Wide Residual Network for Image Recognition

Xiang Cui[1]*

Abstract
Conventional machine learning algorithms have limited ability to process the natural object image data in their raw form. Then the deep learning methods especially the convolutional neural networks(ConvNets) came out, which using multiple layers of nonlinear processing units to do the feature extraction and transformation starting with raw input. With composition of enough layers, very complex functions can be learned and could detect the object in the image quite well. However, deeper neural networks are hard to optimize and could cause the training accuracy degradation problems. Then the residual network learning frameworks(ResNet) which are easy to optimize was proposed and could improve accuracy performance from considerably increased depth. But very deep ResNet has a problem of diminishing feature reuse, hence we used a novel wide ResNet(WRN) which decrease depth and increase width of residual networks to get a better result for the image recognition.
For this final project, I will do the object image recognition for the CIFAR-10 Image dataset with 60000 images using a 16-layer-deep wide residual networks. And I will show this wide residual networks could have better test accuracy compared with the benchmark result using SVM, Random Forest, conventional ConvNets and the original ResNet.

Keywords
Image Object Recognition — Convolutional Neural Network — Wide Residual Network

[1] *Department of Statistics, University of Illinois at Urbana Champaign, Champaign, IL*
***Corresponding author**: xiangc5@illinois.edu

Contents

Introduction

Machine learning algorithms have influenced many aspects of modern life: from online search to social media, from text mining to recommendation systems on electronic commerce website. And it's also increasingly present in our daily life products such as smartphones, televisions and even cars. Machine learning algorithms are also used to identify objects in the images. However, the conventional machine learning algorithms have limited ability to deal with the natural image data in their raw form. It requires a well-designed feature extraction system to transform the raw data into a suitable feature vector from which the machine learning algorithm or classifier could detect the patterns from the input.

Then the deep learning methods neural networks came out with multiple hidden layers, simple obtained by composing many non-linear simple modules that each transform the raw input data at one level. This allows the machine to automatically discover the patterns that are needed for the classification from the raw data. Through composition of many non-linear modules, very complex functions could be learned which showed great performance in the classification or detection. For image classification, deep convolutional neural networks(ConvNets) like LeNet [13], AlexNet[12] and VGGNet [15] have led to a series of breakthroughs. The ConNets could enjoy the increasing high-performance computing powers such as GPUs and the distributed clusters[2] [3] as well as the current large scale public image repositories.

Deep ConvNets could integrate different level of features and the level of features could be enriched by the depth of the network. And there are evidence that could show that the network depth is important and image recognition could benefit from deep networks[15] [18]. However, with deep networks, the degradation problems arose. With the depth of a network increasing ,the accuracy first increases to get saturated and then unexpectedly degrades[5][17]. This degradation problem is not caused by overfitting, since a deeper network might lead to lower training accuracy. This degradation problem shows that the deeper networks are not easy to optimize. Then the deep residual network(ResNet)[6] framework was proposed. Instead of directly optimizing the desired underlying mapping, they tried to optimize the residual mapping, which is the desired mapping minus the input. And it's easier to optimize the residual mapping than to optimize the original desired mapping since to the extreme, they could optimize the residual mapping to zero which is equivalent to fit an identity mapping. Through this residual framework, the deep ResNet could gain more training accuracy from deeper network.

But for the deep ResNet, each single digit percent of improved training accuracy could cost nearly double the network depth and training very deep ResNet could have a diminishing feature reuse problem. Hence people came up with an idea to widen the convolutional layers in the ResNet by adding more feature planes. The wide residual networks(WRNs)[19] came out. Using the WRNs means we try to increase the width of ResNets by adding more feature planes in each convolution layer instead of just increasing the depth of the ResNet.

For this final project, I trained wide residual networks with 16 convolutional layers for the image recognition of the CIFAR-10 dataset and compared with the benchmark results from Machine learning algorithm SVM and Random Forest, conventional Convolutional Neural Networks like AlexNet and also original ResNet. The CIFAR-10 dataset used consists of 60000 32×32 color images in 10 classes with 6000 images per class. There are 50000 training images and 10000 testing images. The best testing accuracy from the wide residual networks is 91.8%, which has a great improvement compared with the benchmark result.

