Multidigit Number Detection in Natural Images

Ruchi Neema[a,d], Saurabh kumar[b,d], Samvat Rastogi[c,d]

[a]*rneema@indiana.edu*
[b]*kumarsau@indiana.edu*
[c]*samrasto@indiana.edu*
[d]*Indiana University, Bloomington*

Abstract

Character Recognition in images is a vital task as it provides conversion of characters in the real world to the characters the virtual world. The basic version of this problem is the Optical Character Recognition (OCR), which works well for a scanned image with fixed size characters in a specified boundary. The advanced version of this problem in recognizing multiple characters in unconstrained natural images. We try to solve the problem of detecting unknown number of digits, with varying shape and orientation, in natural images. For this task we have used the publicly available Street View Housing Number (SVHN) data set. We have built a Convolutional Neural Network (CNN) model called '11Class' for this classification task. The entire problem has 3 major parts namely Localization, Segmentation and Classification. 11Class model integrates segmentation and classification via use of CNN that works directly on the cropped multi-digit housing numbers. This model consists of 3 convolutional layers and 2 fully connected layers. We have developed a base model based on a CNN with 2 layers. Base model is used to compare accuracy results.

Keywords: CNN, SVHN, softmax, multi-digit, regularization, 11Class

1. Introduction

The main problem that this paper is trying to solve is detection of multi-digit. Classifying one digit at a time is a basic version of this problem, but in real world images we don't know the number of digits in each image. This creates problem as you need an extra task of segmentation associated with it. Different orientation for digits make the problem even tougher.

Firstly, we do the data curation step. We take the publicly available Street View House Numbers (SVHN) data set [5]. The curation includes formation of cropped images and creating a data frame of the labels associated with each image. We use grey scale as the raw images are colored. The length of digits range from 1 to 5, for our data set. This data is different from the MNIST data set, as it not centered around digits [4].

We apply two approaches to solve the recognition problem. 11Class approach uses 3 convolutional layer and flattened layer followed by 2 fully connected layers. In our approach, the segmentation is not done. Instead the we use the the cropped images with unknown

number of digits. 11Class approach we assume that there 'n' digits in the image and each digits is represented by 11 classes. The 10 classes show the digit number (0-9). The last class shows the absence of digit on that location. We also increase the convolutional layers from 3 to 5 for our main approach.

To better analyze our results we have built a base model, loosely similar to the one used in (Goodfellow et al.,2013). This helps to create a comparative study of our results. This approach uses 2 convolutional layers and the output layer consists of 6 softmax function. The first function is used to determine the number of digits in the image (1-5) and the other 5 functions are used to determine each digit. If the length is less than 5, the subsequent softmax functions are turned off [1].

Finally, we compare the training and testing accuracy with respect to the number of iterations. We do this for both the approaches.

2. Related Work

The classic method for classification and recognition of digits in natural images uses unsupervised learning approaches (Netzer et al.,2011). Using Street View Housing Number dataset[reference], various feature representation and classification models were evaluated. The major inspiration of this experiment was to compare features, which are commonly used in computer vision systems, generated by feature learning systems and by hand-constructed feature representations. Unsupervised learning techniques like Histograms-of-Oriented-Gradients (HOG), Binary Features (WDCH), K-Means and Stacked Sparse Auto-Encoder were used which highest accuracy of 90.6% for K-Means and lowest accuracy of 63.3% in case of Binary Features (WDCH). Paper also discusses the human accuracy with respect to the height of images in terms of pixels [3].

CNNs have groups of neurons with tied parameters. A CNN based technique (Goodfellow et al.,2013) is used to recognize arbitrary multi-digit numbers from Street View imagery. In this approach, a CNN based deep neural network, is proposed, that works directly on image pixels which unifies the traditional approach of localization, segmentation, and recognition. Using DistBelief implementation of deep net, they were able to attend an accuracy of 96% in recognizing complete street numbers from publicly available SVHN dataset. On a per-digit recognition, an accuracy of 97.84% was achieved, improving the state of the art system. This system have recognized over 100 million physical street numbers from Street View Imagery till 2014 [1].

Another paper (Guo et al.,2014) has very similar approach to Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks paper but it uses Hidden Markov Model to model the image and then uses CNNs to model the appearance of digit. Hence, here CNNs were combined with probabilistic graphical model to resolve the problem. This paper also compares the accuracy of CNN-HMM, Histograms-of-Oriented-Gradients (HOG), K-Means and Stacked Sparse Auto-Encode for single digit recognition. The paper concludes on discussing the usage of more sophisticated probabilistic models than HMM, e.g. CRF and pending investigation of the problem with different and better architecture of CNNs [2].

3. Dataset

The Street View House Numbers (SVHN) data set is an image data set which is based on house numbers in Google Street View Imagery. One of the primary usage of this data set is to develop machine learning and image recognition algorithms with minimal data pre-processing. It has over 600,00 labelled images [5].

Each numerical digit is assigned a class. There are total of 10 classes. '1' for digit 1, '9' for digit 9 and '10' for digit 0. Maximum length sequence of a house number is 5. There are 73257 digits for training, 26032 digits for testing, and 531131 additional digits for more training. There are two data formats:

- Original images with character level bounding boxes.
- MNIST-like 32-by-32 images centered around a single character (many of the images do contain some distractors at the sides).

For this project, we are using the first format. Images in this format are the original, variable-resolution, color house-number images. Each digit in the image is bounded within a bounding box. The bounding box information are stored in digitStruct.mat. A struct called digitStruct with the same length as the number of original images is stored in digitStruct. Each element in this struct has the following fields:

- name: String; contains the file name of the corresponding image.
- bbox: Struct array; contains the position, size and label of each digit bounding box in the image.

The data curation process for creating cropped images using raw images and bbox information, is shown in figure 1

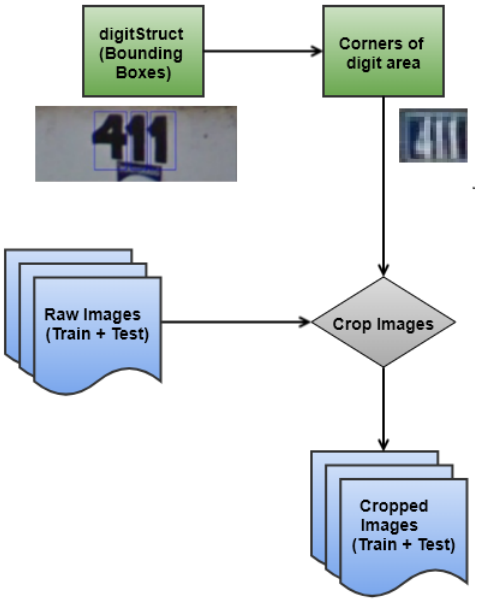

Figure 1: Data Curation

The distribution of count of images with different number of digits is shown in figure 2. The figures shows the distribution for both the train and test sets.

