CNN vs Bi-directional Network: Can Bi-directional Training be helpful in Image Classification Task?

Renhao Tan

Research School of Computer Science,
Australian National University
u6211458@anu.edu.au

Abstract. This paper generally conducts several performance comparisons of conventional multi-layer perceptron (MLP), bi-directional neural networks (BDNN) and convolutional neural network (CNN) on different classification tasks in a large-scale image dataset. Additionally, a hybrid training method with the combination of bi-directional model and CNN is proposed and investigated to see whether such a method of training can bring improvements on prediction performance. To obtain features from raw images, Residual Network (ResNet) and CNN play the role of feature extractor in the experiment, and then the extracted features are fed to MLP and BDNN for training. In the experiment, CNN is also used to predict the label of images directly and compare with conventional MLP and BDNN. According to the obtained results, CNN can outperform the other models on image classification task, and the combination of CNN and bi-directional training can slightly improve the performance comparing with the combination of convolution layers and MLP.

Keywords: Bi-directional model · Convolutional neural network · Image classification.

1 Introduction

With the progress of the society and technology, digital images are almost everywhere in the world and have become to a major component of our daily life. Quoting an old saying, 'An image is more than a thousand words', which implies us image is a capsule filled with information. Traditional image analysis and classification are mostly done by human, while such a process is time-consuming and requires much effort for human, especially in an era with tons of images to analyze. Thankfully, modern deep learning models and neural networks are capable of analyzing and extracting valid features in digital images in a short period of time. Nowadays, such advanced models are widely applied to classifying images, such as human face recognition, object detection and autonomous vehicles. One of the most popular deep learning model for image classification is convolutional neural network [LeCun & Bengio, 1995]. In this paper, a CNN model is implemented to perform classification and feature extraction in images.

Convolution networks (CNN) are proved to be valid for processing grid-like data such as images, while there is few research of training a bi-directional neural network [Nejad & Gedeon, 1995] for classification on images. In order to examine the discriminative capability of a bi-directional model (BDNN) on massive image data, an experiment of training BDNN to classify items is conducted to explore this matter. Moreover, conventional multi-layer perceptron (MLP) and uni-directional model are implemented in order to compare with the bi-directional model. If the implemented BDNN can outperform the uni-directional model and MLP on the classification task, such a technique could be applied more widely in order to pursue a better prediction performance when training on an image dataset.

To conduct the aforementioned experiment, a large-scale synthetic dataset of vehicle images, VehicleX [Yao *et al.*, 2020], is selected for the models to train on. There are 75,516 samples in VehicleX, where each sample in the dataset is an image of a specific vehicle captured by a camera. Among all these samples, a total number of 1,362 different vehicles and 11 types of vehicles are captured under different circumstances (e.g. light intensity, angle) by the camera. In this paper, CNN, conventional MLP, BDNN and a uni-directional model are constructed to test performance on 2 different classification tasks, which are classifying the 1,362 vehicles and predicting the type of a given vehicle (e.g. sedan, SUV, van, etc.). Since conventional MLP, BDNN and the uni-directional model cannot effectively classify on raw image inputs, Residual network [He *et al.*, 2016] and CNN are used to extract features from raw images and fed the extracted features to the other models for classification. The evaluation metrics of the models would be the prediction accuracy.

In this paper, the model performance are compared in 2 different classification tasks: one is predicting vehicle ID class (1,362 classes in total), and the other one is predicting the vehicle type class (11 in total). For vanilla MLP, BDNN and the uni-directional model, the ResNet extracted features are input data, while CNN would be trained on the raw input images. For the hybrid training of CNN + MLP and CNN + BDNN, CNN is considered as a feature extractor, and the extracted features by CNN would be used to train the MLP and BDNN. Training methods and evaluation results would be explained and reported in detail in the following sections.

