A New Supervised Classification Model for Modified MNIST

Yuxiang Ma, Qifan Wang, Shenshun Yao 260709204

McGill University, Montreal, QC, CAN

Abstract

MNIST dataset is a large system of database which contains numeric digits from 0-9 for training image processing system. In this project, we were tasked to design and validate a supervised classification model that finds the number occupied the most space in each image from Modified MNIST dataset. Each image in Modified MNIST dataset contains more than one digit. Our model is based on modifying the structure of ResNet and using different hyper-parameter setting. The results were evaluated by hold-out validation. The best model of us is a modified ResNet which reported an accuracy score of 97.066 on the final test set.

1. Introduction

We performed an image analysis prediction in this project based on a modified MNIST dataset. Our goal was to find which number occupies the most space in the image from our modified MNIST dataset.

We started by introducing some related Neural networks inspiring models such as AlexNet, VGG and ResNet. Then, we used hold-out validation to separate the dataset into training set and validation set. Moreover, we preprocessed the training set by taking affine transformation with random shearing on training set and normalizing gray scale values on both training set and validation set. Our network design followed the design of ResNet, with a modified structure of ResNet-50 to fit the modified MNIST dataset. One highlight of our model is to use 3×3 kernel instead of the original 7×7 kernal which leads to high stability, lower number of converge epochs, and higher accuracy. We also found that although shearing increases accuracy on validation set, the convergence rate of loss and stability would decrease. Because of the limited computational power, we regrettably have not tried the much deeper network structures e.g. ResNet-101 and ResNet152. We may also investigate how the depth of network structure affect the prediction results.

2. Related work

There are various Neural networks for image classification, including AlexNet, ResNet, and VGGNet. Understanding of concepts from these models enhanced our design process and model quality.

AlexNet[5] introduced a deep convolutional neural network with several tricks. The authors concluded that Deep CNN with ReLUs train several times faster than their equivalents with tanh or sigmoid units; Training on multiple GPUs takes less time than a single GPU and Dropout exhibits substantial overfitting but increases the number of iterations required to converge. The key idea of Dropout is to randomly drop units (along with their connections) from the neural network during training [9]. These conclusions inspired us for our model construction.

Email addresses: `yuxiang.ma@mail.mcgill.ca` (Yuxiang Ma), `qifan.wang2@mail.mcgill.ca` (Qifan Wang), `shenshun.yao@mcgill.ca` (Shenshun Yao 260709204)

[1]This is the report for miniproject 3 of Comp 551 Applied Machine learning, written by group 41.

[2]All three members contributed to the project quite equally. Qifan Mainly contributed to model construction, Yuxiang and Shenshun ran the experiments and tested the results. The report were written together.

VGG[8] mainly motivated us in evaluation of networks of increasing depth using an architecture using very small (3×3) convolution filters which makes the improvement over AlexNet by replacing large kernel-sized filters(11 and 5 in the first and second convolutional layer, respectively) with multiple 3×3 kernel-sized filters consecutively.

ResNet[3] mainly introduced a so-called "identity shortcut connection" that skips one or more layers, such that the networks are easier to be optimized, and accuracy are improved by increasing depths. We chose this model as our basic model, many detail have to be discussed.

3. Dataset and setup

Modified MNIST dataset consists of 50000 images of handwritten digits, in which 40000 training images are labeled with the largest handwritten digit in the image, and 10000 images are unlabeled and to be predicted by the trained model. Images are gray scale and 64×64 size such that it can be represented by a $1 \times 64 \times 64$ tensor, in which every entry ranges from 0 to 255. Training images were divided into training set (39000 images with labels) and validation set (1000 images with labels). For hyperparameter tuning and model selection, hold-out validation based on validation set was used.

Different preprocessing approaches were used for training set and validation set. For training set, we took affine transformation with random shearing in range of -15 to +15 degree, normalization of gray scale values between -1 to 1 for transformed images, and the transformed images were divided into batches with size 64 . These setup approaches refer to [7]. In comparison, for the validation set, we only used normalization of gray scale values to -1 to 1 for transforming images, with batch size of 128.

4. Proposed approach

Our network design follows the design of ResNet. We chose ResNet-50 as underlaying structure. We modified the structure of ResNet-50 to fit the modified MNIST dataset. Details of our network structure will be discussed in the following subsections.

