loss: 1.9449 - acc: 0.2073 -
val_loss: 1.9149 - val_acc: 0.2079
Epoch 6/10
60000/60000 [==============================] - 6s 103us/step - loss: 1.9571 - acc: 0.2038 -
val_loss: 2.0062 - val_acc: 0.2036
Epoch 7/10
60000/60000 [==============================] - 6s 103us/step - loss: 2.0425 - acc: 0.2015 -
val_loss: 1.9419 - val_acc: 0.2087
Epoch 8/10
60000/60000 [==============================] - 6s 103us/step - loss: 1.9552 - acc: 0.2049 -
val_loss: 1.9911 - val_acc: 0.1831
Epoch 9/10
60000/60000 [==============================] - 6s 103us/step - loss: 1.9635 - acc: 0.2073 -
val_loss: 1.9083 - val_acc: 0.2100
Epoch 10/10
60000/60000 [==============================] - 6s 103us/step - loss: 1.9195 - acc: 0.2112 -
val_loss: 1.8967 - val_acc: 0.2109
```

In [50]:

```
score = model_sigmoid.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
Test score: 1.896732205581665
Test accuracy: 0.2109
```



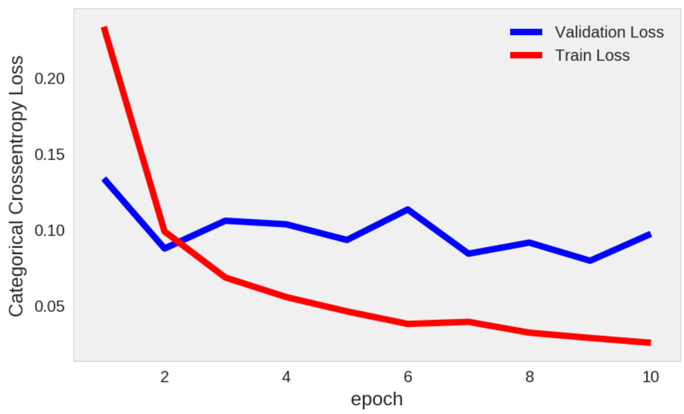
In [52]:

```
# Multilayer perceptron

# https://arxiv.org/pdf/1707.09725.pdf#page=95
# for relu layers
# If we sample weights from a normal distribution N(0,σ) we satisfy this condition with
σ=√(2/(ni).
```

```
# h1 =>  σ=√(2/(fan_in) = 0.062  => N(0,σ) = N(0,0.062)
# h2 =>  σ=√(2/(fan_in) = 0.125  => N(0,σ) = N(0,0.125)
# out =>  σ=√(2/(fan_in+1) = 0.120  => N(0,σ) = N(0,0.120)

model_relu = Sequential()
model_relu.add(Dense(700, activation='relu', input_shape=(input_dim,), kernel_initializer=RandomNor
mal(mean=0.0, stddev=0.053, seed=None)))
model_relu.add(Dense(600, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.057
, seed=None)))
model_relu.add(Dense(500, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.063
, seed=None)))
model_relu.add(Dense(400, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.070
, seed=None)))
model_relu.add(Dense(300, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.081
, seed=None)))
model_relu.add(Dense(200, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.1,
seed=None)))
model_relu.add(Dense(100, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.141
, seed=None)) )
model_relu.add(Dense(output_dim, activation='softmax'))

model_relu.summary()


model_relu.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model_relu.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, valid
ation_data=(X_test, Y_test))


score = model_relu.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_30 (Dense)             (None, 700)               549500    
_________________________________________________________________
dense_31 (Dense)             (None, 600)               420600    
_________________________________________________________________
dense_32 (Dense)             (None, 500)               300500    
_________________________________________________________________
dense_33 (Dense)             (None, 400)               200400    
_________________________________________________________________
dense_34 (Dense)             (None, 300)               120300    
_________________________________________________________________
dense_35 (Dense)             (None, 200)               60200     
_________________________________________________________________
dense_36 (Dense)             (None, 100)               20100     
_________________________________________________________________
dense_37 (Dense)             (None, 10)                1010      
=================================================================
Total params: 1,672,610
```

```
Total params: 1,672,610
Trainable params: 1,672,610
Non-trainable params: 0
_________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 7s 115us/step - loss: 0.2335 - acc: 0.9282 -
val_loss: 0.1335 - val_acc: 0.9578
Epoch 2/10
60000/60000 [==============================] - 6s 99us/step - loss: 0.0985 - acc: 0.9701 -
val_loss: 0.0873 - val_acc: 0.9723
Epoch 3/10
60000/60000 [==============================] - 6s 100us/step - loss: 0.0682 - acc: 0.9793 -
val_loss: 0.1056 - val_acc: 0.9722
Epoch 4/10
60000/60000 [==============================] - 6s 98us/step - loss: 0.0553 - acc: 0.9827 -
val_loss: 0.1033 - val_acc: 0.9704
Epoch 5/10
60000/60000 [==============================] - 6s 103us/step - loss: 0.0458 - acc: 0.9865 -
val_loss: 0.0929 - val_acc: 0.9753
Epoch 6/10
60000/60000 [==============================] - 6s 103us/step - loss: 0.0375 - acc: 0.9892 -
val_loss: 0.1130 - val_acc: 0.9737
Epoch 7/10
60000/60000 [==============================] - 6s 102us/step - loss: 0.0389 - acc: 0.9887 -
val_loss: 0.0839 - val_acc: 0.9772
Epoch 8/10
60000/60000 [==============================] - 6s 99us/step - loss: 0.0318 - acc: 0.9910 -
val_loss: 0.0912 - val_acc: 0.9778
Epoch 9/10
60000/60000 [==============================] - 6s 100us/step - loss: 0.0283 - acc: 0.9917 -
val_loss: 0.0792 - val_acc: 0.9805
Epoch 10/10
60000/60000 [==============================] - 6s 101us/step - loss: 0.0251 - acc: 0.9928 -
val_loss: 0.0970 - val_acc: 0.9791
Test score: 0.09696072832358123
Test accuracy: 0.9791
```



In [53]:

```
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])
```

```
Test score: 0.2991245568871498
Test accuracy: 0.9164
```

The model is clearly overfitting. So, let us do batch normalisation

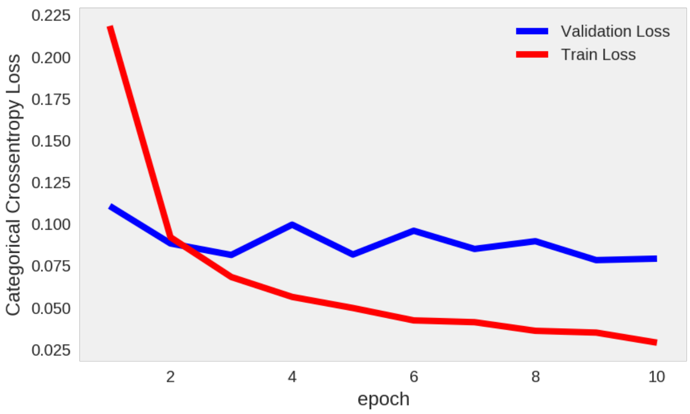
In [54]:

```
# Multilayer perceptron
# https://arxiv.org/pdf/1707.09725.pdf#page=95
# for relu layers
# If we sample weights from a normal distribution N(0,σ) we satisfy this condition with
σ=√(2/(ni).
# h1 =>  σ=√(2/(fan_in) = 0.062  => N(0,σ) = N(0,0.062)
# h2 =>  σ=√(2/(fan_in) = 0.125  => N(0,σ) = N(0,0.125)
# out =>  σ=√(2/(fan_in+1) = 0.120  => N(0,σ) = N(0,0.120)
```

```
# out => σ=√(2/(fan_in+1) = 0.120 => N(0,σ) = N(0,0.120)
from keras.layers.normalization import BatchNormalization


model_relu = Sequential()

model_relu.add(Dense(700, activation='relu', input_shape=(input_dim,), kernel_initializer=RandomNor
mal(mean=0.0, stddev=0.053, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(600, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.057
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(500, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.063
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(400, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.070
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(300, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.081
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(200, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.1,
seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(100, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.141
, seed=None)) )
model_relu.add(BatchNormalization())
model_relu.add(Dense(output_dim, activation='softmax'))

model_relu.summary()


model_relu.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model_relu.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, valid
ation_data=(X_test, Y_test))

score = model_relu.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_38 (Dense)             (None, 700)               549500    
_________________________________________________________________
batch_normalization_1 (Batch (None, 700)               2800      
_________________________________________________________________
dense_39 (Dense)             (None, 600)               420600    
_________________________________________________________________
batch_normalization_2 (Batch (None, 600)               2400      
_________________________________________________________________
dense_40 (Dense)             (None, 500)               300500    
```

```
_________________________________________________________________
batch_normalization_3 (Batch (None, 500)               2000      
_________________________________________________________________
dense_41 (Dense)             (None, 400)               200400    
_________________________________________________________________
batch_normalization_4 (Batch (None, 400)               1600      
_________________________________________________________________
dense_42 (Dense)             (None, 300)               120300    
_________________________________________________________________
batch_normalization_5 (Batch (None, 300)               1200      
_________________________________________________________________
dense_43 (Dense)             (None, 200)               60200     
_________________________________________________________________
batch_normalization_6 (Batch (None, 200)               800       
_________________________________________________________________
dense_44 (Dense)             (None, 100)               20100     
_________________________________________________________________
batch_normalization_7 (Batch (None, 100)               400       
_________________________________________________________________
dense_45 (Dense)             (None, 10)                1010      
=================================================================
Total params: 1,683,810
Trainable params: 1,678,210
Non-trainable params: 5,600
_________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 16s 262us/step - loss: 0.2183 - acc: 0.9338 - val_l
oss: 0.1106 - val_acc: 0.9674
Epoch 2/10
60000/60000 [==============================] - 14s 226us/step - loss: 0.0919 - acc: 0.9712 - val_l
oss: 0.0882 - val_acc: 0.9724
Epoch 3/10
60000/60000 [==============================] - 14s 227us/step - loss: 0.0681 - acc: 0.9789 - val_l
oss: 0.0814 - val_acc: 0.9745
Epoch 4/10
60000/60000 [==============================] - 14s 229us/step - loss: 0.0562 - acc: 0.9820 - val_l
oss: 0.0995 - val_acc: 0.9700
Epoch 5/10
60000/60000 [==============================] - 14s 237us/step - loss: 0.0496 - acc: 0.9844 - val_l
oss: 0.0816 - val_acc: 0.9764
Epoch 6/10
60000/60000 [==============================] - 15s 244us/step - loss: 0.0422 - acc: 0.9861 - val_l
oss: 0.0958 - val_acc: 0.9740
Epoch 7/10
60000/60000 [==============================] - 15s 243us/step - loss: 0.0411 - acc: 0.9866 - val_l
oss: 0.0849 - val_acc: 0.9773
Epoch 8/10
60000/60000 [==============================] - 14s 237us/step - loss: 0.0360 - acc: 0.9883 - val_l
oss: 0.0896 - val_acc: 0.9756
Epoch 9/10
60000/60000 [==============================] - 14s 227us/step - loss: 0.0349 - acc: 0.9885 - val_l
oss: 0.0782 - val_acc: 0.9779
Epoch 10/10
60000/60000 [==============================] - 13s 223us/step - loss: 0.0289 - acc: 0.9904 - val_l
oss: 0.0792 - val_acc: 0.9798
Test score: 0.07915055781649426
Test accuracy: 0.9798
```



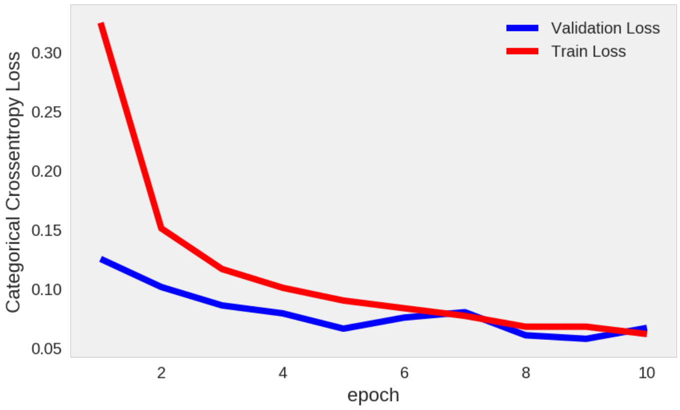
Adding Dropout

In [55]:

```
# Multilayer perceptron
# https://arxiv.org/pdf/1707.09725.pdf#page=95
# for relu layers
# If we sample weights from a normal distribution N(0,σ) we satisfy this condition with
σ=√(2/(ni).
# h1 =>  σ=√(2/(fan_in) = 0.062  => N(0,σ) = N(0,0.062)
# h2 =>  σ=√(2/(fan_in) = 0.125  => N(0,σ) = N(0,0.125)
# out =>  σ=√(2/(fan_in+1) = 0.120  => N(0,σ) = N(0,0.120)
from keras.layers.normalization import BatchNormalization
from keras.layers import Dropout


model_relu = Sequential()

model_relu.add(Dense(700, activation='relu', input_shape=(input_dim,), kernel_initializer=RandomNor
mal(mean=0.0, stddev=0.053, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dropout(0.5))
model_relu.add(Dense(600, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.057
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(500, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.063
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(400, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.070
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(300, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.081
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(200, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.1,
seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(100, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.141
, seed=None)) )
model_relu.add(BatchNormalization())
model_relu.add(Dense(output_dim, activation='softmax'))

model_relu.summary()


