DenseResNet: A Neural Network Based Method for Artifact Reduction in Video Compression

Mengwei Yuan
460101132

Shirui Cheng
460304157

He Zhao
460290609

Abstract

The compression technology has been developed and significantly improved recent years with the demand of high speed transmission particularly in the video domain. However, a high compression rate is usually achieved by sacrificing the video quality. Many researches has been conducted to investigate the restoration methods which reduced the compression artifact so that compensates the drawback of video compression. As one of the most innovative and effective approach, the neural network based artifact removal achieves high scores in many restoration tasks. In this study, three network models including ARCNN, FastARCNN and DnCNN are analysed and tested by a dataset in JPEG and HEVC format. Based on the analysis result and theoretical hypothesis, a new model named DenseResNet is proposed and tested by the same dataset, which shows a better result compared with classical network models. The study also illustrates the potential of neural network approach in restoration tasks of many other types of compression format.

1. Introduction

The digital communication technology has been dramatically grown in past decade and has reached a large number of end users. Data, as one of the most valuable resources generated and transmitted by the communication system, become more and more diverse in terms of its format and contents. As a consequence, the amount and the size of data is also increased, particularly for the data in video format. According to the estimation of Cisco, the video will take more than 80% of data that is transmitted through the internet in the near future [1]. Although the infrastructure of communication system is well developed and keep being improved recent years, the demand of high-speed transmission is still difficult to achieve without the compression technology. However, there is a trade-off by using different compression technologies [2]. The lossless compression reserves the quality of the original data with zero distortion but with a limited compression rate. On the other hand, lossy compression is more popular for its high compression rate but with the cost of sacrificing the data quality (Figure 1). Therefore, some restoration methods are proposed by studies as a compensation to the distortion created by the lossy compression.

Figure 1: Distortion introduced by lossy compression method. Original image (left) and compressed image by JPEG method (right)

In the video and image domain, one of the data restoration approaches is by utilizing the neural network model. In this paper, we analysis and evaluate some state-of-the-art neural network models in artifact reduction tasks. Then based on the previous literature and initial experiments result, a best model is selected as our backbone structure for further improvement. By exploring training strategies, we improve the performance of that network. In addition, we propose a new model which achieves a better performance comparing with the classical models.

In data pre-processing, we theoretically compare the effect of some data augmentation methods and make some hypothesis on them, which are later proved experimentally by evaluating the result of model on a separate test dataset.

In the experiment, hyper-parameters such as batch size, initial learning rate, kernel size and network structures are investigated. The study also compares the performance of model trained by different learning rate schedulers including cosine, multi-step and exponential scheduler.

The following sections of this report are arranged as below. Section 2 provides a synthesis of the existing artifact reduction theory and methods. Section 3 introduces the backbone structure of benchmark and proposed network. Section 4 illustrates training methods including the selection of pre-processing techniques and hyper-parameters. Section 5 focuses on the result analysis and comparison between different training strategies. Finally, a conclusion summaries the project and some insight for future exploration will be indicated in Section 6 and 7.

2. Previous Work

Many classical network models has been invented and applied on the artifact reduction tasks such as artifact reduction convolutional neural network (ARCNN), FastARCNN and DnCNN [3][4][5]. These models utilizes serval techniques including residual learning [6], non-linearity enhancing [7] and layer decomposition [8], which improves the model capacity and ability therefore helps the models achieve better performance than others.

2.1. ARCNN

The ARCNN is a four-layer shallow model which is consist of four convolutional layers with the kernel size being 9, 7, 1 and 5 respectively (Figure 2). This model has the advantage of fast processing speed but limited in the performance compared with other deeper models such as DnCNN or CAS-CNN [4].

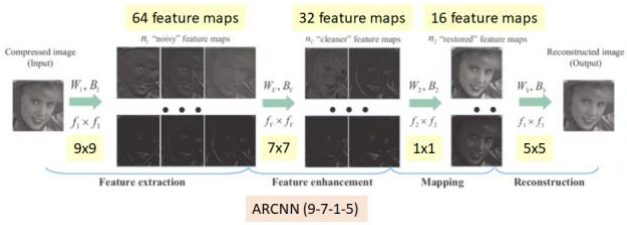

Figure 2: Architecture of ARCNN.

