George Joseph, Shobhik Bhadraray and Shahnawaz Ahmed

# A CNN based number recognition tool

## I. Introduction

Identification of text in natural environments is a difficult problem. Convolution neural networks have shown promising trends towards prediction of text and this work will explore the prediction of digits of various lengths using the SVHN dataset. We will use TensorFlow as the backend for building the convolution neural network with Keras.

## II. Data

The dataset we used is the Street View House Numbers (SVHN) dataset. It consists of images with labeled digits and bounding box information for each digit.

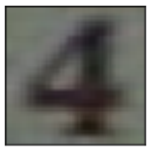
### *A.Description*

We have the following counts for numbers in the data set.

| Digits | Number |
|---|---|
| 1 | 5137 |
| 2 | 18130 |
| 3 | 8691 |
| 4 | 1434 |
| 5 | 10 |

### *B.Preprocessing images*

The preprocessing involves stitching together individual bounding boxes and generating numbers of various lengths. We assume that the largest sequence in this data set is of length 5. We use vectors of length 6 as the output for each image with the first 5 elements denoting the digits and the last element denoting the length of sequence. All images are resized to 50x50. This was performed using the Python Image Library using a cubic spline interpolation.



### *C.One hot encoding of target*

The target vectors are initially of dimension 6 with the first 5 elements representing the number and the 6th element representing the sequence length. ‘0’ is represented by 10 and a 0 in the target vector represents that the digit is not present.

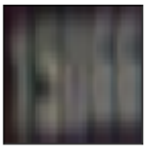
target = [4, 5, 4, 0, 0, 3]

The target vectors are one hot encoded and the vectors of length 6 are now converted into matrices of shape 6 x 11. Each digit is now represented by a 11 dimensional vector.

### *D.Data augmentation*

Since the proportion of data was skewed, we generate more data for 3, 4 and 5 digit numbers for training. For creating more 3 digit numbers we take the existing two digit numbers, randomly select a digit from the sequence and then randomly append it either to the front or back of the two digit sequence. We similarly use the original 3 digit sequences to create the new 4 digit sequences and use the original 4 digit sequences to create the new 5 digit sequences





| Digits | Number |
|---|---|
| 1 | 5137 |
| 2 | 18130 |
| 3 | 26821 |
| 4 | 10124 |
| 5 | 1434 |

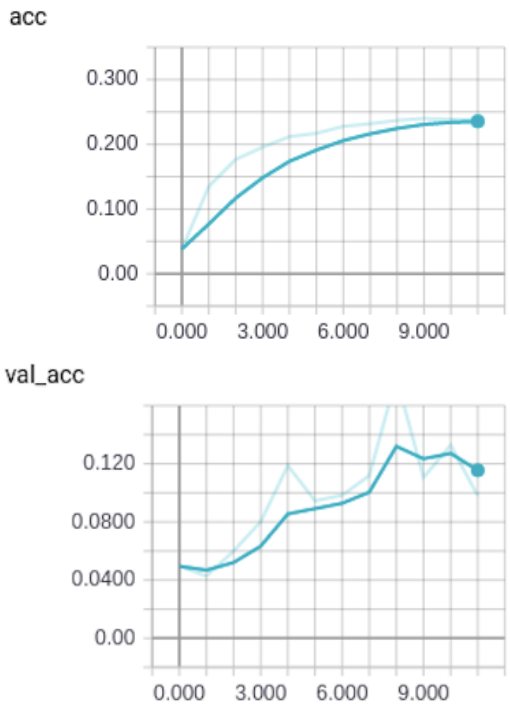
## III. Model I: Mini

We use a simple model for the initial prediction where we input all the images and try to predict the digits and length of the sequence simultaneously.

### *A.Architecture*

2 Convolution layers
1 dense layer

*B.Results*



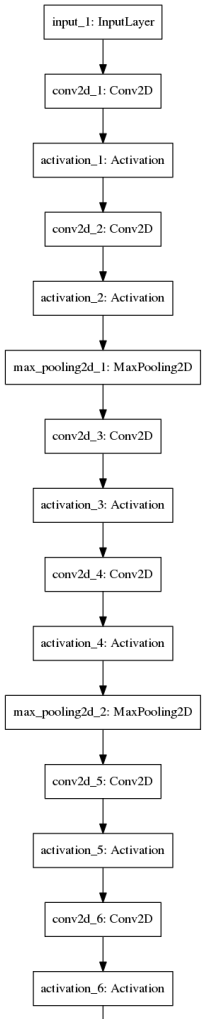

The validation accuracy plots show that the training is not smooth.