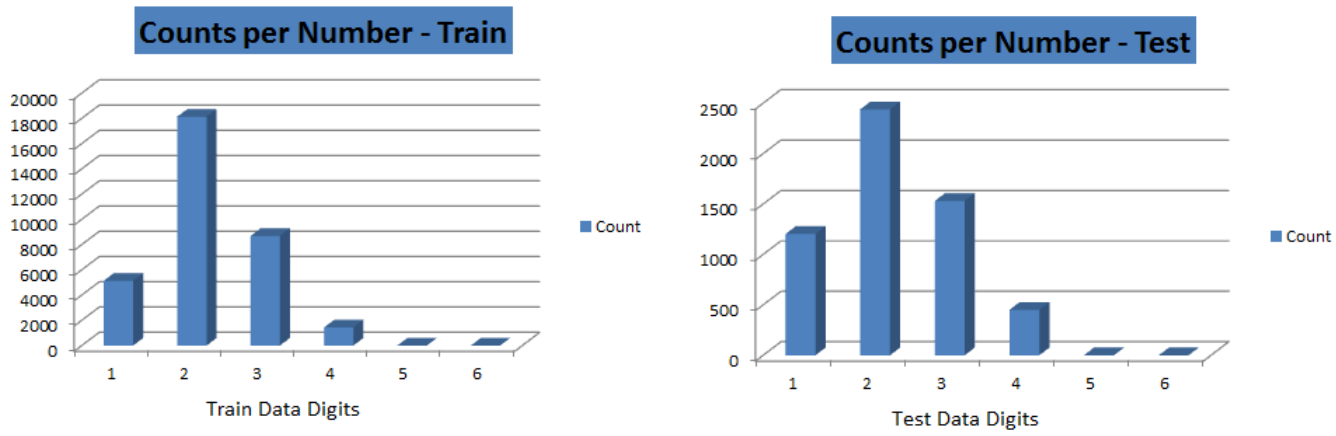

Figure 2: Count of Different Digit Numbers in Data

4. Model and Implementation

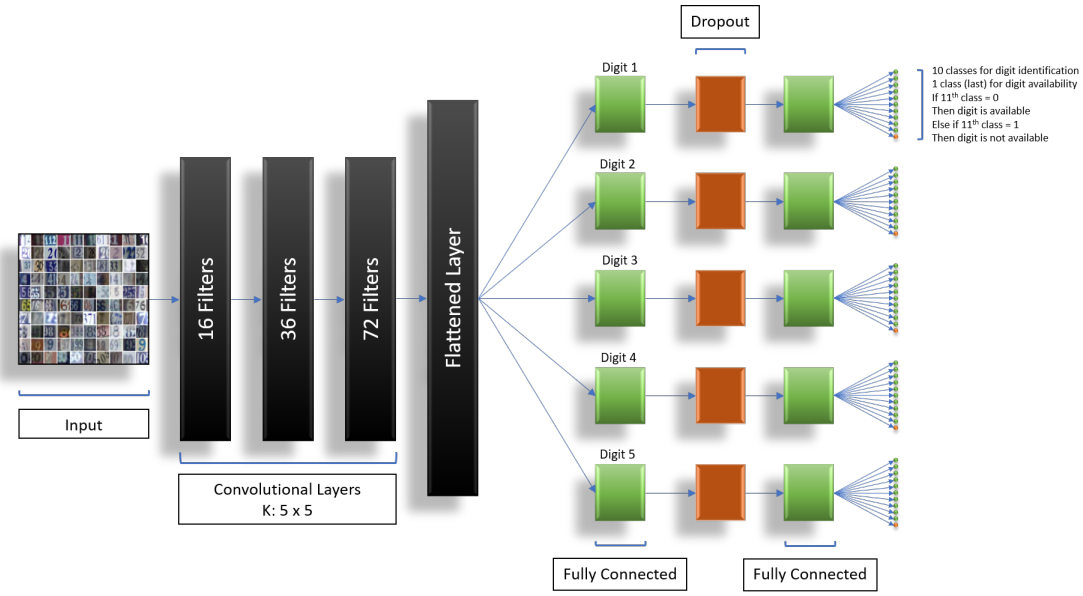

Figure 3: CNN 11Class Model Architecture

4.1. 11Class Algorithm

The main idea behind this algorithm is to solve the problem of digit recognition using a single convolution neural network. The goal of the CNN is to find the sequence S of digits

given an image. This algorithm assumes that the sequence S consist of maximum 'n' digits where each digit can be divided into 11 classes, 10 classes belongs to digits number (0-9) and the last class indicates the absence of digit on that location. This type of class structure automatically accommodates for the length of the digits and reduces the need to define a separate random variable for the length parameter as done in previous state-of-art algorithm. Through our experiments, we found better training and testing accuracy with this algorithm as compared to algorithm 1 with just 3 convolution layers in both the architectures. We believe that this change is publishable and also looking for your feedback on the same.