2 Method

2.1 Data Preprocessing

VehicleX is a large-scale image dataset of vehicles with 1,362 different vehicles, 11 vehicle types, 45,438 training samples, 14,936 validation samples and 15,142 testing samples. One advantage of synthetic datasets is that the distribution of data is much likely to be more balanced than real world datasets. As for VehicleX, the distribution of 1,362 vehicle classes and the 11 vehicle type classes are relatively balanced. To further reduce the slight distribution imbalance in the given dataset, a random sampling technique is decided to be adopted. Such a decision is also due to hardware and time constraints. For the proposed random sampling method, $\lfloor \frac{\mathbf{N_i}}{10} \rfloor + 1$ samples are selected from each vehicle class (1,362 in total) in the whole dataset, where $\mathbf{N_i}$ denoted as the total number of samples in vehicle class $\mathbf{i}$. The intuition behind this sampling technique is to further balance the distribution of training data and decrease the effort on subsequent training process. Randomly selection among data is to make the sampling data unbiased with the inliers and outliers of each vehicle class can both be selected for training without losing generality. After applying the random selection technique, a training dataset of 5,083 instances is obtained. The statistics of original and random selected datasets on number of instances of each vehicle class on vehicle ID classification task are shown in Table 1, we can see that the variance and standard deviation among number of instances in each is largely decreased, and the newly obtained dataset can be considered as more balanced. Note that the random selection technique is also applied on the original validation and testing data, and the total number of instances in each dataset is shown in Table 2.

Table 1. Dataset Statistics of #Samples in each Vehicle Class

Dataset	**Mean**	**Std.**	**Variance**	**Min.**	**Max.**
Original	33.36	4.70	22.12	20	48
VehicleID Random Selected	3.89	0.54	0.30	3	5

However, there is some imbalance in the distribution of vehicle type classes, while the standard deviation of number of samples in each class is more than 1,000. In order to tackle with this issue, a fixed number of samples are randomly selected from each vehicle type classes. That is, the distribution among vehicle type classes in the new randomly selected dataset follows a uniform distribution, which means the numbers of samples in different classes are equivalent. Therefore, the dataset would not be biased as before. Note that the pixel values in the images are all normalized between 0 and 1. Due to hardware constraint, all vehicle images for CNN is converted to grayscale images and then resize from 256 by 256 to 64 by 64. The numbers of instances in training, validation and testing sets on vehicle type classification task are listed in Table 2.

Table 2. Total #Instances in each Dataset

Dataset	**Train**	**Validation**	**Test**
Original	45,438	14,936	15,142
VehicleID Random Selected	5,304	2,268	2,332
Vehicle Type Random Selected	4,543	1,485	1,507

2.2 Training with Convolutional Neural Network

Convolutional neural network (CNN) takes grid-like data (e.g. images) as input, and then extracts features from such data by convolution layers, eventually train a linear-layer-based MLP using the extracted features to perform classification or prediction. In this paper, a CNN model with 2 convolution layers and 2 linear layers is implemented to perform image classification. The general structure of implemented CNN model is illustrated in Figure 1.

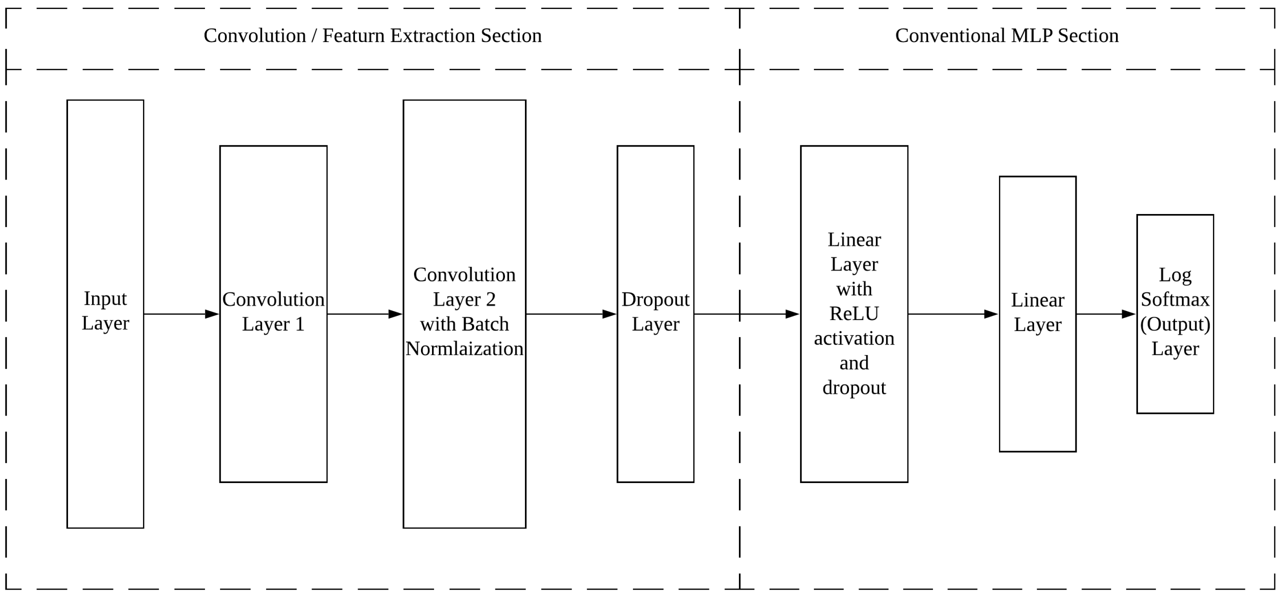
Convolution / Featurn Extraction Section
Conventional MLP Section
Input Layer
Convolution Layer 1
Convolution Layer 2 with Batch Normlaization
Dropout Layer
Linear Layer with ReLU activation and dropout
Linear Layer
Log Softmax (Output) Layer

Fig. 1. Structure of Implemented CNN model.

