neural_networks

March 24, 2019

In [1]:
```
import warnings
warnings.filterwarnings("ignore")

from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers.convolutional import Convolution2D, MaxPooling2D
from keras.utils import np_utils
import numpy as np
from keras import backend as K
K.set_image_dim_ordering('th')
```

```
Using TensorFlow backend.
```

In [2]:
```
from keras.datasets import mnist
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

In [3]:
```
num_pixels = X_train.shape[1] * X_train.shape[2]
n_channels = 1 #All picutures are in shades of gray
def preprocess(matrix):
    return matrix.reshape(matrix.shape[0],
                          n_channels,
                          matrix.shape[1],
                          matrix.shape[2]).astype('float32') /255.
```

In [4]:
```
print(X_train.shape)
print(X_train.dtype)
print(np.max(X_train))
```

```
(60000, 28, 28)
uint8
255
```

In [5]:
```
X_train, X_test = preprocess(X_train), preprocess(X_test)
```

In [6]:
```
print(X_train.shape)
print(X_train.dtype)
print(np.max(X_train))
```

```
(60000, 1, 28, 28)
float32
1.0
```

```
In [7]: y_train = np_utils.to_categorical(y_train)
        y_test = np_utils.to_categorical(y_test)
        num_classes = y_train.shape[1]
        print(y_train.shape)
        print(y_train.shape[1])
```

```
(60000, 10)
10
```

```
In [8]: def basline_model():
            model = Sequential()
            model.add(Flatten(input_shape=(1,28,28)))
            model.add(Dense(num_pixels, activation='relu', kernel_initializer="normal"))
            model.add(Dense(num_classes, activation='softmax', kernel_initializer="normal"))
            model.compile(loss='categorical_crossentropy',optimizer='adam', metrics=['accuracy
            return model
```

```
In [9]: def convolution_small():
            model = Sequential()
            model.add(Convolution2D(32,5,5,border_mode='valid',input_shape = (1,28,28),activati
            model.add(MaxPooling2D(pool_size=(2,2)))
            model.add(Dropout(0.2))
            model.add(Flatten())
            model.add(Dense(128,activation='relu'))
            model.add(Dense(num_classes,activation='softmax'))
            model.compile(loss='categorical_crossentropy',optimizer='adam', metrics=['accuracy
            return model
```

```
In [10]: def convolution_large():
             model = Sequential()
             model.add(Convolution2D(30,5,5, border_mode='valid',input_shape = (1,28,28),activa
             model.add(MaxPooling2D(pool_size=(2,2)))
             model.add(Convolution2D(15,3,3, activation='relu'))
             model.add(MaxPooling2D(pool_size=(2,2)))
             model.add(Dropout(0.2))
             model.add(Flatten())
             model.add(Dense(128, activation='relu'))
             model.add(Dense(50, activation='relu'))
             model.add(Dense(num_classes,activation='softmax'))
             model.compile(loss='categorical_crossentropy',optimizer='adam', metrics=['accuracy
             return model
```

```
In [11]: np.random.seed(101)
         models = [('Baseline', basline_model()),
```

```
                ('Small', convolution_small()),
                ('Large', convolution_large())]
        for name, model in models:
            print("Model: %s" % name)
            model.fit(X_train, y_train, validation_data=(X_test, y_test),
                      nb_epoch=10, batch_size=100, verbose=2)
            scores = model.evaluate(X_test, y_test, verbose=0)
            print("Base error: %f" % (100-scores[1]*100))
            print("_"*20)
```

