Course: DD2424 - Assignment 1
Exercise 1: Training a multi-linear classifier

Álvaro Orgaz Expósito

April 14, 2019

For this assignment I will just use data in the CIFAR 10 file *data batch 1* for training, the file *data batch 2* for validation and the file *test batch* for testing. Each batch contains 10000 images (columns) of 3072 pixels (rows) and 10 possible labels (targets). In figure 1 we can see the first 5 images of each label for *batch test*. And in figure 2 we can find the distribution of labels by images batch of CIFAR 10. Importantly the class distribution in training validation and test is approximately uniform.

I will convert the images data to double format and divide it by 255 to convert the data to be between 0 and 1 since normalization helps when training the networks.

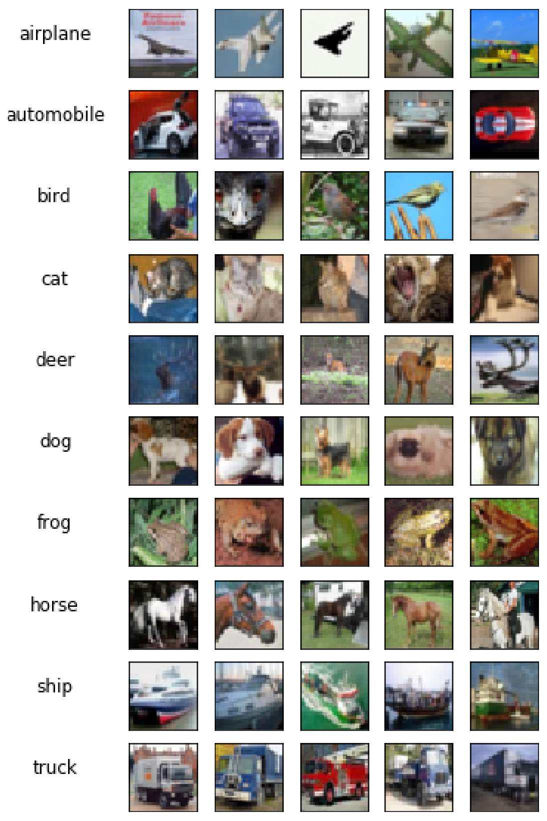


Figure 1: First 5 images of each label in the batch test.

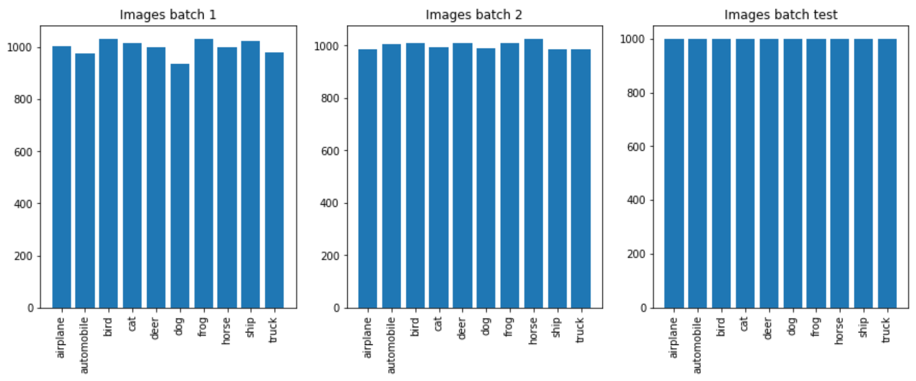


Figure 2: Distribution of labels by images batch of CIFAR 10.

In this assignment, I will use a neural network with one input layer (as many nodes as images pixels) and one output layer (as many nodes as possible image labels) with the loss function cross entropy (classification metric) and an L2 regularization term on the weight matrix. It is a simple network where no hidden layer will be used and the predicted class corresponds to the label with the highest predicted probability. Then, the shape of the weights matrix is as many rows as output nodes and as many columns as input nodes. And for the bias only 1 column with as many rows as output nodes.

Then, for computing the gradients of the cross entropy loss function and an L2 regularization term in mini-batch mode of gradient descent (back propagation) I have used the following equations by data batches, where D is the number of input features, N is the number of training samples and C is the number of possible output labels.