2.2. FastARCNN

Similar with the ARCNN, the FastARCNN is also a shallow model which has an additional layer with kernel size being 1 between the first and second layer of ARCNN (Figure 3). This configuration significantly reduced the number of parameters stored in the model while maintain the same coverage range [8]. This layer decomposition also enhance the non-linearity of the model so that improve the accuracy [7]. However, this model has same limitation on the restoration performance and even worse than the ARCNN on some artifact removal tasks [4].

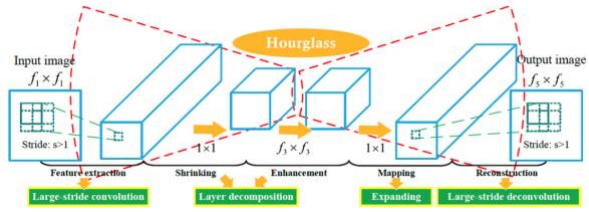

Figure 3: Architecture of FastARCNN

2.3. DnCNN

Another network model, DnCNN, resolves the limitation on the restoration performance by making the network deeper and more trainable parameters. The DnCNN baseline model has 15 convolutional layers with kernel size being 3 (Figure 4). This configuration effectively increase the capacity of the model and allow more latent features to be extracted.

The novelty of this model is that there is an identity map between the input and the output so that the model's behavior is changed to learn the residual between input and ground truth instead of predicting the ground truth directly [9]. In addition, as DnCNN is 15-layer network, this residual connection also significantly mitigates the vanishing gradient problem which is caused by small gradient after many times of mathematical computation on the features throughout a number of layers propagation and usually happened in deep network models [6].

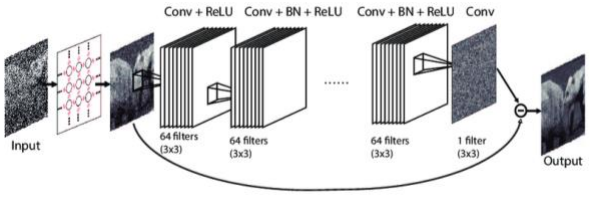

Figure 4: Architecture of DnCNN.

2.4. Dense Connection

On the side of the network structures, another advanced structure, the dense connections which also benefits the network model and alleviates some issues during the network training. The dense connection maps the output from a unit to the input of all the following units. Instead taking the output from previous unit, it concatenates all information from both dense connection and previous output as input. (Figure 5). This type of connection also prevent the vanishing gradient problem. Furthermore, it provides opportunities for feature reuse and therefore reduced the number of parameters substantially. In addition, This shortcut path enhance the propagation of features in each dense unit [10]. This architecture achieve a noticeable performance when applying to the artifact removal tasks [11].

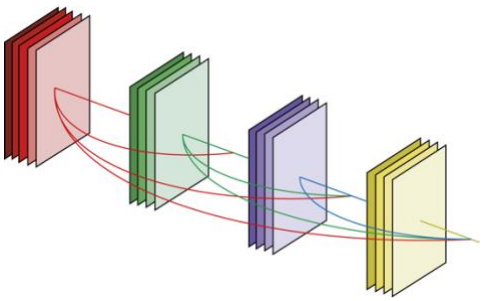

Figure 5: Dense connection

3. Backbone Structure

Based on the previous literatures and initial experiment (see Section 5), the DnCNN has the best performance and enormous potential to handle artifact reduction tasks. We also propose another new structure which combines the advantage of DnCNN and dense connection with the name of DenseResNet. Therefore, the DenseResNet and DnCNN are selected together as our backbone structures for further experiment.

3.1. DnCNN Structure

The DnCNN backbone model contains 15 convolutional layers in a sequence with 3*3 kernel size and 1 pixel padding around the boundary for each of them. The first 14 layers has 64 output channels, and the last layer has 3 output channels which matches the image dimension (RGB format). In addition, a ReLU activation function is added after each of the first 14 convolutional layers. The output and input are connected via an identity map (Figure 6).

