Report on Cifar-10 Classification with MLP and CNN

千英卓 2017013622

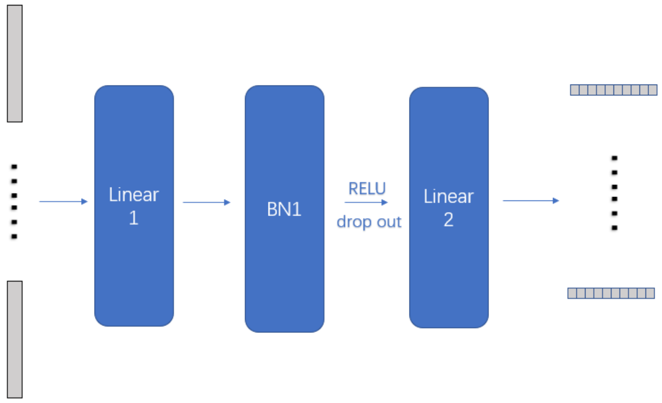
1 Introduction

This task aims to implement a Cifar-10 classification with MLP and CNN. We're expected to finish the implementation of TA-provided models, based on TensorFlow. Techniques such as drop out and batch normalization are also required.

2 Model Architecture

The architecture of my best performing model is depicted below:

MLP:

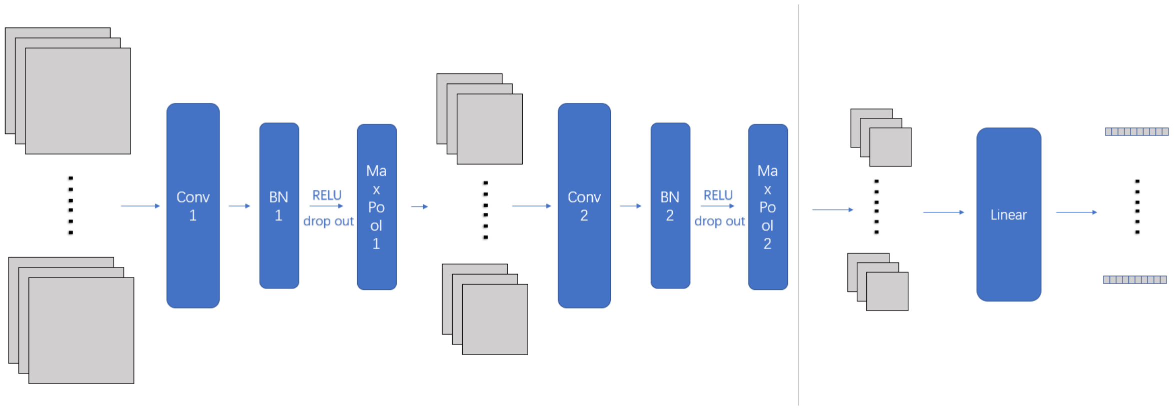


layers	input_shape	output_shape
Linear1	[batch_size, 3072]	[batch_size, 768]
Linear2	[batch_size, 768]	[batch_size, 10]

parameter	value

parameter	value
batch size	500
drop_out rate	0.3

CNN:



parameter	value
Conv1 kernel_size	[3, 3]
Conv1 filter_size	128
Maxpool1 kernel_size	[2, 2]
Maxpool1 stride	[2, 2]
Conv2 kernel_size	[3, 3]
Conv2 filter_size	32
Maxpool2 kernel_size	[2, 2]
Maxpool2 stride	[2, 2]
Linear input shape	[batch_size, 2048]
Linear output shape	[batch_size, 10]
drop_out rate	0.6
batch size	500

3 Result

The performance of my best model is depicted below:
MLP:

Time used	Validation Accuracy	Test Accuracy
279s	0.5604	0.5513

CNN:

Time used	Validation Accuracy	Test Accuracy
18421s	0.7051	0.6984

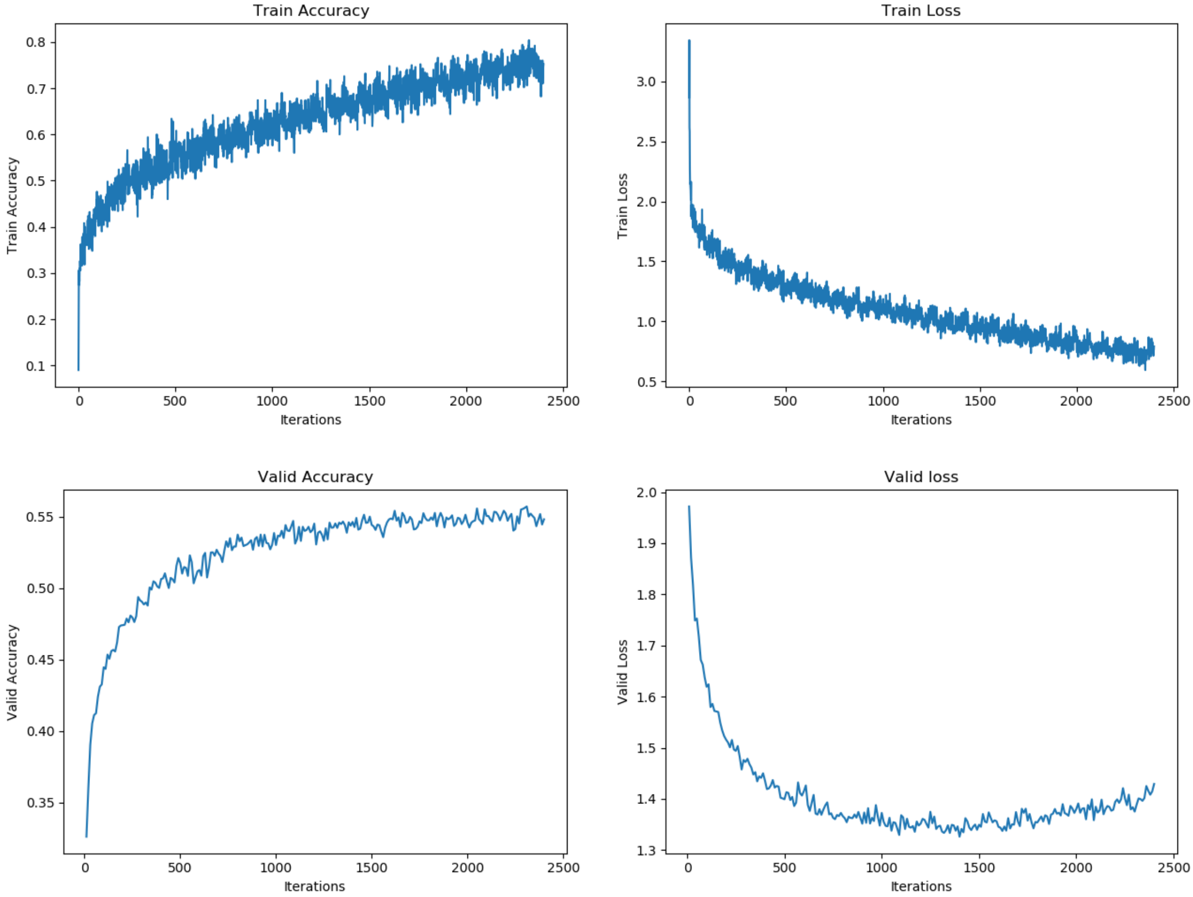
4 Questions

1. The arguments in *model.forward()* is filled as below

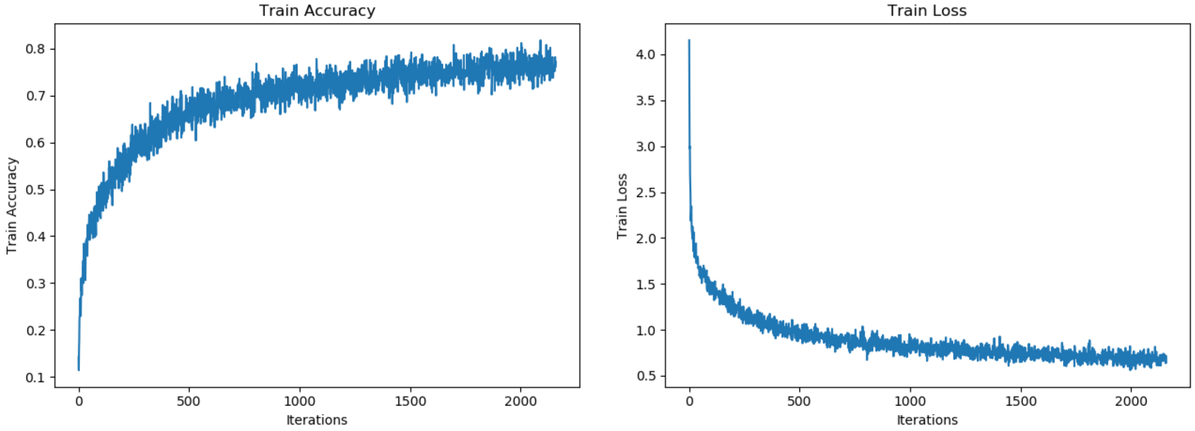
   ```
   `self.loss, self.pred, self.acc = self.forward(is_train=True, reuse=False)`
   `self.loss_val, self.pred_val, self.acc_val = self.forward(is_train=False, reuse=True)`
   ```

