Multi-Class Image Classification with Deep Learning

MSDS 458 – Assignment 2

Siyuan Liu

Email: siyuanliu2022@u.northwestern.edu

February 4th, 2022

Abstract

Multi-class image classification has been one of application areas under computer vision domain where deep learning has achieved great progress in last decade together with object detection and image segmentation. As one of deep learning model structure, Convolutional Neural Network (CNN) has been proved to outperform other models especially in image classification because of its advantageous capability to extract features from the input images and evolve based on self-learning. In this research, the CIFAR dataset, a collection of color images evenly distributing across 10 class objects including bird, cat, automobile and so on, has been used to training and testing of multiple DNN and CNN models. The primary objective of this research is to compare the performance of CNN versus DNN on image classification, to experiment with differentiated model structure in terms of number of layers and regularization, and to explore how number of layers and regularization such as dropout and batch normalization impact the model performance.

Key Words:

Multi-class image classification; Neural network; Deep learning; Convolutional Neural Network (CNN)

1. Introduction

Visual object recognition has been one of focused areas where a lot of artificial intelligence studiers attempt to train machine to mimic human being capability of detecting and recognizing images which vary in position, scale, pose and so on. Recently, deep neural networks have achieved impressive progress in this area, while CNN outperforms others due to its advantageous capability of extracting features from images through self-learning (Tien 2018). In the research, several experiments are conducted to experiment with neural network including DNN and CNN on CIFAR-10 dataset as shown in Figure 1, which is one of the well-known dataset of natural images.

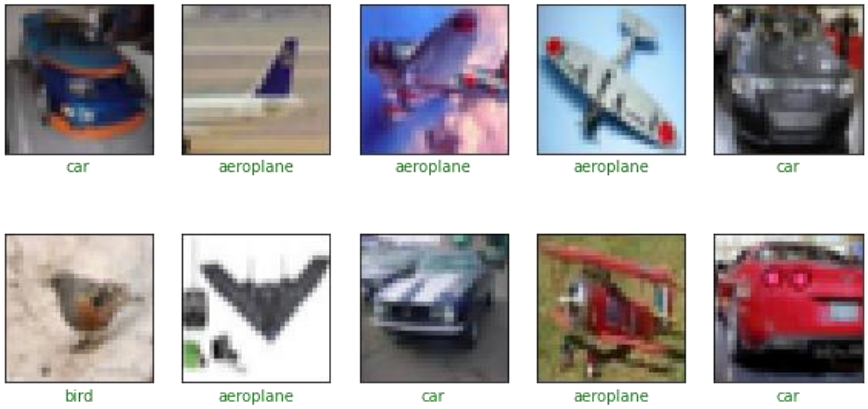


Figure 1. Images from the CIFAR-10 training set.

The main object of conducting these experiments is to make fair comparison between Convolutional Neural Network (CNN) against typical Dense Neural Network (DNN) to explore how specialized feature learning capabilities enable it to excel in image recognition. While at a more granular level, multiple model structures with variety of number of layers including convolutional and pooling layers are experimented to explore the impact on model performance. Additionally, regularization such as dropout and batch normalization, and data augmentation are applied on the base model to testify whether these add-on can help improve model performance.

2. Literature review

Image classification is one of prevalent and fundamental areas where deep learning has proved to achieve impressive progress. The VGGNet has improved the model performance upon AlexNet structure since it won the Large Scale Visual Recognition Challenge in 2012 (Simonyan and Zisserman, 2014). GoogLeNet has improved the performance by enhancing (Szegedy, Liu, and Sermanet, 2015). ResNet developed by He and his team in 2016 has reduced the difficulty of training deep convolutional neural networks by introducing shortcut connections and residual representation during training. In 2017, DenseNet was proposed with each layer obtaining additional inputs from all preceding layers and passes on its own feature maps to all the subsequent layers (Huang et al, 2017)

3. Method

3.1. Data Collection & Exploration

The CIFAR-10 dataset consists of 60,000 32X32 color images, which evenly distributed with 6000 instances across 10 classes. The dataset has been split into training set with 50,000 images and 10,000 test images. The classes are completely mutually exclusive.

3.2 Data Preprocessing

Data normalization is a necessary step to take before the image data is fed into classification training. Given all pixels are in color-scale ranging from 0 to 255, the data is normalized to 0 to 1 simply by dividing all pixel data by 255. The dataset, which is by default in 4 dimension, has been reshaped to flat array for DNN model training.