1. Methods

1.1 Convolutional Neural Network

The ConvNets are deep learning algorithms that are designed to directly process the raw data that comes in the form of multiple arrays, like the CIFAR10 image dataset which are color images composed of three 2 dimension arrays that includes pixel information from three color channels. The typical ConvNet are formatted as a series of hidden stage layers. For the first few hidden stages, it usually composes two types of layers: the convolutional layers and the pooling layers. First the convolutional layers are organized as feature maps. Within the feature maps, each unit could connect the local patches in the previous layer's feature map through a set of weights. These weights are called filter bank. All the units in the same feature map share the same filter bank. Through the feature maps, the convolutional layer could detect the local conjunction from the previous layer. Secondly, the pooling layer plays the role of merging similar features inside an area into one. The typical pooling unit is the maximum pooling unit which computes the maximum of a local patch of units. And typically, the results from the convolution layer would pass through a nonlinear units like ReLU[14] and then pass into the pooling layer. After several layers of convolution, non-linearity and pooling, the remaining part includes several convolutional layers and fully-connected layers that could compute the scores for each class. A typical ConvNets(AlexNet) is shown in Figure 1. The weights inside each filter banks are easy to train through a Backpropagation gradient descent optimization method.

The conventional ConvNets framework has lead to a series of breakthrough in the image recognition, like LeNet [13], AlexNet[12] and VGGNet [15]. The success comes from the key idea behind ConNets which take advantage of the natural figures that have local connections, shared weights and the use of pooling as well as the efficient use of GPUs, ReLUs[14] and the regularization techniques dropout[16].

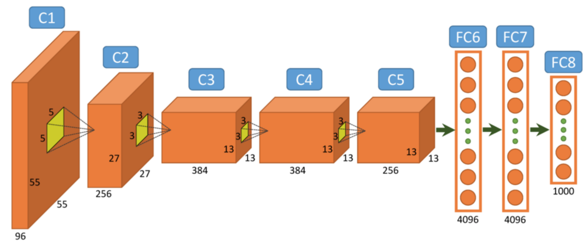

Figure 1. Typical ConvNets(AlexNet)[9]

1.2 Residual Network

For the conventional ConvNets structure, when the deeper networks begin to converge, a degradation problem comes out. As the depth of the ConvNets increasing, we could find that the accuracy first increases until gets saturated and then degrades unexpectedly. And this is not an overfitting problem since with more Convolutional layers, the training error increases. This degradation problem shows that the a too deep ConvNets is not easy to optimize. Because we could imagine that with more convolutional layers added to the ConvNets, the new added layers are just identity mapping and the already added layers are keep the same as before, then a deeper model could not produce higher training error. But the degradation problem truly was reported in [5][17]. To deal with this problem, the Residual Network framework(ResNet) was proposed[6].

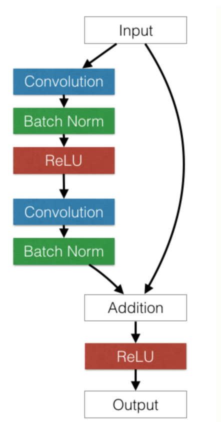

Figure 2. Typical ResNet Block

Suppose the desired underlying mapping is $\mathcal{H}(x)$, where x denotes the input to these layers. We could hypothesis that if multiple layers could approximate complicated functions, the it's equivalent to approximate the residual function $\mathcal{F}(x) = \mathcal{H}(x) - x$. This means instead of letting the hidden