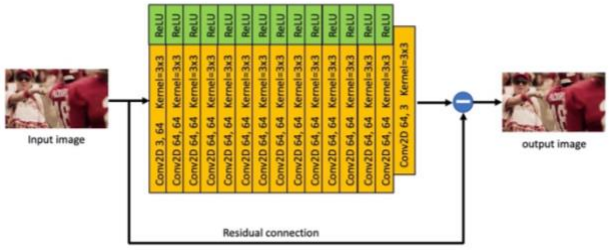

Figure 6: DnCNN backbone structure

3.2. DenseResNet Structure

The DenseResNet backbone model contains 14 convolutional layers in a sequence. Similarly with DnCNN, each layer in this model also contains the kernel with size of 3*3 and 1 pixel padding. Also, a residual connection is established between the input and output. Differently from DnCNN, the DenseResNet binds 3 adjacent convolutional layers as a convolutional block and densely connects the blocks together. With this configuration, the densely connected outputs are concatenated before outputting to the ReLU activation (Figure 7).

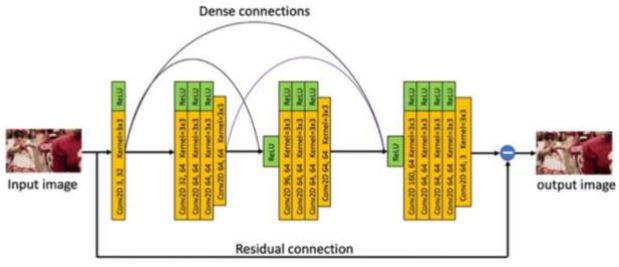

Figure 7: DenseResNet backbone structure

4. Methodology

By using the backbone structures in previous section, we identify the training strategies and make initial settings for experiments including some preprocessing techniques, loss function and evaluation metrics.

4.1. Data Preprocessing

There are 4 different data augmentation techniques are tested including the random cropping, center cropping, flipping (both vertical and horizontal) and ratating. The input images are first preprocessed by these operations and then passed to the model. The center crop and random crop generate a cropped image with height and width being half of the original image. The flip and rotating probability is set to be 0.5. In the experiment, 4 different combinations of these operations (random crop, center crop, random crop + flip and random crop + flip + rotate) are tested (Figure 8).

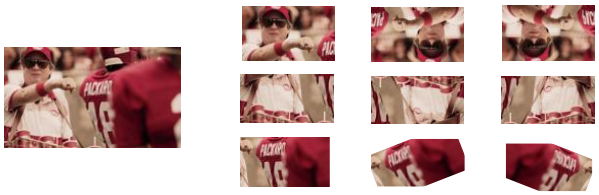

Figure 8: Examples of data augmentation methods

4.2. Loss function

The loss function during the model training is selected to be the mean square error (MSE) loss. The MSE is calculated by the equation:

$$MSE(x,X) = \frac{1}{3*m*n}\sum_{i=1}^{3}\sum_{j=1}^{m}\sum_{k=1}^{n}(X_{i,j,k} - x_{i,j,k})^2$$

where m and n are the width and height of the image; x is the predicted image and X is the ground truth image.

Since the output of such generative model is an image which has the same size and format with the ground truth image, the mean square error could perfectly reflect the pixel-wise difference between the prediction and ground truth image.

4.3. Evaluation Matric

To evaluate the artifact removal performance, the peak signal-to-noise ratio (PSNR) is employed to measure the difference between reconstructed images and ground truth images. The PSNR is calculated by the equation:

$$PSNR(x,X) = 20 * log_{10}(\frac{1}{MSE(x,X)})$$

5. Experiment and Result

5.1. Dataset

The JPEG images are selected as our dataset since the performance is generally better after initially training the model by JPEG images. In the later experiments, the validation images are fixed to be a 80-image dataset so that the input PSNR is unchanged each time during the validation. This makes the performance of model to be easily tracked and compared. In order to increase the speed of training, the number of images in training set is modified to be half of the provided JPEG images in the initial experiment for 3 network models. For rest experiment, the training set is the full set of provided JPEG images.

