

贊同117

號
分享

卷積神經網路C++ 從零開始實現



山與水你和我

體系結構小白，影像處理小白

117 人贊同了該文章

目前建構卷積神經網路（CNN）一般直接用Pytorch、Tensorflow 等深度學習框架，很簡單。但如果是手寫反向傳播過程，情況就比多層感知機（MLP）網路複雜多了，因為不只是矩陣相乘。

號▲ 贊同117

號▼

號● 29 則評論

號✈ 分享

號♥ 喜歡

號★ 收藏

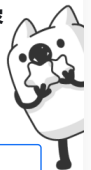
號📄 申請轉載

號...

登入即可查看超5億專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。

立即登入/註冊



- 公式看的實在腦袋疼，不好理解，一大堆的 \sum ，變數還特別多，最後自己實現才發現，卷積的時間複雜度還挺高，好幾層for；
- 卷積層的反向傳播，從輸出回傳的梯度 δ^l ，求輸入的梯度 δ^{l-1} 存在一個權重矩陣rot180 的操作，而且還需要對梯度 δ^l 填滿padding 的問題，請參閱卷積神經網路(CNN)反向傳播演算法- 劉建平Pinard - 部落格園（劉建平老師的部落格強烈推薦），如下

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \delta_{11} & \delta_{12} & 0 \\ 0 & \delta_{21} & \delta_{22} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} * \begin{pmatrix} w_{22} & w_{21} \\ w_{12} & w_{11} \end{pmatrix} = \begin{pmatrix} \nabla a_{11} & \nabla a_{12} & \nabla a_{13} \\ \nabla a_{21} & \nabla a_{22} & \nabla a_{23} \\ \nabla a_{31} & \nabla a_{32} & \nabla a_{33} \end{pmatrix}$$

例子

這個例子沒有錯，但只是一種特殊情況，如果步長stride 大於1，就不僅僅是外圍填充0 了，還需要對 δ^l 資料之間也做填充0，具體過程和轉置卷積的前向過程一模一樣。但即使真的可以這樣做，寫對了，做padding 消耗也是很大的，假設stride = 2，則大約有 $\frac{3}{4}$ 的計算都是跟0 做乘法，是無意義的計算，個人以為不可取，個人改用了其他想法。

後續一步步實現，遇到的也不僅僅這兩個問題，一一克服，整個過程的關鍵就是 - - **硬著頭皮，老實畫圖，別看公式（因人而異）**。

整個過程中，又回顧了一些C++ 的坑與優化技巧，對卷積的前向以及後向過程和如何搭建深度學習流程有了更清晰的認識，收穫頗豐。

提綱

[山與水你和我：卷積神經網路（一）tensor 定義](#)

[山與水你和我：卷積神經網路（二）從影像到tensor](#)

[山與水你與我：卷積神經網路（三）ReLU 層](#)

[山與水你和我：卷積神經網路（四）池化層](#)

[山與水你和我：卷積神經網路（五）卷積層](#)

[山與水你和我：卷積神經網路（六）全連接層](#)

登入即可查看**超5億**專業優質內容


超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



山與水你和我：卷積神經網路（八）訓練CNN

程式碼

<https://github.com/hermosayhl/CNN>

號  github.com/hermosayhl/CNN

環境

1. 視窗11
2. \geq C++17 (TDM GCC 10.3.0)
3. OpenCV 4.5.2
4. 建置工具Cmake

數據集

採用的小型影像分類資料集，從cat-dog-panda資料集剔除cat（cat和dog相對較難），然後從CUB-200 bird資料集中隨機抽出1000張鳥類影像，湊成三分類的小型數據集。train : valid : test 比例8:1:1。

網路模型

本人也不知道是什麼網路結構，隨便設計的（能跑就行），只有捲積層、最大池化層、ReLU層、Softmax層、Linear全連接層，比AlexNet要簡單，接受的輸入大小是224x224x3，輸出3個值，經過softmax得到機率，損失函數是交叉熵，最佳化方法是SGD隨機梯度下降，最終在測試集上大概可以達到0.91的準確率，不高，但至少跑通了。