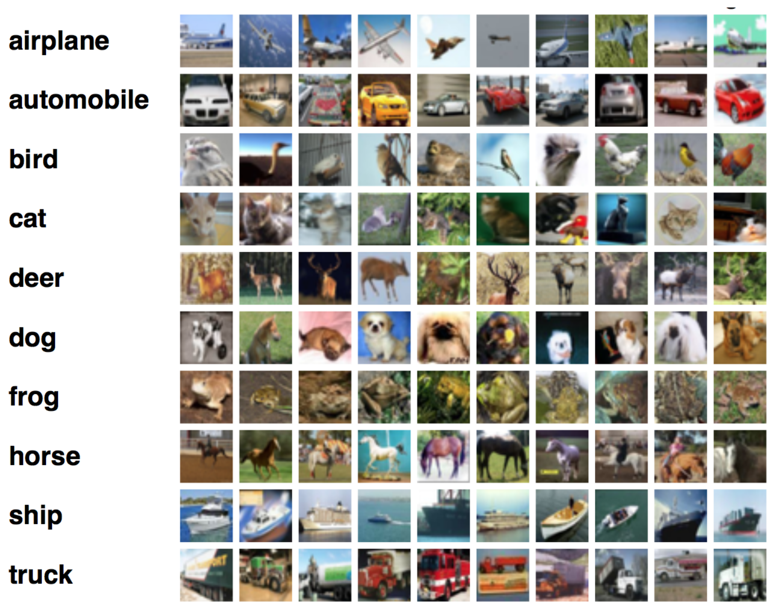

Figure 3. Sample Images[11]

layers approximate $\mathcal{H}(x)$, we could let these hidden layers to approximate $\mathcal{F}(x)$, the residual part. The motivation behind this ResNet framework is as we have discussed, if the added layers could be optimized as identity mapping, then a deeper model could not lead to higher training error. So the degradation problem indicates that the current optimization algorithm could approximate identity mappings with multiple linear and nonlinear layers. But with this novel ResNet framework, if the identity mapping is the optimal, then the optimization solver could just let the weights inside multiple layers to be zero, which means to let $\mathcal{F}(x) = 0$, then the original underlying mapping is $\mathcal{H}(x)$ could be optimized as identity mapping. With this framework, a deeper model could not have higher training error. In general, the identity mapping might not be the optimal mapping, but the ResNets framework could help us precondition the degradation problem.

A typical ResNet Block is show in Figure 2. Formally, each ResNets block could be defined as:

$$y = \mathcal{F}(x) + x \tag{1}$$

where x and y are the input and output of the ResNet block. And the $\mathcal{F}(x)$ residual mapping is learned by left part in Figure 2. If the dimension of x and $\mathcal{F}(x)$ are the same, then the operation $\mathcal{F}(x) + x$ could just be performed using shortcut connection and element-wise addition. If there is a dimension increase from x to $\mathcal{F}(x)$, then a projection shortcut using a 1×1 convolution layer could be applied to match the dimension and then do the element-wise addition in equation 1.

1.3 Wide Residual Network

The ResNets with identity mapping could allow us to train very deep ConvNets, however each single digits of training accuracy improvement could cost almost doubling the depth of the network. As gradient goes through the deep networks, only a few ResNets blocks have learned useful representations and also many blocks might have very little information that could make contributions to the final output. This problem has been formulated as a diminishing feature reuse problem[17] which makes the deep networks take a long time to train.

To increase the power of the ResNets blocks, people came up a possible simple way: adding more feature planes in each convolutional layers to widen the convolutional layers. Instead of the original ResNet which using thin convolutional layers with small number of feature planes and make the network very deep, the Wide Residual Networks(WRNs) was proposed to decrease the depth and increase the width of the residual networks by adding more feature planes in each convolution layer.

Let's first give one parameters that we need to define the wide residual networks: the width parameter k, which multiplies the number of feature planes in each convolutional layers and corresponding to the k in table 1. The general structure of the wide residual networks are shown in table 1. It consists of the initial layer **Conv1** followed by three groups of residual network blocks **Conv2**, **Conv3** and **Conv4**. At the end, we could apply the average pooling layer and the final fully connected classification layer. For each convolutional layer, we use the convolution filter size 3×3 which corresponds to the 3×3 in table 1. The feature plane size of the initial layer **Conv1** is fixed and width parameter k scales the width which is the number of feature planes used in group **Conv2**, **Conv3** and **Conv4**. And the parameter N in table 1 shows the number of ResNets blocks we will use for each group. At the end, we will use the average pooling with filter size 8×8 and the final classification layer. we could see that the original ResNets framework[6][7] is just set the width parameter k here as 1 or 2. And for each single block in **Conv2**, **Conv3** and **Conv4**, it follows the similar structure as in Figure 2

