Changyeol Lee

Final Project Report

1) **Fit training set well:**
   a. **What metric(s) are you using to evaluate your training performance (training accuracy, recall, precision, false positives, etc.)? Explain why these metrics are useful in your data context.**

   I use accuracy because my dataset has an equal number of samples belonging to each class. I also use logarithmic loss because it is useful when a model needs to perform multi-class classification.

   b. **Comparing to human level performance (or any other benchmark in your data context), how well is your model fitting the training data?**

   The 0^{th} model (Reference model) shows 0.7045 of accuracy and 0.8522 of loss.

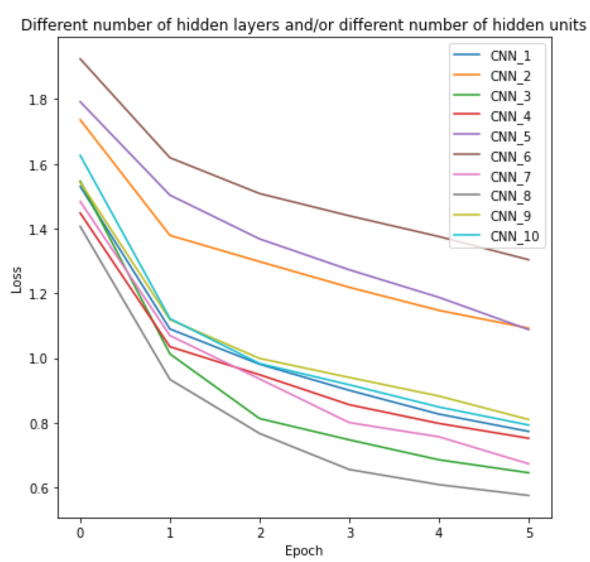
0 (Reference model)	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**
		8	3,3	1,1	**0.7045**
	Pooling layers 1	pool_size	strides		
		3,3	1,1		**Loss**
		layer			**0.8522**
	Fully connected layers 1	64			

   Compared to human level performance, my reference model has a lower degree of accuracy.

   c. **Can you improve your training performance using a bigger neural network? Report a table in which you show the training performance for a total of ten different models (with different number of hidden layers and/or different number of hidden units).**

   Yes, the 8^{th} model (Reference model) shows 0.7922 of accuracy and 0.575 of loss. So, the 8^{th} model shows the higher accuracy and lower loss.

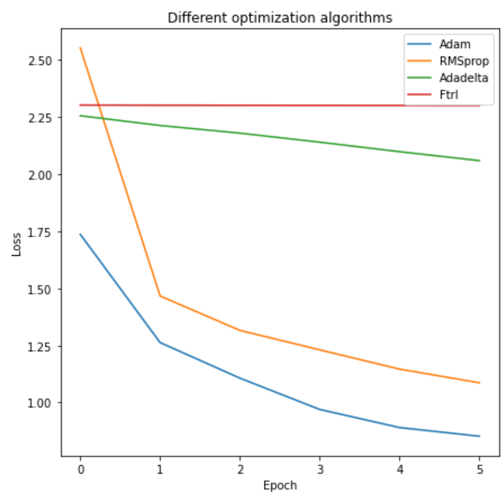
1	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**	2	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**
		8	3,3	1,1	**0.7227**			4	5,5	1,1	**0.6015**
	Pooling layers 1	pool_size	strides				**Pooling layers 1**	pool_size	strides		
		3,3	1,1		**Loss**			3,3	1,1		**Loss**
		layer			**0.7729**			layer			**1.0918**
	Fully connected layers 1	128					**Fully connected layers 1**	256			
3	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**	4	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**
		8	3,3	1,1	**0.7687**			8	3,3	1,1	**0.7298**
	Pooling layers 1	pool_size	strides				**Pooling layers 1**	pool_size	strides		
		3,3	1,1		**Loss**			3,3	1,1		**Loss**
		layer						layer			
	Fully connected layers 1	128			**0.6451**		**Fully connected layers 1**	256			**0.7517**
	Fully connected layers 2	64					**Fully connected layers 2**	64			
5	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**	6	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**
		4	5,5	1,1	**0.5972**			4	5,5	1,1	**0.5065**
	Pooling layers 1	pool_size	strides				**Pooling layers 1**	pool_size	strides		
		2,2	2,2		**Loss**			2,2	2,2		**Loss**
		layer						layer			
	Fully connected layers 1	128			**1.0875**		**Fully connected layers 1**	256			**1.3034**
	Fully connected layers 2	64					**Fully connected layers 2**	64			
7	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**	8	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**
		8	3,3	1,1				8	3,3	1,1	
	Pooling layers 1	pool_size	strides		**0.7556**		**Pooling layers 1**	pool_size	strides		**0.7922**
		3,3	1,1					3,3	1,1		
	Convolution layers 2	filter	kernel_size	strides			**Convolution layers 2**	filter	kernel_size	strides	
		8	3,3	1,1	**Loss**			8	3,3	1,1	**Loss**
	Pooling layers 2	pool_size	strides				**Pooling layers 2**	pool_size	strides		
		3,3	1,1					3,3	1,1		
		layer			**0.6728**			layer			**0.575**
	Fully connected layers 1	128					**Fully connected layers 1**	256			
	Fully connected layers 2	64					**Fully connected layers 2**	64			
9	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**	10	**Convolution layers 1**	filter	kernel_size	strides	**Accuracy**
		4	5,5	1,1				4	5,5	1,1	
	Pooling layers 1	pool_size	strides		**0.7075**		**Pooling layers 1**	pool_size	strides		**0.7108**
		2,2	2,2					2,2	2,2		
	Convolution layers 2	filter	kernel_size	strides			**Convolution layers 2**	filter	kernel_size	strides	
		4	5,5	1,1	**Loss**			4	5,5	1,1	**Loss**
	Pooling layers 2	pool_size	strides				**Pooling layers 2**	pool_size	strides		
		2,2	2,2					2,2	2,2		
		layer			**0.8097**			layer			**0.7925**
	Fully connected layers 1	128					**Fully connected layers 1**	256			
	Fully connected layers 2	64					**Fully connected layers 2**	64			