model_relu.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model_relu.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, valid
ation_data=(X_test, Y_test))

score = model_relu.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-
packages/keras/backend/tensorflow_backend.py:3445: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future
version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_46 (Dense)             (None, 700)               549500    
_________________________________________________________________
batch_normalization_8 (Batch (None, 700)               2800      
_________________________________________________________________
dropout_1 (Dropout)          (None, 700)               0         
_________________________________________________________________
dense_47 (Dense)             (None, 600)               420600    
_________________________________________________________________
batch_normalization_9 (Batch (None, 600)               2400      
_________________________________________________________________
dense_48 (Dense)             (None, 500)               300500    
_________________________________________________________________
batch_normalization_10 (Batc (None, 500)               2000      
_________________________________________________________________
dense_49 (Dense)             (None, 400)               200400    
_________________________________________________________________
batch_normalization_11 (Batc (None, 400)               1600      
_________________________________________________________________
dense_50 (Dense)             (None, 300)               120300    
_________________________________________________________________
batch_normalization_12 (Batc (None, 300)               1200      
_________________________________________________________________
dense_51 (Dense)             (None, 200)               60200     
_________________________________________________________________
batch_normalization_13 (Batc (None, 200)               800       
_________________________________________________________________
dense_52 (Dense)             (None, 100)               20100     
_________________________________________________________________
batch_normalization_14 (Batc (None, 100)               400       
_________________________________________________________________
dense_53 (Dense)             (None, 10)                1010      
=================================================================
Total params: 1,683,810
Trainable params: 1,678,210
Non-trainable params: 5,600
_________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 17s 279us/step - loss: 0.3247 - acc: 0.8996 - val_l
oss: 0.1250 - val_acc: 0.9622
Epoch 2/10
60000/60000 [==============================] - 14s 231us/step - loss: 0.1508 - acc: 0.9534 - val_l
oss: 0.1012 - val_acc: 0.9676
Epoch 3/10
60000/60000 [==============================] - 13s 225us/step - loss: 0.1164 - acc: 0.9633 - val_l
oss: 0.0856 - val_acc: 0.9731
Epoch 4/10
60000/60000 [==============================] - 13s 222us/step - loss: 0.1005 - acc: 0.9677 - val_l
oss: 0.0789 - val_acc: 0.9737
Epoch 5/10
60000/60000 [==============================] - 13s 225us/step - loss: 0.0897 - acc: 0.9718 - val_l
oss: 0.0659 - val_acc: 0.9794
Epoch 6/10
60000/60000 [==============================] - 14s 227us/step - loss: 0.0832 - acc: 0.9737 - val_l
oss: 0.0753 - val_acc: 0.9778
Epoch 7/10
60000/60000 [==============================] - 14s 238us/step - loss: 0.0768 - acc: 0.9749 - val_l
oss: 0.0799 - val_acc: 0.9764
Epoch 8/10
60000/60000 [==============================] - 15s 242us/step - loss: 0.0676 - acc: 0.9781 - val_l
oss: 0.0603 - val_acc: 0.9804
Epoch 9/10
60000/60000 [==============================] - 14s 240us/step - loss: 0.0676 - acc: 0.9780 - val_l
oss: 0.0573 - val_acc: 0.9826
Epoch 10/10
60000/60000 [==============================] - 14s 231us/step - loss: 0.0614 - acc: 0.9796 - val_l
oss: 0.0666 - val_acc: 0.9793
Test score: 0.06658548916075378
```

```
Test accuracy: 0.9793
```



In [0]:

```
from keras.optimizers import Adam,RMSprop,SGD
def best_hyperparameters(activ, optimizer):

    model = Sequential()
    model.add(Dense(512, activation=activ, input_shape=(input_dim,), kernel_initializer=RandomNorma
l(mean=0.0, stddev=0.062, seed=None)))
    model.add(Dropout(0.5))
    model.add(Dense(128, activation=activ, kernel_initializer=RandomNormal(mean=0.0, stddev=0.125,
seed=None)) )
    model.add(Dense(output_dim, activation='softmax'))


    model.compile(loss='categorical_crossentropy', metrics=['accuracy'], optimizer = optimizer)

    return model
```

In [69]:

```
# https://machinelearningmastery.com/grid-search-hyperparameters-deep-learning-models-python-keras
/

activ = ['sigmoid','relu']
optimizer = ['SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', 'Nadam']

from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import GridSearchCV

model = KerasClassifier(build_fn=best_hyperparameters, epochs=nb_epoch, batch_size=batch_size, verb
ose=0)
param_grid = dict(activ=activ, optimizer = optimizer)


# if you are using CPU
# grid = GridSearchCV(estimator=model, param_grid=param_grid, n_jobs=-1)
# if you are using GPU dont use the n_jobs parameter

grid = GridSearchCV(estimator=model, param_grid=param_grid)
grid_result = grid.fit(X_train, Y_train)

print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
    print("%f (%f) with: %r" % (mean, stdev, param))
```

```
Best: 0.977250 using {'activ': 'relu', 'optimizer': 'RMSprop'}
0.766750 (0.012041) with: {'activ': 'sigmoid', 'optimizer': 'SGD'}
0.966733 (0.000997) with: {'activ': 'sigmoid', 'optimizer': 'RMSprop'}
0.945483 (0.001336) with: {'activ': 'sigmoid', 'optimizer': 'Adagrad'}
0.949500 (0.001073) with: {'activ': 'sigmoid', 'optimizer': 'Adadelta'}
0.968433 (0.001700) with: {'activ': 'sigmoid', 'optimizer': 'Adam'}
0.957700 (0.000534) with: {'activ': 'sigmoid', 'optimizer': 'Adamax'}
0.973700 (0.001592) with: {'activ': 'sigmoid', 'optimizer': 'Nadam'}
0.931167 (0.002786) with: {'activ': 'relu', 'optimizer': 'SGD'}
```

```
0.977250 (0.001431) with: {'activ': 'relu', 'optimizer': 'RMSprop'}
0.972800 (0.001296) with: {'activ': 'relu', 'optimizer': 'Adagrad'}
0.975817 (0.001319) with: {'activ': 'relu', 'optimizer': 'Adadelta'}
0.976650 (0.001089) with: {'activ': 'relu', 'optimizer': 'Adam'}
0.975050 (0.001087) with: {'activ': 'relu', 'optimizer': 'Adamax'}
0.975983 (0.000849) with: {'activ': 'relu', 'optimizer': 'Nadam'}
```

i so wanted to try hyperparam tuning, but it took so much time even for a simple model. So, i will skip it for now

Increasing no of epochs to see how my results differ

In [0]:

```
# some model parameters

output_dim = 10
input_dim = X_train.shape[1]

batch_size = 128
nb_epoch = 200
```

with drpoout and batch normalisation

In [71]:

```
# Multilayer perceptron
# https://arxiv.org/pdf/1707.09725.pdf#page=95
# for relu layers
# If we sample weights from a normal distribution N(0,σ) we satisfy this condition with
σ=√(2/(ni).
# h1 =>  σ=√(2/(fan_in) = 0.062  => N(0,σ) = N(0,0.062)
# h2 =>  σ=√(2/(fan_in) = 0.125  => N(0,σ) = N(0,0.125)
# out =>  σ=√(2/(fan_in+1) = 0.120  => N(0,σ) = N(0,0.120)
from keras.layers.normalization import BatchNormalization
from keras.layers import Dropout


model_relu = Sequential()

model_relu.add(Dense(700, activation='relu', input_shape=(input_dim,), kernel_initializer=RandomNor
mal(mean=0.0, stddev=0.053, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dropout(0.5))
model_relu.add(Dense(600, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.057
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(500, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.063
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(400, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.070
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(300, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.081
, seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(200, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.1,
seed=None)))
model_relu.add(BatchNormalization())
model_relu.add(Dense(100, activation='relu', kernel_initializer=RandomNormal(mean=0.0, stddev=0.141
, seed=None)) )
model_relu.add(BatchNormalization())
model_relu.add(Dense(output_dim, activation='softmax'))

model_relu.summary()


model_relu.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model_relu.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, valid
ation_data=(X_test, Y_test))
```

```
score = model_relu.evaluate(X_test, Y_test, verbose=0)
print('Test score:', score[0])
print('Test accuracy:', score[1])

fig,ax = plt.subplots(1,1)
ax.set_xlabel('epoch') ; ax.set_ylabel('Categorical Crossentropy Loss')

# list of epoch numbers
x = list(range(1,nb_epoch+1))

# print(history.history.keys())
# dict_keys(['val_loss', 'val_acc', 'loss', 'acc'])
# history = model_drop.fit(X_train, Y_train, batch_size=batch_size, epochs=nb_epoch, verbose=1, va
lidation_data=(X_test, Y_test))

# we will get val_loss and val_acc only when you pass the paramter validation_data
# val_loss : validation loss
# val_acc : validation accuracy

# loss : training loss
# acc : train accuracy
# for each key in histrory.histrory we will have a list of length equal to number of epochs

vy = history.history['val_loss']
ty = history.history['loss']
plt_dynamic(x, vy, ty, ax)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_193 (Dense)            (None, 700)               549500    
_________________________________________________________________
batch_normalization_15 (Batc (None, 700)               2800      
_________________________________________________________________
dropout_47 (Dropout)         (None, 700)               0         
_________________________________________________________________
dense_194 (Dense)            (None, 600)               420600    
_________________________________________________________________
batch_normalization_16 (Batc (None, 600)               2400      
_________________________________________________________________
dense_195 (Dense)            (None, 500)               300500    
_________________________________________________________________
batch_normalization_17 (Batc (None, 500)               2000      
_________________________________________________________________
dense_196 (Dense)            (None, 400)               200400    
_________________________________________________________________
batch_normalization_18 (Batc (None, 400)               1600      
_________________________________________________________________
dense_197 (Dense)            (None, 300)               120300    
_________________________________________________________________
batch_normalization_19 (Batc (None, 300)               1200      
_________________________________________________________________
dense_198 (Dense)            (None, 200)               60200     
_________________________________________________________________
batch_normalization_20 (Batc (None, 200)               800       
_________________________________________________________________
dense_199 (Dense)            (None, 100)               20100     
_________________________________________________________________
batch_normalization_21 (Batc (None, 100)               400       
_________________________________________________________________
dense_200 (Dense)            (None, 10)                1010      
=================================================================
Total params: 1,683,810
Trainable params: 1,678,210
Non-trainable params: 5,600
_________________________________________________________________
Train on 60000 samples, validate on 10000 samples
Epoch 1/200
60000/60000 [==============================] - 24s 393us/step - loss: 0.3282 - acc: 0.8968 - val_l
oss: 0.1216 - val_acc: 0.9630
Epoch 2/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.1486 - acc: 0.9533 - val_l
oss: 0.0969 - val_acc: 0.9717
Epoch 3/200
60000/60000 [==============================] - 14s 235us/step - loss: 0.1206 - acc: 0.9619 - val_l
oss: 0.0855 - val_acc: 0.9749
Epoch 4/200
```

```
60000/60000 [==============================] - 14s 241us/step - loss: 0.1006 - acc: 0.9680 - val_l
oss: 0.0673 - val_acc: 0.9795
Epoch 5/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0896 - acc: 0.9720 - val_l
oss: 0.0766 - val_acc: 0.9777
Epoch 6/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0849 - acc: 0.9729 - val_l
oss: 0.0915 - val_acc: 0.9733
Epoch 7/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0769 - acc: 0.9748 - val_l
oss: 0.0667 - val_acc: 0.9799
Epoch 8/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0697 - acc: 0.9778 - val_l
oss: 0.0625 - val_acc: 0.9814
Epoch 9/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0645 - acc: 0.9796 - val_l
oss: 0.0713 - val_acc: 0.9804
Epoch 10/200
60000/60000 [==============================] - 14s 236us/step - loss: 0.0606 - acc: 0.9803 - val_l
oss: 0.0653 - val_acc: 0.9812
Epoch 11/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0557 - acc: 0.9819 - val_l
oss: 0.0571 - val_acc: 0.9829
Epoch 12/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0543 - acc: 0.9827 - val_l
oss: 0.0609 - val_acc: 0.9828
Epoch 13/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0530 - acc: 0.9827 - val_l
oss: 0.0621 - val_acc: 0.9825
Epoch 14/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0491 - acc: 0.9841 - val_l
oss: 0.0636 - val_acc: 0.9814
Epoch 15/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0454 - acc: 0.9856 - val_l
oss: 0.0658 - val_acc: 0.9809
Epoch 16/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0430 - acc: 0.9859 - val_l
oss: 0.0666 - val_acc: 0.9814
Epoch 17/200
60000/60000 [==============================] - 14s 235us/step - loss: 0.0414 - acc: 0.9861 - val_l
oss: 0.0591 - val_acc: 0.9817
Epoch 18/200
60000/60000 [==============================] - 14s 236us/step - loss: 0.0413 - acc: 0.9865 - val_l
oss: 0.0711 - val_acc: 0.9796
Epoch 19/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0387 - acc: 0.9872 - val_l
oss: 0.0567 - val_acc: 0.9846
Epoch 20/200
60000/60000 [==============================] - 16s 266us/step - loss: 0.0366 - acc: 0.9876 - val_l
oss: 0.0599 - val_acc: 0.9827
Epoch 21/200
60000/60000 [==============================] - 16s 268us/step - loss: 0.0366 - acc: 0.9876 - val_l
oss: 0.0610 - val_acc: 0.9848
Epoch 22/200
60000/60000 [==============================] - 16s 265us/step - loss: 0.0350 - acc: 0.9882 - val_l
oss: 0.0626 - val_acc: 0.9822
Epoch 23/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0338 - acc: 0.9886 - val_l
oss: 0.0584 - val_acc: 0.9834
Epoch 24/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0315 - acc: 0.9891 - val_l
oss: 0.0576 - val_acc: 0.9828
Epoch 25/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0305 - acc: 0.9901 - val_l
oss: 0.0570 - val_acc: 0.9847
Epoch 26/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0297 - acc: 0.9901 - val_l
oss: 0.0589 - val_acc: 0.9818
Epoch 27/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0268 - acc: 0.9909 - val_l
oss: 0.0583 - val_acc: 0.9854
Epoch 28/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0283 - acc: 0.9905 - val_l
oss: 0.0661 - val_acc: 0.9846
Epoch 29/200
60000/60000 [==============================] - 14s 234us/step - loss: 0.0257 - acc: 0.9916 - val_l
oss: 0.0599 - val_acc: 0.9834
```