- Initial input and output data:

$$\underset{[D\times N]}{X} \qquad \underset{[C\times N]}{Y} \tag{1}$$

- Forward pass (predictions):

$$\underset{[C\times N]}{S} = \underset{[C\times D]}{W} \times \underset{[D\times N]}{X} + \underset{[C\times 1]}{b} \tag{2}$$

$$\underset{[C\times N]}{P} = SoftMax\left(\underset{[C\times N]}{S}\right) \quad where \quad SoftMax(S_{ji}) = \frac{exp(S_{ji})}{\sum_{c=1}^{C} exp(S_{ci})} \tag{3}$$

- Loss function (cross entropy) to minimise respect to weights:

$$J = L + \lambda||W||^2 = \frac{1}{N}\sum_{i=1}^{N} -log\left(\underset{[1\times C]}{Y_{:i}^T} \times \underset{[C\times 1]}{P_{:i}}\right) + \lambda\sum_{c,d}^{C,D} W_{cd}^2 \tag{4}$$

- Backward pass (vanilla updates equal to $-\eta$ times the following gradients):

$$\frac{\partial\lambda||W||^2}{\partial W} = 2\lambda \underset{[C\times D]}{W} \quad and \quad \frac{\partial L}{\partial W} = \frac{1}{N}\left(\underset{[C\times N]}{P} - \underset{[C\times N]}{Y}\right) \times \underset{[N\times D]}{X^T} \tag{5}$$

$$then \quad \frac{\partial J}{\partial W} = \frac{\partial L}{\partial W} + \frac{\partial\lambda||W||^2}{\partial W} \quad and \quad \frac{\partial J}{\partial b} = \frac{\partial L}{\partial b} = \frac{1}{N}\left(\underset{[C\times N]}{P} - \underset{[C\times N]}{Y}\right) \times \underset{[N\times 1]}{1} \tag{6}$$

I want to check that the gradients computed in the gradients function correspond to the correct gradients. To do this, I compare the gradient obtained by the network (analytical method) with the gradient computed with the finite difference method (numerical method), after initializing the weight and bias with same Gaussian random (zero mean and standard deviation 0.01) for both cases. As error measures I will apply the absolute error and relative error in two different scenarios:

- Firstly, I study the gradients of the 20 first dimensions of the first image of *data batch 1* without regularization. The result is:
  - For weights W: 100.0% of absolute errors below 1e-6 and the maximum absolute error is 1.5539e-08
  - For bias b: 100.0% of absolute errors below 1e-6 and the maximum absolute error is 4.6837e-08
  - For weights W: 100.0% of relative errors below 1e-6 and the maximum relative error is 1.3081e-07
  - For bias b: 100.0% of relative errors below 1e-6 and the maximum relative error is 2.2443-07
- Secondly, I study the gradients of all dimensions of the first 5 images of *data batch 1* with regularization. The result is:
  - For weights W: 100.0% of absolute errors below 1e-6 and the maximum absolute error is 4.7962e-08
  - For bias b: 100.0% of absolute errors below 1e-6 and the maximum absolute error is 5.8252e-08
  - For weights W: 92.8255% of relative errors below 1e-6 and the maximum relative error is 0.0096
  - For bias b: 100.0% of relative errors below 1e-6 and the maximum relative error is 4.4302e-07

To evaluate the learning process I use the metrics loss and accuracy (fraction of well-predicted images or images with the highest predicted class equal to the true) after each epoch of the vanilla mini-batch gradient descent algorithm. In figure 3 we can find the learning curves (both metrics for training and validation sets) for the 4 parametrization cases proposed.

$$W^{t+1} = W^t - \eta \frac{\partial J\left(Batch^{t+1}\right)}{\partial W} \quad where \quad \frac{\partial J\left(Batch^{t+1}\right)}{\partial W} = \frac{1}{N_{Batch^{t+1}}} \sum_{i=1}^{N_{Batch^{t+1}}} \frac{\partial J_i}{\partial W} \tag{7}$$

$$b^{t+1} = b^t - \eta \frac{\partial J\left(Batch^{t+1}\right)}{\partial b} \quad where \quad \frac{\partial J\left(Batch^{t+1}\right)}{\partial b} = \frac{1}{N_{Batch^{t+1}}} \sum_{i=1}^{N_{Batch^{t+1}}} \frac{\partial J_i}{\partial b} \tag{8}$$

On the one side, a large learning rate (η=0.1) leads to instability in the training process and the loss does not seem to converge. On the other side, regularization increases training error and simultaneously reduces the gap between training and validation error curves. In this case, the net effect is always a higher validation error if I increase λ which is obviously not what I want but definitely what we can expect given that ridge regression usually only makes sense if we train models with high capacity on relatively small datasets.