While training the CNN model, mini-batch training with a batch size of 64 is applied to prevent overfitting. Note that in vehicle type classification task, the first convolution layer extract 8 features / filters from each input image (64 × 64 pixels), and the second convolution layer take the output of previous layer as input and generates 16 features per image. Then these 16 extracted features would transform to an 1-D vector as input for the first linear layer in the conventional MLP section of the model. Since the total number of instances in random selected dataset of vehicle ID classification task is greater than that of the vehicle type classification task, the number of extracted features per image is reduced (2 features in first convolution layer, and 4 in the second one). Such a decision is also due to hardware constraints. Note that max pooling is applied for both convolution layers.

As shown in the structure of the implemented CNN (see Figure 1), dropout technique is applied to prevent overfitting and batch normalization [Loffe & Szegedy, 2015] is adopted to accelerate the training of model to convergence. The dropout technique basically sets a proportion of weights in a layer to 0 during training, and the weights to drop would be selected randomly. The aim of such a technique is to enforce the model not overly depend on specific neurons. Another intention would be adding noise to the model, so the model could be more robust to noise instead of overfitting to training examples. To validate this, we conduct experiments on the validation loss of the model with different dropout rate in the vehicle type classification task. The model is trained for 200 epochs with a learning rate of 1×10^{-5}, and the results are plotted in Figure 2. We can see that 0 or smaller dropout rate may make the model overfit with training data earlier than higher rate, and a dropout rate of 0.5 obtain the lowest loss.

Internal covariate shift is a problem that most deep neural networks suffering from in batch training. Such a problem is raised while training: as the neural network goes deeper and deeper, the linear and non-linear transformation may change the original distribution of the data, and consequently lead to slower convergence. Batch normalization normalizes the input and makes the input follow a distribution of 0 mean and standard deviation of 1. Such an operation can make each sample equally important for training instead of one sample with a relatively large (by value) input while the others with a small-value inputs. The advantage of batch normalization is leading the model to faster convergence and stablizing the distribution of input. Additionally, such a technique can implicitly regularize the model against the noise from each mini-batch. To validate the effectiveness of batch normalization, the performances of CNN model (train for 200 epochs with a learning rate of 1×10^{-5}) with batch normalization and without batch normalization in vehicle type classification task are compared, and the results are shown in Figure 3. According to the plots, we can see that both training and validation loss with batch normalization is much lower than that without batch normalization. Such an experiment is repeated for 5 times, and the obtained average test accuracy with batch normalization is 47.87%, while the one without batch normalization is only 37.09%. The plots and obtained statistics have shown the effectiveness of batch normalization in mini-batch training, and such a technique can obviously accelerate the training of model to convergence.

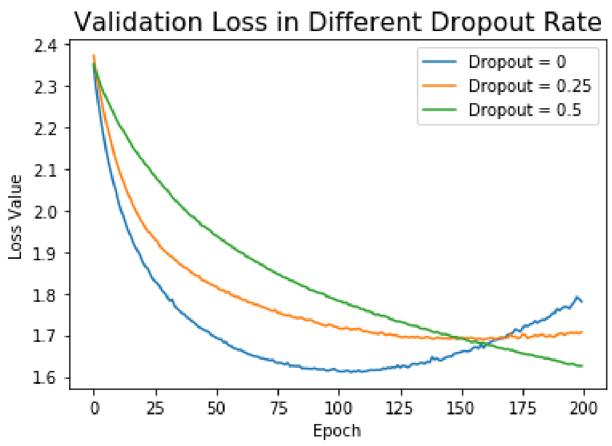

Fig. 2. Validation Loss in Different Dropout Rate in CNN Training.

