Evaluating Impacts of Weight Sharing and Auxiliary Loss on Neural Networks

Bohan WANG
bohan.wang@epfl.ch

Ke WANG
k.wang@epfl.ch

Anqi Hou
anqi.hou@epfl.ch

***Abstract*—Weight sharing and auxiliary loss are two techniques that can improve neural networks. In the report, the impact of implementing weight sharing, and of the use of an auxiliary loss are tested using the MNIST database.**

I. Introduction

In the project, the performances of different architectures to compare two digits visible in a two-channel image are evaluated. Each architecture aims to predict whether the first digits is less or equal to the second digits. The input is of tensor dimension $N \times 2 \times 14 \times 14$, the prediction target is of dimension $N \times 1$, with value 0 or 1, and the predicted class dimension is $N \times 2$, with value 0-9. Seven different neural network architectures are constructed to compare the performances and test the influence of weight loss and auxiliary loss on prediction accuracy. All experiments are done with 1,000 pairs for training and testing. All architectures are estimated through 15 runs where both data and weight initialization are randomized.

Section II briefly introduces several important methods necessary to implement the project. Section III details the 7 models. Section IV shows the results. Section V compares and discusses the results of different models.

II. Methods

Weight Sharing: In weight sharing method, only one shared set of model parameters is trained for different layers which greatly reduces the number of parameters involved in a model [2]. For the two input networks, models force them to share same weights in all layers, meaning that training one network updates parameters for both. The second network is trained using those same weights. It can not only help the model converge better but also let the model generalise better.

Auxiliary Loss: The auxiliary loss adds extra gradient flow by attaching small network to the output of that module during backpropogation, helping to reduce gradient vanishing problem and to increase training stability. In the project, the prediction loss for the prediction class is taken as the auxiliary loss.

Implementation All neural networks are built using the Pytorch package. We set the number of epochs to 25, since 25 epochs are enough for networks to converge. SGD optimizer is the optimizer for all models. Instead of applying SGD on the entire training data, Mini-batches are used to get higher efficiency. During the back propagation, Cross Entropy loss is selected to compute total loss, as well as auxiliary loss.

III. Models

- **CNN, SiameseWS, SiameseWSAL1, SiameseWSAL2**

CNN The convolutional neural networks (CNN) model is the base architecture without implementing weight sharing and auxiliary loss. The input to CNN is with tensor dimension, $N \times 2 \times 14 \times 14$ (N is the mini batch size). The first layer is a convolutional layer, activated by Relu. After that, a 2d batch normalization is applied to make artificial neural networks faster and more stable. The second layer is another convolutional layer, activated by Relu and followed by a batch normalization. Then we add a layer of type Max pooling to down sample the input image. The last two layers are fully connected layers. The output of the first fully connected layer is also activated by ReLu. Finally, the output of the model is 2 dimensional.

Siamese network version 1: The model is constructed based on the previous CNN model, adopting the weight sharing method. One main difference is that this model splits the pair input into two tensors each with tensor dimenstion, $N \times 1 \times 14 \times 14$. It has the same layers as the CNN method. The number of the input channel at the first convolution layer is 1 for each tensor. One more step after activating the output layer in the CNN model is to concatenate the two 1D tensors. And two additional fully connected layers are added as the final output layers.

Siamese network version 1 auxiliary losses: The auxiliary loss measures the accuracy of digit class prediction for each image. Without auxiliary loss, only the accuracy of the output target 0 or 1 matters. However, with auxiliary loss, the neural networks is trained based on three prediction results.

Siamese network version 2 This model still adopts both weight sharing and auxiliary loss, but this model is not as deep as the previous model. The model only considers the two auxiliary losses generated from predicting class of two digits. Then, the predicted classes of the two digits are used to determine the target. However, the loss obtained from the predicted output target is not considered during the back propagation. Thus, a less deep

model is produced. The neural networks architecture is the same as the CNN model.