The architecture for the problem is given in Figure 3. It consists of 3 convolution layer with 16 filters of 5*5 in first layer, 36 filters of 5*5 in second layer and 72 filters of 5*5 in third layer. We also implemented max pooling in each of these layers and took dropout with probability of 0.1 in all the fully connected layer of five digits. We trained our model with training and testing batch size of 128 and learning rate of 0.00001. We separated data into 75% training set and 25% of testing set.

We define the summation of cross entropy for each digit as a loss function. One more innovation in our approach is the way we define one-hot encoded labels for the training data based on the number of digits present in the image. Note that we have maximum of 5 digits in our image set, if the number of digits are less than 5 then we assume that they are the first three digit in our network and last two digit belongs to 11$\hat{\text{t}}$h class. This rule helps us during testing phase and helps us to find the number of digits based on the absence of the first digit in the sequence. In future, we are planning to enhance the model using Hidden Markov Chain and find the number of digits based on the probabilities obtained by our CNN model.

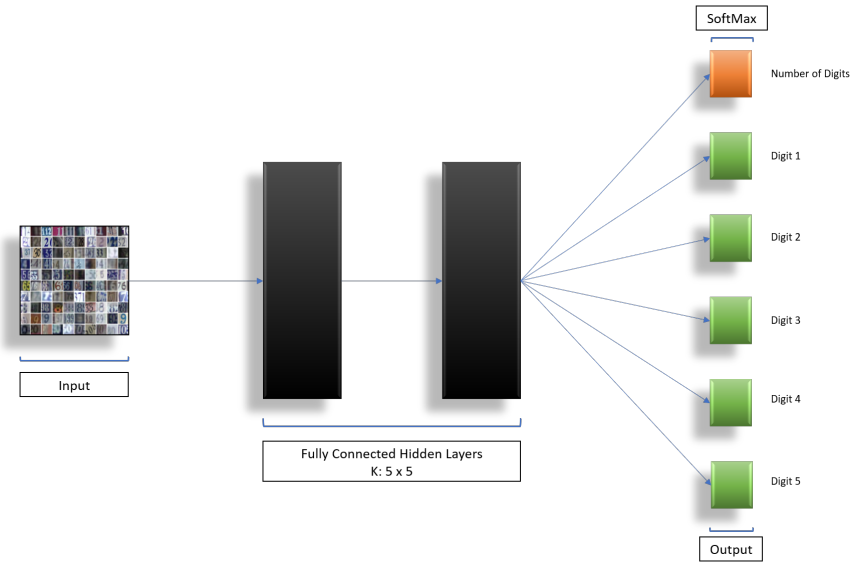

Figure 4: CNN Base Model Architecture

4.2. Base Algorithm

The base model is a CNN with 2 convoluted layer and an output layer with 6 sigmoid functions. The architecture is shown in figure 4. This model has 16 filters of 5*5 in first layer, 36 filters of 5*5 in second layer. We have used a learning rate of 0.001 and 50000 iterations for this model.

The detailed algorithm used for pre-processing and building the CNN model is shown below. This includes the various functions used graph for creating neural network in tensorflow.

Algorithm 1 Pre-processing SVHN data set and Building CNN model

procedure DATA_INPUT(*data_location*)
    images ← Read all images
    images ← Convert to grey scale
    return images

procedure DEFINE_PARAMETERS
    $Learning\_Rate, iterations, batch\_size$ ← Values
    $Network\_Parameters$ ← Value
    $Weights$ ← Calculate weights for layer1 and layer2
    $Biases$ ← Calculate biases for layer1 and layer2
    return parameters

procedure DEFINE_CNN
    $Normalize$ ← Batch Normalization
    $Convolution\_layer\_1$ ← RELU ← Maxpool
    $Convolution\_layer\_2$ ← RELU ← Maxpool
    $Fully\_Connected\_layer$
    $Output\_layer$ ← 6 Softmax
    return graph

