COMP 551 MINI PROJECT 4

Shelley Xia, Yiwei Cao, Mingze Li

1 Introduction

Ensemble methods, a machine learning approach that uses multiple algorithms, have been shown to achieve better accuracy than any of the constituent machine learning algorithms alone [1]. In the work done by An et al. [2], three different convolutional neural networks (CNN) models with different kernel sizes are employed. The corresponding accuracy of each of the three CNN models alone and in ensembles is computed and compared. It is found that an ensemble composed of three different CNN models (termed heterogeneous ensemble) performs better than an ensemble composed of three of the same CNN models (termed homogeneous ensemble). Both of the ensembles perform better than any of the three CNN models alone. In the present report, we replicate the work done by An et al. [2] for the reproducibility challenge.

Following their work, we test the performance of the three models alone and in ensembles (both heterogeneous and homogeneous) on the same dataset, the widely-used MNIST handwritten digit data set. Our results confirm their findings and reach the same accuracy as theirs. Our results, too, show that ensembles perform better than any of the single models, and a heterogeneous ensemble outperforms the homogeneous ensemble.

2 Scope of reproducibility

The paper we chose aimed to improve the image classification accuracy on the MNIST dataset by using ensemble networks.

Main claims of the original paper:

- An ensemble of three CNN networks with the same architecture (a homogenous ensemble network) predicts labels for the MNIST dataset with higher accuracy than a single network.
- An ensemble of three CNN networks with different kernel sizes (a heterogenous ensemble network) predicts labels for the MNIST dataset with higher accuracy than single networks and homogenous ensemble networks.
- The best test accuracy can be achieved by using a two-level ensemble method, where three different homogeneous ensemble networks are combined to form a heterogenous ensemble of ensemble networks.

Additionally, the paper also examined the impact of network structure, data augmentation, and batch normalization. They found that 1) networks without max pooling can achieve higher accuracy in general, 2) using both translation and rotation to augment the data yields higher accuracy than using only one of the augmentation methods and not augmenting the data at all, and 3) removing batch normalization decreases test accuracy.

We focused on reproducing the first two main claims.

3 Methodology

In all experiments, we used the author's code as basis. In our replication studies, we used the identical code provided by the authors. We ran them according to the ReadME in the github repository. For exploration including hyperparameter tunning and ablation studies, We touched up on the code to make some minor changes such as the number of epoches and trials to comply with our available computational resources. Hardware-wise, we used 2 virtual machines, one with GPU and one with CPU, to complete the following experiments in the results section.

3.1 Model descriptions

The models and algorithms used in this paper were based off three Convolutional Neural Networks (CNN). These three basis networks have kernel size of 3, 5, and 7 respectively, so they are denoted by M3, M5, and M7. The number of

33rd Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

layers in each network is adjusted such that three networks have a similar number of inputs in the final fully-connected layer. Specifically, M3 has 10 convoluted layers, M5 has 5 convoluted layers, and M7 has 4 convoluted layers. In our ablation studies that will be presented later, we tried to change the number of convoluted layers. All convoluted layers use ReLU as the activation function. Thus far, individuals networks are not special. The uniqueness of the paper comes from "ensembles", which employs a similar idea as random forests. Ensemble networks are groups of multiple models used together when making predictions, usually achieved by majority voting. There are 4 types of ensemble strategies discussed by the paper: M3+M3+M3, M5+M5+M5, M7+M7+M7, M3+M5+M7. The paper also explores the effects of batch normalization and data augmentation through translations and rotations. To maximize model accuracy, all three models are trained on augmented data and include batch normalizations in all layers.

3.2 Datasets

The dataset used in the paper is the famous benchmark dataset MNIST handwritten digits. The task is to classify images with handwritten digits to its correct class from 0 to 9. It is comprised of 60,000 training images and 10,000 testing images. Each image is a 28x28 pixel grayscale image. In order to improve model learning, the authors use data augmentation techniques such as image translations and rotations. They found that using both augmentation schemes improve model performances. Their findings on impacts of data augmentation can be found in the paper, so we did not investigate the effects of data augmentation due to computational limitations. Both the raw and processed dataset can be downloaded from `https://github.com/ansh941/MnistSimpleCNN/tree/master/data/MNIST`