d. Can you improve your training performance using a better optimization algorithm? Report a table in which you show the training performance for at least three different optimization algorithms.

No, Adam optimization shows the best performance.

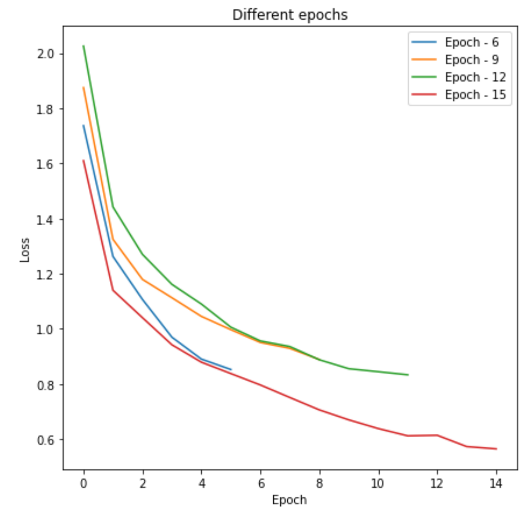
	Accuracy	Loss
Adam	0.7045	0.8522
RMSprop	0.6057	1.0863
Adadelta	0.274	2.0592
Ftrl	0.1124	2.3007



e. Can you improve your training performance using a higher number of iterations (epochs)? Report a table in which you show the training performance for at least three different number of epochs.

Yes, the model with 15 epochs shows the best performance.

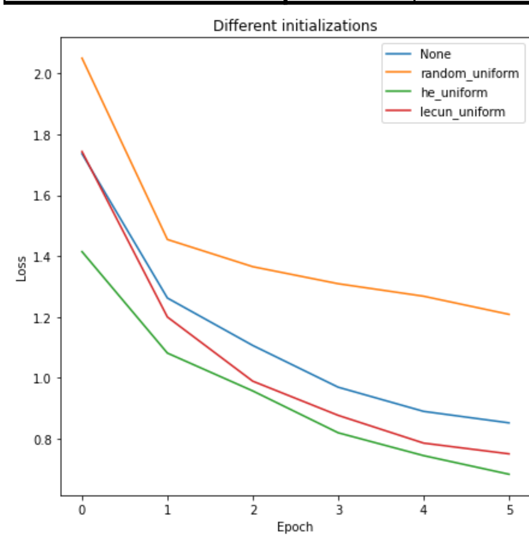
epochs	Accuracy	Loss
9	0.6781	0.8884
12	0.703	0.8331
15	0.7999	0.5644



f. **Can you improve your training performance using a better weight initialization? Report a table in which you show the training performance for at least three different weight initializations.**

Yes, the model using he_uniform shows the best performance.

Weight initializations	Accuracy	Loss
random_uniform	0.5752	1.2087
he_uniform	0.7588	0.6835
lecun_uniform	0.7375	0.7505



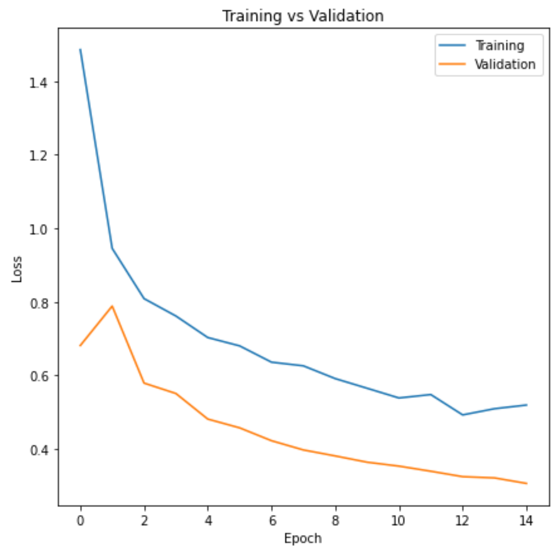
g. **Explain which model is the best model fitting your training set? Use this model for the next part.**

The 8th model using adam optimizer, 15 epochs and he_uniform weight initialization is the best model fitting my training set. The model shows the highest accuracy and the lowest loss.

2) **Fit validation set well:**

a. **Compared to training performance, how well is your model fitting the validation data? Explain if you have an overfitting problem.**

I have the underfitting issue, because the training set shows high bias, while the validation set shows low variance.

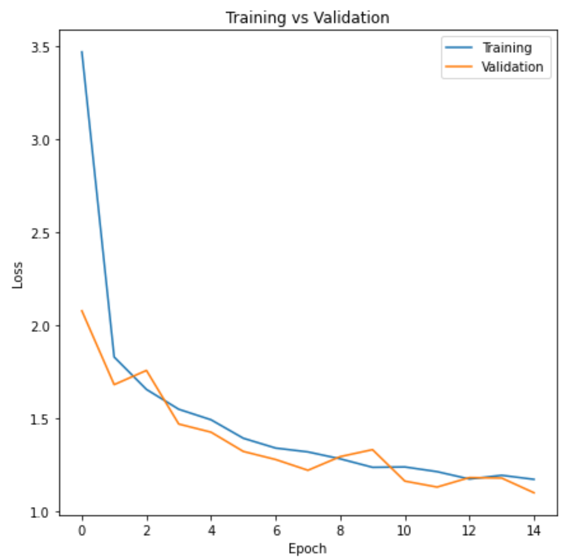


b. **Can you improve your validation performance using L2 regularization? Report a table in which you show the validation performance for five different penalty (λ) rates.**

Yes, I can improve the validation performance, using 0.005, 0.01, 0.05 and 0.1 of penalty rates. The model with 0.01 of penalty rates shows the best performance.

penalty (λ) rates	training		validation	
	Accuracy	Loss	Accuracy	Loss
0.001	0.7698	0.7581	0.6678	0.9607
0.005	0.7097	1.0192	0.7263	0.994
0.01	0.666	1.1716	0.6907	1.0997
0.05	0.557	1.3977	0.5637	1.3386
0.1	0.5537	1.4479	0.5693	1.4719

The model with 0.01 of penalty rates

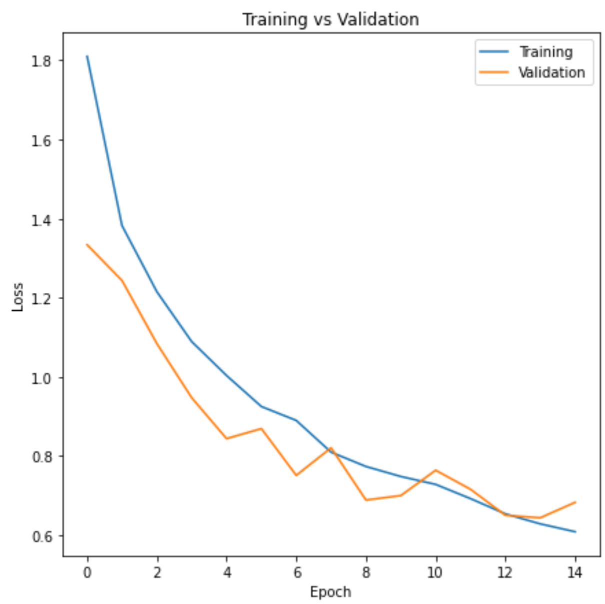


c. **Can you improve your validation performance using dropout regularization? Report a table in which you show the validation performance for five different dropout rates.**

Yes, I can improve the validation performance, using 0.3 and 0.4 of dropout rates. The model with 0.3 of penalty rates shows the best performance.

Dropout rates	training		validation	
	Accuracy	Loss	Accuracy	Loss
0.1	0.8435	0.4376	0.7937	0.6025
0.2	0.8106	0.5351	0.8106	0.6228
0.3	0.7825	0.609	0.7596	0.6832
0.4	0.8007	0.5784	0.7711	0.6516
0.5	0.791	0.6134	0.7196	0.8058

The model with 0.3 of dropout rates



d. Can you improve your validation performance using a mixture of dropout regularization and L2 regularization? Report a table in which you show the validation performance for five different combinations.