```
Epoch 30/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0251 - acc: 0.9919 - val_l
oss: 0.0591 - val_acc: 0.9849
Epoch 31/200
60000/60000 [==============================] - 14s 235us/step - loss: 0.0246 - acc: 0.9917 - val_l
oss: 0.0590 - val_acc: 0.9848
Epoch 32/200
60000/60000 [==============================] - 14s 233us/step - loss: 0.0251 - acc: 0.9917 - val_l
oss: 0.0582 - val_acc: 0.9840
Epoch 33/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0239 - acc: 0.9925 - val_l
oss: 0.0569 - val_acc: 0.9851
Epoch 34/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0223 - acc: 0.9927 - val_l
oss: 0.0609 - val_acc: 0.9844
Epoch 35/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0214 - acc: 0.9929 - val_l
oss: 0.0605 - val_acc: 0.9844
Epoch 36/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0210 - acc: 0.9927 - val_l
oss: 0.0545 - val_acc: 0.9857
Epoch 37/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0171 - acc: 0.9944 - val_l
oss: 0.0642 - val_acc: 0.9852
Epoch 38/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0219 - acc: 0.9927 - val_l
oss: 0.0621 - val_acc: 0.9843
Epoch 39/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0200 - acc: 0.9932 - val_l
oss: 0.0578 - val_acc: 0.9848
Epoch 40/200
60000/60000 [==============================] - 14s 232us/step - loss: 0.0190 - acc: 0.9939 - val_l
oss: 0.0553 - val_acc: 0.9868
Epoch 41/200
60000/60000 [==============================] - 14s 236us/step - loss: 0.0185 - acc: 0.9941 - val_l
oss: 0.0536 - val_acc: 0.9861
Epoch 42/200
60000/60000 [==============================] - 15s 249us/step - loss: 0.0199 - acc: 0.9933 - val_l
oss: 0.0619 - val_acc: 0.9856
Epoch 43/200
60000/60000 [==============================] - 16s 269us/step - loss: 0.0177 - acc: 0.9942 - val_l
oss: 0.0521 - val_acc: 0.9864
Epoch 44/200
60000/60000 [==============================] - 16s 267us/step - loss: 0.0167 - acc: 0.9944 - val_l
oss: 0.0588 - val_acc: 0.9856
Epoch 45/200
60000/60000 [==============================] - 16s 262us/step - loss: 0.0171 - acc: 0.9943 - val_l
oss: 0.0567 - val_acc: 0.9870
Epoch 46/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0155 - acc: 0.9945 - val_l
oss: 0.0657 - val_acc: 0.9840
Epoch 47/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0181 - acc: 0.9943 - val_l
oss: 0.0605 - val_acc: 0.9846
Epoch 48/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0157 - acc: 0.9944 - val_l
oss: 0.0557 - val_acc: 0.9861
Epoch 49/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0156 - acc: 0.9947 - val_l
oss: 0.0582 - val_acc: 0.9848
Epoch 50/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0143 - acc: 0.9951 - val_l
oss: 0.0602 - val_acc: 0.9843
Epoch 51/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0151 - acc: 0.9949 - val_l
oss: 0.0564 - val_acc: 0.9856
Epoch 52/200
60000/60000 [==============================] - 14s 234us/step - loss: 0.0142 - acc: 0.9952 - val_l
oss: 0.0610 - val_acc: 0.9855
Epoch 53/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0161 - acc: 0.9946 - val_l
oss: 0.0557 - val_acc: 0.9857
Epoch 54/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0151 - acc: 0.9950 - val_l
oss: 0.0571 - val_acc: 0.9857
Epoch 55/200
60000/60000 [==============================] - 14s 235us/step - loss: 0.0144 - acc: 0.9950 - val_l
```