4.1. ResNet

ResNet was first proposed by [3], and achieve beat-human performance. It will achieve super performance by adding more layers in the network. Their final model has 152 layers. However, as shown in their paper, adding more layers to the CNN model caused degradation problem. They used a residual learning to tackle this problem. Residual learning adds a short cut from the input layer to the output layer of a stack of network layers. ResNet uses an identity mapping for short cut. Identity mapping only performs an element-wise addition function[3], yet the short cut do not add any parameter or computational complexity to the model when solving the degradation problem. Another advantage is that the short cut allows gradient propagate directly from the output layer to the input layer[4].

4.2. Modification

The input for the ResNet was a 224×224 pixels image with 3 channels from ImageNet in the original paper [3], while our data is 64×64 pixels image with 1 channel, therefore we modify the first convolutional layer of the ResNet-50. Since our input data size is smaller compare to the ImageNet dataset, we need to use a smaller kernel to extract the feature. Also, the max poling may drop some important information, for the sharpness of the original modified MNIST data is not high. Adding a max pooling layer may emit some important feature. The detail of our first layer includes: The original one is a 7×7 kernel with stripe sets to 2, follow a max pooling layer of size 3×3 with stride 2. We change it to a 3×3 kernel with stripe 2, without a max pooling layer. We also change the padding correspondingly. Our experiments show that by changing the kernel size to 3×3 and remove the max pooling layer contribute to 2% of accuracy in the validation sets.

We have also tried ResNet-101 and ResNet-154 in our experiments. Due to the limited computing power, we use the original setting of the model. We do not change the kernel size of the first layer and keep the max pooling layer. The result in validation set is 1% less compare to our final model.

4.3. Implementation

The image size is unchanged for input.we decided to perform data augmentation since we found some digits are tilt. We use random shear[3] of range -15 to 15 degree for data augmentation. We also normalized our data to $[-1, 1]$. We use standard SGD as our optimizer with momentum. The initial learning rate is set to 0.3, with momentum 0.1. After the validation error flat, we decay the learning rate to 0.1 and change the momentum to 0.9 follow the setting in the original paper. Again, when the validation error plateaued, we decay the learning rate to 0.001 with momentum 0.9. We use weight decay of 0.0001 in our model. The model is trained 45 epochs in respecting to the whole training set. For weight initialization, we follow the paper [2].

Pytorch[7] only have transform function on image data set. The transform function does not support transform on tensordataset directly. Therefore, we add a wrapper class for the tensordataset class. The wrapper class getitem function will perform the transformation for image and return the transformed image and original label.

5. Results

Cross Entropy loss was used as loss criteria. We ran the following experiments to get the best performance model, and the figures illustrated the results.

5.1. Shearing

We started by comparing effect of shearing and non-shearing on accuracy and loss at preprocessing level. From red and green curves in figure 1, Loss of 3×3 kernel without shearing decreased faster than loss of 3×3 kernel with shearing in range 1 epoch to 10 epochs, yet both of the losses remained stable after 20 epochs. Similarly, from red and green curves in figure 2, percentage of correctly classified examples on validation set increased faster when shearing was not applied at preprocessing, and both of the ratios converged after 20 epochs. In terms of numerical accuracy, percentage of correctly classified images with shearing oscillated from 96.2% to 96.4%, while accuracy remained at 96.1% for last 22 epochs when shearing not applied. Therefore, shearing increased accuracy on validation set while decreasing the convergence rate of loss and stability.

5.2. Kernel

Regarding the size of kernel at first layer, we compared original ResNet 7×7 kernel and modified ResNet 3×3 kernel. According to red and blue curves in figure 1, loss of 3×3 kernel decreased slower but more stable, while 7×7 kernel reached a spike at second epoch at 3.1144 and decreased with oscillations. However, according to red and blue curves in figure 2, accuracy ratio (fraction of correctly classified validation examples) increased slower in 7×7 kernel, which was inconsistent with result from loss, and this increase was still not stable with oscillations. For both 7×7 kernel and $3\times$ kernel, loss and accuracy converged before 30 epochs. For final accuracy, 7×7 kernel only achieved 93.2% accuracy, while 3×3 kernel achieved 96.4% accuracy. Therefore, 3×3 kernel has high stability, shorter convergence epoch, and higher accuracy than 7×7 kernel.

