Street View Housing Number Digit Recognition

Neural Networks

Domain

Autonomous Vehicles, Neural Networks

Business Context

Recognizing multi-digit numbers in photographs captured at street level is an important component of modern-day map making. A classic example of a corpus of such street-level photographs is Google's Street View imagery composed of hundreds of millions of geo-located 360-degree panoramic images.

The ability to automatically transcribe an address number from a geo-located patch of pixels and associate the transcribed number with a known street address helps pinpoint, with a high degree of accuracy, the location of the building it represents. More broadly, recognizing numbers in photographs is a problem of interest to the optical character recognition community.

While OCR on constrained domains like document processing is well studied, arbitrary multi-character text recognition in photographs is still highly challenging. This difficulty arises due to the wide variability in the visual appearance of text in the wild on account of a large range of fonts, colours, styles, orientations, and character arrangements.

The recognition problem is further complicated by environmental factors such as lighting, shadows, specularities, and occlusions as well as by image acquisition factors such as resolution, motion, and focus blurs. In this project, we will use the dataset with images centred around a single digit (many of the images do contain some distractors at the sides). Although we are taking a sample of the data which is simpler, it is more complex than MNIST because of the distractors.

Objective

We will build a digit classifier on the SVHN (Street View Housing Number) dataset.

Dataset description

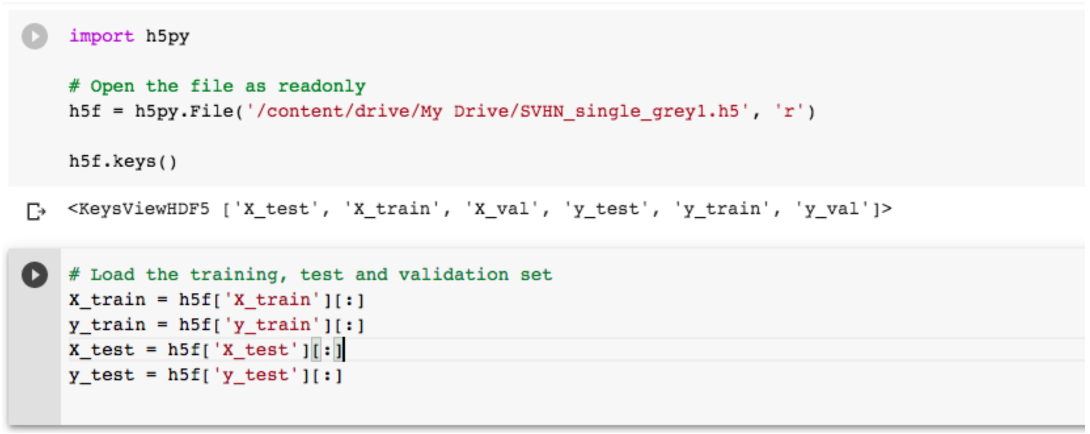
SVHN is a real-world image dataset for developing machine learning and object recognition algorithms with the minimal requirement on data formatting but comes from a significantly harder, unsolved, real-world problem (recognizing digits and numbers in natural scene images). SVHN is obtained from house numbers in Google Street View images.

Some examples of the samples from this dataset are:

Where the labels for each of this image are the prominent number in that image i.e. 2,6,7 and 4 respectively.

The dataset has been provided in the form of h5py files. You can read about this file format here: http://docs.h5py.org/en/stable/high/dataset.html

Code to load the dataset



```
import h5py

# Open the file as readonly
h5f = h5py.File('/content/drive/My Drive/SVHN_single_grey1.h5', 'r')

h5f.keys()
```

```
<KeysViewHDF5 ['X_test', 'X_train', 'X_val', 'y_test', 'y_train', 'y_val']>
```

```
# Load the training, test and validation set
X_train = h5f['X_train'][:]
y_train = h5f['y_train'][:]
X_test = h5f['X_test'][:]
y_test = h5f['y_test'][:]
```

Acknowledgement

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y. Ng Reading Digits in Natural Images with Unsupervised Feature Learning NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011. PDF
http://ufldl.stanford.edu/housenumbers as the URL for this site when necessary

Steps/Grading Policy

The objective of the project is to learn how to implement a simple image classification pipeline based on a deep neural network and understand the basics of Image Classification

- Read the data from the h5py file and understand the train/test splits (5 points)
- Reshape and normalize the train and test features so that the same can be fed for model building. We need to feed a 2D tensor into the model and currently we have a 3D tensor.(5 points)
- One hot encode the labels for train and test data (6 points)
- Define the model architecture using TensorFlow with a flatten layer followed by dense layers with activation as ReLu and softmax (7 points)
- Compile the model with loss as categorical cross-entropy and adam optimizers. Use accuracy as the metric for evaluation (7 points)
- Fit and evaluate the model. Print the loss and accuracy for the test data (10 points)
- Plot the training loss, validation loss vs number of epochs and training accuracy, validation accuracy vs number of epochs plot and write your observations on the same. (10 points)

Further Questions (Optional)

Can you try changing a few hyperparameters such as number of layers in the network or number of units in a hidden layer or try different activation functions in the hidden layers and see if you get better results than the one in this notebook?

Learning Outcomes

- Neural networks
- Tensorflow
- Computer Vision
- Hyperparameter Tuning
- Working with Image dataset
- Digit Classification