20220515-机器学习

1.学习内容

1.1 CNN类

2.结果描述

1.学习内容

1.1 CNN类

CNN.h C++ 复制代码

```cpp
#pragma once
#ifndef CNN_H_
#define CNN_H_
/*
1)读取训练数据并将其存储进一个Matrix对象中

2)



存储模型参数


*/


#include "Matrix.h"
#include <string>
#include <fstream>
#include <iostream>
#include <vector>
#pragma warning(disable:4996)

class CNN
{
public:
    CNN();
    std::vector<std::vector<uint8_t>> GetFeature(std::string feature_file);
    std::vector<uint8_t> GetLabel(std::string label_file);
    uint32_t convert_to_little_endian(const unsigned char* bytes);
    std::vector<std::vector<float>> filter_inl(int num_f,int num_r, int num_c);
    void conv_layer(Matrix FeatureData,std::vector<std::vector<float>> filter);
    /*
    void train();
    float train_loss();
    float test_loss();
    void optimization();

    void pooling_layer();
    */
private:
    uint32_t numPic;
```

```
43        uint32_t numPixel;
44
45
46
47    };
48
49    #endif
```

CNN.cpp C++ 复制代码

```cpp
#include "CNN.h"

CNN::CNN()
{

}

std::vector<std::vector<uint8_t>> CNN::GetFeature(std::string feature_file)
{
    FILE* fp = fopen(feature_file.c_str(), "r");
    while (!fp)
    {
        std::cout << "Can not open feature file!" << std::endl;
        exit(-1);
    }
    uint32_t header[4]={};
    unsigned char bytes[4];

    for (int i = 0; i < 4; i++)
    {
        if (std::fread(bytes, sizeof(bytes), 1, fp))
        {
            header[i] = convert_to_little_endian(bytes);
        }
    }
    numPic = header[1];
    numPixel = header[2]*2;
    std::cout << numPixel;
    std::vector < std::vector<uint8_t>> featureMat;
    for (int j = 0; j < numPic; j++)
    {
        std::vector<uint8_t> imageF;
        for (int k = 0; k <numPixel; k++)
        {
            uint8_t element[1];
            if (std::fread(element, sizeof(element),1, fp))
            {
                imageF.push_back(element[0]);
                std::cout << k << std::endl;
            }
        }
        featureMat.push_back(imageF);
    }
    return featureMat;
```

```
45  }
46
47  std::vector<uint8_t> CNN::GetLabel(std::string label_file)
48  {
49      FILE* lp = fopen(label_file.c_str(), "r");
50      while (!lp)
51      {
52          std::cout << "Can not open label file" << std::endl;
53          exit(-1);
54      }
55      uint32_t lheader[2]={};
56      unsigned char lbytes[4];
57      for (int i = 0; i < 2; i++)
58      {
59          if (std::fread(lbytes, sizeof(lbytes), 1, lp))
60          {
61              lheader[i] = convert_to_little_endian(lbytes);
62          }
63      }
64      std::vector<uint8_t> labelData;
65      for (int j = 0; j < lheader[1]; j++)
66      {
67          uint8_t lelement[1];
68          if (std::fread(lelement, sizeof(lelement), 1, lp))
69          {
70              labelData.push_back(lelement[0]);
71          }
72      }
73      //std::cout << static_cast<int>(labelData[2]) << std::endl;
74      return labelData;
75  }
76
77
78  uint32_t CNN::convert_to_little_endian(const unsigned char* bytes)
79  {
80      return(uint32_t)(
81          (bytes[0]<<24)|
82          (bytes[1]<<16)|
83          (bytes[2]<<8)|
84          (bytes[3])
85          );
86  }
87
88  std::vector<std::vector<float>> CNN::filter_inl(int num_f, int num_r,
    int num_c)
89  {
90      std::vector<std::vector<float>> filter_matrix;
91      for (int i = 0; i < num_f; i++)
```

```
92      {
93          std::vector<float> filter_array;
94          for (int j = 0; j < num_r + num_c; j++)
95          {
96              float randW = (-1) + 2 * rand() / float(RAND_MAX);
97              filter_array.push_back(randW);
98              std::cout << filter_array[j] << " ";
99          }
100         filter_matrix.push_back(filter_array);
101     }
102     return filter_matrix;
103 }
104
105 void CNN::conv_layer(Matrix FeatureData, std::vector<std::vector<float>>
    filter)
106 {
107     for (int i = 0; i < numPic; i++)
108     {
109
110     }
111 }
```

2.结果描述

由于原先的Matrix类不方便进行互相关运算，不得已还是用了vector，但不知怎的一直报错。暂时还没搞清楚原因。明天继续。