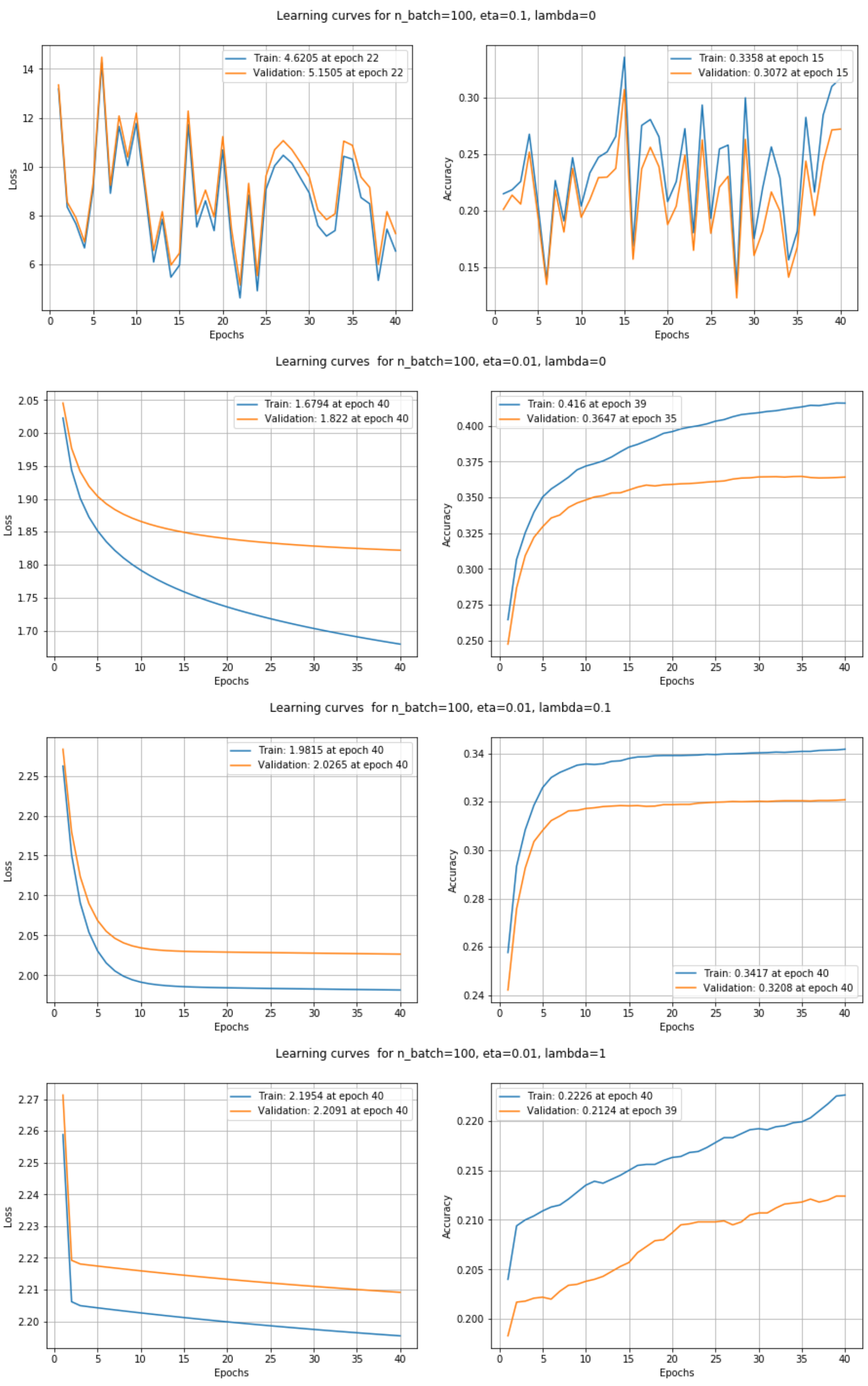


Figure 3: Learning curves for each 4 parametrization case.

In figure 4 we can see what class templates the network has learnt for each output node (weights matrix by rows) and parametrization case. Notice how regularization smooths out the weights, this is to be expected since ridge regression forces the weights to be closer to each other in value (i.e. closer to zero). Also, for some of the classes, we can see the shape of the image or contour of objects described such as for automobile or horse. Also, in figure 5 we can observe the histograms of weights by regularizations where the largest regularizations λ have a lower standard deviation (more centred to 0).

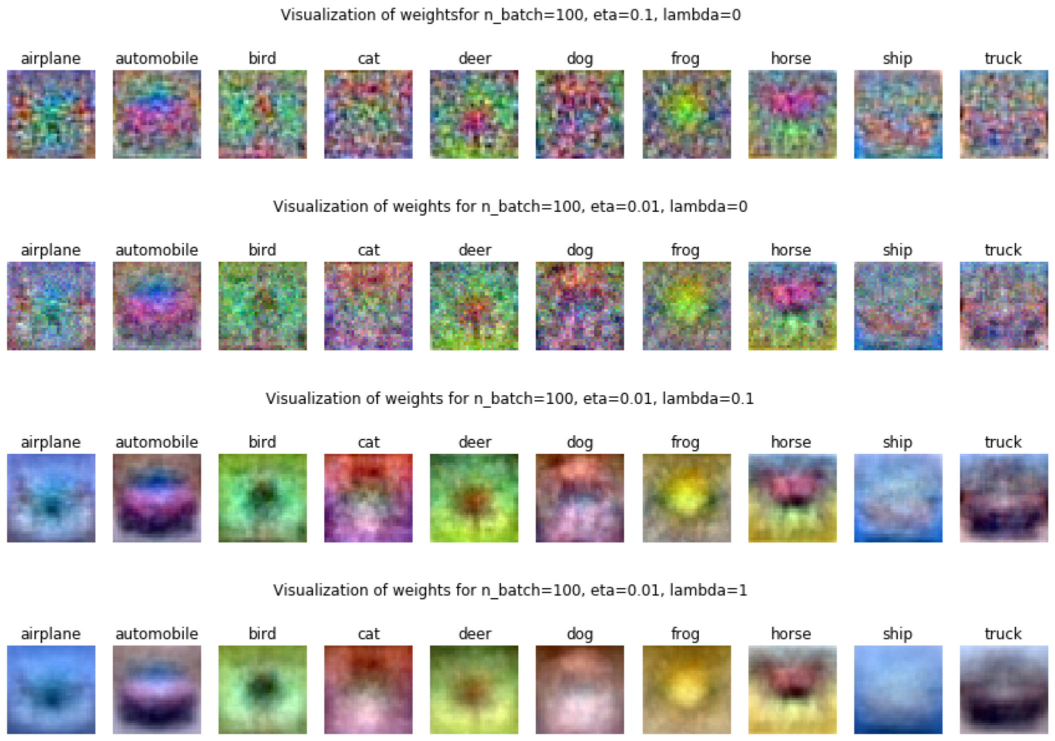


Figure 4: Weights learnt for each 4 parametrization case.

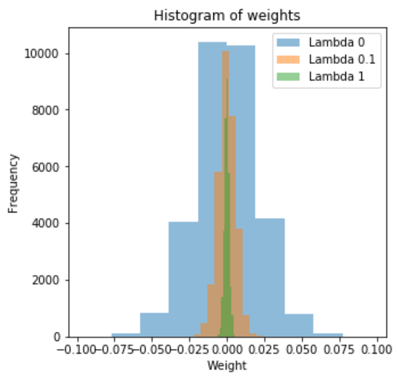
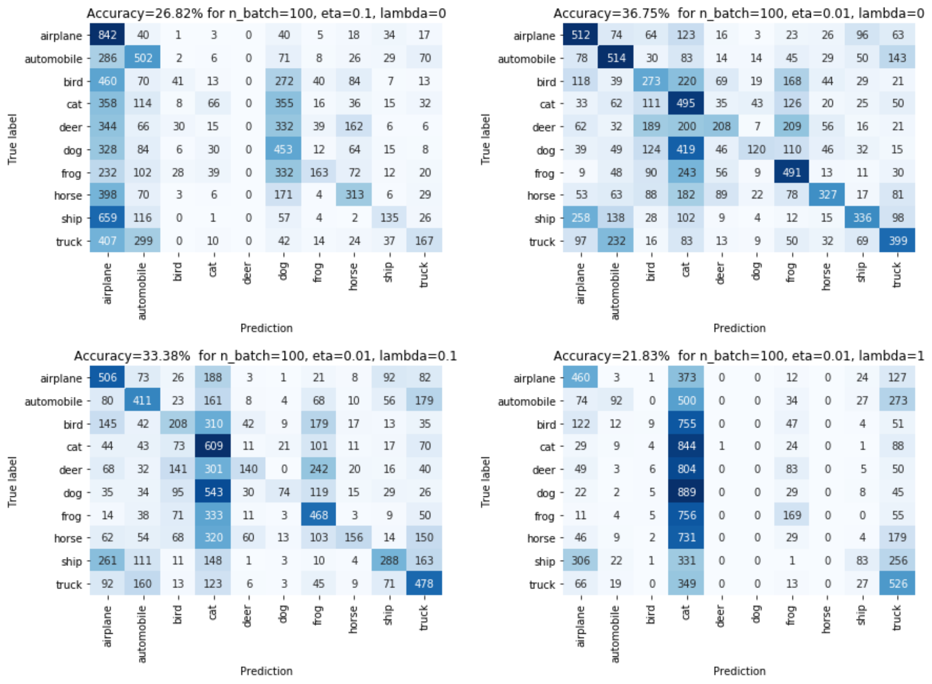