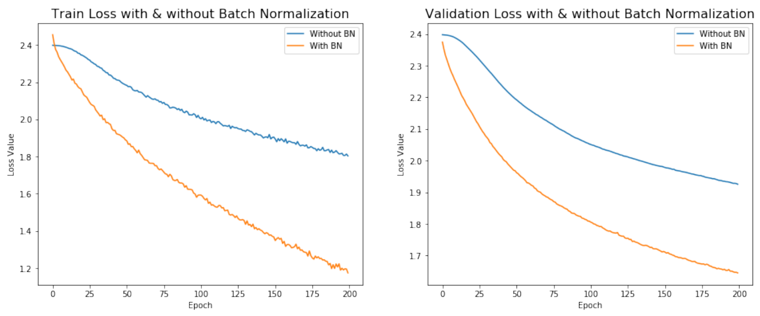

Fig. 3. Train and Validation Loss of with and without Batch Normalization in CNN Training.

2.3 Training with Uni- and Bi-Directional Models

As mentioned in the research of BDNN [Nejad & Gedeon, 1995], BDNN is supposed to be trained in the normal training direction and the reverse direction, then apply conventional error backpropagation algorithm to adjust intrinsic weight matrix of the model based on the loss with respect to forward and backward direction. In this experiment, such methods are adopted and implemented for training, and single hidden layer models of BDNN and uni-directional network are implemented. Note that a bias vector is created to apply between the leftmost layer (input layer in normal training direction) and hidden layer during backward training. Such a bias vector would be learned as a model parameter in training phase.

One challenge of training a BDNN is that we need to make the training functions invertible. For the proposed classification task, the relationship between input and output vectors are many-to-one, since the dataset includes several different instances which belongs to a same vehicle class. In this case, if we apply a BDNN in the reverse direction, the model would be unable to recognize which input vectors to map to with a given vehicle class ID containing multiple instances in the dataset. In order to tackle this issue, an extra attribute is added to the output vector of each instance, denoting as a unique ID for each training instances. Given the unique ID for each instance, the relationship between input and output vectors in the dataset becomes to one-to-one. For better training purpose, the unique IDs of all 5,083 training instances are normalized to values between 0 and 1. The normalization is also performed when adding unique instance IDs on both validation and testing sets.

The selection of loss function is also vital to train the BDNN. In this experiment, 2 different loss functions are applied when training in the forward direction and 1 loss function is applied to train backward. As applied in most classification problem, cross entropy loss is adopted on the forward direction on the original vehicle class and vehicle type class labels (without unique instance ID), while MSE loss is selected on predicting the unique instance ID, since the sample ID is a scalar type data. In the reverse direction, MSE loss is also chosen to measure the distance between the original input vectors and predicted input vectors. When training in the reverse direction, each original vehicle ID class label are transformed to a vector of 1,362 dimensions in vehicle ID classification task (11 dimensions in vehicle type classification task) via one-hot encoding. Feeding the one-dimensional scalar class labels to the model could lead to bad performance, since there is no natural ordering among such class labels. After computing all the different loss values, as all these losses are either using different loss function or applied on different attributes, error-backpropagation are applied to them separately.

In this experiment, the uni-directional neural network is slightly modified in order to accommodate with BDNN for the comparison purpose. To be more specific, similar to the aforementioned training outline of BDNN, the unique instance ID is also added to each output vector in the training set, and the uni-directional model is required to predict the vehicle class and unique sample ID at the same time. Intuitively, cross entropy loss and MSE loss are adopted on vehicle class and unique sample ID, respectively, to adjust intrinsic weight matrices of the model. In the result and discussion section, the performance among uni-directional model, BDNN and conventional MLP would be compared.

2.4 Bi-directional Training for CNN

In this paper, a bi-directional training scheme for CNN is proposed. Such a proposal is generally a combination of convolution layers and BDNN. The original proposal of backward training in BDNN would be considering the output vectors as input and make the model to map the original output vectors with original input vectors. Since convolution operations cannot be invertible, the original proposal has abandoned and been replaced with a new one: pre-train a CNN model, and then only use the convolution layers in pre-trained CNN to extract features from image data, while the extracted features would be the input data for bi-directional model. In this paper, such a hybrid training is named as **CNN+BDNN**. For comparison purpose, a model of **CNN+MLP** is proposed: such a model also used the original convolution layers in pre-trained CNN model as a feature extractor, then the features are trained by a conventional MLP. The MLP component of **CNN+MLP** is identical to the conventional MLP section in Figure 1. The purpose of creating these 2 types of hybrid training is to investigate whether bi-directional training can be helpful in image classification tasks. However, due to hardware constraints, the number of extracted feature in each convolution layer is reduced (first layer extracts 2 features, second one extracts 4 features). The results of these 2 models would be reported in the next section.

3 Result and Discussion

According to the experiment proposed in previous sections, several meaningful results are obtained by training CNN, BDNN, UDNN (uni-directional neural network) and conventional MLP on different pairs of randomly selected dataset. To evaluate the model performance, classification accuracy and cross entropy loss are calculated on selected test sets. 10 different randomly selected datasets are used to evaluate the classification performance of the models mentioned above. For each model, 2 test runs are conducted on each dataset. Note that mini-batch training are performed for all model trainings, and all models have been trained for 200 epochs with a learning rate of 1×10^{-5}. Note that CNN considered the the raw image as input, while the other models used ResNet extracted features from raw images as input.

Table 3. Experimental Results and Comparison Vechicle ID and Vehicle Type Classification Tasks for CNN, conventional MLP, UDNN and BDNN

Task	**Vehicle Type**		**Vehicle ID**	
Model	**Accuracy**	**Loss**	**Accuracy**	**Loss**
MLP	33.00%	1.9076	0.81%	6.7650
UDNN	32.07%	1.9129	0.75%	6.7821
BDNN	32.20%	1.9111	0.81%	6.7786
CNN	**45.53%**	**1.5754**	**1.73%**	6.7384

For the obtained averaged experimental results shown in Table 3, we can see that CNN outperformed other models at testing accuracy in both tasks. This shows the advantage of CNN's ability of extract patterns from image inputs. Aside from that, BDNN performed slightly better than UDNN is accuracy and loss. This may be credited from the bi-directional training method. However, the performance of BDNN and conventional MLP are similar. In a non-naturally invertible prediction problem like this, BDNN may be difficult to show its advantage while comparing with conventional MLP.

Note that the comparison of **CNN+MLP** and **CNN+BDNN** is also conducted, and the results are shown in Table 4. Both **CNN+MLP** and **CNN+BDNN** used extracted features from raw images by convolution layers of a pre-trained CNN model (trained for 200 epochs with a learning rate of 1×10^{-5}) as input data and then trained for 200 epochs with a learning rate of 1×10^{-5}. The experiment also run in 10 different pairs of random selected datasets for both model. Looking at the results of these 2 models, both models obtained similar performances in the 2 classification tasks. That means the advantages of bi-directional training in CNN are not obvious, which is similar to the results in Table 3. The intuition behind this is that bi-directional model may not be able to outperform conventional MLP if the problem is not naturally invertible. Only if the problem is naturally invertible, the advantage of bi-directional training may appear: BDNN obtained slighly better results than UDNN. Conversely, if the problem is not naturally invertible, the conventional MLP is preferred to bi-directional training, since MLP is more easily to implement and BDNN cannot show its advantage in these problems comparing with MLP.

Table 4. Experimental Results and Comparison Vechicle ID and Vehicle Type Classification Tasks for **CNN+MLP** and **CNN+BDNN**

Task	**Vehicle Type**		**Vehicle ID**	
Model	**Accuracy**	**Loss**	**Accuracy**	**Loss**
CNN+MLP	35.61%	1.8857	2.36%	6.5723
CNN+BDNN	35.17%	1.8786	2.48%	6.5900

4 Conclusion and Future Work

In this paper, it is shown that bi-directional models can perform better than uni-directional if the problem is naturally invertible. Conversely, in a image classification problem which is not naturally invertible and contains complex operations like convolution, the advantage of bi-directional training is lost. Additionally, while dealing with grid-like data such as images, convolutinal networks can effectively extract features from the raw grid-like data to perform classification. Using extracted features from image data and then trained by MLP-based model may not acquire performance which is better than convolutional network treating raw images as input.

In the future, bi-directional training can be further validating its usefulness combining with other advanced deep learning models and techniques, such as speech recognition. If the bi-directional training technique can be proved effective in combinations with modern deep learning models, such a training technique can be applied more widely in the industry.

References

1. He, K., Zhang, X., Ren, S. and Sun J. (2016): Deep Residual Learning for Image Recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.
2. Loffe, S. and Szegedy, C. (2015): Batch normalization: accelerating deep network training by reducing internal covariate shift. International Conference on Machine Learning. JMLR.org, 2015, pp. 448–456.
3. Nejad, A. F. and Gedeon, T.D. (1995): BiDirectional Neural Networks and Class Prototypes. Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 3, pp. 1322–1327.
4. Yao, Y., Zheng, L., Yang, X., Naphade, M. and Gedeon, T. (2020): Simulating Content Consistent Vehicle Datasets with Attribute Descent. ECCV.