3.3 Model Construction and Training

Ten neural network models have been constructed with Keras from TensorFlow and experimented for prediction after being trained with the whole dataset, conclusional insights are generated based on model performance comparison across models with different structure such as DNN vs CNN, different number of layers, regularization and so on with more details shown in Figure 2. As top priority, DNN and CNN models are developed and experimented in multiple structures varying by number of layers and regularization. As baseline, Model 1 and 2 are DNN models with 2 and 3 dense layers, to compare against two CNN models

(Model 3 and 4) with same number of layers but using convolutional and pooling layer instead. None of models have factored in any regularization components. Through Model 5 to 8, additional model structure has been added on the baseline model including the Dropout layers with dropout ratio 0.3 after each DNN or CNN pooling layer, and Batch Normalization before final classification dense layer in order to reduce overfitting problems, which is commonly seen in neural network training process. L2 regularization with learning rate 0.001 is also added. With added layers of dropout and batch normalization and classification regularization, we can compare the set of models with model 1/2/3/4 by setting everything else the same and see how these regularization tactics can help further reduce overfitting and improve the prediction accuracy.

Model	Model Structure	Number of Layers	Additional Tunning
Model 1	DNN	2	No regularization
Model 2	DNN	3	No regularization
Model 3	CNN	2	No regularization
Model 4	CNN	3	No regularization
Model 5	DNN	2	Dropout, Batch Normalization, L2 Regularization
Model 6	DNN	3	Dropout, Batch Normalization, L2 Regularization
Model 7	CNN	2	Dropout, Batch Normalization, L2 Regularization
Model 8	CNN	3	Dropout, Batch Normalization, L2 Regularization
Model 9	CNN	2	No regularization, data augmentation layer
Model 10	CNN	3	No regularization, data augmentation layer

Figure 2. Model Design & Structure

Selected hyperparameters are utilized consistently to compile the model structure across all CNN models. For example, in convolutional layer, kernel_size is 3 by 3 with stride 1 to screen the image and extract features. To reduce the computation complexity, max pooling 2 by 2 are built in with stride 2 after each convolutional layer. The number of filters for first convolutional layer is set as 128, 256 for the second convolutional layer, and 512 for the third. Relu is used as activation function for each layer, while Softmax is used in final dense layer as this is a multi-classification problem, so does SparseCategoricalCrossEntropy as loss function. Adam is deployed as optimizer. For comparison purpose, all models are trained on 100 epochs and 512 instances as batch size, while the actual training epochs might vary as early stopping is set with patience 3 to reduce computational time. After convolutional and pooling process has been done with the data, a dense layer with 384 neurons is inserted before the final dense layer with 10 neurons with each associated with one particular image class.

Another technique utilized in this research is data augmentation, which has proved to be capable of increasing the diversity of training set by applying random transformation such as shifting and rotation the image. Model 9 and 10 are derived based on CNN models from Model 3 and 4 with two additional data augmentation layers added before the first convolution layer. Two data augmentation layers are RandomFlip and RandomRotation with rate 0.2, so the input data is transformed and manipulated before it is fed into first convolutional layer for feature extraction.

Another experiment that has been conducted is to run Gridsearch Cross Validation, which specifically focus on optimal hyperparameters of CNN model. To balance out the computation cost, I decided to testify pooling type (max, average), activation function for convolutional layer (sigmoid, tanh, relu), kernel_size ((3,3), (5,5)), and optimizer (‘adam’ , ‘rmsprop’) as shown in Figure 3. To facilitate the search process to land on the conclusion faster, the model with each hyperparameter combination is set to be trained trained on 10 epochs with cross validation in three folds.

```
param_grid = {
    'pool_type': ['max', 'average'],
    'conv_activation': ['sigmoid', 'tanh', 'relu'],
    'kernel_size': [(3,3), (5,5)],
    'epochs': [10],
    'optimizer': ['rmsprop', 'adam']
}
```

Figure 3. GridSearch Cross Validation Parameters

4. Results

Experimental models are evaluated with performance metrics being traced, including accuracy, F1 score, recall, precision and computing time, as summarized in Figure 4. Overall, CNN models outperform DNN models with significant higher accuracy gain across training, validation and testing, although the training of CNN models is more computation expensive. The two CNN models from Model 3 and 4 have gained above 85% training accuracy, and above 70% validation and testing accuracy while the two DNN models from Model 1 and 2 only get to close to 50% as shown in Figure 5.