No, although validation and training lines in some models move together, losses are high, and accuracies are low.

penalty (λ) rates	Dropout rates	training		validation	
		Accuracy	Loss	Accuracy	Loss
0.001	**0.3**	0.6999	0.9897	0.7448	0.8407
0.001	**0.4**	0.7091	0.9953	0.7511	0.8572
0.01	**0.3**	0.5878	1.2894	0.5748	1.2944
0.01	**0.4**	0.5977	1.2934	0.66	1.1576
0.005	**0.3**	0.6525	1.1294	0.6781	1.066

e. Can you improve your validation performance using batch-normalization? Report a table in which you show the validation performance for five different batch-normalizations (one model with default TF batch-normalization parameters and four models with customized parameters).

No, batch-normalization results in overfitting in all models.

batch-normalization	training		validation	
	Accuracy	Loss	Accuracy	Loss
Default	0.9557	0.146	0.7196	0.9984
epsilon = 0.005	0.955	0.1487	0.6767	1.3878
momentum = 0.95	0.952	0.1482	0.4559	3.48
e=0.005, m=0.95	0.9178	0.252	0.6552	1.43
e=0.01	0.9625	0.1243	0.6663	1.34

f. Can you improve your validation performance using a mixture of batch-normalization and dropout regularization? Report a table in which you show the validation performance for five different combinations.

No, the mixture of batch-normalization and dropout regularization causes overfitting in all models.

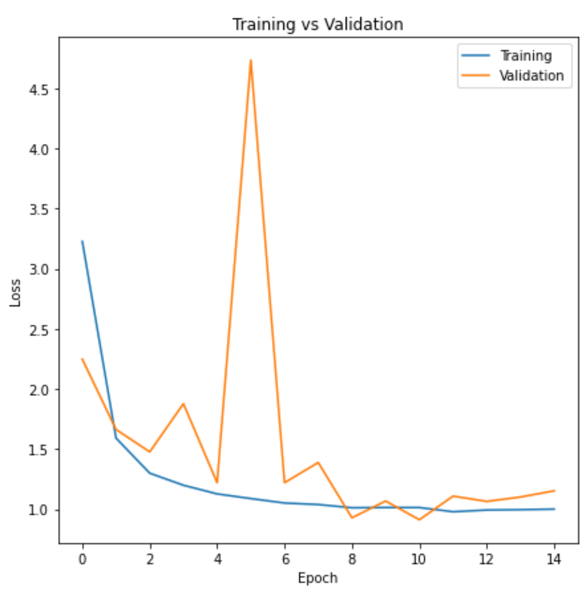
batch-normalization	Dropout rates	training		validation	
		Accuracy	Loss	Accuracy	Loss
Default	**0.3**	0.8015	0.5607	0.5096	1.8656
e=0.01	**0.3**	0.8144	0.528	0.6085	1.1294
e=0.005	**0.3**	0.8364	0.4675	0.7641	0.6783
Default	**0.4**	0.8148	0.5331	0.5411	1.6441
e=0.01	**0.4**	0.7962	0.5901	0.6048	1.4172

g. Can you improve your validation performance using a mixture of batch-normalization and dropout regularization and L2 regularization? Report a table in which you show the validation performance for five different combinations.

Yes, the model with 0.01 of epsilon in batch-normalization, 0.3 of dropout rate, and 0.01 of penalty rate in L2 regularization shows the good performance for the validation.

batch-normalization	Dropout rates	penalty (λ) rates	training		validation	
			Accuracy	Loss	Accuracy	Loss
e=0.01	**0.3**	**0.01**	0.7248	1.0005	0.6767	1.1529
e=0.005	**0.3**	**0.01**	0.7267	1.0074	0.5789	1.6057
e=0.01	**0.3**	**0.005**	0.7384	0.9259	0.6137	1.2392
e=0.01	**0.4**	**0.01**	0.7213	1.0966	0.667	1.26
e=0.01	**0.4**	**0.005**	0.7197	0.9811	0.6133	1.2477

e=0.01, dropout = 0.3, penalty rate = 0.01



h. **Explain which model is the best model fitting your validations set? Use this model for the next part.**
The model with 0.3 of dropout rate is the best model fitting my validations set, because the model has the highest accuracy score and the lowest loss score.

3) **Best model:**
   a. **Explain which model would be your final (best) model and why. Compare the training performance, validation performance and test performance.**

   The model using 0.3 of dropout rate is the final model, because the model shows the highest accuracy and the lowest loss among all the models. Also, the model fit my validation set. The model works well with test set, as well.

Performance	Training	Validation	Test
Accuracy	0.8882	0.7918	0.7807
Loss	0.3254	0.6337	0.6596