登入即可查看**超5億**專業優質內容

超5千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



知乎

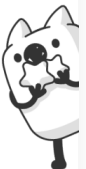
首發於
電腦視覺基礎

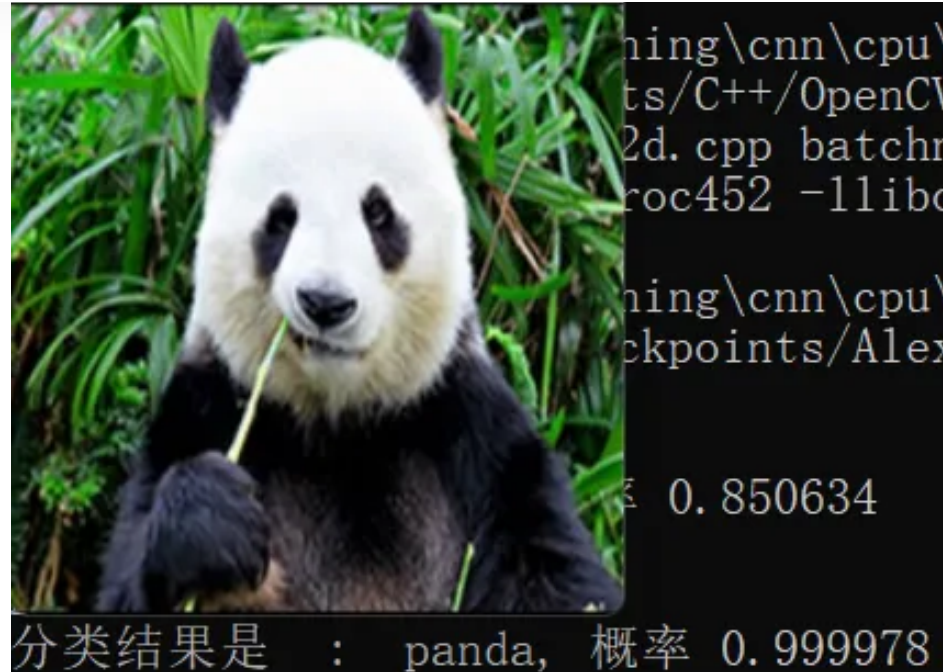
切换模式



登入即可查看超5億專業優質內容

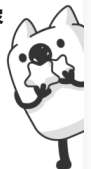
超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。





登入即可查看[超5億](#)專業優質內容

超5 千萬創作者的優質提問、專業回
答、深度文章和精彩影片盡在知乎。



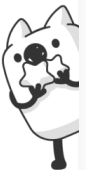


後面雖然也寫了BatchNorm 層、DropOut 層，訓練是沒問題的，這兩個的前向和反向傳播都對，但valid 和test 階段，過度擬合了。。。按照網路上許多說法嘗試，但都失敗了，遺留問題。

後面也嘗試了Grad-CAM 視覺化神經網絡，實現跟論文裡的細節些許不一樣，例子如下，分類為bird

登入即可查看**超5億**專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。





參考

編輯於2023-02-08 21:40 · IP 屬地上海

卷積神經網路 (CNN) C++ 影像分類

寫下你的評論...