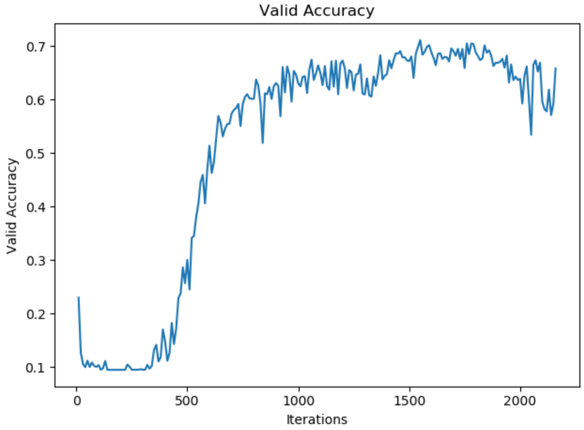
   For batch normalization layer, *is_train* determines whether the mean and variance used in normalization is from current batch or an internal storage of the mean value during training. Therefore, when training, *is_train* should be set to True, meaning that it uses the mean and variance from currently trained batch; otherwise it should be False, meaning that it uses the stored mean value and that the result of this batch will not be counted in the internal storage.
   For drop out layer, the drop out techniques should be applied in training, but not in testing. Thus *is_train* should be True in training and False otherwise.
   reuse determines whether to use the existing weights or to create new ones. So for both layers it should be False in training and True in testing.

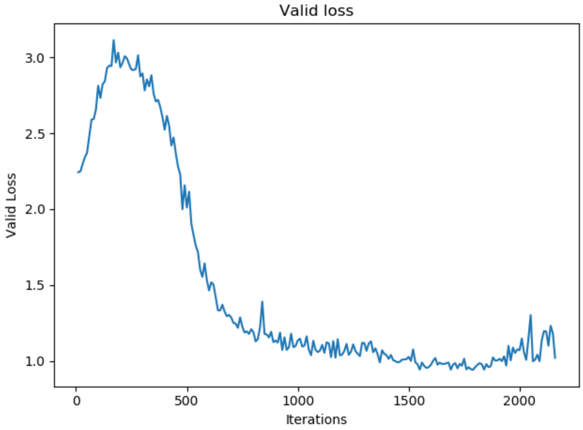
2. The loss and accuracy during training of my best model is ploted below.
   MLP:


Train Accuracy
Train Accuracy
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
500
1000
1500
2000
2500
Iterations
Train Loss
Train Loss
0.5
1.0
1.5
2.0
2.5
3.0
0
500
1000
1500
2000
2500
Iterations
Valid Accuracy
Valid Accuracy
0.35
0.40
0.45
0.50
0.55
0
500
1000
1500
2000
2500
Iterations
Valid loss
Valid Loss
1.3
1.4
1.5
1.6
1.7
1.8
1.9
2.0
0
500
1000
1500
2000
2500
Iterations

CNN:


Train Accuracy
Train Accuracy
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0
500
1000
1500
2000
Iterations
Train Loss
Train Loss
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
500
1000
1500
2000
Iterations





3. The result is shown above. 4. MLP without batch normalization:

Time used	Validation Accuracy	Test Accuracy
258s	0.5436	0.5373

CNN without batch normalization:

Time used	Validation Accuracy	Test Accuracy
5982s	0.7276	0.7367

The use of batch normalizaiton is intended to address the problem of ICS(internal covariate shift). It normalizes the output of every layer, preventing the gradient from vanishing or exploding. As can be seen, MLP implemented with batch normalization technique reached a higher accuracy using slightly shorter time. However, the technique seems to have a negative effect in CNN model, which is confusing.

5.Results of different drop rate is shown below:

MLP:

drop out rate	Validation Accuracy	Test Accuracy
0.3	0.5614	0.5598
0.4	0.5614	0.5434
0.5	0.5614	0.5486

drop out rate	Validation Accuracy	Test Accuracy
0.6	0.5614	0.5458
0.7	0.5614	0.5513

6.Because the distribution of samples in train set and validation set is different. If the accuracy of train set is considerably higher than that of validation set, it indicates that the distribution is different and the drop out rate should set higher to acquire better generalization ability.