Group Name	Block type
conv1	$\left[3 \times 3 \quad 16\right]$
conv2	$\left[\begin{array}{cc} 3 \times 3 & 16 \times k \\ 3 \times 3 & 16 \times k \end{array}\right] \times N$
conv3	$\left[\begin{array}{cc} 3 \times 3 & 32 \times k \\ 3 \times 3 & 32 \times k \end{array}\right] \times N$
conv4	$\left[\begin{array}{cc} 3 \times 3 & 64 \times k \\ 3 \times 3 & 64 \times k \end{array}\right] \times N$
avg-pool	$\left[8 \times 8\right]$

Table 1. Typical Wide ResNet

In the rest of this paper, I will use the following notation: WRN$-n-k$, which denotes a wide residual network framework with the depth parameter as k and the total convolutional layers used as n.

2. Numerical Results and Discussion

2.1 The Dataset

The dataset we deal with for this final project is The CIFAR-10 dataset[11] which consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. They were collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The classes are completely mutually exclusive. There is no overlap between different classes. The name of the 10 classes and also some sample images for each of the 10 classes are shown in Figure 3.

2.2 Data Preprocessing

For the data preprocessing, we applied the normalization and Zero Components Analysis(ZCA) whitening. The image we deal with is tiny images which only have dimension 32×32. And the tiny images usually have strong two-way correlations between the nearby pixels. So before applied the deep learning algorithms to the image data, it is better to remove the second-order correlation first. This could force the model to concentrate on the higher-order correlations without being distracted with the two-way correlations. This could make the model more likely to discover the regularities inside the image.

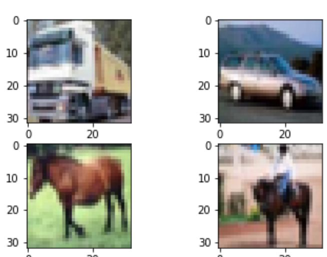

Figure 4. Before Processing

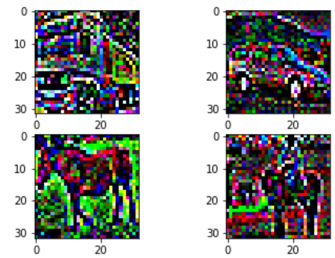

Figure 5. After Processing

The ZCA whitening could use Singular Value Decomposition(SVD) to compute $\hat{\Sigma}^{-\frac{1}{2}}$ (where $\hat{\Sigma}$ is the sample correlation matrix) and multiply the original matrix by this $\hat{\Sigma}^{-\frac{1}{2}}$ to remove the second-order correlation. Four sample images from the original images are shown in Figure 4. And the corresponding images after the preprocessing are shown in Figure 5. Compared the images before and after the preprocessing, we could the see the two-way correlation has been removed and the nearby pixels don't have similar regularities after the preprocessing. And using the preprocessed image, our model could focus more on the shape of the object inside the image to classify the corresponding object. Also we could find that the images before processing are blurry, this is because the image sizes are too small and this could limit the potential performance of our model.

2.3 Experimental Results

For the experimental result, I will compare the performance of Wide Residual Networks on the CIFAR10 dataset with the benchmark results provided by SVM, Random Forest, conventional ConvNets and original ResNets. Compared to the benchmark results, we will find that the wide ResNets could have a really great performance in the image recognition for this CIFAR10 datasets.

Support Vector Machine In machine learning, support vector machines(SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Formally, SVM constructs a set of hyperplanes in a high-dimensional space, which can be used for classification. And a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class[4][8]. Using the sklearn package in python, I train the SVM model on the CIFAR10 training dataset. The result I got was training accuracy 38.15% and test accuracy 34.33%. We could see the SVM performs really bad for the image recognition, even for the training dataset.

Random Forest For the conventional machine learning algorithm, the Random Forests perform best on this dataset. Random forests[1] are an ensemble learning method that could be used for classification, which could operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes for classfication of the individual trees. Using the sklearn package in python, I train the Random Forest model on the CIFAR10 training dataset. And I got training accuracy is 100% but the best test accuracy I got was 46.74%. We could see that the Random Forest could perform better compared with the SVM. Especially, we could see that the Random Forest have a really good performance on the training dataset, but its test accuracy performance(46.74% test accuracy) is still not that good for an image recognition.