5.2. Initial Experiment

The initial experiment of 3 models are conducted with the default setting, which are batch size of 8 with 0.0001 learning rate, multistep learning rate scheduler and trained by 30 epochs with center crop data augmentation methods. The result (Table 1) shows that DnCNN has a much better performance than ARCNN and FastARCNN. To further confirm that the negative improvements of ARCNN is not caused by unoptimized parameters. Another set of experiment are conducted which varies the training batch size of ARCNN from 2 to 8. The result (Table 2) proves that the limitation of ARCNN comparing with the DnCNN, which is matched the previous statement in Section 2.3 concluded by reviewing the literature.

Model	ARCNN	FastARCNN	DnCNN
PSNR improvement (dB)	-0.1351	-1.55	**1.15**

Table 1: Initial experiment result of 3 different models.

Batch size	2	4	8
PSNR improvement (dB)	0.5543	0.4783	-0.1351

Table 2: Performance of ARCNN by different batch size.

5.3. Training Strategies Optimization

Data Augmentation Considering the dataset is still limit even if the full image set is used for training, some data augmentation methods is implemented which are expected to significantly increases the diversity of data without actually collecting new data. The performance of DnCNN model trained by different augmentation method is illustrated in Table 3. It could be noticed that the random crop contribute the majority of improvement over other augmentation methods because the random crop potentially increase the amount of data with a factor of 60 (the training is finished by 60 epochs and iterates 60 times of full dataset) while reserving the features in compressed image. The rotation may change the local feature of a compressed image and flipping does not significantly enlarges the data set. Based on these performance, the later experiments is set to be only with the random crop operation before input.

Augmentation methods	Center crop	Random crop	Random crop + flip	Random crop + flip + rotate
PSNR improvement (dB)	1.15	**1.2609**	1.2604	1.2596

Table 3: performance of DnCNN by different data augmentation method.

Learning Rate Scheduler Three commonly used schedulers are compared in this study: Multi-step, Exponential and Cosine. The default scheduler is Multi-step, in which the learning rate is multiplied by a factor of 0.2 on every milestone which is 10 epochs in this experiment. However, in order to tune the multi-step scheduler to its optimal settings, a large number of experiment need to be run to identify each steps. Another drawback of this type of scheduler is that the learning rate will never reach 0 so that hinder the learning step to reach the optimal point to some extent (Figure 9). On the contrary, the exponential scheduler adjust the learning rate to an extremely small value quickly within few epochs (Figure 9) where the model may not converged yet and still require larger learning rate, which make the training less effective. The cosine scheduler perfectly solve the problem of previous two schedulers. While maintaining a high learning rate at the first half of the training, it quickly reduces the rate at the second half and finally reach to 0 (Figure 9). The manner allows the model to be trained effectively at the beginning and fine-tuned at the end. The cosine scheduler, therefore, are selected to be the optimal and used in rest of experiment.

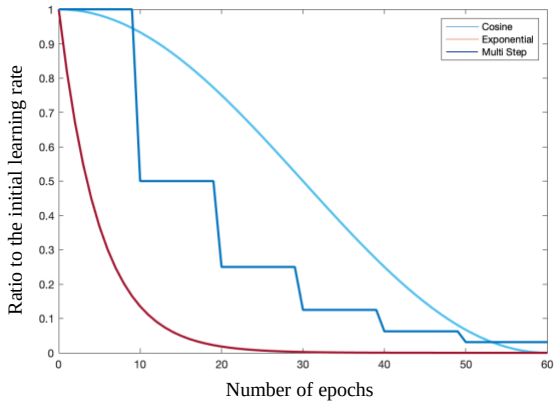

Figure 9: Comparison between different LR schedulers.

Batch Size & Learning Rate The experiment first fix the batch size to be 16 and adjust the learning from 0.0001 to 0.001. The result (Table 4) shows that the optimal choice for 16 batch size is the learning rate of 0.0003 at the beginning. The larger learning rate makes the gradient descent step bouncing around the optimal point, and smaller

Model	DnCNN							DenseResNet			
Learning rate scheduler	Multi step [step=7]	Multi step [step=10]	Cosine								
Batch size	16					8	4	16	8	4	2
Initial learning rate	0.0001	0.001	0.001	0.0003	0.001	0.0002	0.0001	0.0003	0.0002	0.0001	0.0001
PSNR improvement (dB)	0.8874	1.2601	1.5259	1.6858	0	1.6744	**1.7044**	1.5743	1.6781	1.7730	**1.8224**

