Handwritten Digit Recognition

Mayank Gupta (2020CSY7571) , Soumen Basu (2019CSZ8406)
IIT Delhi

Abstract

In this work, we have explored neural architectures to recognize handwritten digits and image-based sequences. For individual digit recognition, we explore the traditional LeNet and other neural architectures for classification. We further propose a neural network architecture, which unifies a CNN-based feature extraction and a RNN-based sequence modeling. The proposed architecture is end-to-end trainable, effective, and low on model size - making it practical for real-world applications. We experimentally show the efficacy of the proposed architecture on a private dataset of handwritten digits.

1. Introduction

Handwritten digit recognition and image-based sequence detection are classic problems in computer vision. Ever since Yann LeCun proposed LeNet [7], the community has seen a plethora of contributions regarding digit recognition. However, visual objects such as scene text or handwriting tend to occur in sequences rather than occurring individually. In contrast to general object recognition, identifying such sequences requires the models to predict a series of object labels instead of a single label. Thus, the identification of digit sequences can be formulated as a sequence recognition problem. Additionally, sequences are often variable in length. As a result, DCNN models like [6] cannot be directly applied to sequence prediction, as such models usually mandate the inputs and outputs to be of fixed dimensions. The works in [1, 11] detect individual characters initially in scene texts and then recognize these characters with DCNN models trained with labeled character images. Recurrent neural networks (RNN) models can capture in their hidden state temporal and spatial features within a sequence of inputs and are popular for sequence recognition tasks. RNNs do not need the position of elements in a sequence during training or testing. Usually, a preprocessing step to convert an input object image into a sequence of features is essential for RNN models. However, such preprocessing steps are independent of the subsequent components making an end-to-end trainable network infeasible. Shi et al [8] proposed CRNN architecture which was end-to-end trainable. We build on the works of [8] and make a small yet robust network to recognize the image-based digit sequences effectively. Additionally, we also provide detailed experimental analysis and design modifications for improving the individual digit recognition.

In summary, our main contribution is as following:

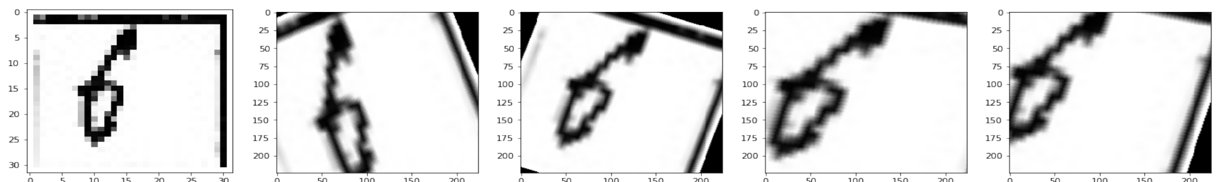
1. We present extensive experimental analysis of LeNet for recognizing single handwritten digits. We further propose design changes to improve the results obtained by the traditional LeNet architecture.

2. We propose and evaluate experimentally an end-to-end trainable CNN and RNN based neural architecture to recognize image-based sequence of handwritten digits.

2. Related Work

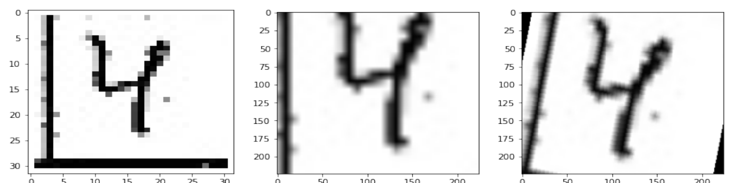
Handwritten digit recognition is a very popular problem and many ideas have been proposed to solve it. While many methods have been applied for this task, Convolutional Neural Networks (CNNs) have achieved better performance than other methods. LeCun et al. proposed the LeNet-5 architecture which achieved remarkable results on the MNIST dataset [7]. A work by Ciresan et al. used a committee of CNNs to achieve much better results on the same dataset [3].