## IV. Model II: Fork

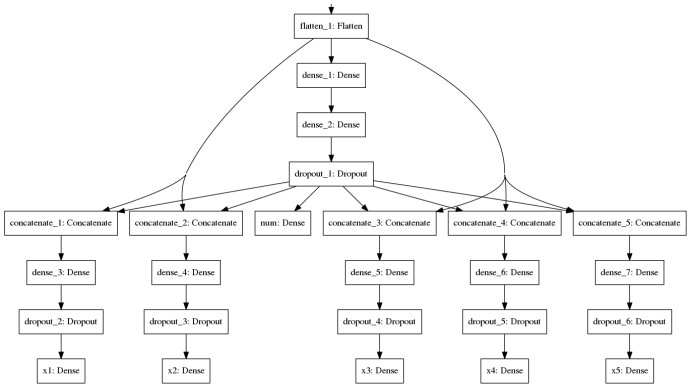
We now have 6 different softmax classifiers which learn feature from the same convolution neural network. One of the classifiers - numtower is for identifying the number of digits whose output is taken as one on the inputs by 5 different classifiers which predict digits at each position.

*A.Architecture*

The first part of the model has the following architecture:



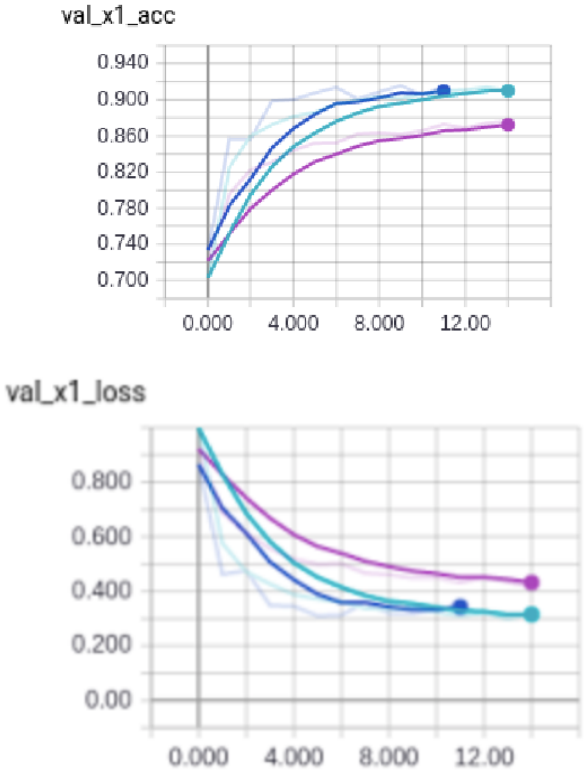

The second part of the model has the following architecture:



*B.Tuning hyperparameters*

We have 4, 6 and 8 convolution layers and compare the performance in predicting the first digit.

The model with 6 and 8 convolution layers give higher accuracies than the model with 4 layers. We choose the model with 6 layers as the accuracy saturates at the same value for both 6 layered and 8 layered models.




| | |
|---|---|
| Pink | 4 Convolution layers |
| Green | 6 Convolution layers |
| Blue | 8 Convolution layers |

*B.Results*

We get an accuracy of 95% for the number tower, which predicts the number of digits in the image. The individual accuracies for all digits are the following

| Digits | Accuracy |
|---|---|
| 1 | 90.38 |
| 2 | 89.63 |
| 3 | 93.49 |
| 4 | 98.47 |
| 5 | 99.96 |

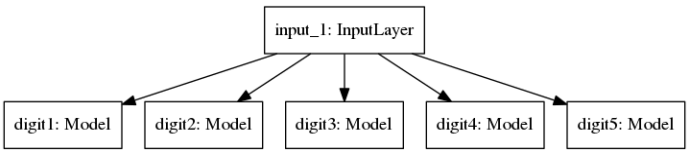
The final accuracy for the full prediction on validation set is 81%

There is a higher accuracy for prediction of 4th and 5th digit as the network predicts it to be 0 most of the times and the dataset is skewed and contains very few 4 and 5 digit numbers. In order to address this issue we augment the dataset to create mode 4 and 5 digit numbers and create a new model for prediction of each digit separately with different training data.

## V. ION CANNON: INDEPENDENT TRAINING FOR DIGITS

We train a new model with augmented data which has more 4 digit and 5 digit numbers. The first part of the input layer remains the same as the previous model which is fed to 5 different layers for prediction of each digit. The first three digits are trained using a combination of the original dataset and augmented data. While training the model for the 4th and 5th digit we feed equal number of instances containing 4 and 5 digit numbers and non 4 and 5 digit numbers. This is done to make the network predict the 4th and 5th digit with better accuracy.

