Name: Nattapat Juthaprachakul, Student ID: 301350117

CMPT-726 Machine Learning: Assignment 2

1. **Softmax for multi-class classification**

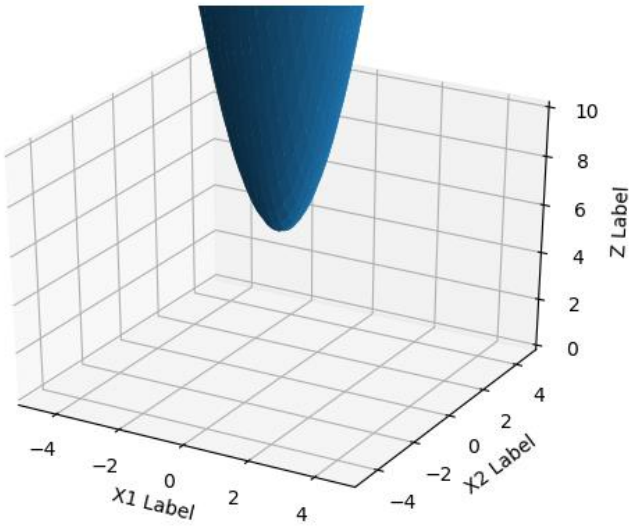
   1.1 ANS: The probability at the green point for each class is equally likely to be any of these 3 classes (0.33 percent for each class).

   1.2 ANS: The probability of input (green point) depends on which direction the green dot heads to (moving along the line in this case). For example, if the green dot moves downward along the red line, the probability of input x being class 1 and 3 (region 1 and 3) is more likely than class 2 (region 2). Also, the more the green dot moves downward, the more unlikely the input is classified as class 2 (region 2). In sum, when green dot moves along red line downward, the probability of input classified to be either class 1 or 3 is equally likely but unlikely to be class 2. This logic applies to both moving leftward and rightward as well. (Moving leftward probability of input being classified to be either class 2 and 3 is equally likely but unlikely to be class 1 and moving rightward probability of input being classified to be either class 1 and 2 is equally likely but unlikely to be class 3.)

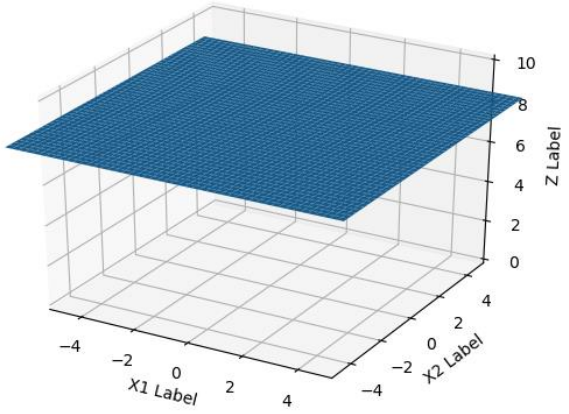
   1.3 ANS: The probability of inputs being classified as the following class depends on the region that the green dot is in. For example, if the green dot is in region 1, the input is likely to be classified as class 1 than classified as class 2 and 3.

2. Generalized Linear model for classification

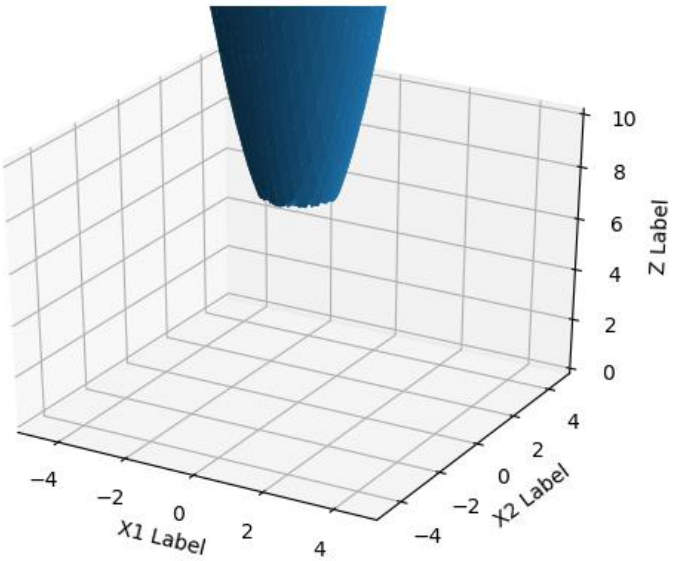
2.1



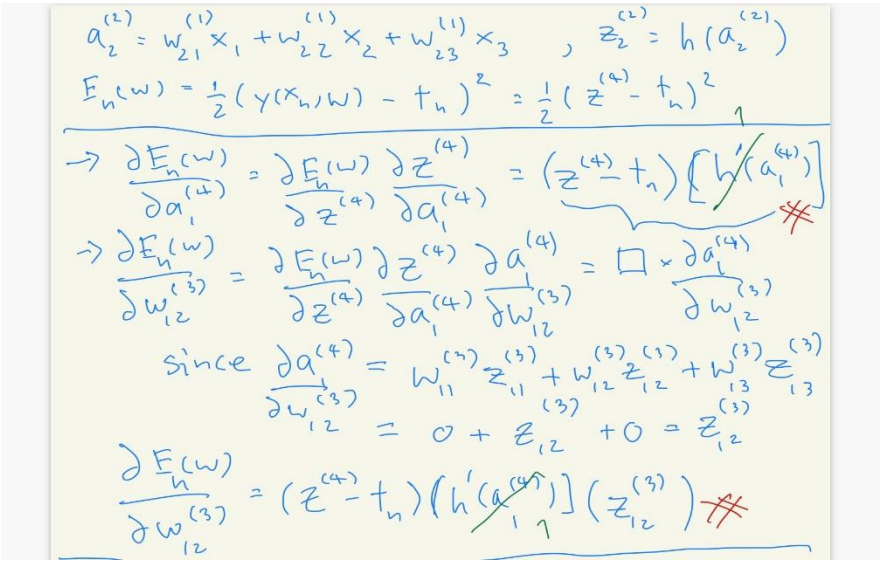
2.2



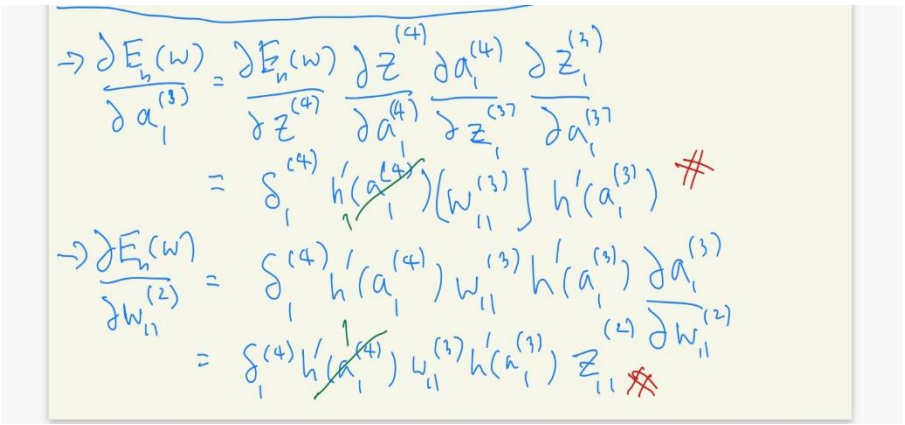
2.3



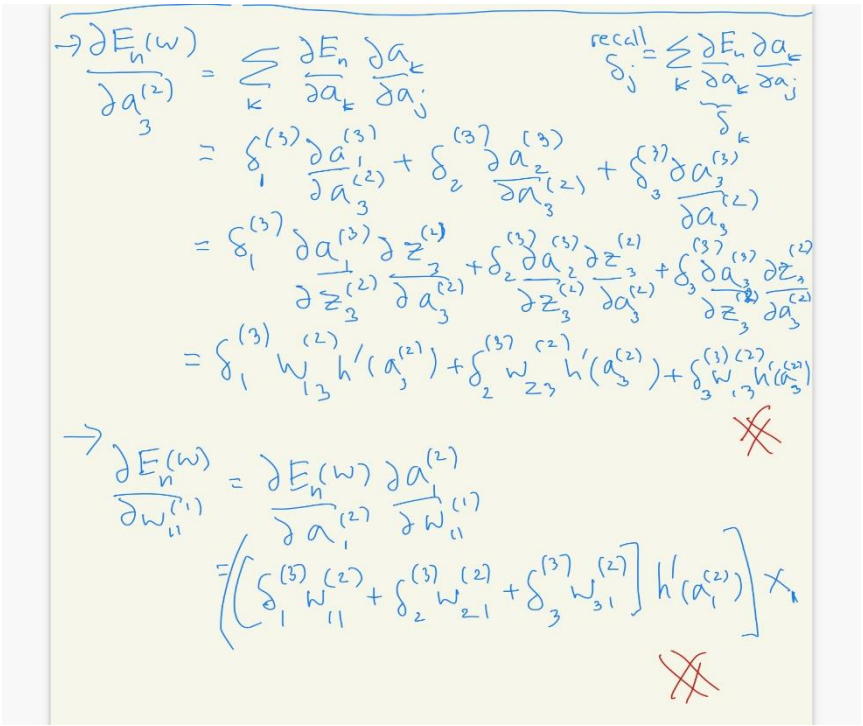
3. 3.1)ANS:



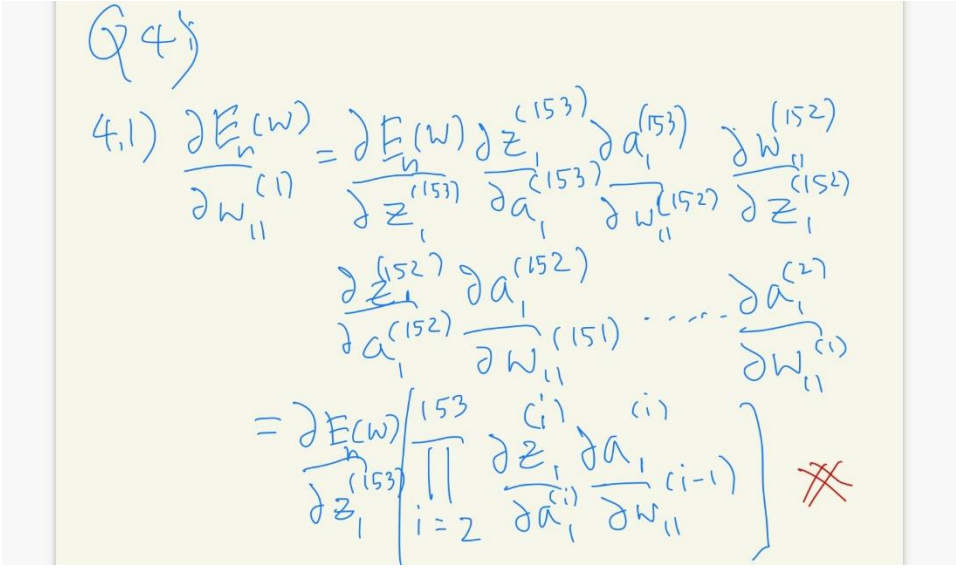
3.2)ANS:



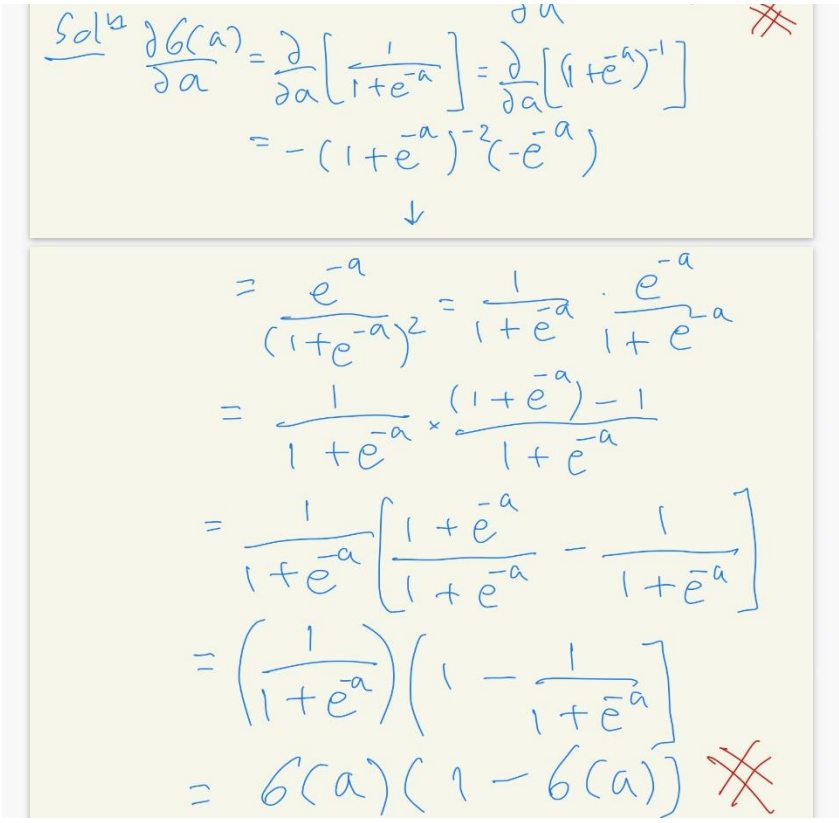
3.3)ANS:

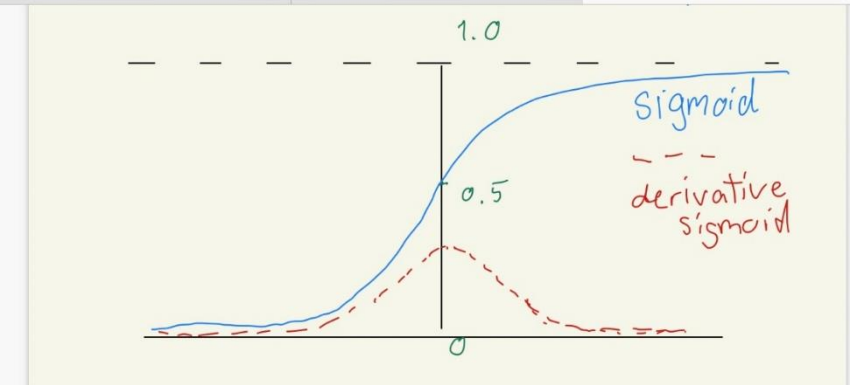


4. 4.1)ANS

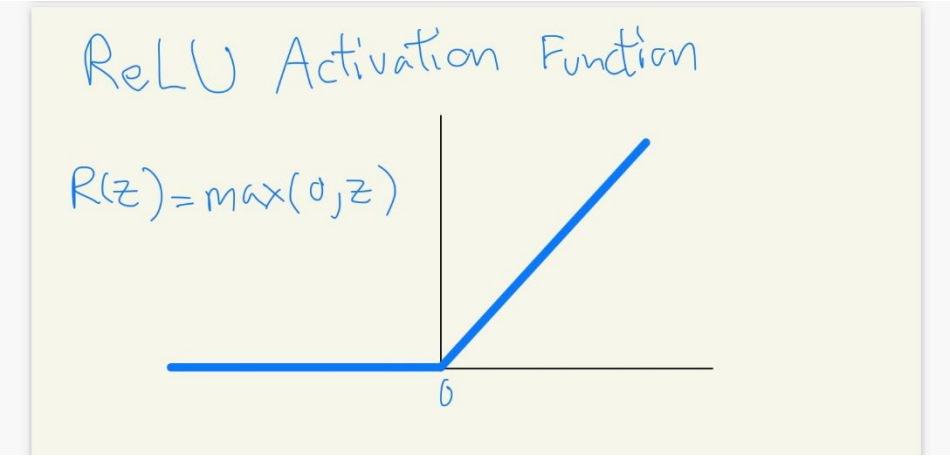


4.2)ANS: The learning will be very slow if the learning rate is very small and the area of the update is in the top and bottom curve (very flat area). The update is slow because the derivative is very small (small rate of change/update). For the softmax function, the gradient of weight could be small or zero when we do backpropagate to modify the weight to minimize the cost function through many layers and many connected nodes as the majority of derivative value of sigmoid lies between 0 and 0.25 (dotted red curve in the graph below), the multiplication of number between 0 and 1 many times will make the values smaller over time and become zero in some connection.





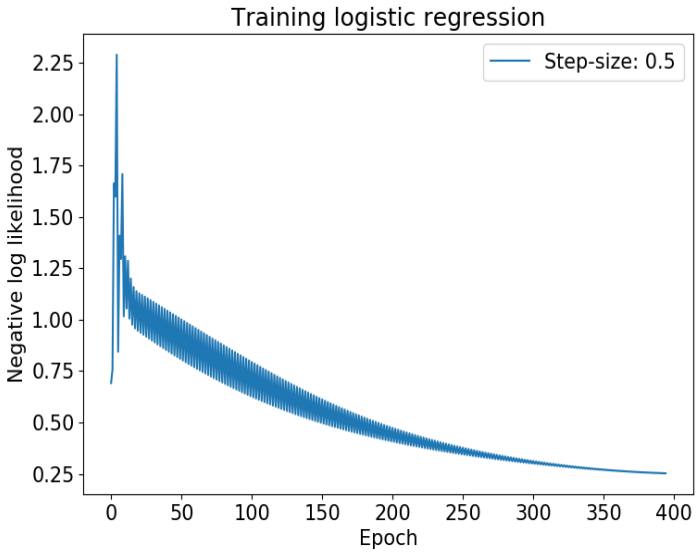
4.3)ANS: When the inputs to ReLU is equal or smaller than zero (the large negative bias term is learned which makes weighted sum of inputs becomes zero or negative), the output of ReLU will be zero, making the derivative become zero as well.

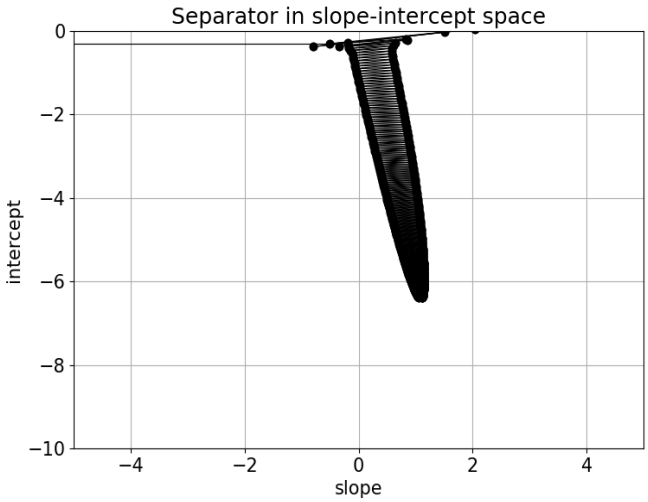


4.4)ANS: The gradient of weight could be zero when we do update/backpropagate to modify the weight to minimize the cost function through many layers and many connected nodes (bipartite). This might be the result when the majority of output from ReLU could become zero as the summation of weight term and bias is zero or negative. The function gradient at zero becomes zero as well.(the gradient descent learning will not alter the weights) This situation is called 'dead' ReLU.

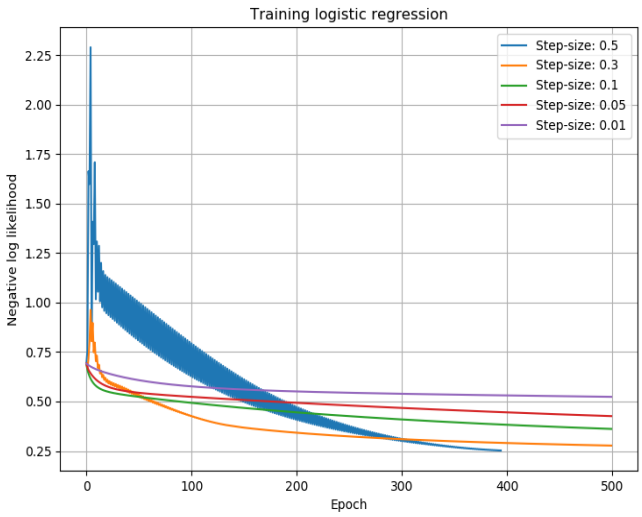
5. 5.1)ANS: When learning rate is too large, it causes drastic weight updates which lead to divergent behaviors (the oscillating curves) observed in the left and right pictures.

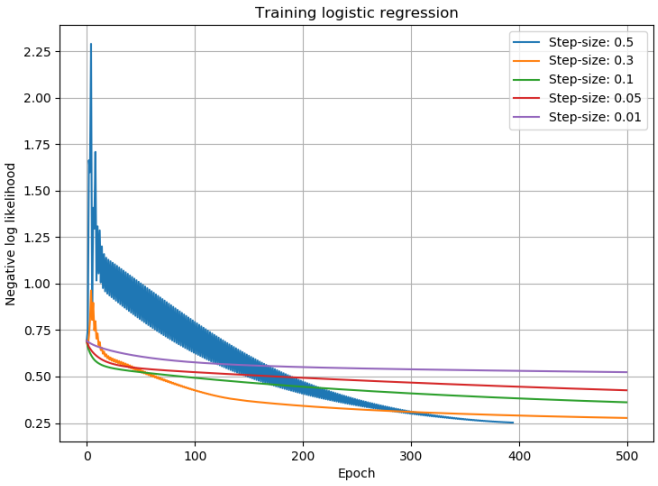
The performance of the model (such as its loss on the training dataset) will oscillate over training epochs. Oscillating performance is caused by weights that diverge.



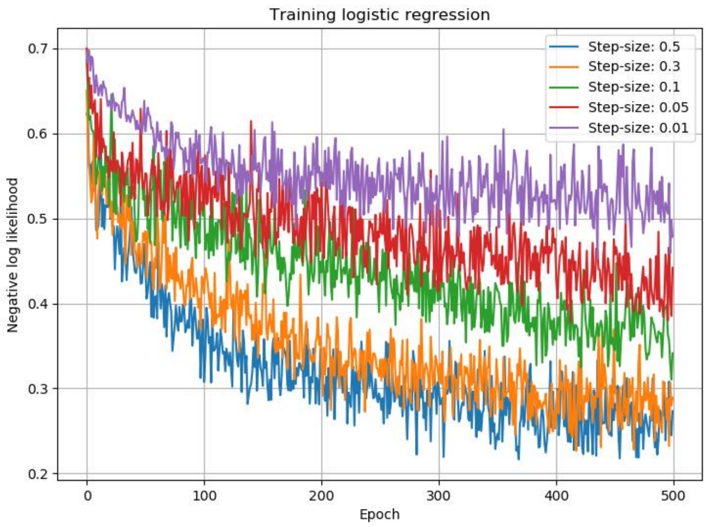


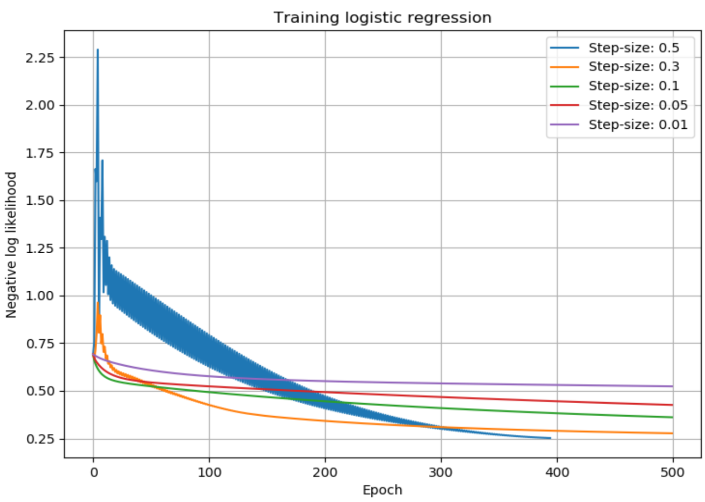
5.2)ANS: in our case, the 0.5 learning rate finds the lowest training error at the fewest number of epoch. As we can see the larger the learning rate is, the faster the training becomes (given the same training error).

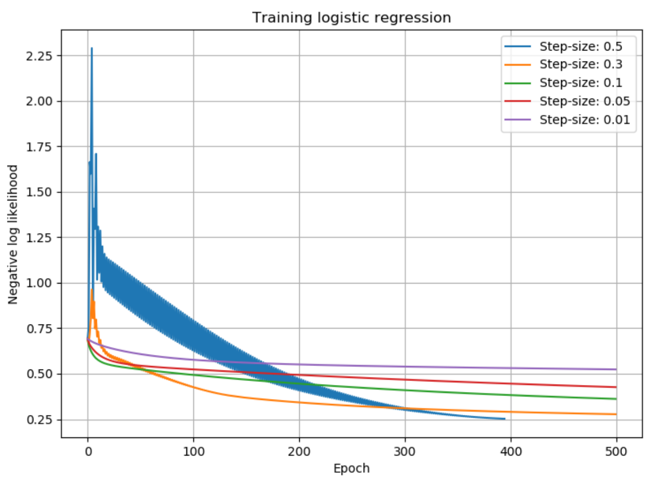




5.3)ANS: in our example, it is not so obvious that which update technique is better than another in term of speed(number of epoch) and performance(error rate). We can see that for the SGD graph is very oscillating since it calculates one data point then makes an update. However, in the setting with big data sets, Gradient Descent takes time to calculate cost or gradient as it needs to sum over all data points. Nonetheless, we do not need to have exact gradient to minimize the cost function in a given iteration. The approximation of gradient is enough; therefore, the Stochastic Gradient descent approximates the gradient using just only one data point at a time which in turn saves lots of time compared to summing over all data.







6. **Fine-Tuning a Pre-trained Network**

Settings: gaming laptop with a single Nvidia GTX 1050Ti

6.1) Main task that I do:

-Write a Python function to be used at the end of training that generates HTML output showing each test image and its classification scores. You could produce an HTML table output for example. (You can convert the HTML output to PDF or use screen shots.)

6.2) Other tasks:

-Try applying L2 regularization to the coefficients in the small networks we added.

-Try modifying the structure of the new layers that were added on top of ResNet20.

The main code is to evaluate the models and to save loss and classification score (max value from softmax output) for each test image. The next code generates each test image with its classification score and saves them all in our specified folder. The final code is to generate HTML table and convert it into PDF. I attached the filename: image_score_html.html and test_image.pdf.

I also created new additional layers (modifying the original assignment code of just one layer) on top of ResNet20 as follow: [feed forward layer -> batch normalization -> feed forward layer -> softmax output] with dropout technique, L2 regularization(weight decay) and specifiable number of hidden nodes.

Results: different epochs on training set of 50k inputs and 1 epoch on test set of 10k inputs.

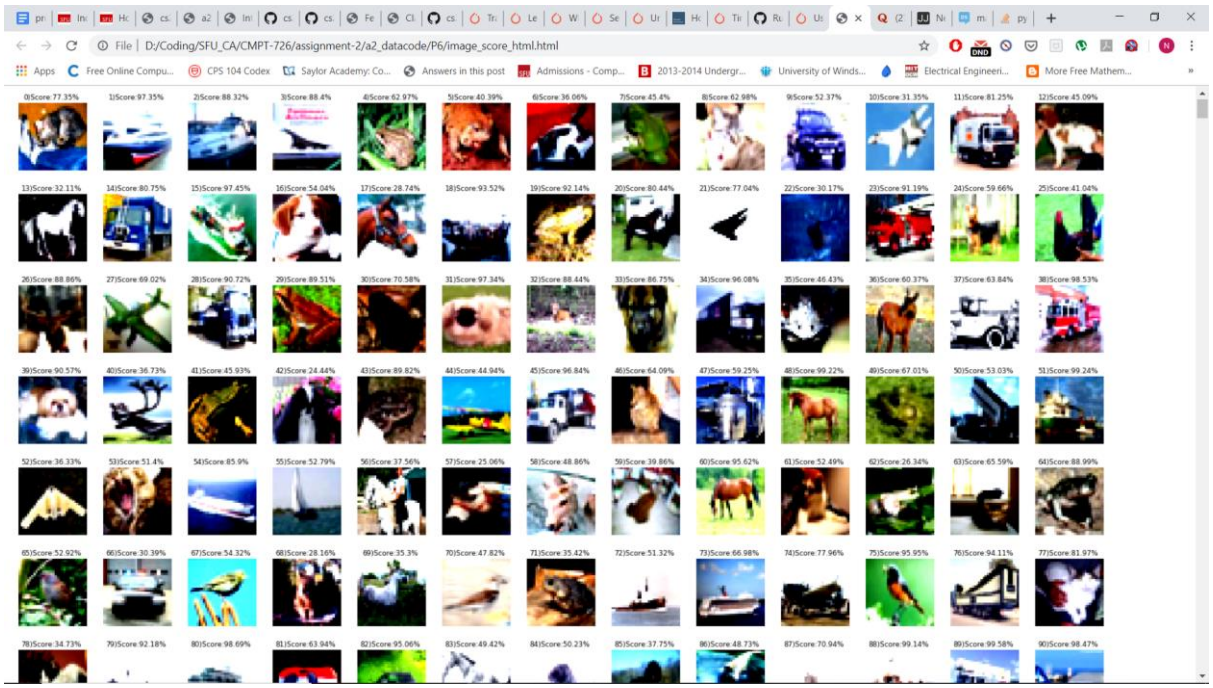
Note: Original feed-forward layer(64 nodes) denotes as fc(64)

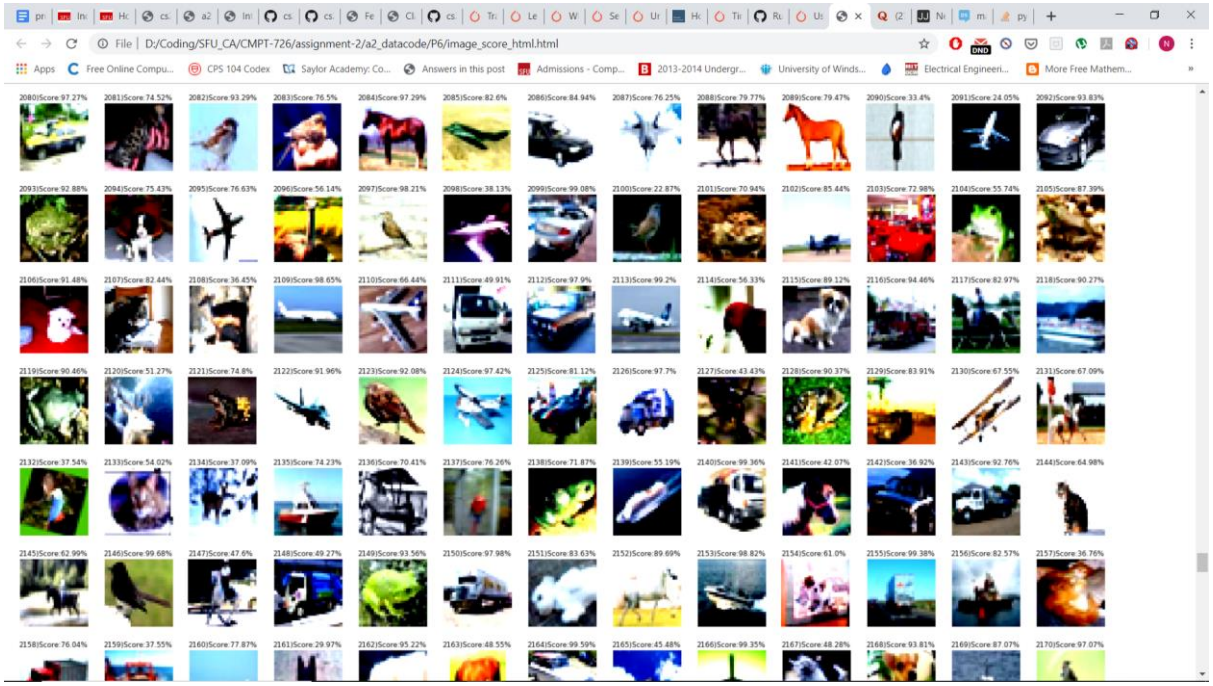
Additional feed-forward layer(256 nodes) denotes as fc(256)

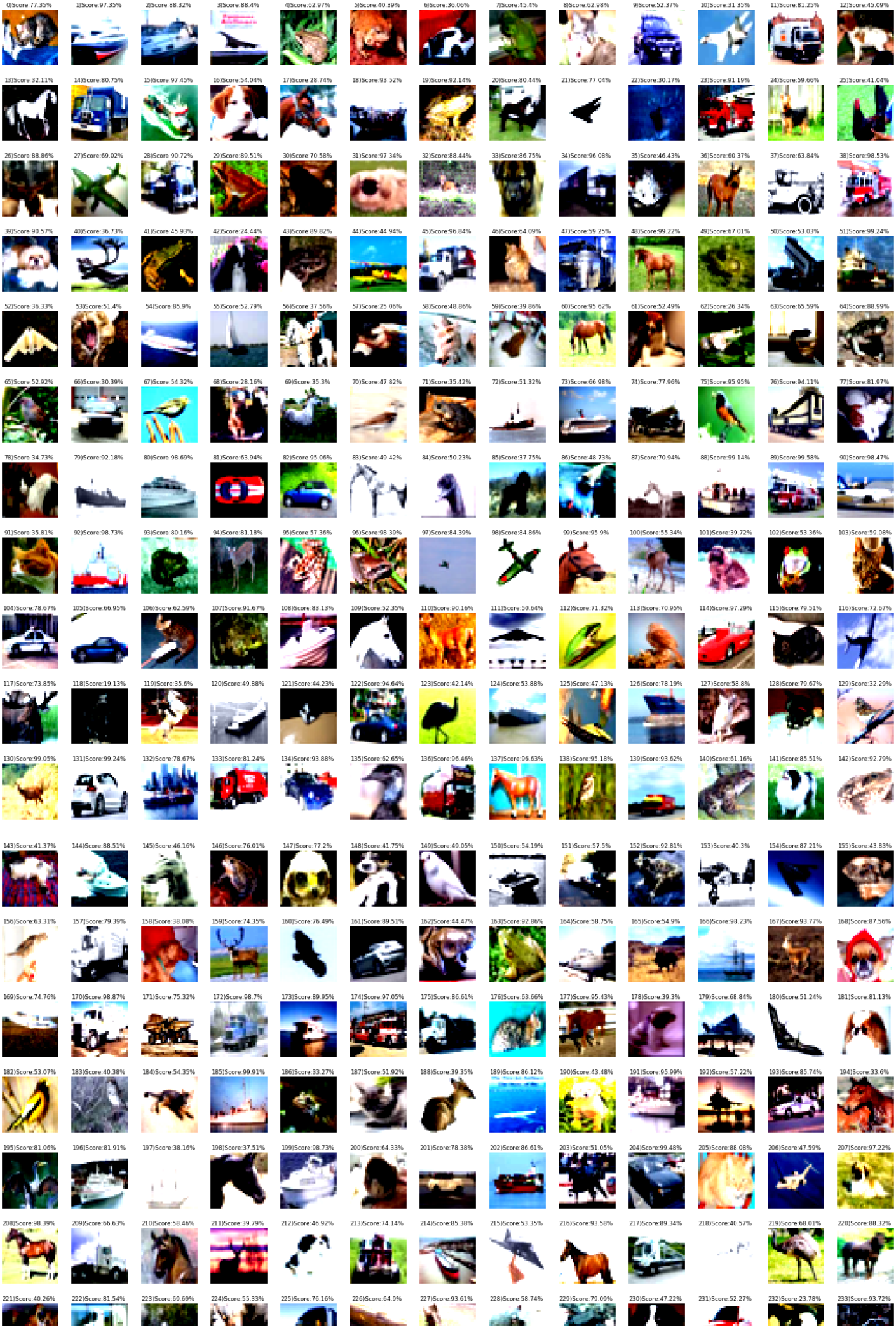
1. 1 epoch training, ResNet20 + fc(64)
   Running optimization on: fc(64)
   Accuracy: 65.10%
2. 10 epoch training, ResNet20 + fc(64)
   Running optimization on: fc(64)
   Accuracy: 68.55%
3. 10 epoch training, ResNet20 + fc(64), L2 regularization(0.001)
   Running optimization on: fc(64)
   Accuracy: 68.65%
4. 10 epoch training, ResNet20 + fc(64) + fc(256), L2 regularization(0.001)
   Running optimization on: fc(256)
   Accuracy: 68.52%
5. 10 epoch training, ResNet20 + fc(64)+ fc(256)
   Running optimization on: ResNet20 + fc(256)
   Accuracy: 84.46%
6. 10 epoch training, ResNet20 + fc(64)
   Running optimization on: ResNet20 + fc(64)

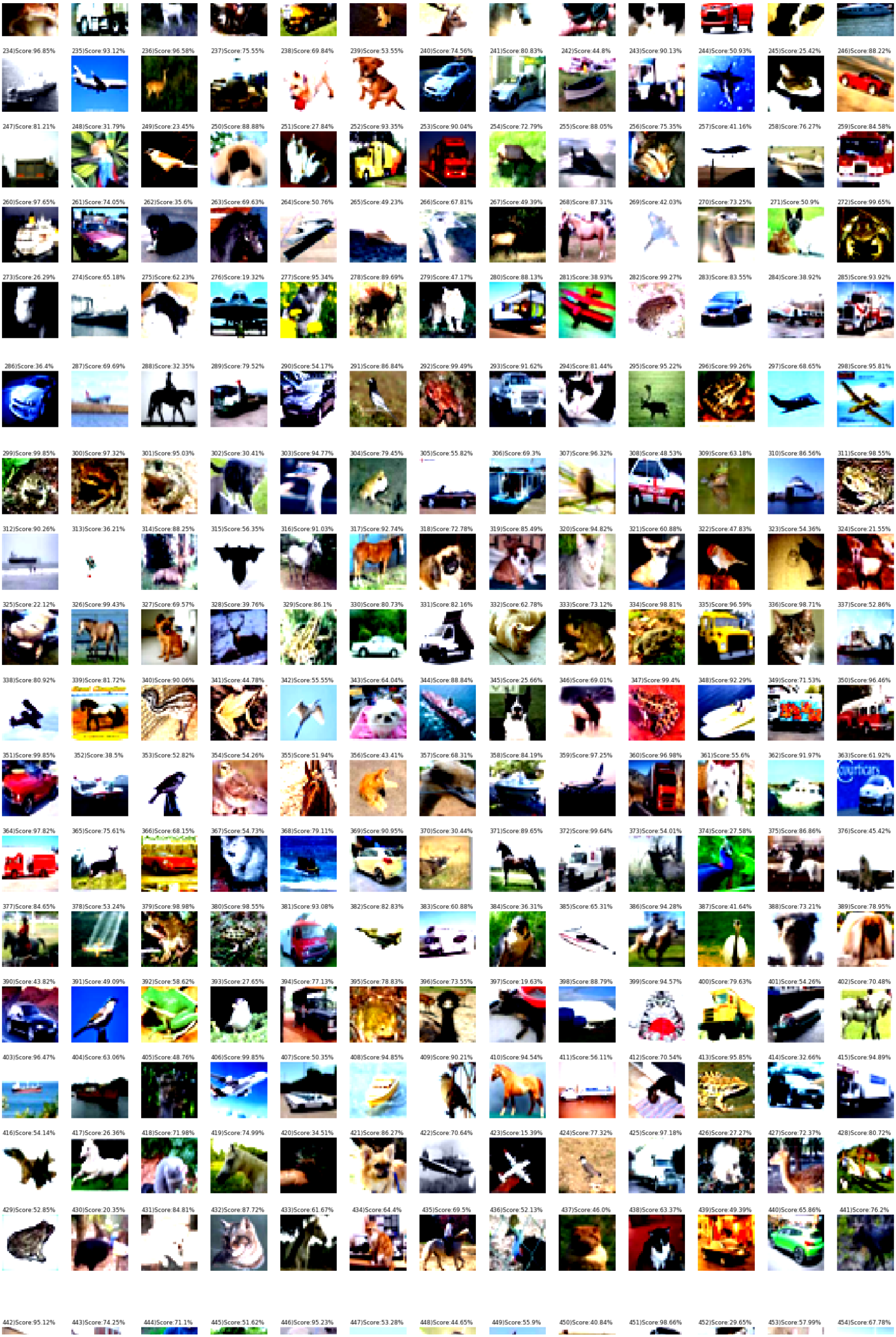
Accuracy: 84.44%

7. 10 epoch training, ResNet20 + fc(64)
   Running optimization on: ResNet20
   Accuracy: 80.33%
8. 10 epoch training, ResNet20 + fc(64), dropout at 0.2
   Running optimization on: ResNet20 + fc(64)
   Accuracy: 84.55%






0)Score:77.35%
1)Score:97.35%
2)Score:88.32%
3)Score:88.4%
4)Score:62.97%
5)Score:40.39%
6)Score:36.06%
7)Score:45.4%
8)Score:62.98%
9)Score:52.37%
10)Score:31.35%
11)Score:81.25%
12)Score:45.09%
13)Score:32.11%
14)Score:80.75%
15)Score:97.45%
16)Score:54.04%
17)Score:28.74%
18)Score:93.52%
19)Score:92.14%
20)Score:80.44%
21)Score:77.04%
22)Score:30.17%
23)Score:91.19%
24)Score:59.66%
25)Score:41.04%
26)Score:88.86%
27)Score:69.02%
28)Score:90.72%
29)Score:89.51%
30)Score:70.58%
31)Score:97.34%
32)Score:88.44%
33)Score:86.75%
34)Score:96.08%
35)Score:46.43%
36)Score:60.37%
37)Score:63.84%
38)Score:98.53%
39)Score:90.57%
40)Score:36.73%
41)Score:45.93%
42)Score:24.44%
43)Score:89.82%
44)Score:44.94%
45)Score:96.84%
46)Score:64.09%
47)Score:59.25%
48)Score:99.22%
49)Score:67.01%
50)Score:53.03%
51)Score:99.24%
52)Score:36.33%
53)Score:51.4%
54)Score:85.9%
55)Score:52.79%
56)Score:37.56%
57)Score:25.06%
58)Score:48.86%
59)Score:39.86%
60)Score:95.62%
61)Score:52.49%
62)Score:26.34%
63)Score:65.59%
64)Score:88.99%
65)Score:52.92%
66)Score:30.39%
67)Score:54.32%
68)Score:28.16%
69)Score:35.3%
70)Score:47.82%
71)Score:35.42%
72)Score:51.32%
73)Score:66.98%
74)Score:77.96%
75)Score:95.95%
76)Score:94.11%
77)Score:81.97%
78)Score:34.73%
79)Score:92.18%
80)Score:98.69%
81)Score:63.94%
82)Score:95.06%
83)Score:49.42%
84)Score:50.23%
85)Score:37.75%
86)Score:48.73%
87)Score:70.94%
88)Score:99.14%
89)Score:99.58%
90)Score:98.47%
91)Score:35.81%
92)Score:98.73%
93)Score:80.16%
94)Score:81.18%
95)Score:57.36%
96)Score:98.39%
97)Score:84.39%
98)Score:84.86%
99)Score:95.9%
100)Score:55.34%
101)Score:39.72%
102)Score:53.36%
103)Score:59.08%
104)Score:78.67%
105)Score:66.95%
106)Score:62.59%
107)Score:91.67%
108)Score:83.13%
109)Score:52.35%
110)Score:90.16%
111)Score:50.64%
112)Score:71.32%
113)Score:70.95%
114)Score:97.29%
115)Score:79.51%
116)Score:72.67%
117)Score:73.85%
118)Score:19.13%
119)Score:35.6%
120)Score:49.88%
121)Score:44.23%
122)Score:94.64%
123)Score:42.14%
124)Score:53.88%
125)Score:47.13%
126)Score:78.19%
127)Score:58.8%
128)Score:79.67%
129)Score:32.29%
130)Score:99.05%
131)Score:99.24%
132)Score:78.67%
133)Score:81.24%
134)Score:93.88%
135)Score:62.65%
136)Score:96.46%
137)Score:96.63%
138)Score:95.18%
139)Score:93.62%
140)Score:61.16%
141)Score:85.51%
142)Score:92.79%
143)Score:41.37%
144)Score:88.51%
145)Score:46.16%
146)Score:76.01%
147)Score:77.2%
148)Score:41.75%
149)Score:49.05%
150)Score:54.19%
151)Score:57.5%
152)Score:92.81%
153)Score:40.3%
154)Score:87.21%
155)Score:43.83%
156)Score:63.31%
157)Score:79.39%
158)Score:38.08%
159)Score:74.35%
160)Score:76.49%
161)Score:89.51%
162)Score:44.47%
163)Score:92.86%
164)Score:58.75%
165)Score:54.9%
166)Score:98.23%
167)Score:93.77%
168)Score:87.56%
169)Score:74.76%
170)Score:98.87%
171)Score:75.32%
172)Score:98.7%
173)Score:89.95%
174)Score:97.05%
175)Score:86.61%
176)Score:63.66%
177)Score:95.43%
178)Score:39.3%
179)Score:68.84%
180)Score:51.24%
181)Score:81.13%
182)Score:53.07%
183)Score:40.38%
184)Score:54.35%
185)Score:99.91%
186)Score:33.27%
187)Score:51.92%
188)Score:39.35%
189)Score:86.12%
190)Score:43.48%
191)Score:95.99%
192)Score:57.22%
193)Score:85.74%
194)Score:33.6%
195)Score:81.06%
196)Score:81.91%
197)Score:38.16%
198)Score:37.51%
199)Score:98.73%
200)Score:64.33%
201)Score:78.38%
202)Score:86.61%
203)Score:51.05%
204)Score:99.48%
205)Score:88.08%
206)Score:47.59%
207)Score:97.22%
208)Score:98.39%
209)Score:66.63%
210)Score:58.46%
211)Score:39.79%
212)Score:46.92%
213)Score:74.14%
214)Score:85.38%
215)Score:53.35%
216)Score:93.58%
217)Score:89.34%
218)Score:40.57%
219)Score:68.01%
220)Score:88.32%
221)Score:40.26%
222)Score:81.54%
223)Score:69.69%
224)Score:55.33%
225)Score:76.16%
226)Score:64.9%
227)Score:93.61%
228)Score:58.74%
229)Score:79.09%
230)Score:47.22%
231)Score:52.27%
232)Score:23.78%
233)Score:93.72%