[3]https://pytorch.org/docs/stable/torchvision/transforms.html

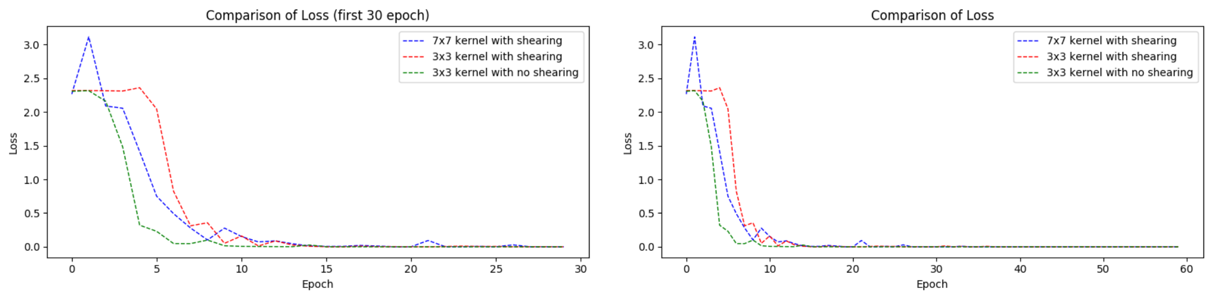

Figure 1: Comparison of cross entropy loss among 7×7 kernel and 3×3 kernel and 3×3 kernel without shearing

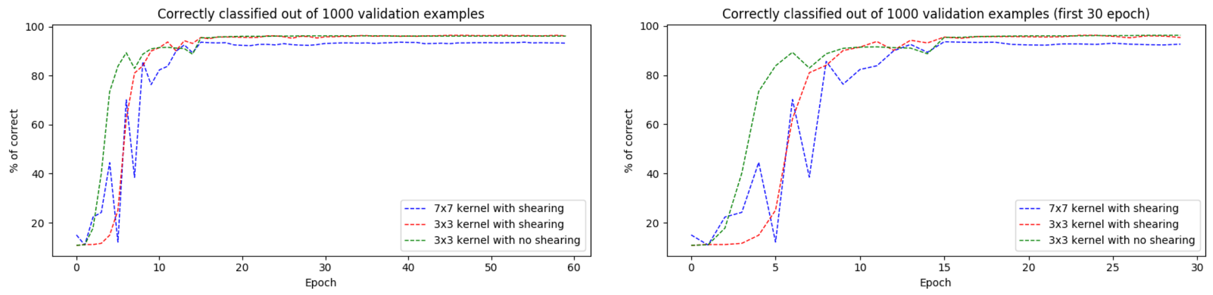

Figure 2: Comparison of percentage of correctly classified examples among 7×7 kernel and 3×3 kernel and 3×3 kernel without shearing

6. Discussion and Conclusion

In this project, we developed a model for classifying largest handwritten digit character in MNIST dataset. The model was based on modified ResNet on first layer input channel and kernel size. Random affine transform (shearing) and normalization were used to preprocess data, and mini-batching and batch normalization were used when training the model to improve accuracy and rate of convergence. 2 model variants including 3×3 kernel and 7×7 kernel were compared and validated with held out validation set with size 1000. According to result on validation set, we concluded that modified ResNet with 3×3 kernel and random shearing increased prediction accuracy, for 3×3 kernel retained more information of original image during filtering, and random shearing avoided overfitting by adding randomness. This model achieved 0.97066 on Kaggle Leaderboard.

First, it is a good practice to maintain as much information at the first layer by maintaining the size of output larger or equal to size of input. This can be done by properly calculating kernel size, padding, and stride according to formulas under description of Conv2D function on website[4]. Note that if stride is 1, the padding required for a layer that size of outputs is the same as size of input can be calculated by $\frac{(\text{kernel size}-1)}{2}$. According to our result, more information leads to lower number of epochs before convergence and higher accuracy.

Second, using randomized method at preprocessing stage on training set can increase prediction accuracy, probably by preventing overfitting of the training set. Randomized linear transforms forced the learning process towards learning the more general features, rather than some overfitting features, leading to a lower variance and a more reasonable bias. This is an example of bias-variance trade off at preprocessing stage. Also data augmentation can teach model invariants in the data domain[1].

[4]https://pytorch.org/docs/stable/torchvision/transforms.html

In the future, with higher computational power, we could try training ResNet-152 and ResNet-101 to investigate relation between depth of network and prediction accuracy. Also, the accuracy and convergence of traditional machine learning method, for example, SVM, on MNIST dataset could also be investigated [6].

References

[1] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

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

[6] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[7] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[8] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[9] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.