29 則評論

預設 最新



阿夫戈

題目太大了😓，以為真是從零開始，你這是從99開始啊😂

2022-02-24

號 回覆 號 7



山與水你和我 作者



2022-02-24

號 回覆 號 喜歡



小布丁

你好想請問下這個熱力圖怎麼畫點

2022-03-12

號 回覆 號 喜歡



哈哈

哈哈 哈哈 哈哈 哈哈 哈哈 哈哈 哈哈 哈哈 哈哈 哈哈

登入即可查看超5億專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



2023-11-10

號● 回覆 號♥ 喜歡

 流風曉

訓練了一晚上，勉強能夠二分類，感覺不穩定/

2023-10-08

號● 回覆 號♥ 喜歡

 山與水你和我 作者

🤔我這個太過時了，建議選擇更好的程式碼🤔，例如kuiperinfer

2023-11-10

號● 回覆 號♥ 喜歡

 流風曉

目前最大的問題還是那個，我想把它改成cpu和gpu同時能協同訓練的，特別是多線程，目前手上的是i9 12900h 20線程的玩意，想讓他加速處理一下。

2023-10-08

號● 回覆 號♥ 喜歡

 流風曉

改了一下，用來做圖片方向分類，效果不錯。🤖二分類打算做成四分類

2023-10-08

號● 回覆 號♥ 喜歡

 流風曉

大佬，我把你的程式加了一個多線程在inference.cpp 這裡，遇到一個全域變數No_grad 導致運算出錯，有什麼辦法可以把這個全域變數轉換為單執行緒自身使用的嗎？

2023-09-12

號● 回覆 號♥ 喜歡

 山與水你和我 作者

std::thread 接受std::function 包裝的函數，例如lambda 中用“取值”

```
自動線程0 = std::thread([No_grad](){  
    // 做一點事  
});
```

這樣取得的No_grad 就是線程本身使用的，如果需要多個線程同步，就需要考慮很多了，比如改成[&No_grad] 獲取變量，而且No_grad 要聲明為volatile 防止被優化到各線程可見的寄存器中，還要考慮讀寫競爭🤖

2023-09-12

號● 回覆 號♥ 喜歡

 流風曉

大佬我用mingw成功的編譯出來了，使用vs編譯的時候，bug超出1000個🤖請教一下怎麼用vs編譯。

2023-09-09

號● 回覆 號♥ 喜歡

 流風曉 ▸ 山與水你和我

非常感謝，我用的是線程池加速用的是用paddleocr中提取的線程池，非常好用最近學

▲

×

登入即可查看超5億專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。




```
#include <vector>
#include <queue>
#include <memory>
#include <thread>
#include <mutex>
#include <condition_variable>
#include <future>
#include <functional>
#include <stdexcept>

class ThreadPool {
public:
    ThreadPool(size_t);
    template<class F, class... Args>
    auto enqueue(F&& f, Args&&... args)
    -> std::future<typename std::result_of<F(Args...)>::type>;
    ~ThreadPool();
private:
    // need to keep track of threads so we can join them
    std::vector< std::thread > workers;
    // the task queue
    std::queue< std::function<void()> > tasks;

    // synchronization
    std::mutex queue_mutex;
    std::condition_variable condition;
    bool stop;
};

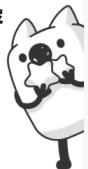
// the constructor just launches some amount of workers
inline ThreadPool::ThreadPool(size_t threads)
: stop(false)
{
    for (size_t i = 0; i < threads; ++i)
        workers.emplace_back(
            [this]
            {
                for (;;)
                {
                    std::function<void()> task;

                    {
                        std::unique_lock<std::mutex> lock(this->queue_mutex);
                        this->condition.wait(lock,
```



登入即可查看**超5億**專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



```
    this->tasks.pop(),
    }

    task();
    }
    };
    }

    // add new work item to the pool
    template<class F, class... Args>
    auto ThreadPool::enqueue(F&& f, Args&&... args)
    -> std::future<typename std::result_of<F(Args...)>::type>
    {
        using return_type = typename std::result_of<F(Args...)>::type;

        auto task = std::make_shared< std::packaged_task<return_type> > (> (
            std::bind(std::forward<F>(f), std::forward<Args>(args)...)
        ));

        std::future<return_type> res = task->get_future();
        {
            std::unique_lock<std::mutex> lock(queue_mutex);

            // don't allow enqueueing after stopping the pool
            if (stop)
                throw std::runtime_error("enqueue on stopped ThreadPool");

            tasks.emplace([task]() { (*task)(); });
        }
        condition.notify_one();
        return res;
    }

    // the destructor joins all threads
    inline ThreadPool::~~ThreadPool()
    {
        {
            std::unique_lock<std::mutex> lock(queue_mutex);
            stop = true;
        }
        condition.notify_all();
        for (std::thread& worker : workers)
            worker.join();
    }
}
```