```
WARNING:tensorflow:From d:\python\lib\site-packages\tensorflow\python\framework\op_def_library
Instructions for updating:
Colocations handled automatically by placer.
WARNING:tensorflow:From d:\python\lib\site-packages\keras\backend\tensorflow_backend.py:3445: 
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
Model: Baseline
WARNING:tensorflow:From d:\python\lib\site-packages\tensorflow\python\ops\math_ops.py:3066: to
Instructions for updating:
Use tf.cast instead.
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
 - 8s - loss: 0.2296 - acc: 0.9337 - val_loss: 0.1124 - val_acc: 0.9682
Epoch 2/10
 - 8s - loss: 0.0904 - acc: 0.9731 - val_loss: 0.0827 - val_acc: 0.9743
Epoch 3/10
 - 8s - loss: 0.0566 - acc: 0.9828 - val_loss: 0.0642 - val_acc: 0.9786
Epoch 4/10
 - 8s - loss: 0.0378 - acc: 0.9887 - val_loss: 0.0692 - val_acc: 0.9779
Epoch 5/10
 - 8s - loss: 0.0259 - acc: 0.9924 - val_loss: 0.0722 - val_acc: 0.9782
Epoch 6/10
 - 8s - loss: 0.0208 - acc: 0.9938 - val_loss: 0.0639 - val_acc: 0.9805
Epoch 7/10
 - 8s - loss: 0.0148 - acc: 0.9958 - val_loss: 0.0661 - val_acc: 0.9801
Epoch 8/10
 - 8s - loss: 0.0112 - acc: 0.9966 - val_loss: 0.0634 - val_acc: 0.9818
Epoch 9/10
 - 8s - loss: 0.0098 - acc: 0.9972 - val_loss: 0.0734 - val_acc: 0.9782
Epoch 10/10
 - 8s - loss: 0.0089 - acc: 0.9971 - val_loss: 0.0802 - val_acc: 0.9795
Base error: 2.050000
--------------------
Model: Small
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
 - 41s - loss: 0.1883 - acc: 0.9453 - val_loss: 0.0611 - val_acc: 0.9811
Epoch 2/10
```

```
 - 41s - loss: 0.0623 - acc: 0.9807 - val_loss: 0.0418 - val_acc: 0.9866
Epoch 3/10
 - 44s - loss: 0.0435 - acc: 0.9866 - val_loss: 0.0378 - val_acc: 0.9872
Epoch 4/10
 - 41s - loss: 0.0338 - acc: 0.9897 - val_loss: 0.0391 - val_acc: 0.9868
Epoch 5/10
 - 45s - loss: 0.0260 - acc: 0.9919 - val_loss: 0.0315 - val_acc: 0.9907
Epoch 6/10
 - 44s - loss: 0.0213 - acc: 0.9931 - val_loss: 0.0329 - val_acc: 0.9901
Epoch 7/10
 - 44s - loss: 0.0166 - acc: 0.9945 - val_loss: 0.0399 - val_acc: 0.9882
Epoch 8/10
 - 43s - loss: 0.0137 - acc: 0.9955 - val_loss: 0.0395 - val_acc: 0.9892
Epoch 9/10
 - 44s - loss: 0.0133 - acc: 0.9954 - val_loss: 0.0339 - val_acc: 0.9904
Epoch 10/10
 - 41s - loss: 0.0108 - acc: 0.9963 - val_loss: 0.0369 - val_acc: 0.9901
Base error: 0.990000
--------------------
Model: Large
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
 - 42s - loss: 0.2746 - acc: 0.9121 - val_loss: 0.0632 - val_acc: 0.9800
Epoch 2/10
 - 41s - loss: 0.0781 - acc: 0.9764 - val_loss: 0.0473 - val_acc: 0.9843
Epoch 3/10
 - 41s - loss: 0.0578 - acc: 0.9822 - val_loss: 0.0379 - val_acc: 0.9881
Epoch 4/10
 - 41s - loss: 0.0473 - acc: 0.9857 - val_loss: 0.0305 - val_acc: 0.9909
Epoch 5/10
 - 41s - loss: 0.0420 - acc: 0.9871 - val_loss: 0.0265 - val_acc: 0.9913
Epoch 6/10
 - 41s - loss: 0.0356 - acc: 0.9887 - val_loss: 0.0288 - val_acc: 0.9912
Epoch 7/10
 - 41s - loss: 0.0326 - acc: 0.9898 - val_loss: 0.0297 - val_acc: 0.9905
Epoch 8/10
 - 41s - loss: 0.0297 - acc: 0.9901 - val_loss: 0.0226 - val_acc: 0.9930
Epoch 9/10
 - 41s - loss: 0.0273 - acc: 0.9912 - val_loss: 0.0266 - val_acc: 0.9914
Epoch 10/10
 - 42s - loss: 0.0241 - acc: 0.9922 - val_loss: 0.0247 - val_acc: 0.9926
Base error: 0.740000
--------------------
```

In [ ]: