

Study Report

Shuo Xu

August 19, 2018

Abstract

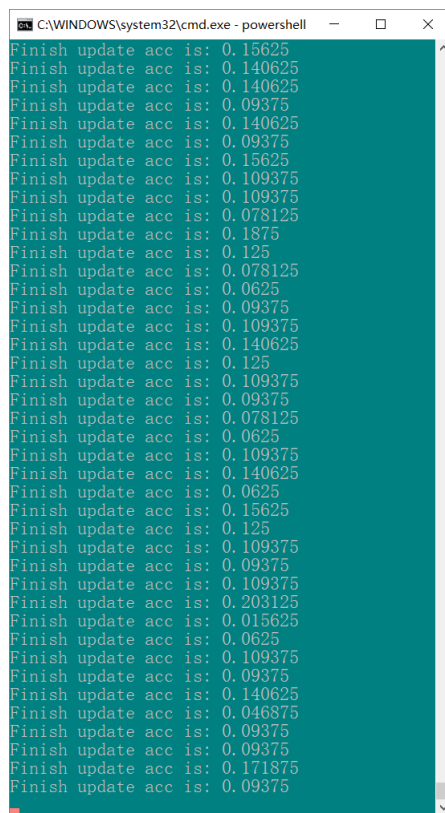
Briefly describe the problems I've encountered and the solutions I've come up with. And the leetcode.

New Problems with CNN

After I asked you that's problems, I simplified my code and decided to run it through the 60,000 photos. I set the learning rate to 0.01 and set the batch size to 64. In the process of running the program I found that training a batch of data actually need three seconds. I've simplified the program as much as I can, but the time is still huge.

After running the program for half an hour, I realized that the batch success rate was below 0.2, and I realised there was a problem inside the program. At first I thought it was my learning rate set is not reasonable enough, so I changed the learning rate and run the program several times. But things aren't getting any better.

Then I ran the program two times and recorded the initial output.



```
C:\WINDOWS\system32\cmd.exe - powershell
Finish update acc is: 0.15625
Finish update acc is: 0.140625
Finish update acc is: 0.140625
Finish update acc is: 0.09375
Finish update acc is: 0.140625
Finish update acc is: 0.09375
Finish update acc is: 0.15625
Finish update acc is: 0.109375
Finish update acc is: 0.109375
Finish update acc is: 0.078125
Finish update acc is: 0.1875
Finish update acc is: 0.125
Finish update acc is: 0.078125
Finish update acc is: 0.0625
Finish update acc is: 0.09375
Finish update acc is: 0.109375
Finish update acc is: 0.140625
Finish update acc is: 0.125
Finish update acc is: 0.109375
Finish update acc is: 0.09375
Finish update acc is: 0.078125
Finish update acc is: 0.0625
Finish update acc is: 0.109375
Finish update acc is: 0.140625
Finish update acc is: 0.0625
Finish update acc is: 0.15625
Finish update acc is: 0.125
Finish update acc is: 0.109375
Finish update acc is: 0.09375
Finish update acc is: 0.109375
Finish update acc is: 0.203125
Finish update acc is: 0.015625
Finish update acc is: 0.0625
Finish update acc is: 0.109375
Finish update acc is: 0.09375
Finish update acc is: 0.140625
Finish update acc is: 0.046875
Finish update acc is: 0.09375
Finish update acc is: 0.09375
Finish update acc is: 0.171875
Finish update acc is: 0.09375
```

Although the learning rate is different, but each time the output is the same as the picture. So I checked the back-propagation part and found that the gradient disappeared. The calculated derivatives were generally less than 0.1. This is a question that should be first thought of. I am looking for suitable initialization mode.

This week I'm also working on improving my code, using classes to write every layer of the neural network, trying to write the form of a tensorflow framework. But the problem of gradient disappearing is still waiting for me to solve.

Leetcode

Word Pattern

- Description

Given a pattern and a string str, find if str follows the same pattern.

Here follow means a full match, such that there is a bijection between a letter in pattern and a non-empty word in str.

- Solution

At first, use istream to create a vector from the input str. Use .size() to get the number of the input pattern and the vector we have created. If they are not equal, return false. Then use two map for the next step, the first one is char corresponds to string another one is string corresponds to char. Use a index i to traverse the string and the vector. When meet new elements, use the find function with the two new elements as the keys and end function of map to judge whether elements are stored. If they are not stored, add them to the map. If they are stored, judge whether the values of two map correspond to each other. If not, return false. There is another special case, only one element was stored. In this case, we only need to return false.

- Code

```
1 class Solution {
2 public:
3     bool wordPattern(string pattern, string str) {
4         istream iss(str);
5         istream_iterator<string> Itbegin = istream_iterator<string>(iss);
6         istream_iterator<string> Itend = istream_iterator<string>();
7         vector<string> strs(Itbegin, Itend);
8         int len1 = pattern.size(), len2 = strs.size();
9         if(len1 != len2)
10            return false;
11         map<char, string> mp1;
12         map<string, char> mp2;
13         for(int i = 0; i < len1; i++){
14             auto it1 = mp1.find(pattern[i]);
15             auto it2 = mp2.find(strs[i]);
16             if(it1 == mp1.end() && it2 == mp2.end()){
17                 mp1[pattern[i]] = strs[i];
18                 mp2[strs[i]] = pattern[i];
19             }
20             else if(it1 != mp1.end() && it2 != mp2.end()){
21                 if(mp1[pattern[i]] != strs[i] || mp2[strs[i]] != pattern[i])
```

```
22                                     return false;
23                                 }
24                             else
25                                 return false;
26                             }
27                         return true;
28                     }
29 };
```