```
oss: 0.0572 - val_acc: 0.9859
Epoch 56/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0139 - acc: 0.9953 - val_l
oss: 0.0603 - val_acc: 0.9851
Epoch 57/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0137 - acc: 0.9954 - val_l
oss: 0.0564 - val_acc: 0.9864
Epoch 58/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0129 - acc: 0.9956 - val_l
oss: 0.0620 - val_acc: 0.9849
Epoch 59/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0133 - acc: 0.9956 - val_l
oss: 0.0608 - val_acc: 0.9843
Epoch 60/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0122 - acc: 0.9960 - val_l
oss: 0.0576 - val_acc: 0.9857
Epoch 61/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0122 - acc: 0.9959 - val_l
oss: 0.0605 - val_acc: 0.9869
Epoch 62/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0135 - acc: 0.9953 - val_l
oss: 0.0592 - val_acc: 0.9867
Epoch 63/200
60000/60000 [==============================] - 15s 247us/step - loss: 0.0099 - acc: 0.9969 - val_l
oss: 0.0603 - val_acc: 0.9857
Epoch 64/200
60000/60000 [==============================] - 15s 251us/step - loss: 0.0119 - acc: 0.9961 - val_l
oss: 0.0604 - val_acc: 0.9851
Epoch 65/200
60000/60000 [==============================] - 15s 257us/step - loss: 0.0135 - acc: 0.9956 - val_l
oss: 0.0580 - val_acc: 0.9850
Epoch 66/200
60000/60000 [==============================] - 16s 267us/step - loss: 0.0113 - acc: 0.9961 - val_l
oss: 0.0613 - val_acc: 0.9852
Epoch 67/200
60000/60000 [==============================] - 16s 265us/step - loss: 0.0119 - acc: 0.9960 - val_l
oss: 0.0536 - val_acc: 0.9866
Epoch 68/200
60000/60000 [==============================] - 15s 251us/step - loss: 0.0114 - acc: 0.9965 - val_l
oss: 0.0607 - val_acc: 0.9863
Epoch 69/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0109 - acc: 0.9965 - val_l
oss: 0.0628 - val_acc: 0.9860
Epoch 70/200
60000/60000 [==============================] - 14s 236us/step - loss: 0.0114 - acc: 0.9962 - val_l
oss: 0.0597 - val_acc: 0.9855
Epoch 71/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0115 - acc: 0.9961 - val_l
oss: 0.0585 - val_acc: 0.9863
Epoch 72/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0115 - acc: 0.9962 - val_l
oss: 0.0550 - val_acc: 0.9866
Epoch 73/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0104 - acc: 0.9965 - val_l
oss: 0.0603 - val_acc: 0.9866
Epoch 74/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0096 - acc: 0.9969 - val_l
oss: 0.0649 - val_acc: 0.9857
Epoch 75/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0102 - acc: 0.9966 - val_l
oss: 0.0690 - val_acc: 0.9856
Epoch 76/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0091 - acc: 0.9971 - val_l
oss: 0.0574 - val_acc: 0.9871
Epoch 77/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0109 - acc: 0.9965 - val_l
oss: 0.0579 - val_acc: 0.9873
Epoch 78/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0083 - acc: 0.9974 - val_l
oss: 0.0599 - val_acc: 0.9876
Epoch 79/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0087 - acc: 0.9969 - val_l
oss: 0.0616 - val_acc: 0.9864
Epoch 80/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0110 - acc: 0.9965 - val_l
oss: 0.0555 - val_acc: 0.9871
Epoch 81/200
```

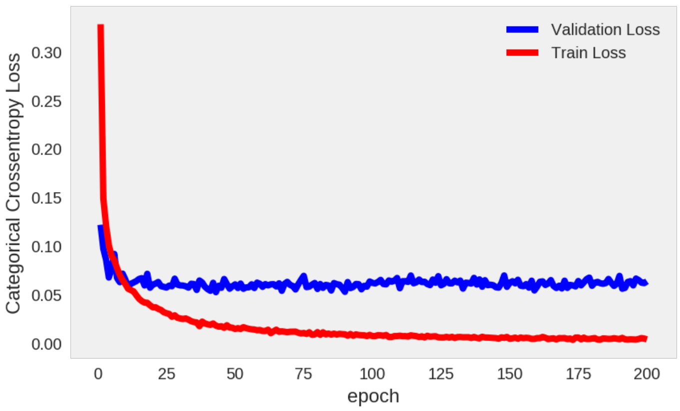
```
Epoch 81/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0082 - acc: 0.9973 - val_l
oss: 0.0606 - val_acc: 0.9866
Epoch 82/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0109 - acc: 0.9964 - val_l
oss: 0.0566 - val_acc: 0.9871
Epoch 83/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0087 - acc: 0.9971 - val_l
oss: 0.0596 - val_acc: 0.9869
Epoch 84/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0095 - acc: 0.9970 - val_l
oss: 0.0591 - val_acc: 0.9863
Epoch 85/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0085 - acc: 0.9973 - val_l
oss: 0.0539 - val_acc: 0.9878
Epoch 86/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0095 - acc: 0.9967 - val_l
oss: 0.0616 - val_acc: 0.9865
Epoch 87/200
60000/60000 [==============================] - 16s 261us/step - loss: 0.0086 - acc: 0.9973 - val_l
oss: 0.0604 - val_acc: 0.9873
Epoch 88/200
60000/60000 [==============================] - 16s 270us/step - loss: 0.0093 - acc: 0.9971 - val_l
oss: 0.0599 - val_acc: 0.9861
Epoch 89/200
60000/60000 [==============================] - 16s 265us/step - loss: 0.0089 - acc: 0.9970 - val_l
oss: 0.0565 - val_acc: 0.9862
Epoch 90/200
60000/60000 [==============================] - 15s 253us/step - loss: 0.0088 - acc: 0.9970 - val_l
oss: 0.0525 - val_acc: 0.9876
Epoch 91/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0074 - acc: 0.9975 - val_l
oss: 0.0626 - val_acc: 0.9855
Epoch 92/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0091 - acc: 0.9971 - val_l
oss: 0.0563 - val_acc: 0.9875
Epoch 93/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0073 - acc: 0.9975 - val_l
oss: 0.0575 - val_acc: 0.9874
Epoch 94/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0087 - acc: 0.9972 - val_l
oss: 0.0606 - val_acc: 0.9863
Epoch 95/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0081 - acc: 0.9973 - val_l
oss: 0.0604 - val_acc: 0.9864
Epoch 96/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0078 - acc: 0.9973 - val_l
oss: 0.0551 - val_acc: 0.9862
Epoch 97/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0078 - acc: 0.9975 - val_l
oss: 0.0587 - val_acc: 0.9865
Epoch 98/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0069 - acc: 0.9975 - val_l
oss: 0.0578 - val_acc: 0.9869
Epoch 99/200
60000/60000 [==============================] - 14s 235us/step - loss: 0.0082 - acc: 0.9974 - val_l
oss: 0.0632 - val_acc: 0.9858
Epoch 100/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0069 - acc: 0.9976 - val_l
oss: 0.0618 - val_acc: 0.9879
Epoch 101/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0069 - acc: 0.9977 - val_l
oss: 0.0612 - val_acc: 0.9867
Epoch 102/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0079 - acc: 0.9975 - val_l
oss: 0.0626 - val_acc: 0.9856
Epoch 103/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0077 - acc: 0.9976 - val_l
oss: 0.0648 - val_acc: 0.9856
Epoch 104/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0072 - acc: 0.9974 - val_l
oss: 0.0606 - val_acc: 0.9865
Epoch 105/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0081 - acc: 0.9974 - val_l
oss: 0.0602 - val_acc: 0.9872
Epoch 106/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0060 - acc: 0.9982 - val_l
oss: 0.0643 - val_acc: 0.9869
```

```
oss: 0.0645 - val_acc: 0.9869
Epoch 107/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0060 - acc: 0.9980 - val_l
oss: 0.0629 - val_acc: 0.9870
Epoch 108/200
60000/60000 [==============================] - 15s 249us/step - loss: 0.0069 - acc: 0.9978 - val_l
oss: 0.0640 - val_acc: 0.9870
Epoch 109/200
60000/60000 [==============================] - 16s 271us/step - loss: 0.0069 - acc: 0.9975 - val_l
oss: 0.0668 - val_acc: 0.9865
Epoch 110/200
60000/60000 [==============================] - 16s 270us/step - loss: 0.0073 - acc: 0.9975 - val_l
oss: 0.0562 - val_acc: 0.9877
Epoch 111/200
60000/60000 [==============================] - 16s 271us/step - loss: 0.0068 - acc: 0.9978 - val_l
oss: 0.0635 - val_acc: 0.9866
Epoch 112/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0069 - acc: 0.9978 - val_l
oss: 0.0634 - val_acc: 0.9872
Epoch 113/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0064 - acc: 0.9982 - val_l
oss: 0.0627 - val_acc: 0.9867
Epoch 114/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0077 - acc: 0.9974 - val_l
oss: 0.0694 - val_acc: 0.9862
Epoch 115/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0072 - acc: 0.9978 - val_l
oss: 0.0613 - val_acc: 0.9863
Epoch 116/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0069 - acc: 0.9976 - val_l
oss: 0.0623 - val_acc: 0.9864
Epoch 117/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0060 - acc: 0.9981 - val_l
oss: 0.0652 - val_acc: 0.9860
Epoch 118/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0067 - acc: 0.9980 - val_l
oss: 0.0626 - val_acc: 0.9865
Epoch 119/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0055 - acc: 0.9983 - val_l
oss: 0.0629 - val_acc: 0.9870
Epoch 120/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0072 - acc: 0.9977 - val_l
oss: 0.0605 - val_acc: 0.9865
Epoch 121/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0063 - acc: 0.9981 - val_l
oss: 0.0594 - val_acc: 0.9873
Epoch 122/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0066 - acc: 0.9978 - val_l
oss: 0.0652 - val_acc: 0.9858
Epoch 123/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0068 - acc: 0.9978 - val_l
oss: 0.0619 - val_acc: 0.9865
Epoch 124/200
60000/60000 [==============================] - 14s 236us/step - loss: 0.0056 - acc: 0.9982 - val_l
oss: 0.0687 - val_acc: 0.9857
Epoch 125/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0055 - acc: 0.9982 - val_l
oss: 0.0592 - val_acc: 0.9878
Epoch 126/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0055 - acc: 0.9983 - val_l
oss: 0.0607 - val_acc: 0.9873
Epoch 127/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0061 - acc: 0.9980 - val_l
oss: 0.0656 - val_acc: 0.9863
Epoch 128/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0053 - acc: 0.9982 - val_l
oss: 0.0613 - val_acc: 0.9875
Epoch 129/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0062 - acc: 0.9981 - val_l
oss: 0.0613 - val_acc: 0.9876
Epoch 130/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0050 - acc: 0.9985 - val_l
oss: 0.0640 - val_acc: 0.9869
Epoch 131/200
60000/60000 [==============================] - 16s 262us/step - loss: 0.0059 - acc: 0.9981 - val_l
oss: 0.0622 - val_acc: 0.9873
Epoch 132/200
60000/60000 [==============================] - 16s 261us/step - loss: 0.0060 - acc: 0.9981 - val_l
```

```
60000/60000 [==============================] - 16s 261us/step - loss: 0.0060 - acc: 0.9981 - val_l
oss: 0.0641 - val_acc: 0.9855
Epoch 133/200
60000/60000 [==============================] - 15s 254us/step - loss: 0.0057 - acc: 0.9983 - val_l
oss: 0.0558 - val_acc: 0.9871
Epoch 134/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0058 - acc: 0.9981 - val_l
oss: 0.0620 - val_acc: 0.9865
Epoch 135/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0058 - acc: 0.9981 - val_l
oss: 0.0617 - val_acc: 0.9875
Epoch 136/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0050 - acc: 0.9985 - val_l
oss: 0.0610 - val_acc: 0.9871
Epoch 137/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0061 - acc: 0.9980 - val_l
oss: 0.0669 - val_acc: 0.9873
Epoch 138/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0050 - acc: 0.9985 - val_l
oss: 0.0600 - val_acc: 0.9865
Epoch 139/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0046 - acc: 0.9986 - val_l
oss: 0.0654 - val_acc: 0.9864
Epoch 140/200
60000/60000 [==============================] - 15s 247us/step - loss: 0.0062 - acc: 0.9977 - val_l
oss: 0.0577 - val_acc: 0.9886
Epoch 141/200
60000/60000 [==============================] - 14s 240us/step - loss: 0.0055 - acc: 0.9981 - val_l
oss: 0.0645 - val_acc: 0.9871
Epoch 142/200
60000/60000 [==============================] - 14s 235us/step - loss: 0.0054 - acc: 0.9981 - val_l
oss: 0.0594 - val_acc: 0.9865
Epoch 143/200
60000/60000 [==============================] - 14s 234us/step - loss: 0.0052 - acc: 0.9981 - val_l
oss: 0.0599 - val_acc: 0.9875
Epoch 144/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0051 - acc: 0.9984 - val_l
oss: 0.0593 - val_acc: 0.9864
Epoch 145/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0049 - acc: 0.9984 - val_l
oss: 0.0573 - val_acc: 0.9875
Epoch 146/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0044 - acc: 0.9986 - val_l
oss: 0.0569 - val_acc: 0.9874
Epoch 147/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0056 - acc: 0.9983 - val_l
oss: 0.0614 - val_acc: 0.9869
Epoch 148/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0051 - acc: 0.9983 - val_l
oss: 0.0694 - val_acc: 0.9860
Epoch 149/200
60000/60000 [==============================] - 14s 238us/step - loss: 0.0061 - acc: 0.9981 - val_l
oss: 0.0576 - val_acc: 0.9870
Epoch 150/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0042 - acc: 0.9986 - val_l
oss: 0.0618 - val_acc: 0.9870
Epoch 151/200
60000/60000 [==============================] - 14s 239us/step - loss: 0.0047 - acc: 0.9986 - val_l
oss: 0.0634 - val_acc: 0.9875
Epoch 152/200
60000/60000 [==============================] - 15s 243us/step - loss: 0.0053 - acc: 0.9984 - val_l
oss: 0.0613 - val_acc: 0.9868
Epoch 153/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0042 - acc: 0.9987 - val_l
oss: 0.0650 - val_acc: 0.9877
Epoch 154/200
60000/60000 [==============================] - 16s 259us/step - loss: 0.0055 - acc: 0.9983 - val_l
oss: 0.0588 - val_acc: 0.9875
Epoch 155/200
60000/60000 [==============================] - 16s 265us/step - loss: 0.0048 - acc: 0.9984 - val_l
oss: 0.0583 - val_acc: 0.9875
Epoch 156/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0052 - acc: 0.9986 - val_l
oss: 0.0605 - val_acc: 0.9866
Epoch 157/200
60000/60000 [==============================] - 14s 241us/step - loss: 0.0049 - acc: 0.9986 - val_l
oss: 0.0568 - val_acc: 0.9871
Epoch 158/200
```