Conventional ConvNets For the conventional convolutional networks, I trained the preprocessed image datasets with the famous framework LeNet [13], AlexNet[12] and VGGNet [15]. I found the AlexNet worked best on this image dataset. The framework of the AlexNet is shown in Figure 1. We could see that the AlexNet is composed of 5 convolutional layers (C1 to C5 in the figure), followed by two fully connected layers (FC6 and FC7). And the last layer is the final softmax output layer (FC8). It was initially trained to recognize 1000 different classes and now we use it to classify the CIFAR10 dataset with 10 different classes. Using the Tensorflow framework, I trained the AlexNet on our dataset for 200 epochs, using the batch size 100 and with data augmentation like shift and rotation of the image. The results of the training and testing accuracy after each epoch is shown in Figure 6. After 200 epochs, the training accuracy is 91.48% and the best test accuracy is 88.73%. We could see that after 100 epochs, the test accuracy just fluctuate around 88% even though the train accuracy keeps increasing. Hence, we don't need to keep training for more epochs. And 88.73% test accuracy is a pretty big improvement compared with the traditional machine learning algorithms SVM and Random Forest.

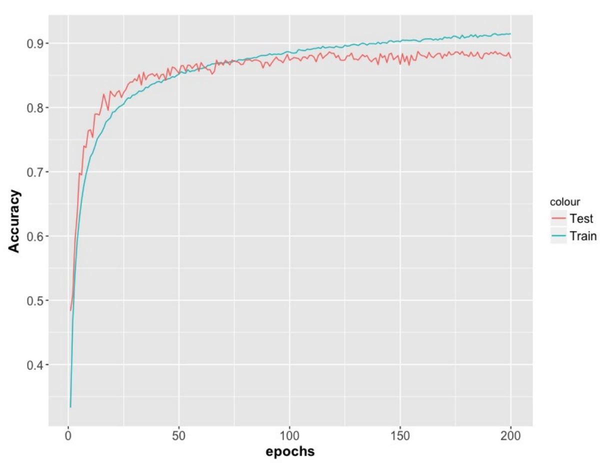

Figure 6. AlexNet

Original ResNet For the original residual network, I trained a deep residual network with total 46 convolutional layers. The ResNet is deep enough and if adding more convolutional layers, there are no improvement on the test accuracy. The ResNet framework we run is shown in Table 2. The conv. filter size are all 3×3. For **Conv2**, **Conv3** and **Conv4**, we run each of them with 6 ResNet blocks with 32, 64 and 128 as the corresponding number of convolution feature planes used in each layer. For the initial **Conv1**, we use 16 feature planes.

There are total 46 convolutional layers used in this ResNet, which includes the 9 layers used for the feature size expanding. Using the Tensorflow framework, I trained the ResNets on our dataset for 200 epochs, using the batch size 100 and with data augmentation like shift and rotation of the image. The results of the training and testing accuracy after each epoch is shown in Figure 7. After 200 epochs, the training accuracy is 99.32% and the best test accuracy is 90.00%. We could see since we used a ResNet framework to run a deep network, the training accuracy could quickly go beyond 99%. And the test accuracy could be improved to 90%, which has more than 1% improvement compared with the conventional ConvNets. This improvement comes from the benefit of the ResNet framework that could let us run the network deeper.

Group Name	Block type
conv1	$\begin{bmatrix} 3\times3 & 16 \end{bmatrix}$
conv2	$\begin{bmatrix} 3\times3 & 32 \\ 3\times3 & 32 \end{bmatrix} \times 6$
conv3	$\begin{bmatrix} 3\times3 & 64 \\ 3\times3 & 64 \end{bmatrix} \times 6$
conv4	$\begin{bmatrix} 3\times3 & 128 \\ 3\times3 & 128 \end{bmatrix} \times 6$
avg-pool	$\begin{bmatrix} 8\times8 \end{bmatrix}$