Build Graph for Deep Net
$Calculate\_Cost$ ← softmax_cross_entropy_with_logits
$Optimize$ ← Cost, Learning Rate
Calculate ← Accuracy
$Tensorflow\_session$ ← Initialize Variables
$CalculateAccuracy$ ← Accuracy For Different iterations
$Tensorflow\_session$ ← Run Optimizer
$BatchAccuracy$ ← Calculate Batch Accuracy for Train and Test

5. Results and Discussion

We train the model on BigRed and it took approximately 12 hours to train the complete model. Some of the correctly classified digits and incorrectly classified digits are shown in the figure 5. We can clearly see that the incorrectly classified images are at least correctly

classifying the correct total number of digits. Also, four and five digit numbers are not properly classified. This is attributed to the number of training images in the data set. The data set is highly biased and only has around 1000 images for four digits and around 10 samples for five digit numbers.

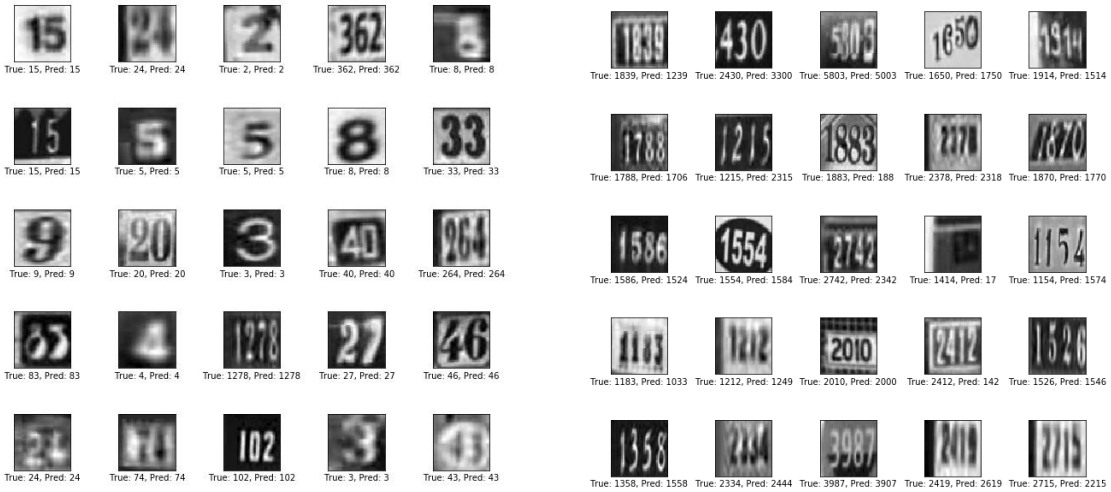

Figure 5: Left: Correct Prediction $\longleftrightarrow$ Right: Incorrect Prediction

Figure 6 represents the training and testing accuracy as a function of number of iterations (or epoch) and also shows the loss function during training. We can see that testing accuracy is almost constant after reaching 65 percentage while training accuracy reached upto 90%. We experimented with many different combinations such as adding more dropouts(different probabilities of dropout such as 0.4 for digit 4 and digit 5 and 0.3 for digit 1,2 and 3), adding regularization in weights and increasing the number of convolution layers to 4 but none of them helps.This shows that the model already learned the features from training set but unable to predict the features in testing set which clearly shows the lack of data. Therefore, in future, we are planning to enhance the data set by augmentation and try to increase the accuracy.

The base model provides results with less iterations. Though the training accuracy did not reach 90%, it was mostly above 80%. The number of iterations required for achieving such accuracy is 50000. The test set accuracy was above 60%. The accuracy for training set and test set is shown in the figure 7. From the results we can see that our 11Class model performed better than our base class model.

6. Conclusion

In this project, we propose a novel architecture for the problem of digit recognition. We find that the architecture obtained a better accuracy as compared to state-of-art approaches with the same number of convolution layers. The 11Class model provided accuracy 0f about 65% for test set.