*A.Architecture*




*A.Results*

The data used for validation consists of a combination of augmented 4 digit numbers and original data. Thus we have more number of 4 digit numbers than the original dataset.

The accuracy in prediction of number of digits on the validation set is 95.4%.

Due to the skewed nature of the dataset, the accuracies in prediction for 4th and 5th digit is lower.

| Digits | Accuracy |
|---|---|
| 1 | 90.17 |
| 2 | 88.018 |
| 3 | 86.31 |
| 4 | 82.34 |
| 5 | 68.5 |

## VI. The complete pipeline

We use the model - fork for the final testing. It gives an accuracy of 81% on the validation set.

### *A.Image preprocessing*

We obtain cropped images of various sizes which contain number of varying sequences. In order to pass it to our model we resize all images to 50x50 using a cubic spline interpolation.

### *B.Prediction*

The resized images are then analysed and the model predicts the values of all 6 elements of the output vector - the digits and the sequence length.

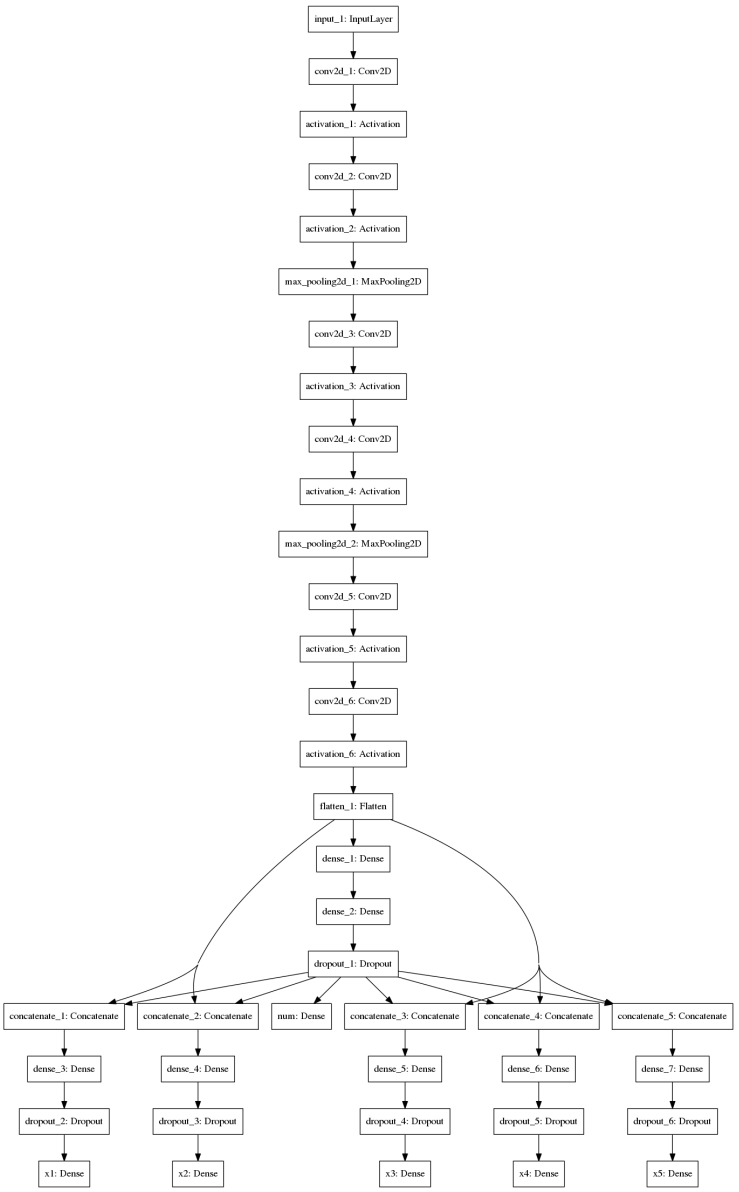
### *C.Results*

An accuracy of 81% is obtained on the final test set using the model - Fork. As the final test had proportions of numbers of various length similar to the training dataset.

## Conclusions

We have demonstrated the capability of a Convolution Neural Network in predicting numbers of various lengths with an accuracy of 81%. As the dataset was skewed, we generated our own data and analysed the performance of our models using both the original and augmented data. We get the best performance for the model - Fork.

The Ion cannon model should perform better when tested against a dataset which has a good proportion of numbers of various lengths.




## References

1. Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, Vinay Shet (2013, Dec). Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. Available: https://arxiv.org/abs/1312.6082