EE590: Deep Learning - Mini-project1

Runke Zhou, Shanci Li, Ruizhi Luo
Master of Communication Systems, EPFL, Switzerland

I. Project Introduction

The goal of this project is to test the performance of different models on the task of comparing two digits in a pair of images. Data used in this project contains multiple pairs of images for digits from the MNIST dataset. Models are expected to predict whether the first image is less or equal to the second. The main objective of this project is to show the improvement on the performance after using weight sharing and auxiliary loss. The performance of different models on this task are compared with the baseline model, the simple Multi Layer Perceptron (MLP).

II. Dataset Structure

The data used in this project is from the MNIST handwritten digits dataset. The training set and test set both contains 1000 pairs of digits. Each set of data is made up of input, targets and classes. To be specific, the input is multiple pairs of 14*14 grayscale images. The target is the label indicating whether the first image is lesser or equal to the second with 0 and 1. The classes is the class of two digits from 0 to 9.

III. Implementation of Basic Deep Learning Models

A. Multi-Layer Perceptron (MLP)

Multi-Layer Perceptron(MLP) is used as the baseline in this project. The basic model of Multi-Layer Perceptron in our project is composed of three full connected layer. The structure of the MLP is shown in Figure 1. The input is reshaped into [1,392] and then enters into three full connected layers to output the binary prediction of the input. The cross-entropy loss is used to compute the loss between the prediction and the true target. The performance of Multi-Layer Perceptron (MLP) can approach 97.9% on the training set and 76.5% on the test set on average.

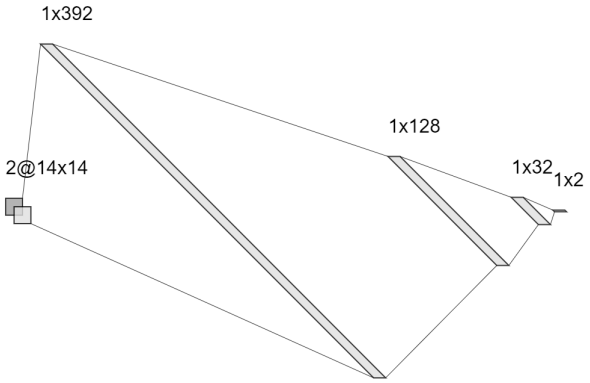

Fig. 1: The structure of Multi-Layer Perceptron

B. Basic Convolutional Neural Networks

A more effective basic model is Convolutional Neural Network. It's more widely used in image classification tasks to extract 2D features. The convolutional layer is more suitable to extract the features of images than the simple full connected layer. The basic convolutional neural network contains two convolutional layers, two max-pooling layers and two full connected layers. The structure of the CNN model is shown in Figure 2. The loss is also calculated by cross-entropy loss. The performance of Convolutional Neural Network(CNN) can approach 99.99% on the training set and 81.62% on the test set on average.

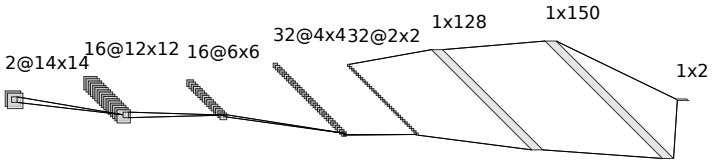

Fig. 2: The structure of Basic CNN

IV. Improvement on Basic Models with Weight Sharing or Auxiliary Losses

Basic models, MLP and CNN , are very basic network models to extract information from images.

But they are not very suitable for this specific project. Thus we add weight sharing and apply auxiliary losses to basic convolutional neural network which can be specially designed for this specific project.

A. CNN with weight sharing

Weight sharing is a very useful technique in deep learning to use the same weights for two layers or two networks and reduce the number of parameters needed in a network model. An important feature of our task is that every input data of our network is a pair of images which both contain handwritten digits from the MNIST dataset. In addition, the task of our network is to compare the pair of digits, which implies that the siamese neural network[1] is very suitable for this situation.

The siamese neural network is made up of two neural networks shared the same weights in parallel to process two input vectors and compare their output in the end[1]. Inspired by the siamese neural network, we design our model to process two separate images from the pair of input and feed into two identical neural networks and concatenate their output to process the final result. The architecture of our model is shown in Figure 3 without implementing auxiliary losses. The performance of CNN with weight sharing can approach 100% on the training set and 84.16% on the test set on average.

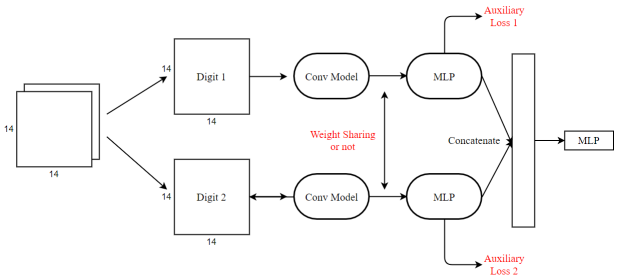

Fig. 3: The structure of CNN_WS, CNN_AL and CNN_WS_AL

B. CNN with auxiliary losses

The effectiveness of networks relies on extracted features from input data. With multiple convolution layers in the basic CNN, the key problem is that local features extracted in the first layers of the network might vanish in the training process. Auxiliary loss is one of effective measures to solve this problem[2].