3.3 Hyperparameters

The hyperparameters associated with network architecture are discussed in section 3.1 and manipulated in the ablation studies. Here we focus on the hyperparameters associated with training the networks and data augmentation.
In the original paper, the authors used the Adam optimizer. The learning rate was initialized to 0.001, with a decaying factor of 0.98. The batch size was 120, and each model (M3, M5, and M7) were trained for 30 trials, each trial with different initial parameters. This yields 30 networks for each type of model. In each trial, the model was trained for 150 epochs.
The authors also augmented the original dataset by performing random translation and rotation. For translation, an image is shifted vertically and horizontally, and the size of the shift is randomly selected between 0 and 20% of the image size. For rotation, the image is rotated either clockwise or counterclockwise, and the size of the rotation is randomly selected between 0 and 20 degrees.
For the hyperparameter tuning section in our project, we decided to first test the effect of changing the optimizer to RMSprop and setting the initial learning rate to 0.01. We trained model M3 and model M5 with this setup, and each model was trained for 3 trials and with 50 epochs in each trial. The reduction in the number of trials and the number of epochs is mainly for the purpose of reducing runtime, but can also be seen as a form of ablation study. When we were training model M7, the server crashed down. Therefore, unfortunately, we were unable to obtain the results for the heterogeneous ensemble network after hyperparameter tuning. We also planned on trying out other optimizers such as Adagrad and SGD, as well as testing out different learning rate decaying factor, batch size, and parameters for data transformation. However, we were unable to perform these experiments due to lack of time. In the results section, we predict how the results will change due to these manipulations.

3.4 Experimental setup and code

We directly downloaded the code from the github url the authors provided: `https://github.com/ansh941/MnistSimpleCNN/tree/master/code`. We then installed the required packages using pip: torch, torchvision, and torchsummary. To train one network instance, such as M5, we ran the command "python3 train.py –seed=0 –trial=1 –kernel_size=5 –gpu=0 –logdir=modelM5". To train a group of networks of the same type (same kernel), we modified the number of "–trial" to train multiple models by setting random seed to different values. Results were recorded in a "logs" folder. Next, we ran the test using "python3 test.py –seed=0 –trial=10 –kernel_size=5 –logdir=modelM5" to save the indices of the misclassified images in a corresponding text file for future use by ensembles. We repeated these steps for M3 and M7. Lastly, we ran "python3 homo_ensemble.py –kerne_size=5" for homogeneous ensembles and "python3 ensemble.py" for heterogeneous ensembles. The above steps conclude the replication experiments. To conduct ablation studies and hyperparamter tunning, we modified the code accordingly and ran code in a similar fashion.

3.5 Computational requirements

We had two virtual machines set up to run these experiments, one with GPU and one with CPU. The replication experiments and hyperparameter tunning used the GPU whereas the ablation studies used the CPU. For replication, one

set of training of one model (M3 or M5 or M7) with 150 epochs and 10 trials took on average 8-9 hours. In comparison, testing and ensembles took rather negligible time. The total number of GPU hours spent for replication was around 27 hours. Hyperparameter tunning had similar computational requirements. As for ablation studies done in the CPU, one set of training of M3 with 50 epochs and 3 trials took around 20 hours. We surmised this was due to the steep increase in the number of parameters to train when removing one convoluted layer. M5 and M7 with 50 epochs and 3 trials took on average 10 hours. In general, we found that running with CPU was much slower than with GPU.

4 Results

Overall, our experiment results reproduce the general trends of results presented in the original paper. Yet numbers differ slightly: our test accuracy for the ablation studies was slightly higher than original results. We acknowledge that this nuance could be due to numerical issues and randomness. We conclude that our results support the main claims of the original paper.