Table 4: Performance of DnCNN and DenseResNet models under different experiment conditions

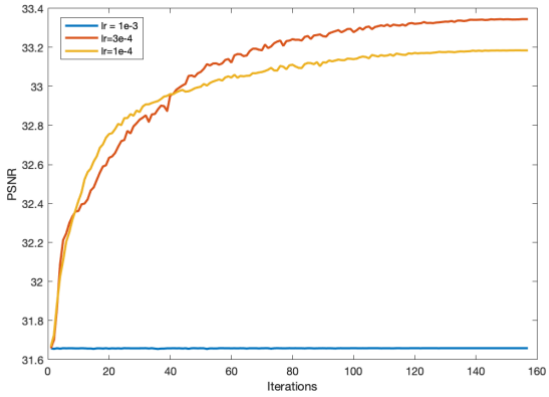

Figure 10: PSNR curves with different initial learning rate

learning rate converges early at a local instead of the global optimal (Figure 10).

Batch Size Batch size is the number of samples that model used to update its parameters each time. Within same number of epochs, larger batch size needs fewer iterations so that requires less amount of time for training and vice versa. From previous experiments, a batch size of 16 with learning rate being 0.0003 achieves the best performance. However, by observing the training curve of this settings in Figure 10 (red line), it shows that the model does not fully converged at the end of training and still has the trend of increasing. Therefore, the batch size is further adjusted to be smaller. As a consequence, the model will take more step during the training which potentially increase the probability to reach the optimal point.

Some research states that when changing batch size, learning rate should be changing simultaneously to achieve similar performance [12][13]. Hence the learning rate should be reduced together with the batch size. The experiment set the batch size and learning rate pairs indicated in Table 4. From the result, a trend can be noticed from the experiment that smaller batch size generally achieves better performance (Figure 11).

5.4. DenseResNet Tuning

As the number of layers and kernel size of DenseResNet is similar with the DnCNN model, the same settings are applied to the DenseResNet for training. The experiment fine-tune the model by selecting an optimal batch size and corresponding initial learning rate (Figure 12). The result is

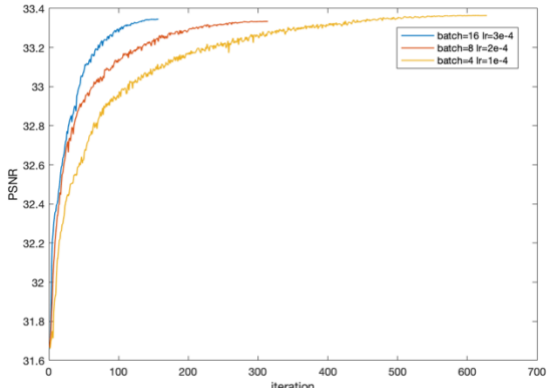

Figure 11: PSNR curves of DnCNN model with different initial learning rate and corresponding batch size

illustrated in Table 4, which suggests that the batch size of 2 outperform others with the best performance of 1.8224 PSNR improvement on the validation dataset. This result matches the observation from previous studies [14]. It states that the randomness introduced by using small batch size during the training may help the gradient decent jump out of the local optimal point and performed like an annealing process.

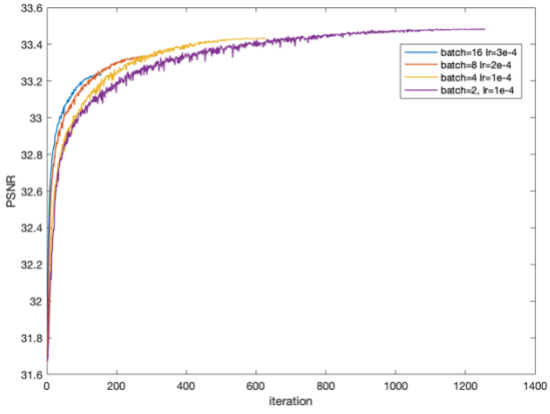

Figure 12: PSNR curves of DenseResNet model with different initial learning rate and corresponding batch size

5.5. Result Visualization

By visualizing the result (Figure 13), it could be noticed that, interestingly, although the deep generative model improves the image quality, it creates some secondary

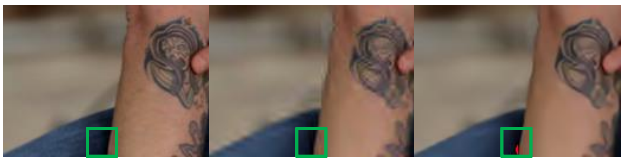

Figure 13: The secondary artifact generated by the neural network model. The ground truth image (left), the input image (middle) and the output image (right)

artifact which increases the degree of distortion in local area. This shows a direction for the future research.