The structure of CNN with auxiliary losses is shown in Figure 3 without weight sharing. As shown in Figure 3, there are three loss functions including two auxiliary loss functions and one principle loss function. Provided with classes of input data, we extract the output of two separate images in a pair and calculate the auxiliary loss of the predicted class and the true class of two digits in a pair. Three loss functions are all computed using the cross-entropy loss. The performance of CNN with auxiliary losses can approach 100% on the training set and 86.71% on the test set on average.

C. CNN with weight sharing and auxiliary losses

The final model is built with implementing both weight sharing and applying auxiliary losses. The architecture of the final model is shown in Figure 3 with both weight sharing and auxiliary losses. The weight sharing is implemented between the convolutional layers and the MLP on two separate digits. Auxiliary losses are calculated from the predicted class on two separate digits and their true labels. The performance of CNN with weight sharing and auxiliary losses can approach 100% on the training set and 92.42% on the test set on average.

V. Results and Discussion

A. Model Optimization

After building models, many methods are used to improve the training process and the final performance of models.

- **Batch Normalization:** Applying batch normalization can normalize the inputs of specific layers and make the training of networks faster and more stable. Batch normalization can also prevent the over-fitting of the model.
- **Learning Rate Scheduler:** The learning rate of some models are set to decay based on the number of epochs. Using this method can improve the efficiency of training by making the step of learning large in the beginning and reduce to lower learning rate after few epochs to prevent the model from fluctuation in the end of training.
- **Hyper-parameter Tuning:**A key problem in deep learning is to tune a large number of hyper parameters to optimize designed models. In our project, we apply grid search to find the

best hyper-parameters in our model. In each experiment, we run the model for 15 iterations with 45 epochs in each iteration. The optimal hyper-parameters and the average accuracy of testing set for each model is shown in Table 1.

	MLP	CNN	CNN_WS	CNN_AL	CNN_WS_AL
lr	0.01	0.01	0.05	0.05	0.05
batch size	100	50	50	50	100
Test accuracy	76.5%	81.62%	84.16%	86.71%	**92.42**%

TABLE I: Comparison between performance of models

As we can conclude from the table, CNN has slightly better performance than the baseline model, MLP. Adding weight sharing and auxiliary losses can both improve the performance of the model.

B. Effect of Weight Sharing and Auxiliary Loss

To compare the improvement on basic models with weight sharing and auxiliary loss, we evaluate these models for 15 iterations and compare their accuracy on test set. The result is shown in Figure 4 and Table 2.

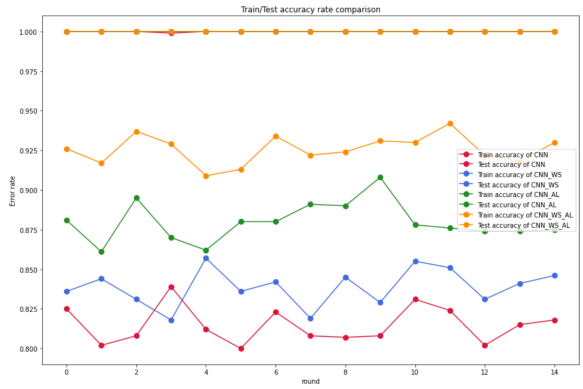

Fig. 4: Comparison of Effect of weight sharing and auxiliary losses

Models	Average Accuracy	std
CNN	81.62%	1.38%
CNN_WS	84.16%	0.86%
CNN_AL	86.71%	1.29%
CNN_WS_AL	92.42%	1.07%

TABLE II: The average and std accuracy of models

As shown in the table and the figure, although both weight sharing and auxiliary losses can improve the performance of model, applying auxiliary losses have better performance than just applying weight sharing. So to obtain the best performance, we apply both weight sharing and auxiliary losses and obtain the accuracy of 92.42% $\pm$1.07%.

VI. Conclusion

The main objective of this project is to test the performance of different models on comparing two digits from the MNIST dataset and evaluate the influence of technique, weight sharing and auxiliary loss, on the performance of model. In this project, we use the multi-layer percepton as the baseline, which achieves the accuracy of 76.5%. What's more, a basic CNN model is implemented, achieving 81.62%, which shows that basic CNN is a nice basic model for this project. With implementing weight sharing and auxiliary loss, the performance can be improved a lot. With weight sharing, the performance can be improved to 84.16%. It shows that for this kind of task to compare two similar objects, models implementing weight sharing, like siamese model[1], can have better performance. With auxiliary loss, the performance can be improved to 86.71%. This shows that models introducing auxiliary losses with local features of neural network can help to improve the performance of models. Finally, to obtain the best model, the performance can be improved to 92.42%.

Another important feature is that except MLP, our models all tends to over-fit, because the number of data in our project is comparatively small compared with the large number of parameters in networks. So other techniques used to prevent over-fitting may possibly improve the performance of models.

References

[1] D. Chicco, *Siamese Neural Networks: An Overview*. New York, NY: Springer US, 2021, pp. 73–94. [Online]. Available: https://doi.org/10.1007/978-1-0716-0826-5_3

[2] M. M. Bejani and M. Ghatee, "Least auxiliary loss-functions with impact growth adaptation (laliga) for convolutional neural networks," *Neurocomputing*, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221001880