Table 2. ResNet

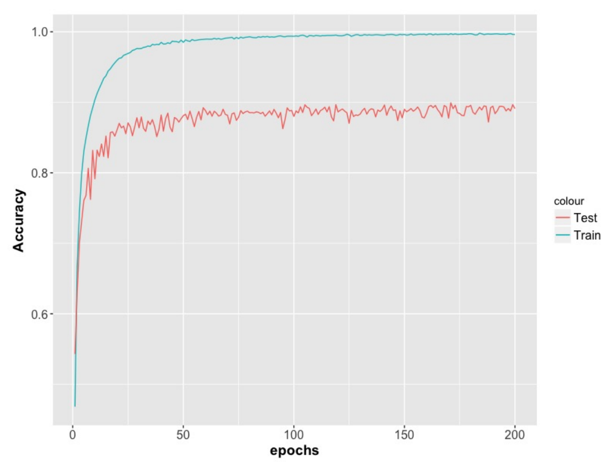

Figure 7. Original ResNet

Wide ResNet For the Wide Residual Network(WRN), due to the constraint of my GPU memory size, I could only run a WRN-16-10, which is set $N = 1$ and $k = 10$ in Table 1. The network has total 16 convolution layers, which is a shallow network.

Using the same batch size and data augmentation as above, I run for 200 epochs. The results of the training and testing accuracy after each epoch is shown in Figure 8. After 200 epochs, the training accuracy is 99.81% and the best test

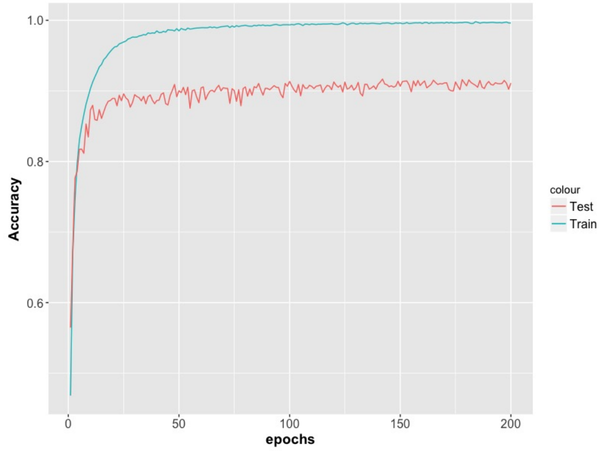

Figure 8. WRN-16-10

accuracy is 91.68%. Compared to the Original ResNet with 46 layers, the test accuracy could be improved from 90.00% to 91.68%. For the original ResNet and the WRN-16-10 we used, they have almost the same parameter size and the same training time, the test accuracy has been improved for more than 1%, this comes from the benefits of the great width of the WRN in each convolution layer to let us gain more information and regularities from the figure.

Comparison Finally, let me compare the test result from all of the methods we used above. For all the deep learning method, I all used the Adam optimizer[10] with step size 0.001. The programs are all running using a NVIDIA GeForce GTX 1060 graphics with 3GB GDDR5 memory size. The test accuracy after each epoch is shown in Figure 9. And for the best test accuracies achieved for the above 5 methods are shown in Figure 10.

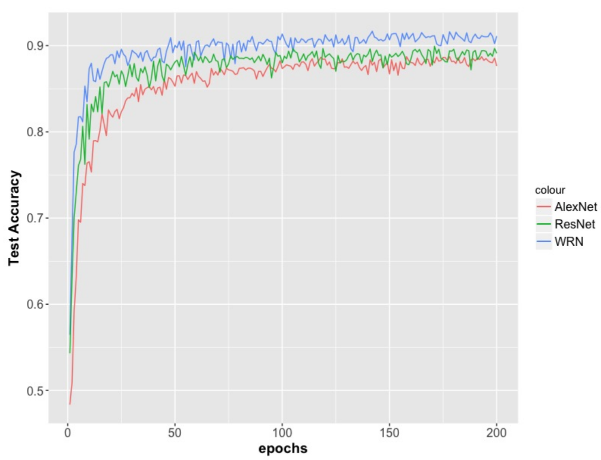

Figure 9. Test Accuracy Comparison

Based on the results, we could find that the deep learning algorithm led to a pretty big breakthrough for the image recognition and improved the test accuracy for the CIFAR10 dataset from around 50% to around 90%. And the ResNets framework could let the network run deeper without the degradation

SVM	Random Forest	AlexNet	ResNet	WRN
34.33	46.74	88.73	90	91.68

Figure 10. Test Accuracy Comparison

problem and could improve the test accuracy compared with the conventional ConvNets. And compared with ResNets, the wide residual networks could improve the performance by add more feature planes in each convolution layer to make the network wider and achieve the test accuracy at around 92%, which is almost the same image recognition accuracy as human. This WRN method could provide a new state-of-the-art prediction accuracy for such kind image recognition dataset.