234)Score:96.85%
235)Score:93.12%
236)Score:96.58%
237)Score:75.55%
238)Score:69.84%
239)Score:53.55%
240)Score:74.56%
241)Score:80.83%
242)Score:44.8%
243)Score:90.13%
244)Score:50.93%
245)Score:25.42%
246)Score:88.22%
247)Score:81.21%
248)Score:31.79%
249)Score:23.45%
250)Score:88.88%
251)Score:27.84%
252)Score:93.35%
253)Score:90.04%
254)Score:72.79%
255)Score:88.05%
256)Score:75.35%
257)Score:41.16%
258)Score:76.27%
259)Score:84.58%
260)Score:97.65%
261)Score:74.05%
262)Score:35.6%
263)Score:69.63%
264)Score:50.76%
265)Score:49.23%
266)Score:67.81%
267)Score:49.39%
268)Score:87.31%
269)Score:42.03%
270)Score:73.25%
271)Score:50.9%
272)Score:99.65%
273)Score:26.29%
274)Score:65.18%
275)Score:62.23%
276)Score:19.32%
277)Score:95.34%
278)Score:89.69%
279)Score:47.17%
280)Score:88.13%
281)Score:38.93%
282)Score:99.27%
283)Score:83.55%
284)Score:38.92%
285)Score:93.92%
286)Score:36.4%
287)Score:69.69%
288)Score:32.35%
289)Score:79.52%
290)Score:54.17%
291)Score:86.84%
292)Score:99.49%
293)Score:91.62%
294)Score:81.44%
295)Score:95.22%
296)Score:99.26%
297)Score:68.65%
298)Score:95.81%
299)Score:99.85%
300)Score:97.32%
301)Score:95.03%
302)Score:30.41%
303)Score:94.77%
304)Score:79.45%
305)Score:55.82%
306)Score:69.3%
307)Score:96.32%
308)Score:48.53%
309)Score:63.18%
310)Score:86.56%
311)Score:98.55%
312)Score:90.26%
313)Score:36.21%
314)Score:88.25%
315)Score:56.35%
316)Score:91.03%
317)Score:92.74%
318)Score:72.78%
319)Score:85.49%
320)Score:94.82%
321)Score:60.88%
322)Score:47.83%
323)Score:54.36%
324)Score:21.55%
325)Score:22.12%
326)Score:99.43%
327)Score:69.57%
328)Score:39.76%
329)Score:86.1%
330)Score:80.73%
331)Score:82.16%
332)Score:62.78%
333)Score:73.12%
334)Score:98.81%
335)Score:96.59%
336)Score:98.71%
337)Score:52.86%
338)Score:80.92%
339)Score:81.72%
340)Score:90.06%
341)Score:44.78%
342)Score:55.55%
343)Score:64.04%
344)Score:88.84%
345)Score:25.66%
346)Score:69.01%
347)Score:99.4%
348)Score:92.29%
349)Score:71.53%
350)Score:96.46%
351)Score:99.85%
352)Score:38.5%
353)Score:52.82%
354)Score:54.26%
355)Score:51.94%
356)Score:43.41%
357)Score:68.31%
358)Score:84.19%
359)Score:97.25%
360)Score:96.98%
361)Score:55.6%
362)Score:91.97%
363)Score:61.92%
364)Score:97.82%
365)Score:75.61%
366)Score:68.15%
367)Score:54.73%
368)Score:79.11%
369)Score:90.95%
370)Score:30.44%
371)Score:89.65%
372)Score:99.64%
373)Score:54.01%
374)Score:27.58%
375)Score:86.86%
376)Score:45.42%
377)Score:84.65%
378)Score:53.24%
379)Score:98.98%
380)Score:98.55%
381)Score:93.08%
382)Score:82.83%
383)Score:60.88%
384)Score:36.31%
385)Score:65.31%
386)Score:94.28%
387)Score:41.64%
388)Score:73.21%
389)Score:78.95%
390)Score:43.82%
391)Score:49.09%
392)Score:58.62%
393)Score:27.65%
394)Score:77.13%
395)Score:78.83%
396)Score:73.55%
397)Score:19.63%
398)Score:88.79%
399)Score:94.57%
400)Score:79.63%
401)Score:54.26%
402)Score:70.48%
403)Score:96.47%
404)Score:63.06%
405)Score:48.76%
406)Score:99.85%
407)Score:50.35%
408)Score:94.85%
409)Score:90.21%
410)Score:94.54%
411)Score:56.11%
412)Score:70.54%
413)Score:95.85%
414)Score:32.66%
415)Score:94.89%
416)Score:54.14%
417)Score:26.36%
418)Score:71.98%
419)Score:74.99%
420)Score:34.51%
421)Score:86.27%
422)Score:70.64%
423)Score:15.39%
424)Score:77.32%
425)Score:97.18%
426)Score:27.27%
427)Score:72.37%
428)Score:80.72%
429)Score:52.85%
430)Score:20.35%
431)Score:84.81%
432)Score:87.72%
433)Score:61.67%
434)Score:64.4%
435)Score:69.5%
436)Score:52.13%
437)Score:46.0%
438)Score:63.37%
439)Score:49.39%
440)Score:65.86%
441)Score:76.2%
442)Score:95.12%
443)Score:74.25%
444)Score:71.1%
445)Score:51.62%
446)Score:95.23%
447)Score:53.28%
448)Score:44.65%
449)Score:55.9%
450)Score:40.84%
451)Score:98.66%
452)Score:29.65%
453)Score:57.99%
454)Score:67.78%