```
Epoch 158/200
60000/60000 [==============================] - 14s 236us/step - loss: 0.0041 - acc: 0.9986 - val_l
oss: 0.0640 - val_acc: 0.9865
Epoch 159/200
60000/60000 [==============================] - 24s 393us/step - loss: 0.0041 - acc: 0.9987 - val_l
oss: 0.0539 - val_acc: 0.9882
Epoch 160/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0049 - acc: 0.9984 - val_l
oss: 0.0579 - val_acc: 0.9870
Epoch 161/200
60000/60000 [==============================] - 15s 247us/step - loss: 0.0047 - acc: 0.9985 - val_l
oss: 0.0629 - val_acc: 0.9877
Epoch 162/200
60000/60000 [==============================] - 15s 254us/step - loss: 0.0061 - acc: 0.9982 - val_l
oss: 0.0633 - val_acc: 0.9861
Epoch 163/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0052 - acc: 0.9984 - val_l
oss: 0.0596 - val_acc: 0.9869
Epoch 164/200
60000/60000 [==============================] - 15s 242us/step - loss: 0.0038 - acc: 0.9987 - val_l
oss: 0.0611 - val_acc: 0.9878
Epoch 165/200
60000/60000 [==============================] - 15s 250us/step - loss: 0.0044 - acc: 0.9986 - val_l
oss: 0.0646 - val_acc: 0.9871
Epoch 166/200
60000/60000 [==============================] - 15s 255us/step - loss: 0.0046 - acc: 0.9985 - val_l
oss: 0.0585 - val_acc: 0.9881
Epoch 167/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0035 - acc: 0.9989 - val_l
oss: 0.0565 - val_acc: 0.9879
Epoch 168/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0049 - acc: 0.9986 - val_l
oss: 0.0588 - val_acc: 0.9874
Epoch 169/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0047 - acc: 0.9985 - val_l
oss: 0.0560 - val_acc: 0.9881
Epoch 170/200
60000/60000 [==============================] - 15s 249us/step - loss: 0.0049 - acc: 0.9984 - val_l
oss: 0.0640 - val_acc: 0.9873
Epoch 171/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0039 - acc: 0.9987 - val_l
oss: 0.0566 - val_acc: 0.9891
Epoch 172/200
60000/60000 [==============================] - 15s 250us/step - loss: 0.0044 - acc: 0.9986 - val_l
oss: 0.0597 - val_acc: 0.9873
Epoch 173/200
60000/60000 [==============================] - 15s 247us/step - loss: 0.0030 - acc: 0.9990 - val_l
oss: 0.0591 - val_acc: 0.9881
Epoch 174/200
60000/60000 [==============================] - 15s 258us/step - loss: 0.0054 - acc: 0.9984 - val_l
oss: 0.0581 - val_acc: 0.9878
Epoch 175/200
60000/60000 [==============================] - 17s 276us/step - loss: 0.0054 - acc: 0.9982 - val_l
oss: 0.0636 - val_acc: 0.9875
Epoch 176/200
60000/60000 [==============================] - 17s 276us/step - loss: 0.0036 - acc: 0.9989 - val_l
oss: 0.0592 - val_acc: 0.9889
Epoch 177/200
60000/60000 [==============================] - 16s 267us/step - loss: 0.0054 - acc: 0.9983 - val_l
oss: 0.0622 - val_acc: 0.9873
Epoch 178/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0039 - acc: 0.9989 - val_l
oss: 0.0656 - val_acc: 0.9868
Epoch 179/200
60000/60000 [==============================] - 15s 246us/step - loss: 0.0040 - acc: 0.9986 - val_l
oss: 0.0672 - val_acc: 0.9872
Epoch 180/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0045 - acc: 0.9986 - val_l
oss: 0.0587 - val_acc: 0.9881
Epoch 181/200
60000/60000 [==============================] - 15s 252us/step - loss: 0.0049 - acc: 0.9986 - val_l
oss: 0.0619 - val_acc: 0.9872
Epoch 182/200
60000/60000 [==============================] - 15s 249us/step - loss: 0.0034 - acc: 0.9990 - val_l
oss: 0.0628 - val_acc: 0.9878
Epoch 183/200
60000/60000 [==============================] - 15s 249us/step - loss: 0.0032 - acc: 0.9990 - val_l
oss: 0.0614 - val_acc: 0.9882
```

```
oss: 0.0614 - val_acc: 0.9882
Epoch 184/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0045 - acc: 0.9984 - val_l
oss: 0.0608 - val_acc: 0.9872
Epoch 185/200
60000/60000 [==============================] - 15s 245us/step - loss: 0.0043 - acc: 0.9987 - val_l
oss: 0.0615 - val_acc: 0.9870
Epoch 186/200
60000/60000 [==============================] - 15s 244us/step - loss: 0.0040 - acc: 0.9987 - val_l
oss: 0.0658 - val_acc: 0.9873
Epoch 187/200
60000/60000 [==============================] - 14s 237us/step - loss: 0.0042 - acc: 0.9988 - val_l
oss: 0.0618 - val_acc: 0.9886
Epoch 188/200
60000/60000 [==============================] - 15s 247us/step - loss: 0.0048 - acc: 0.9983 - val_l
oss: 0.0586 - val_acc: 0.9889
Epoch 189/200
60000/60000 [==============================] - 15s 251us/step - loss: 0.0042 - acc: 0.9986 - val_l
oss: 0.0607 - val_acc: 0.9870
Epoch 190/200
60000/60000 [==============================] - 15s 253us/step - loss: 0.0038 - acc: 0.9987 - val_l
oss: 0.0689 - val_acc: 0.9862
Epoch 191/200
60000/60000 [==============================] - 15s 253us/step - loss: 0.0051 - acc: 0.9986 - val_l
oss: 0.0557 - val_acc: 0.9882
Epoch 192/200
60000/60000 [==============================] - 15s 253us/step - loss: 0.0036 - acc: 0.9988 - val_l
oss: 0.0566 - val_acc: 0.9885
Epoch 193/200
60000/60000 [==============================] - 15s 254us/step - loss: 0.0034 - acc: 0.9989 - val_l
oss: 0.0627 - val_acc: 0.9876
Epoch 194/200
60000/60000 [==============================] - 15s 250us/step - loss: 0.0036 - acc: 0.9988 - val_l
oss: 0.0635 - val_acc: 0.9877
Epoch 195/200
60000/60000 [==============================] - 15s 248us/step - loss: 0.0035 - acc: 0.9988 - val_l
oss: 0.0591 - val_acc: 0.9872
Epoch 196/200
60000/60000 [==============================] - 15s 254us/step - loss: 0.0034 - acc: 0.9989 - val_l
oss: 0.0662 - val_acc: 0.9867
Epoch 197/200
60000/60000 [==============================] - 16s 270us/step - loss: 0.0040 - acc: 0.9988 - val_l
oss: 0.0646 - val_acc: 0.9867
Epoch 198/200
60000/60000 [==============================] - 16s 272us/step - loss: 0.0049 - acc: 0.9987 - val_l
oss: 0.0620 - val_acc: 0.9869
Epoch 199/200
60000/60000 [==============================] - 16s 267us/step - loss: 0.0044 - acc: 0.9987 - val_l
oss: 0.0615 - val_acc: 0.9865
Epoch 200/200
60000/60000 [==============================] - 15s 252us/step - loss: 0.0035 - acc: 0.9987 - val_l
oss: 0.0634 - val_acc: 0.9862
Test score: 0.0633901542599633
Test accuracy: 0.9862
```



Conclusion:

1. We can clearly see how adding a dropout made a huge difference.

2. The dropout layer clearly made the model not overfit