- **Resnetblock, Resnetblock weight share, Resnetblock weight share and auxiliary loss**

  Resnetblock Typical ResNet models are implemented with double- or triple- layer skips that contain nonlinearities (ReLU) and batch normalization in between [1]. In the project, shortcuts are utilized to jump over some layers. The Resnetblock model is the base model for our second category models before employing weight sharing and auxiliary loss method. Same as the first CNN model, the input to the Resnetblock model is with tensor dimension, $N \times 2 \times 14 \times 14$ (N is the mini batch size). The first layer is a convolutional layer. We do a batch normalization on that layer and activate the layer by Relu. After that another convolutional layer is added with batch normalization. Then, we use shortcut to add the input and the residual, and activate it by Relu. Next, a Max Pooling layer of kernel dimension 2 is used to get an abstract form of image. Finally, we flatten the last step output and add the last two fully connected layers. The first fully connected layer is followed by ReLu activaion function.

  Resnetblock weight share Adopting the method we used in the model Siamese network version 1, we get a Resnet block model with weight sharing. It has the same layer construction as the Resnetblock model. The only difference with Resnetblock model is that the input channel at the first convolution layer is 1.

  Resnetblock weight share and auxiliary loss We applied the same method as the Siamese network version 1 auxiliary losses model.

Below table shows the different architectures.

Models	**Architectures** (Output channels of conv, fc, use_bn)	**Number of parameters**
CNN	(32, 64, 25, True)	59373
Siamese network	(16, 32, 50, 25, True)	46033
Siamese network with auxiliary loss	(16, 32, 50, 25, False)	45937
Siamese network (predict class by comparing predicted digits)	(16, 32, 50, 25, True)	46033
Resnet block	(16, 16, 80, False)	65666
Resnet block with weight sharing	(16, 16, 80, True)	66260
Resnet block with weight sharing and auxiliary loss	(16, 16, 80, True)	66260

TABLE I

We use five folds cross validation to find the hyperparameters. The best mini bacth size is chosen from the set {50, 100, 200} for each model. Also, the best learning rate for SGD to train each model is chosen from {1e-1, 1e-2, 1e-3, 1e-4}. Whether a model uses batch normalization is also determined by cross validation. Finally, the weights for final loss and auxiliary losses are chosen from the set, {[1,1], [1,0.75], [1,0.5], [1,0.25], [1,1.25], [1,1.5], [1,2]}.

IV. Results

In table 2, the average of testing accuracy of different architectures are presented.

Models	**Mean accuracy**	**Standard deviation**
CNN	82%	1%
Siamese network version 1	86%	2%
Siamese network version 1 with auxiliary loss	92%	2%
Siamese network version 2 (predict class by comparing predicted digits)	**97%**	**1%**
Resnet block	79%	2%
Resnet block with weight sharing	86%	1%
Resnet block with weight sharing and auxiliary loss	88%	1%

TABLE II

As shown in table 2, the mean testing accuracy of the baseline model (CNN) is 82% just higher than that of Resnet block model. With weight sharing, both Siamese network and Resnet block with weight sharing can achieve 86% testing accuracy. With addition of auxiliary loss, the performances of weight sharing models can be improved. The accuracy of Siamese network increased from 86% to 92%, and that of Reset block with weight sharing growed to 88%. Noticeably, the highest testing accuracy (97%) is achieved by the Siamese network which predicted the classes of the pairs of 14×14 grayscale images, and compared them to determine whether the first digit was lesser or equal to the second. In addition, all the seven models are stable, because the highest standard deviations of the testing accuracy estimated through 15 rounds is 2%.

In figure 1 and figure 2, the training and testing performances of the models estimated through 15 rounds are shown. It is obvious that the initial losses of Siamese version 2 and Siamese network with auxiliary loss are higher (about 35.0). However, they can achieve higher testing accuracy. Furthermore, the training loss and testing accuracy of Siamese version 2 change smoothly with the increasing epochs. The testing accuracy of other models fluctuates during the 15 rounds estimations. Therefore, the Siamese version 2 is more stable than other models.