For the task of detecting multiple digits/text in an image the commonly used methods are not sufficient. Many works have proposed methods to solve the problem. Sequence to sequence models are a type of model commonly used for this. A work by Shi et al. combines a CNN and a RNN to create a model named Convolutional Recurrent Neural Network (CRNN) which is then used for the task of scene text [8]. Another similar work by Voigtlaender et al. uses a combination of LSTM and convolutional layers for classifying handwritten text [10]. The works in [1, 11] detect individual characters initially in scene texts and then recognize these characters with DCNN models trained with labeled character images.



(a) (b) (c) (d) (e)

Figure 1: Data augmentations pipeline example. (a) Original image (b) - (e) Various augmentations of the same image



(a) (b) (c)

Figure 2: Data augmentations used. (a) Original image (b) Random cropping (c) Random rotation

3. Our Methods

3.1. Individual Digits Classification

LeNet-5: The base model we used a LeNet-5 model [7]. The model consists of 3 convolutional layers with a kernel size of 5x5. We have replaced the subsampling layer with an Average pooling layer. The other architectural details are similar to the ones used in the original paper. Since the original model was designed for the MNIST dataset, the number of output units of the model is 10. We have experimented with both finetuning a model trained on the MNIST dataset as well as training a model from scratch.

VGG-16: According to our experiments, we found the LeNet-5 to be lacking for the current dataset. The lesser number of convolutional layers meant that the model was not powerful enough. The model also used tanh as the activation and not the more preferred ReLU. In order to overcome these issues, we decided to use a better model and chose VGG-16 [9]. We have performed experiments on a basic VGG-16 model and have also made design interventions to account for the specific issues present in the data as well as the model.

Loss function: The loss function used was Cross Entropy loss.

Issues found: The following issues were found to be the cause of the low classification accuracy of the previous models:

1. Some images are not cropped correctly leading to some part of the number being outside the image.

2. Numbers may be rotated at different angles according to a persons handwriting. This leads to the model being confused between similar looking numbers.

3. Due to the small size of the images and relatively small dataset larger models tend to overfit the data easily.

Design Interventions: To counter the identified issues the following changes were added to the base VGG model:

1. Random cropping: To imitate the incorrect cropping, the images were first resized to 256x256 and then random cropping was applied to generate a image of size 224x224

2. Random rotation: To add rotated samples, a random rotation was done from -30 to +30 degrees on the cropped images

3. Batch Normalization: Batch norm layers were added after every convolutional layer of the VGG16 model to improve the convergence and also reduce overfitting.

4. Fine-tuning: The model taken was trained on the ImageNet dataset and we finetuned it on our data in order to provide better results.

3.2. Sequence Classification

We adopt the similar architecture proposed by [8]. The architecture consists of a convolutional feature extractor, the recurrent layer, and a fully connected layer. The convolutional layers automatically extract a feature sequence from each input image. A recurrent network makes predictions for the feature sequence. The fully-connected head translates the predictions by the recurrent layers into a label sequence.