Prediction Result Using the final model of WRN-16-10, we made the prediction for the 10000 images in the test dataset of the CIFAR10 and give the normalized confusion matrix of the prediction in Figure 11. Based on the confusion matrix, we could find the prediction accuracy for the transport like airplane, automobile, ship and truck are all above 90%, which is pretty good. This result shows that the transport images have pretty much information inside the image that could let deep learning algorithm justify the image well. However, for the animal, especially for the cat and dog, the prediction accuracy is only 87%, which is not that good. We could find 9% of the cats are misclassified as dogs and also 9% of the dogs are misclassified as cats. This is the image dataset itself limit the performance of our model, because if you look at the sample image in Figure 4, you could find the image is too small which makes it look too blurry. And it should be really hard to figure out whether it's a dog or a cat inside the image, even for human. On the other hand, we could find that almost no animal images are misclassified as transport and also almost no transport images are misclassified as animal, which is an evidence that shows our final model is really good.

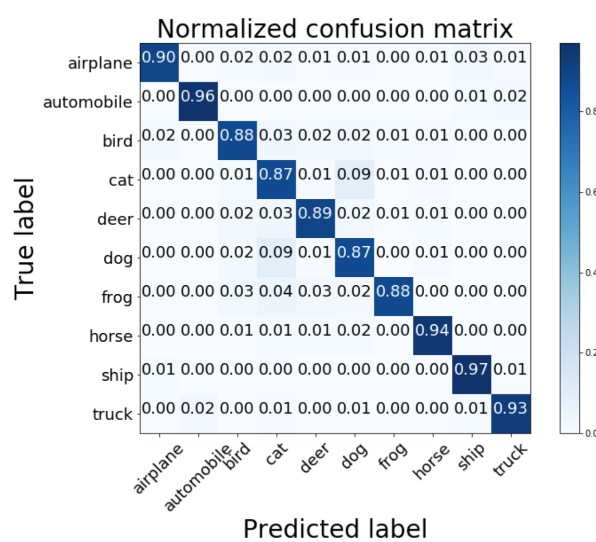

Figure 11. Confusion matrix for the prediction

3. Conclusion

My final results show that using wide residual network has really good performance on the image classification data like CIFAR10. It could outperform the conventional convolutional neural networks as well as the original residual networks with the same parameter size. The best test accuracy we got is 91.68%, which could match the human image recognition ability which has around 92% recognition accuracy. And it is notable that due to GPU memory constraint, we could only run a WRN-16-10 model, I believe that if we could have a better GPU that could run a WRN-28-10 model with deeper networks(more convolutional layers), the recognition accuracy provided by the WRN model could outperform human ability. And this wide residual network architecture could provide state-of-art results on many commonly used benchmark image recognition datasets.

Acknowledgments

I would like to express my special thanks of gratitude to the instructor Prof. Annie Qu for giving me golden opportunity to work on this wonderful project on the deep learning topic. It's such a great learning experience and I really appreciate it.

References

[1] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[2] Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1237. Barcelona, Spain, 2011.

[3] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pages 1223–1231, 2012.

[4] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[5] Kaiming He and Jian Sun. Convolutional neural networks at constrained time cost. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 5353–5360. IEEE, 2015.

[6] 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.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[8] Bernd Heisele, Purdy Ho, and Tomaso Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 688–694. IEEE, 2001.

[9] Fan Hu, Gui-Song Xia, Jingwen Hu, and Liangpei Zhang. Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. *Remote Sensing*, 7(11):14680–14707, 2015.

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[11] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[13] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[14] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[16] 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.

[17] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[18] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. Cvpr, 2015.

[19] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.