	Description	accuracy_train	accuracy_valid	accuracy_test	f1_test	recall_test	precision_test	train_time
model1	DNN-2 layer-no regularization	0.513000	0.4586	0.4744	0.461351	0.4744	0.486725	20.908893
model2	DNN-3 layer-no regularization	0.488000	0.4500	0.4622	0.449745	0.4622	0.476422	8.297839
model3	CNN-2 layer-no regularization	0.917267	0.7126	0.7104	0.708750	0.7104	0.718264	61.643661
model4	CNN-3 layer-no regularization	0.853200	0.7306	0.7289	0.726467	0.7289	0.732115	41.099482
model5	DNN-2 layer-with L2 regularization / Dropout /...	0.442578	0.4164	0.4345	0.430742	0.4345	0.444887	11.816478
model6	DNN-3 layer-with L2 regularization / Dropout /...	0.427933	0.3990	0.4163	0.398141	0.4163	0.432377	9.910189
model7	CNN-2 layer-with L2 regularization / Dropout /...	0.965511	0.7546	0.7529	0.751482	0.7529	0.759941	72.869413
model8	CNN-3 layer-with L2 regularization / Dropout /...	0.958822	0.8008	0.7988	0.796384	0.7988	0.800900	86.047304
model9	CNN-2 layer-Data Augmentation (rotate/flip)	0.583667	0.5616	0.5680	0.548915	0.5680	0.569648	42.727110
model10	CNN-3 layer-Data Augmentation (rotate/flip)	0.578022	0.5576	0.5634	0.549912	0.5634	0.589437	41.628106

Figure 4. Summary Table of Performance Score

Model 8 (CNN: 3 conv2d, 3 max pooling, 3 dropout, 1 batch normalization, with L2 regularization) delivers best performance across all models. The model has achieved 96.55% training accuracy, 80.08% on validation and 79.88% on test dataset, achieving 4% additional accuracy gain with additional one convolutional and pooling layer. The discrepancy of accuracy score between training and validation/testing suggest that overfitting problem still exist to some extent.

Interestingly, if we just look at CNN models, the ones with three layers have performed better than the ones with two layers. Model 4 (CNN: 3 conv2d, 3 max pooling) has achieved 72.89% test accuracy while Model 3 (CNN: 2 conv2d, 2 max pooling) has classified 71.04% of images, the same case applies for Model 7 (CNN: 2 conv2d, 2 max pooling, 2 dropout, 1 batch normalization, with L2 regularization) and Model 8. Another observation is that, after adding dropout and batch normalization layers, the performance has been improved with higher accuracy scores, with evidence that Model 8 has achieved higher accuracy score than Model 4 which has similar fundamental CNN model structure but without regularization layers including dropout and batch normalization. However, this trend has been completely opposite for DNN models. With regularization is added, the test accuracy achieved by Model 5 (43.45%) and 6 (41.63%) have been worse than their baseline peers, Model 1 (47.44%) and 2 (46.22%). This is might be attributed to the information loss brought by feature dropout as our model structure is relatively shallow, which should be investigated in deeper level.

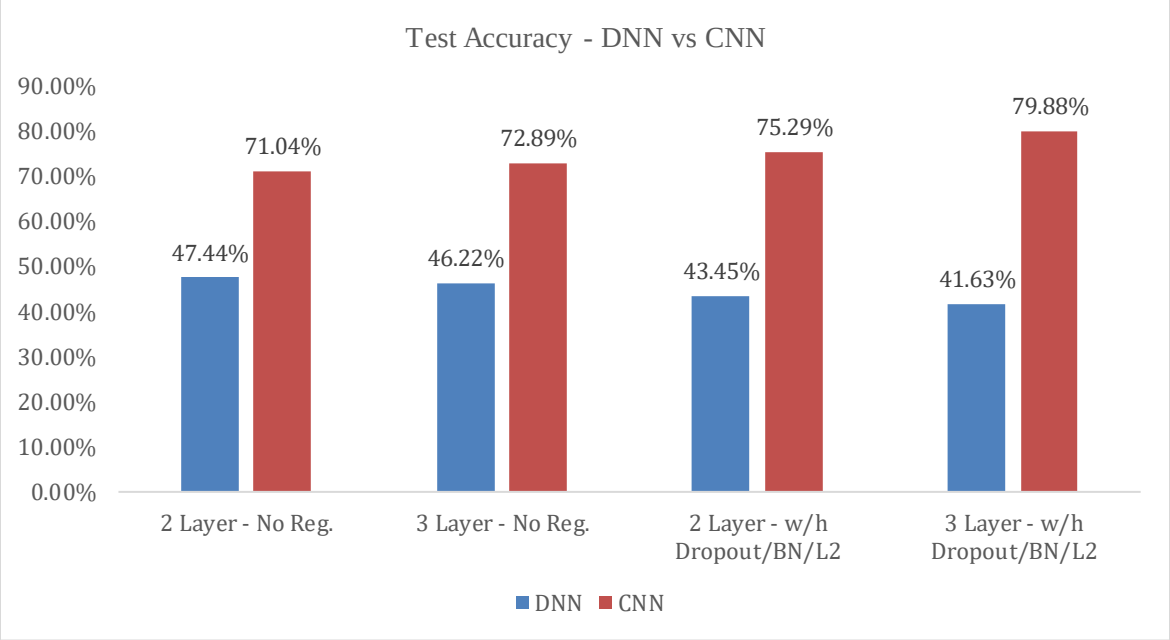


Figure 5. Performance Comparison – CNN vs DNN

Model 9 and 10, which have added additional data augmentation layers into CNN baseline models – Model 3 and 4 did not deliver better performance as expected originally. Model 9 (CNN: 2 conv2d, 2 max

pooling, 1 RandomFlip, 1 RandomRotation) achieves 58.8% test accuracy, while Model 10 (CNN: 3 conv2d, 3 max pooling, 1 RandomFlip, 1 RandomRotation) delivers 56.34% test accuracy only as shown in Figure 6. However, the data augmentation does help reduce overfitting problem. For example, Model 9, which is based on Model 3 as baseline CNN, has only 1.5% accuracy difference between training and test, while Model 3 has approximately 20% (91.73% training, 71.04% test) as shown in Figure 7.

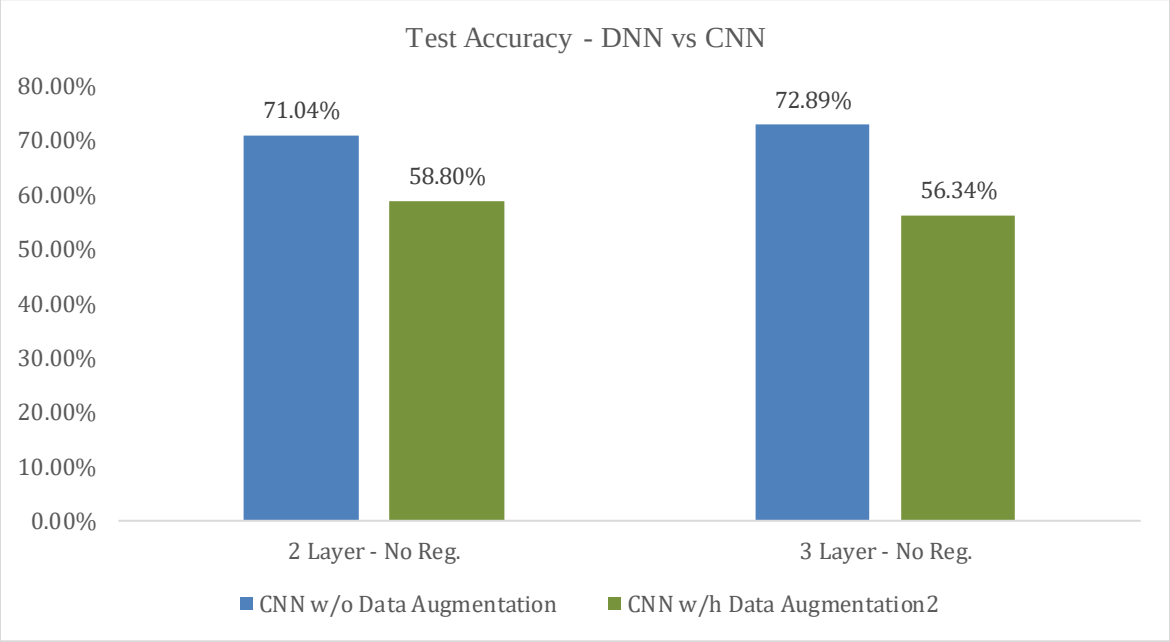
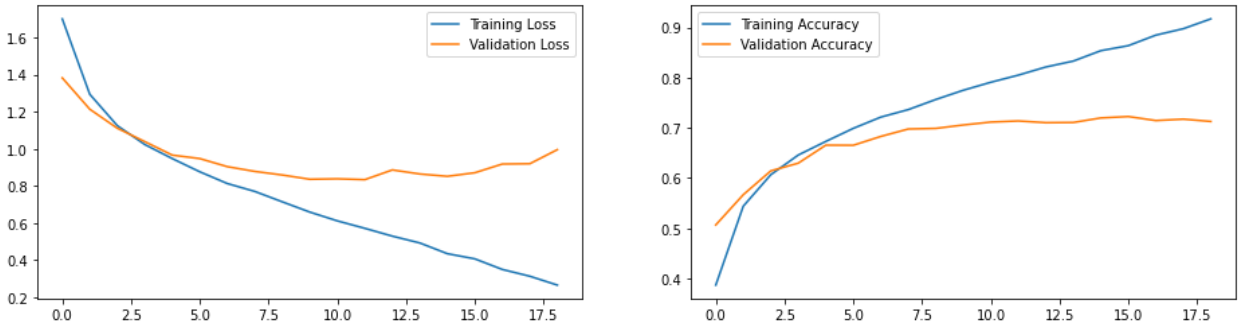