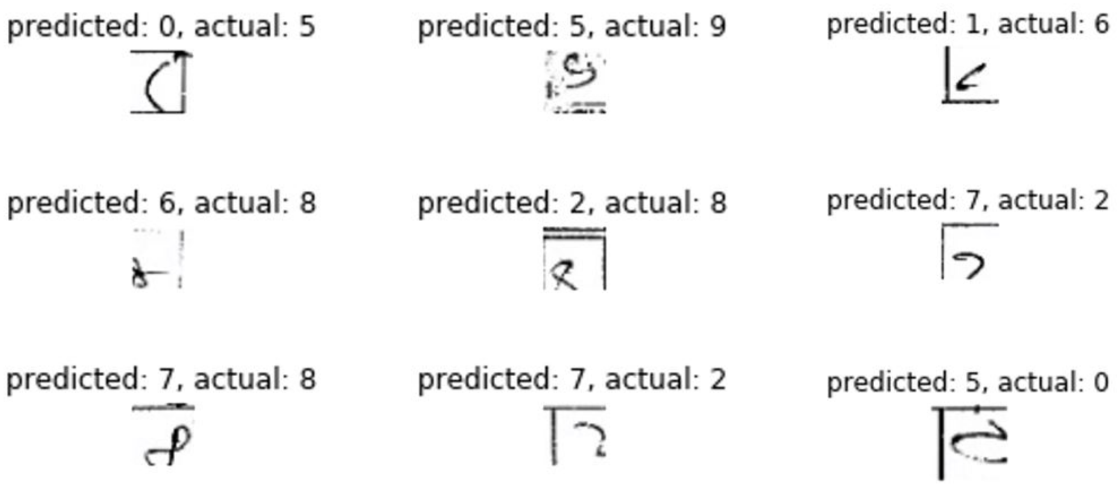


Figure 3: Some of the issues found after examining predictions of the base VGG model. Some of the issue were due to the incorrect cropping of digits (middle row) while some were due to rotation (top right, bottom left). Other incorrect predictions were due to either poor image quality or abnormal handwriting.

Convolutional Feature Extractor: We construct the component of the convolutional feature extractor by adopting the convolutional and pooling layers of VGG. However, our feature extractor has four convolutional layers. Experiments reveal that these layers can effectively extract the required features.

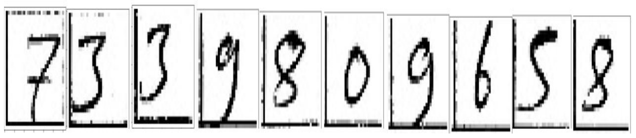
Recurrent Layer: We use a Gated Recurrent Unit [2] in the recurrent layer as it takes fewer parameters than the LSTM [5]. The recurrent layer predicts a label distribution for each frame in a feature sequence. A traditional RNN unit has a self-connected hidden layer between its input and output layers. Each time it receives a frame in the sequence, it updates its internal state h_t with a non-linear function that takes both current input x_t and past state h_{t-1} as its inputs: $h_t = g(x_t, h_{t-1})$. Then the prediction is made based on h_t. We have used a bidirectional GRU with 128 hidden states and two layers.

Data Augmentation: We used the similar augmentations described for the individual digits. We first augment the digits and then reformulate the sequence with the augmented images of the digits (see 4). The augmentation of the data ensures robustness to overfitting.

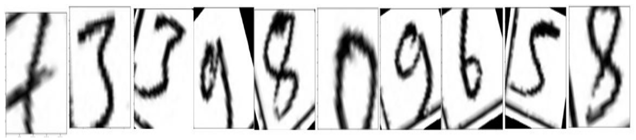
Loss Function: We have used the CTCLoss described by [4].

4. Dataset Description

We have worked on a private dataset of handwritten digits. Each image is a grid of 16×10 cells. Each cell is of size 32×32 pixels, in which image content is of size 30×30 pixels, and a white border of 1 pixel. A black cell implies that there is no image available for that cell. Figure 5 shows a sample image. We have total 989 such images which amounts to 15824 rows of size 10 each. However, 3 rows didn't have any annotations, so we removed those and got total 15821 rows.



(a)



(b)

Figure 4: (a) Original sequence of digits and (b) augmented sequence of digits.

Individual Digits: For individual digit recognition, we extracted the individual digits from each of the images. The individual digits were 158210. After removing the black cells, we got 74595 images of digits between 0 to 9.

Sequence of Digits: For sequence of digit recognition, we extracted the rows and treated them as single data samples. Each such sample is of size 32×320 pixels. We have a total of 15821 such image samples.