6. Conclusion

Through this study, we analysis the performance and behavior of 3 different models, and by applying the dense connection technique, we propose a new models which has advantages over the classical models and achieve a higher score in artifact removal tasks on the given dataset. We also investigated and tuned the training strategies and hyper parameters to the optimal setting within limited training epochs and number of trainable parameters. The proposed model finally achieves a score of average 1.503 PSNR improvement on HEVC and JPEG images.

7. Future Work

Some insight about the further improvement of neural network models on artifact reduction task are inspired during the project. As mentioned in the Section 5, the training set only contains the JPEG compressed images without any HEVC images for the reason that the JPEG has a better training result compared with HEVC. However, the model might be more adaptive on both dataset if both HEVC and JPEG images are mixed in the training set. In this way, the trade-off between two types of images will be eliminated. Another approach is to train two models on HEVC and JPEG dataset respectively while adding a classifier before two generative models. The input image first pass through the classifier so that the model could recognize the format of input image and send the image to the corresponding generative model (Appendix B). If such classifier can be successfully trained with high classification accuracy, this approach could achieve high PSNR improvement on both HEVC and JPEG dataset.

References

[1] Cisco. The Zettabyte Era: Trends and Analysis. Technical report, 2017.

[2] N. Popitsch and A. von Haeseler, "NGC: lossless and lossy compression of aligned high-throughput sequencing data", 2020.

[3] J. Cheolkon, J. Feng, and Z. Li. "Gradient Guided Image Deblocking Using Convolutional Neural Networks.", 2019.

[4] T. Kim, H. Lee, H. Son and S. Lee, "SF-CNN: A Fast Compression Artifacts Removal via Spatial-To-Frequency Convolutional Neural Networks," 2019 *IEEE International Conference on Image Processing (ICIP)*, Taipei, Taiwan, 2019, pp. 3606-3610, doi: 10.1109/ICIP.2019.8803503.

[5] G. Lu, X. Zhang, W. Ouyang, D. Xu, L. Chen and Z. Gao, "Deep Non-Local Kalman Network for Video Compression Artifact Reduction", *IEEE Transactions on Image Processing*, vol. 29, pp. 1725-1737, 2020. Available: 10.1109/tip.2019.2943214.

[6] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[7] L. Min, Q. Chen, and S. Yan. "Network in network." *arXiv preprint* arXiv:1312.4400, 2013.

[8] S. Christian, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. "Going deeper with convolutions." *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1-9. 2015.

[9] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, July 2017.

[10] H. Gao, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. "Densely connected convolutional networks." *In Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700-4708. 2017.

[11] L. Junchi, and J. Deng. "Artifact reduction in brain magnetic resonance imaging (MRI) by means of a dense residual network with K-space Blending (DRN-KB)." *Medical Imaging 2020*: Image Processing. Vol. 11313. International Society for Optics and Photonics, 2020.

[12] G. Priya, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. "Accurate, large minibatch sgd: Training imagenet in 1 hour." *arXiv preprint arXiv:1706.02677*, 2017.

[13] H. Elad, I. Hubara, and D. Soudry. "Train longer, generalize better: closing the generalization gap in large batch training of neural networks." In *Advances in Neural Information Processing Systems*, pp. 1731-1741, 2017.

[14] K. Nitish Shirish, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. Tak Peter Tang. "On large-batch training for deep learning: Generalization gap and sharp minima." *arXiv preprint arXiv:1609.04836*, 2016.

Appendix

A. Group Contribution

The group is working coherently and the work is conducted by all three of us with equal contribution.

B. Classifier Before Models