Windows PowerShell

版权所有 (C) 2016 Microsoft Corporation。保留所有权利。

PS C:\WINDOWS\system32> cd ..

PS C:\WINDOWS> cd ..

PS C:\> cd .\Users\

PS C:\Users> cd .\lxy98\

PS C:\Users\lxy98> cd .\Documents\

PS C:\Users\lxy98\Documents> cd .\MyGitProjects\

PS C:\Users\lxy98\Documents\MyGitProjects> cd .\test\_your\_own-dlsys\

PS C:\Users\lxy98\Documents\MyGitProjects\test\_your\_own-dlsys> cd .\testcase\_my\

PS C:\Users\lxy98\Documents\MyGitProjects\test\_your\_own-dlsys\testcase\_my> ls

目录: C:\Users\lxy98\Documents\MyGitProjects\test\_your\_own-dlsys\testcase\_my

Mode LastWriteTime Length Name

---- ------------- ------ ----

d----- 2017-7-27 下午 02:54 MNIST\_data

d----- 2017-8-7 上午 11:57 tensorwolf

-a---- 2017-8-1 上午 10:12 2728 10\_cnn\_2.py

-a---- 2017-7-25 下午 05:18 360 1\_adder.py

-a---- 2017-7-25 下午 04:25 474 2\_init.py

-a---- 2017-7-25 下午 04:51 690 3\_assign.py

-a---- 2017-7-25 下午 05:10 551 4\_context.py

-a---- 2017-7-28 下午 04:10 1283 5\_mnist\_grad.py

-a---- 2017-7-28 下午 04:31 1267 6\_mnist\_optimizer.py

-a---- 2017-7-31 上午 10:38 3289 7\_ml\_perceptron.py

-a---- 2017-7-31 下午 04:22 1258 8\_adam.py

-a---- 2017-8-7 下午 05:00 2624 9\_cnn\_1.py

-a---- 2017-7-28 上午 11:42 1572 autodiff.py

-a---- 2017-7-31 上午 10:36 1496 checkSMCE.py

-a---- 2017-7-31 上午 08:57 1043 mytest.py

-a---- 2017-8-6 上午 12:29 3002 quick\_cnn1.py

-a---- 2017-8-7 下午 05:00 2729 testConv.py

-a---- 2017-8-7 下午 04:12 817 testConv.py.lprof

-a---- 2017-8-5 下午 07:44 130390 testConv.py.prof

PS C:\Users\lxy98\Documents\MyGitProjects\test\_your\_own-dlsys\testcase\_my> kernprof -l -v .\testConv.py > profile.txt

PS C:\Users\lxy98\Documents\MyGitProjects\test\_your\_own-dlsys\testcase\_my> kernprof -l -v .\9\_cnn\_1.py

Extracting MNIST\_data/train-images-idx3-ubyte.gz

Extracting MNIST\_data/train-labels-idx1-ubyte.gz

Extracting MNIST\_data/t10k-images-idx3-ubyte.gz

Extracting MNIST\_data/t10k-labels-idx1-ubyte.gz

0

Step 0, trainning accuracy 0.16

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

Step 50, trainning accuracy 0.75

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

Step 100, trainning accuracy 0.81

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

Step 150, trainning accuracy 0.82

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

Test accuracy: 0.8874

Wrote profile results to 9\_cnn\_1.py.lprof

Timer unit: 4.27653e-07 s

Total time: 567.053 s

File: .\tensorwolf\executor.py

Function: run at line 24

Line # Hits Time Per Hit % Time Line Contents

==============================================================

24 @profile

25 def run(self, feed\_dict):

26 """

27 Parameters

28 ----------

29 feed\_dict: a dictionary of node->np.ndarray supplied by user.

30

31 Returns

32 -------

33 A list of values for nodes in eval\_node\_list. NDArray or np.ndarray.

34 """

35 206 1762 8.6 0.0 node\_to\_val\_map = {}

36 616 4300 7.0 0.0 for node, value in feed\_dict.items():

37 410 96784 236.1 0.0 node\_to\_val\_map[node] = np.array(value)

38

39 18957 85988 4.5 0.0 for node in self.topo\_order:

40 18751 86876 4.6 0.0 if node in node\_to\_val\_map:

41 410 2008 4.9 0.0 continue

42 #print("Compute: ", node.name)

43 #print("Compute-Type: ", type(node.op))

44 18341 297593 16.2 0.0 input\_vals = [node\_to\_val\_map[n] for n in node.inputs]

45 18341 1325226516 72254.9 99.9 value = node.op.compute(node, input\_vals)

46 # if isinstance(value, np.ndarray):

47 # print("shape:", value.shape)

48 18341 159086 8.7 0.0 node\_to\_val\_map[node] = value

49 # os.system("PAUSE")

50

51 206 2230 10.8 0.0 return [node\_to\_val\_map[n] for n in self.eval\_node\_list]

Total time: 202.836 s

File: .\tensorwolf\ops.py

Function: compute at line 728

Line # Hits Time Per Hit % Time Line Contents

==============================================================

728 @profile

729 def compute(self, node, input\_vals):

730 410 2019 4.9 0.0 return c\_ops.correlate2d(

731 410 1234 3.0 0.0 input=input\_vals[0],

732 410 818 2.0 0.0 filter=input\_vals[1],

733 410 1574 3.8 0.0 strides=node.const\_attr[0],

734 410 474295346 1156817.9 100.0 padding=node.const\_attr[1]

735 )

Total time: 131.751 s

File: .\tensorwolf\ops.py

Function: compute at line 749

Line # Hits Time Per Hit % Time Line Contents

==============================================================

749 @profile

750 def compute(self, node, input\_vals):

751 200 1199 6.0 0.0 return c\_ops.correlate2d(

752 200 692 3.5 0.0 input=input\_vals[2],

753 200 1316 6.6 0.0 filter=np.rot90(np.transpose(

754 200 32397 162.0 0.0 input\_vals[1], (0, 1, 3, 2)), axes=(0, 1), k=2),

755 200 821 4.1 0.0 strides=[1, 1, 1, 1],

756 200 308043451 1540217.3 100.0 padding=node.const\_attr[1]

757 )

Total time: 123.719 s

File: .\tensorwolf\ops.py

Function: compute at line 770

Line # Hits Time Per Hit % Time Line Contents

==============================================================

770 @profile

771 def compute(self, node, input\_vals):

772 # only handle "SAME"

773 400 1488 3.7 0.0 assert node.const\_attr[1] == "SAME"

774 400 1387 3.5 0.0 return c\_ops.conv2d\_filter\_gradient(

775 400 614 1.5 0.0 input=input\_vals[0],

776 400 562 1.4 0.0 gradient=input\_vals[2],

777 400 289294411 723236.0 100.0 ori\_filter=input\_vals[1]

778 )

Total time: 41.5482 s

File: .\tensorwolf\ops.py

Function: compute at line 792

Line # Hits Time Per Hit % Time Line Contents

==============================================================

792 @profile

793 def compute(self, node, input\_vals):

794 410 2361 5.8 0.0 assert len(input\_vals) == 1

795 # check shape

796 410 3008 7.3 0.0 batchs = input\_vals[0].shape[0]

797 410 1002 2.4 0.0 i\_h = input\_vals[0].shape[1]

798 410 904 2.2 0.0 i\_w = input\_vals[0].shape[2]

799 410 926 2.3 0.0 i\_c = input\_vals[0].shape[3]

800 # zero padding

801 410 1192 2.9 0.0 strides = node.const\_attr[1]

802 410 1045 2.5 0.0 ksize = node.const\_attr[0]

803 410 1875 4.6 0.0 o\_h = (i\_h - 1) // strides[1] + 1

804 410 948 2.3 0.0 o\_w = (i\_w - 1) // strides[2] + 1

805 410 1184 2.9 0.0 if node.const\_attr[2] == 'SAME':

806 410 1307 3.2 0.0 z\_h = ((i\_h - 1) // strides[1]) \* strides[1] + ksize[1]

807 410 1074 2.6 0.0 z\_w = ((i\_w - 1) // strides[2]) \* strides[2] + ksize[2]

808 410 3949 9.6 0.0 z = zero\_padding\_func(ori=input\_vals[0], up=(z\_h - i\_h) // 2, down=(z\_h -

i\_h + 1) // 2,

809 410 35632110 86907.6 36.7 left=(z\_w - i\_w) // 2, right=(z\_w - i\_w + 1) // 2)

810 else:

811 raise NotImplementedError

812 410 147704 360.3 0.2 output\_val = np.zeros([batchs, o\_h, o\_w, i\_c])

813 4715 13806 2.9 0.0 for i in range(o\_h):

814 54530 191642 3.5 0.2 for j in range(o\_w):

815 50225 140597 2.8 0.1 output\_val[:, i, j, :] = np.max(

816 50225 61006479 1214.7 62.8 get\_patch(z, i, j, ksize[1], ksize[2], strides), axis=(1, 2))

817 410 792 1.9 0.0 return output\_val

Total time: 9.2138 s

File: .\tensorwolf\ops.py

Function: compute at line 830

Line # Hits Time Per Hit % Time Line Contents

==============================================================

830 @profile

831 def compute(self, node, input\_vals):

832 400 2865 7.2 0.0 assert len(input\_vals) == 2

833 # check shape

834 400 3157 7.9 0.0 batchs = input\_vals[0].shape[0]

835 400 1593 4.0 0.0 i\_h = input\_vals[0].shape[1]

836 400 1547 3.9 0.0 i\_w = input\_vals[0].shape[2]

837 400 1478 3.7 0.0 i\_c = input\_vals[0].shape[3]

838 # zero padding

839 400 1766 4.4 0.0 strides = node.const\_attr[1]

840 400 1645 4.1 0.0 ksize = node.const\_attr[0]

841 400 2738 6.8 0.0 o\_h = (i\_h - 1) // strides[1] + 1

842 400 1534 3.8 0.0 o\_w = (i\_w - 1) // strides[2] + 1

843 400 1735 4.3 0.0 if node.const\_attr[2] == 'SAME':

844 400 2023 5.1 0.0 z\_h = ((i\_h - 1) // strides[1]) \* strides[1] + ksize[1]

845 400 1704 4.3 0.0 z\_w = ((i\_w - 1) // strides[2]) \* strides[2] + ksize[2]

846 400 2545 6.4 0.0 z = zero\_padding\_func(ori=input\_vals[0], up=(z\_h - i\_h) // 2, down=(z\_h -

i\_h + 1) // 2,

847 400 7011419 17528.5 32.5 left=(z\_w - i\_w) // 2, right=(z\_w - i\_w + 1) // 2)

848 else:

849 raise NotImplementedError

850 '''

851 print("i\_size", i\_h, i\_w)

852 print("o\_size:", o\_h, o\_w)

853 print("z\_size:", z\_h, z\_w)

854 '''

855 '''

856 # all up date version

857 output\_val = np.zeros((batchs, z\_h, z\_w, i\_c))

858 for i in range(o\_h):

859 for j in range(o\_w):

860 nw = get\_patch(z, i, j, ksize[1], ksize[2], strides)

861 valid = np.equal(nw, np.max(

862 nw, axis=(1, 2), keepdims=True)).astype(np.float32)

863 get\_patch(output\_val, i, j, ksize[1], ksize[2], strides)[

864 :, :, :, :] = valid \* input\_vals[1][:, i:i + 1, j:j + 1, :]

865 up = (z\_h - i\_h) // 2

866 left = (z\_w - i\_w) // 2

867 output\_val = output\_val[:, up:up + i\_h, left:left + i\_w, :]

868 '''

869

870 # update one version

871 400 51642 129.1 0.2 output\_val = np.zeros((batchs, z\_h, z\_w, i\_c), dtype=np.float32)

872 400 2371 5.9 0.0 c\_ops.max\_pool\_gradient(

873 400 2045 5.1 0.0 gradient=input\_vals[1],

874 400 1088 2.7 0.0 input=z,

875 400 1105 2.8 0.0 output=output\_val,

876 400 1263 3.2 0.0 ksize=ksize,

877 400 14430805 36077.0 67.0 strides=strides

878 )

879 '''

880 for b in range(batchs):

881 for c in range(i\_c):

882 for i in range(o\_h):

883 for j in range(o\_w):

884 get\_patch(output\_val, i, j, ksize[1], ksize[2], strides)[b, :,

:, c].flat[

885 np.argmax(

886 get\_patch(z, i, j, ksize[1], ksize[2], strides)[b, :,

:, c])

887 ] = input\_vals[1][b, i, j, c]

888 '''

889

890 400 6044 15.1 0.0 up = (z\_h - i\_h) // 2

891 400 1453 3.6 0.0 left = (z\_w - i\_w) // 2

892 400 8135 20.3 0.0 output\_val = output\_val[:, up:up + i\_h, left:left + i\_w, :]

893

894 400 1309 3.3 0.0 return output\_val

PS C:\Users\lxy98\Documents\MyGitProjects\test\_your\_own-dlsys\testcase\_my>