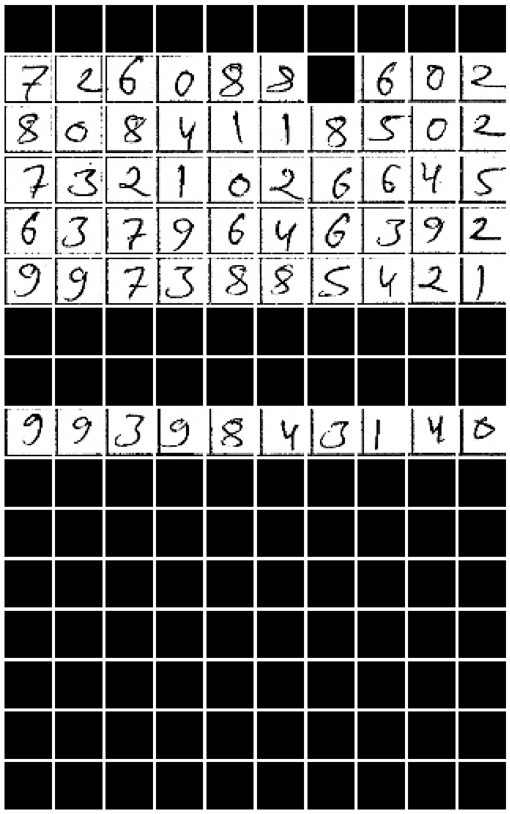


Figure 5: Sample image from the dataset.

5. Experiments and Results

5.1. Implementation Details

We have used PyTorch to implement all our models. The trained models can be found in the Onedrive folder here [1]. All the code can be found here [2]

Individual Digits: For the LeNet architecture, we simplified the original sub-sampling layers using average pooling. For individual digit recognition, we used a standard cross-entropy loss. For training the LeNet model, we used an Adam optimizer with a fixed learning rate of 0.0001, a batch size of 32 for 15 epochs. For the modified VGG-based architecture, we used a Stochastic Gradient Descent optimizer with a momentum of 0.9 and a learning rate of 0.001. The network was trained for 15 epochs with a batch size of 64.

Sequence of Digits: For the CNN-RNN architecture to recognize a sequence, we used the Connectionist Temporal Classification Loss or CTC Loss. The network was trained using an Adam optimizer with a learning rate of 0.001 for 30 epochs. The batch size was 64.

[1] Trained Models

[2] https://github.com/sbasu276/sil801-A1-digits

Data Augmentation: We have used the rotation, resize, random crop on the individual digit images of our dataset. We augmented individual digits of the sequence for a sequence of digits and then aggregated them in the same order to form the sequence back.

Transfer Learning: For the individual digit recognition, we have tried training from scratch as well as using pre-trained networks on MNIST data.

5.2. Evaluation Metric

We have used accuracy as the evaluation metric for the models. We consider the prediction to be correct only if all the digits in the sequence are predicted correctly in the same order for a sequence of digits. We report 5-fold cross validation results for statistical significance.

5.3. Accuracy of Classifiers

The improved VGG model achieves a 5-fold average accuracy of 99.3% on the individual digits. The CNN + RNN model achieves a 5-fold avergae accuracy of 93.7% on the sequence of digits.

Model	**Avg. Accuracy**	**Std. Dev**
LeNet (Baseline)	0.940	0.002
LeNet (Fine-tuned)	0.948	0.003
VGG-16 (Vanilla)	0.990	0.004
VGG-16 (Improved)	0.993	0.005
CNN+RNN (sequence model)	0.937	0.016

Table 1: Comparison of the 5-fold accuracy for different models.

6. Conclusion

In this work we present architectures for classifying both individual handwritten digits as well as a sequence of handwritten digits. The several design interventions chosen to counter the issues present in the dataset allow the models to have improved performance over the standard LeNet baseline. The proposed CNN+RNN architecture allows for accurate classification of a sequence of digits. The CNN+RNN architecture proves to effective and yet lightweight. The experiments done show the efficacy of the proposed methods and how they are able to provide a significant improvement over baseline methods.

References

[1] Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven. Photoocr: Reading text in uncontrolled conditions. In *Proceedings of the ieee international conference on computer vision*, pages 785–792, 2013. 1

[2] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Gated feedback recurrent neural networks. In *International conference on machine learning*, pages 2067–2075. PMLR, 2015. 3

[3] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3642–3649. IEEE, 2012. 1

[4] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376, 2006. 3

[5] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012. 1

[7] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1, 2

[8] Baoguang Shi, Xiang Bai, and Cong Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2298–2304, 2016. 1, 2

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[10] Paul Voigtlaender, Patrick Doetsch, and Hermann Ney. Handwriting recognition with large multidimensional long short-term memory recurrent neural networks. In *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 228–233. IEEE, 2016. 1

[11] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. End-to-end text recognition with convolutional neural networks. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 3304–3308. IEEE, 2012. 1