Figure 6. Performance Comparison – CNN vs DNN



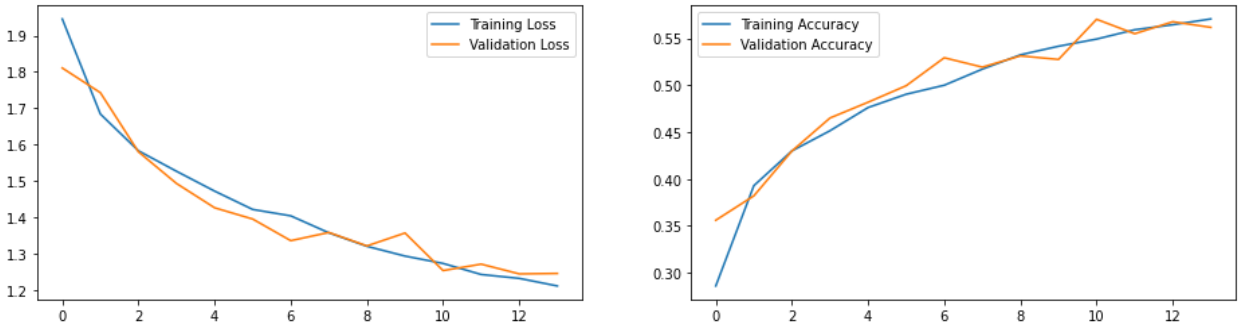


Figure 7. Performance Comparison – CNN with vs without data augmentation (Model 3 vs Model 9)

5. Conclusions

Overall, CNN outperforms DNN specifically in image classification field thanks to its extraordinary capability to extract global and local features based on self-learning. Performance could be optimized and improved potentially by adding more convolutional layers and filters to extract more precise information from images, but at the same time might bring overfitting problem and increase computational complexity. Factoring regularization techniques such as adding dropout layers and batch normalization could potentially help mitigate these problems. Data augmentation could be another option to reduce overfitting, however it has not been proved is help improve model performance in this study.

CNN model could definitely be utilized for face recognition application especially for maximizing its advantage of high accuracy on detecting features and classifying images. However, there might be some challenge with applying it on real time application due to its computational complexity.

Reference:

Ho-Phuoc, Tien. "CIFAR10 to Compare Visual Recognition Performance Between Deep Neural Networks and Humans." (2018): n. pag. Print.

Krizhevsky, A., Sutskever, I. and Hinton, G., 2017. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), pp.84-90.

Han, Y., Zhang, P., Zhuo, T., Huang, W. and Zhang, Y., 2018. Going deeper with two-stream ConvNets for action recognition in video surveillance. *Pattern Recognition Letters*, 107, pp.83-90.

K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, *IEEE conference on computer vision and pattern recognition*, vol. 7, 2016, pp. 770–778.

G. Huang, Z. Liu, L. Van Der Maaten, K. Weinberger. Densely connected convolutional networks, *IEEE Conference on Computer Vision and Pattern Recognition*, 2017 (2017), pp. 2261-2269