V. Discussion

In the project, we compared three types of representative architectures (which are original Convolution networks, residual

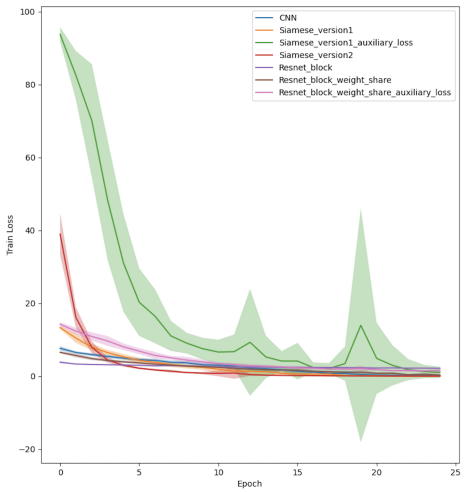

Fig. 1: Comparison of training losses of different architectures

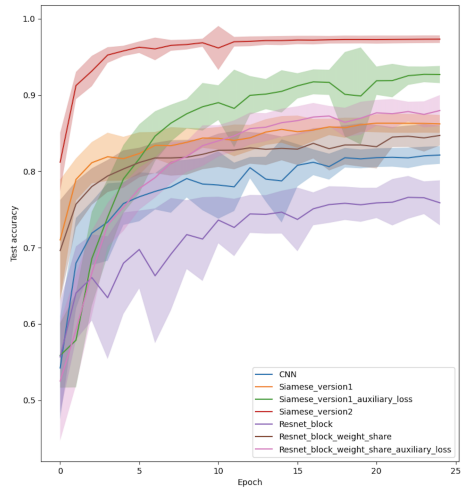

Fig. 2: Comparison of test accuracy of different architectures

networks and Siamese networks). Based on our findings, the standard deviations of test accuracy are low (from 1% to 2%) for each architecture estimated through 15 rounds, where both data and weight initialization are randomized. This benefits from the convolutional layers and pooling layer in each model. The convolutional layers can learn shift-invariant local features and build robust representations of the input image. The pooling layer can make the model more robust to variations in the position of the learned features of convolutional layers.

The performances of the models can be improved through weight sharing. In table 2, both Siamese network and Resnet block with weight sharing can achieve 86% test accuracy. In figure 1, it is noticeable that the training losses of all the models can convergence to 0 or a tiny small value, so the overfitting occurs when testing the models. Weight sharing can reduce the number of parameters in a model, which can help avoid overfitting caused by the complextivity of a model. As shown in table 1, the numbers of parameters of all Siamese network based architectures are fewer than those of other architectures. The increased learnable parameters in Resnet block with weight sharing come form the additional fully connected output layer, compared to Resnet block without weight sharing.

The performances of the models can be also improved through auxiliary losses. As shown in table 2, the Siamese network with auxiliary losses can achieve 92% mean test accuracy. Also, the Resnet block with weight sharing and auxiliary loss can achive 88%. The weighted auxiliary loss (weighted by some value less than 1) from the intermediate layer in a model can help avoid vanishing gradient. Therefore, the auxiliary loss can reduce the number of dead neurons in a model. In our models, we add the auxiliary loss from the full connected layers, which predict the labels of the pairs of the images, to the final loss. When the loss of predicting the target is small, the auxiliary losses can help gradient flow to keep convolution layers learning the representative features of the input images.

Noticeably, the performances of the model can be improved significantly by the Siamese network version 2 which only uses the predicted label of two input images to determine whether the first digit is lesser or equal to the second. The test accuracy of Siamese network version 2 is 97% much higher than those of other models. The Siamese network is simpler to predict the class of the two input images, compared to predict for each pair if the first digit is lesser or equal to the second. This is indicated by figure 1. In figure 1, the training of Siamese network version 2 is faster than those of other models.

Finally, the short connections introduce by residual networks can not improve the performance. As shown in table 2, the test accuracy of Resnet block is lower than that of CNN. Also, the test accuracy of Resnet block with weight loss is lower than that of Siamese network. The short connections can help train deeper networks. In the project, the model is not very deep, so the short connections can not accelerate training and improve generality.

References

[1] *Residual neural network - Wikipedia*. En.wikipedia.org. URL: https://en.wikipedia.org/wiki/Residual_neural_network(accessed:%2020.05.2021).

[2] Ram Sagar. *What Is Weight Sharing In Deep Learning And Why Is It Important*. analyticsindiamag.com. URL: https://analyticsindiamag.com/weight-sharing-deep-learning(accessed:%208.05.2021).