4.1 Results reproducing original paper

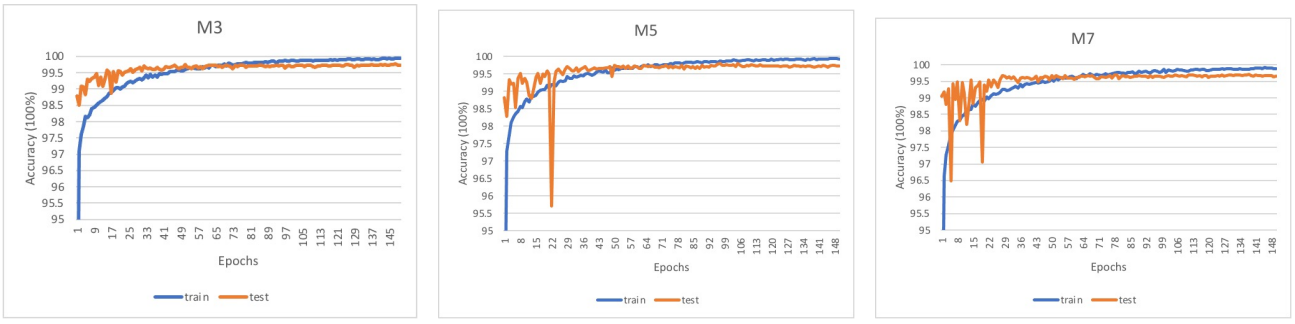
We focused on reproducing two sets of experiments: the individual networks with 150 epochs and 10 trials and the ensemble networks basing off these individual networks. Both sets of experiments replicated the main claims made in the original paper.

4.1.1 Individual Networks

First, we trained individual networks of each type using the same set up described in the paper: identical network design, batch size=120, optimizer=Adam, learning rate=0.001, epoch=150, and trial=10. Results are presented in table 1. They are highly similar to the original results taking into account of uncertainties. Some model differences are also captured: M3 has the best accuracy followed by M5. Yet surprisingly, our results suggest M7 outperforms M3 and M5, which is not the case in the original paper. We also observe the same trend of progressing in test accuracy with respect to epoch, as illustrated in figure 1. In particular, models with larger kernel sizes are more unstable during early epoch and their patterns become the same after 50 epoch.

model	min	avg	max
M3	99.62	99.71	99.78
M5	99.63	99.72	99.80
M7	99.52	99.77	99.90

Table 1: Test accuracy of networks measured between 50 epoch and 150 epoch



(a) M3 (b) M5 (c) M7

Figure 1

4.1.2 Ensemble Networks

There are four types of ensemble networks: combination of three M3 networks, combination of three M5 networks, combination of three M7 networks, and combination of a M3, a M5, and a M7 network. Same as the original paper, we tested 1000 ensemble networks for each type. Our results support the first two main claims in Section 2. As can be seen

from Table 2, the accuracies of the homogeneous ensemble networks (M3+M3+M3, M5+M5+M5, M7+M7+M7) are higher than the individual networks (M3, M5, M7), and the heterogeneous ensemble network (M3+M5+M7) yields the highest accuracy. Our results are also numerically close to the results presented in the paper.

	best	original best
M3	99.81	99.82
M5	99.80	99.80
M7	99.77	99.79
M3+M3+M3	99.84	99.86
M5+M5+M5	99.84	99.86
M7+M7+M7	99.82	99.85
M3+M5+M7	99.85	99.87

Table 2: Test accuracy of networks in our experiment (first column) vs published results (second column)

Figure 2 provides a visualization of the second main claim in Section 2. It shows the distribution of test accuracy of the 1000 ensemble networks for each type. These lines are obtained by fitting a normal distribution to the data. As can be seen from the graph, our results show that the heterogeneous ensemble network (line in blue) generally performs better than the homogeneous networks (i.e. the distribution is further to the right).

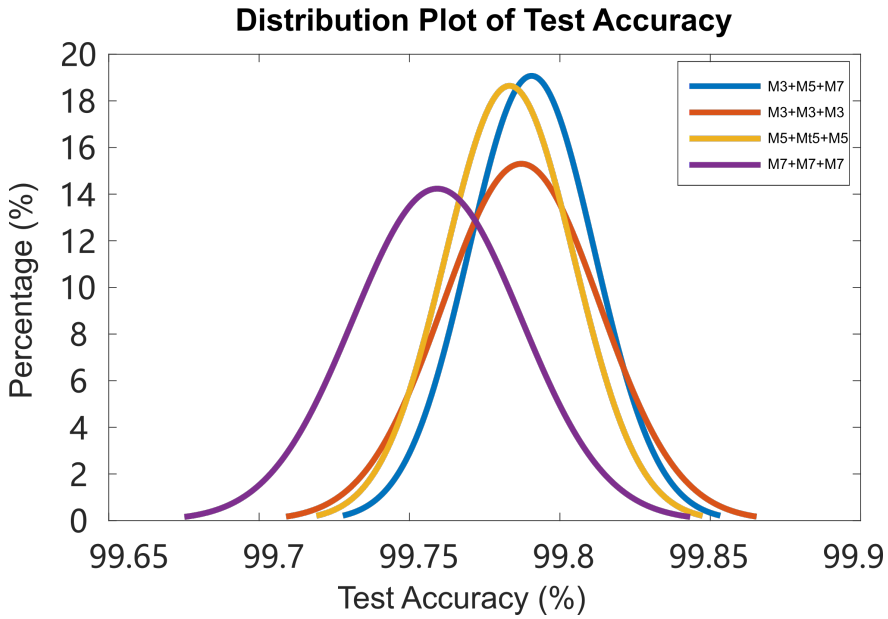


Figure 2

4.2 Results beyond original paper

Some experimental components in the paper are not explicitly specified. For example, the authors do not reason their choices of the structures of the networks: the number of convoluted layers, activation functions, batch sizes. They also did not include why they chose certain hyperparameters values, such as Adam as the gradient descent method, learning rates, and rotational angles in data augmentation. Due to limited computational resources, we were unfortunately unable to investigate all of these factors that might have led to different results. We chose a subset of them: the number of convoluted layers and gradient descent methods.

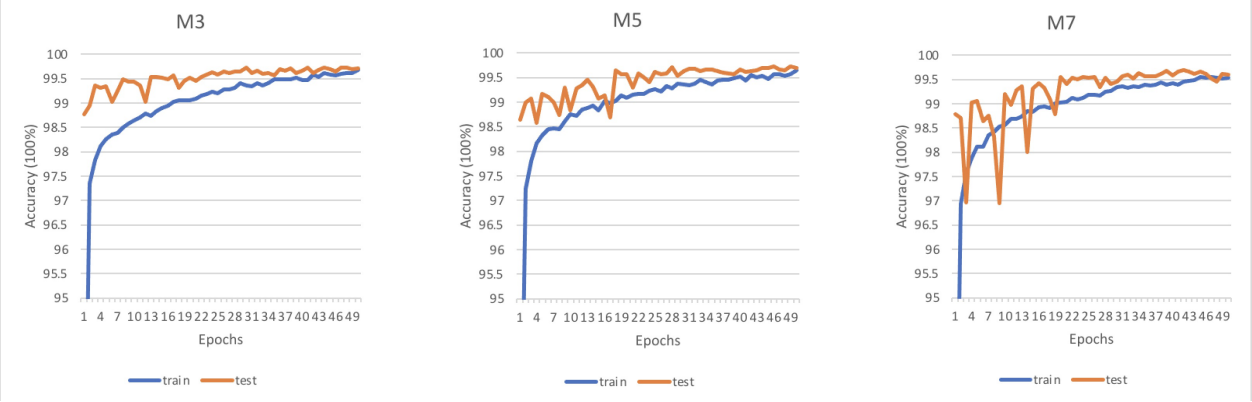
4.2.1 Ablation Studies: Removing a convoluted layer

The authors do not state the reason behind their network design: 10 convoluted layers for M3, 5 convoluted layers for M5, and 4 convoluted layers for M7. We examined the effects of the number of convoluted layers in model accuracy by removing one convoluted layer in each type of model. Note that this reduction still guaranteed comparable feature sizes

input to the final fully connected layer, as required in the paper. We were unable to run experiments under the same settings as the paper due to computational limitations. We trimmed down the number of trials from 10 to 3 and epochs from 150 to 50 because the authors found patterns became similar after 50 epochs. Results of individual networks with such modifications are shown in table 3. Overall model accuracy increases after reducing the number of convoluted layers. Trends of test accuracy with respect to epochs during training are illustrated in figure 3. We found similar trend as the authors that networks with larger kernel size show more instability in early epochs.

model	min	avg	max	best	original avg
M3	98.77	99.52	99.73	99.73	99.45
M5	98.57	99.42	99.73	99.73	99.36
M7	96.95	99.25	99.69	99.69	99.33

Table 3: Test accuracy of networks measured between 0 epoch and 50 epoch in training