Figure 5: Histogram of weights by regularization levels (cases 2, 3 and 4).

In figure 6 we can find the test accuracy and confusion matrix for each parametrization case. Interestingly the networks with low overall accuracy seem to assign mostly a single class to most test samples. Notice that even the best network has a tendency to classify all animals like cats.



Accuracy=26.82% for n_batch=100, eta=0.1, lambda=0

True label \ Prediction	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	842	40	1	3	0	40	5	18	34	17
automobile	286	502	2	6	0	71	8	26	29	70
bird	460	70	41	13	0	272	40	84	7	13
cat	358	114	8	66	0	355	16	36	15	32
deer	344	66	30	15	0	332	39	162	6	6
dog	328	84	6	30	0	453	12	64	15	8
frog	232	102	28	39	0	332	163	72	12	20
horse	398	70	3	6	0	171	4	313	6	29
ship	659	116	0	1	0	57	4	2	135	26
truck	407	299	0	10	0	42	14	24	37	167

Accuracy=36.75% for n_batch=100, eta=0.01, lambda=0

True label \ Prediction	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	512	74	64	123	16	3	23	26	96	63
automobile	78	514	30	83	14	14	45	29	50	143
bird	118	39	273	220	69	19	168	44	29	21
cat	33	62	111	495	35	43	126	20	25	50
deer	62	32	189	200	208	7	209	56	16	21
dog	39	49	124	419	46	120	110	46	32	15
frog	9	48	90	243	56	9	491	13	11	30
horse	53	63	88	182	89	22	78	327	17	81
ship	258	138	28	102	9	4	12	15	336	98
truck	97	232	16	83	13	9	50	32	69	399

Accuracy=33.38% for n_batch=100, eta=0.01, lambda=0.1

True label \ Prediction	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	506	73	26	188	3	1	21	8	92	82
automobile	80	411	23	161	8	4	68	10	56	179
bird	145	42	208	310	42	9	179	17	13	35
cat	44	43	73	609	11	21	101	11	17	70
deer	68	32	141	301	140	0	242	20	16	40
dog	35	34	95	543	30	74	119	15	29	26
frog	14	38	71	333	11	3	468	3	9	50
horse	62	54	68	320	60	13	103	156	14	150
ship	261	111	11	148	1	3	10	4	288	163
truck	92	160	13	123	6	3	45	9	71	478

Accuracy=21.83% for n_batch=100, eta=0.01, lambda=1

True label \ Prediction	airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
airplane	460	3	1	373	0	0	12	0	24	127
automobile	74	92	0	500	0	0	34	0	27	273
bird	122	12	9	755	0	0	47	0	4	51
cat	29	9	4	844	1	0	24	0	1	88
deer	49	3	6	804	0	0	83	0	5	50
dog	22	2	5	889	0	0	29	0	8	45
frog	11	4	5	756	0	0	169	0	0	55
horse	46	9	2	731	0	0	29	0	4	179
ship	306	22	1	331	0	0	1	0	83	256
truck	66	19	0	349	0	0	13	0	27	526

Figure 6: Test accuracy and confusion matrix for each parametrization case.