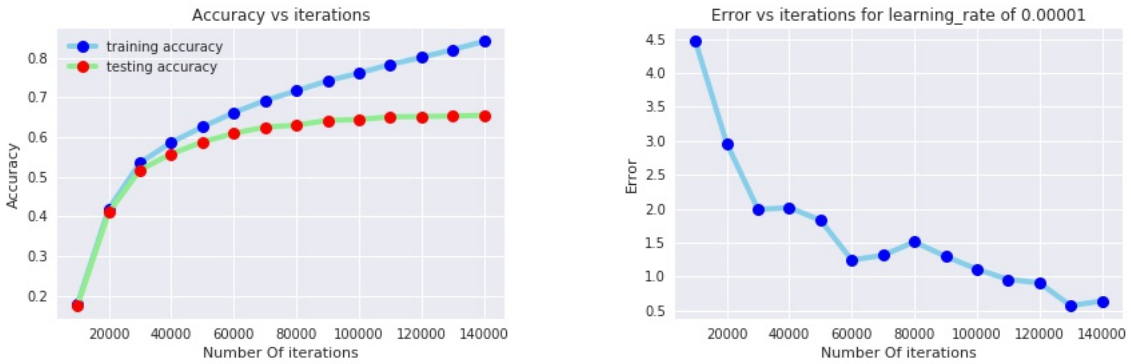

Figure 6: 11Class Model Results→ Left: Training and Testing accuracy. ⟷ Right: Loss function

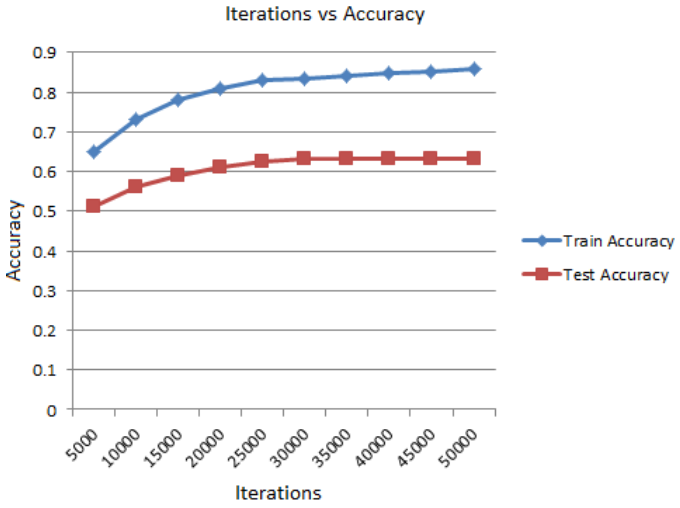

Figure 7: Base Model Results

In future, we are planning to enhance the data set and try to increase the accuracy for the digit recognition problem. The first solution would be to increase the number of layers in the CNN. We can also use transfer learning that is we can take weights from already trained model such as alexnet. Our current model gives the probabilities of each class for each digits. In future, we are planning to enhance the model using hidden markov chain where hidden states are the digits and probabilities are the likelihood obtained by the current CNN architecture.

References

[1] Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay D. Shet. 2013. Multi-digit Number Recognition from Street View Imagery using Deep Convolutional Neural Networks. *CoRR* abs/1312.6082 (2013). arXiv:1312.6082 http://arxiv.org/abs/1312.6082

[2] Q. Guo, J. Lei, D. Tu, and G. Li. 2014. Reading numbers in natural scene images with convolutional

neural networks. In *Proceedings 2014 IEEE International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*. 48–53. https://doi.org/10.1109/SPAC.2014.6982655

[3] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning.

[4] Christopher J.C. Burges Yann LeCun, Corinna Cortes. 2009. THE MNIST DATABASE of handwritten digits. (2009). http://yann.lecun.com/exdb/mnist/

[5] Tao Wang Yuval Netzer. 2011. The Street View House Numbers (SVHN) Dataset. (2011). http://ufldl.stanford.edu/housenumbers