(a) M3 (b) M5 (c) M7

Figure 3

After obtaining results for individual networks, we tested the performances of homogeneous and heterogeneous ensembles. Due to computational constraints, we had only 3 trials for each type of networks. Since each ensemble groups 3 networks, we only had a single result for each type of homogeneous ensembles and 27 (3*3*3) results for heterogeneous ensembles instead of 1000 (10*10*10) in the original paper. Our findings support authors' claim that higher accuracy could be achieved by combining results from three networks of the same type. Similarly, the highest accuracy was observed when combining one network from each type of networks. Results are shown in Table 4.

ensemble strategies	average accuracy	best accuracy
M3+M3+M3	99.80	99.80
M5+M5+M5	99.80	99.80
M7+M7+M7	99.77	99.77
M3+M5+M7	99.79	99.82

Table 4: Test accuracy of ensemble networks

4.2.2 Hyperparameter Tuning

As mentioned in section 3.3, for this part of the project, we were only able to train the M3 and M5 models with a different optimizer, RMSprop, and a different initial learning rate, 0.01. This is due to the lack of computational power and time. The results are shown in Table 5.

We predict that if the learning rate is too large, then the models may fail to converge, and when the learning rate is too small, the model may take too long to train. Using the SGD optimizer would result in more oscillations in the gradient descent process. Setting the data transformation parameters higher (shifting the image further or increasing the degree of rotation) may make the model more robust. Finally, if we test out different kernel sizes, we would expect that larger

	min	avg	best	original avg
M3	34	96.59	99.71	99.45
M7	93.2	99.32	99.74	99.36

Table 5

kernel sizes would lead to greater instability in test accuracy early in the training process, but become more stable in later epochs.

5 Discussion

In general, our experimental results support the claims and findings of the original paper. We were able to replicate most of the experiments the original authors performed except for those whose required computational power and time exceed our given resources in the class. We performed some additional ablation studies and hyperparameter tuning to test the robustness of the proposed algorithms.

One of the most surprising findings of ours is that our ablation studies found that reducing the number of convoluted layers increases test accuracy. We surmised that this was due to less compaction through convolution and thus more features are passed in the fully connected layer.

5.1 What was easy

The paper selected was easy for us to follow and replicate. The concepts involved in the paper, convolutional neural networks and ensemble algorithms, are covered extensively in our course. All of these accumulated knowledge and experience equipped with the adequate power to comprehend the flow and the algorithms of the paper.

Moreover, the code was well modularized and commented on. The given github site even provided thorough instructions on how to run their scripts from the command lines. This made the beginning of our replication experiment go smoothly.

5.2 What was difficult

Estimating the necessary time and computational power and resource needed to run the experiments is difficult. To enhance our computational power, we tried 0) CS department server on mimi 1) our local systems with 0 GPUs 2) Google Colab with 1 GPU 3) the lab server belonging to a lab one of our group members is in 4) Google Cloud Virtual Machine (credits given by COMP551). Though we found that the lab server has the greatest computational power, the sheer amount of computational work required by the chosen experiments made it hard to complete. Thus, we had to drop some of the experiments presented in the original paper. To use our time and computational resources to the fullest, we split up the experiments and run them on both Google Cloud and the lab server simultaneously.

Despite the effort to fully utilize our time and computational power and resources, some of our replication and ablation experiments still took longer than expected. All of the above mentioned limitations led to our inability to fully replicate every experiment.

5.3 Communication with original authors

We did not communicate with the original authors. If we were to communicate with them, we would discuss some of our questions, which includes 1) how they calculated the uncertainties in table 1 of their original manuscript 2) what is the difference between the usage of the words “max” and “best” in their manuscript, as the ambiguity caused some of our confusions.

6 Statement of Contribution

Shelley focused on individual networks and ablation studies; she wrote the methodology section and corresponding parts under results section. Mingze selected the paper, performed some of the replication work, and wrote part of the manuscript. Yiwei reproduced the results, tried to run the hyperparameter tuning experiments, and wrote section 2, part of section 3, and part of section 4.

References

[1] Opitz, D.; Maclin, R. (1999). "Popular ensemble methods: An empirical study". Journal of Artificial Intelligence Research. 11: 169–198. doi:10.1613/jair.614.
[2] An, Sanghyeon Lee, Minjun Park, Sanglee Yang, Heerin So, Jungmin. (2020). An Ensemble of Simple Convolutional Neural Network Models for MNIST Digit Recognition.