登入即可查看**超5億**專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



```
/*

// #include "pool_number.cpp"
#include <thread>
#include <iostream>
#include <chrono>
#include <vector>
using namespace std;

int cpu_number() {
// SYSTEM_INFO sysInfo;
// GetSystemInfo(&sysInfo);
// unsigned int numCores1 = sysInfo.dwNumberOfProcessors;
// return numCores1;

unsigned int numCores = std::thread::hardware_concurrency();
return numCores;
}

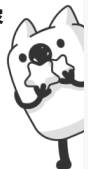
int main(){
cv::waitKey(0);
system("chcp 65001>nul");
auto start = std::chrono::high_resolution_clock::now();
cout << "cpu_number " << cpu_number() << endl;
int pool_size = cpu_number();
//int task_num = 1000;
ThreadPool threadpool(pool_size);
// vector<future<int>> resVec;
vector<future<int>> resVec;

for(int main_number =0;main_number<=1;main_number +=1){
resVec.emplace_back(
threadpool.enqueue(
// 参数要对应，写两次。
[main_number] {return dd(main_number); }
)
);
//main_number = main_number + 1;
}
}
```



登入即可查看[超5億](#)專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



```
},

for (int i = 0; i <= 100; i++) {
    dd(i);
}

// 获取结束时间点
auto end = std::chrono::high_resolution_clock::now();
// 计算代码执行时间 (以毫秒为单位)
auto duration = std::chrono::duration_cast<std::chrono::milliseconds>(end -
start).count();
// 输出执行时间
std::cout << "usetime:" << duration << " ms" << std::endl;
cout << endl;

system("pause");
}
```

*/

2023-09-29

● 回复 ● 喜欢

 山与水你和我 作者

你好，我没咋用过 VS，可能是文件编码格式有问题🤔😓，或许你需要把所有中文注释删了，然后全部保存成 utf-8 格式 应该就可以了

2023-09-10

● 回复 ● 喜欢

展开其他 1 条回复 >



反内卷算法

👍 跟着大佬学习

2023-02-16

● 回复 ● 喜欢



山与水你和我 作者

我這個還是比較新手，提升不夠高。我推薦一個項目，kuiperinfer，一個很適合入門跟提升的深度學習推理框架🤔😓👍，內容比較系統，能學到東西


2023-02-16

號 ● 回覆 號 ● 喜歡

×

登入即可查看超5億專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。



2022年10月21日

號 回覆 號 喜歡



山與水你和我 作者

當初本人很菜，沒考慮到這一點，看大家評論，後續有計劃重構一遍

2022年10月21日

號 回覆 號 喜歡

寫下你的評論...

文章被以下專欄收錄



電腦視覺基礎

最簡單的捲積神經網路、ReLU、BN、目標偵測等



軟體架構



帶你學習AI·深度學習應用之路

創作內容基於深度學習的理論學習與應用開發技術分享

推薦閱讀

卷積神經網路C++ 從零開始實現

作者 | 山與水你和我@知乎山與水你和我：卷積神經網路C++ 從零開始實現目前搭建卷積神經網路（CNN）一般直接用Pytorch、Tensorflow 等深度學習框架，很簡單。但如果是手寫反向傳播過程...

極市平台

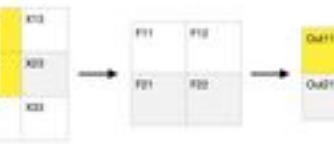
發表於極市平台

C++快速實現2D卷積與池化層

看一些面試會要求手撕一下卷積和池化層，因此花了一小段時間來複習卷積神經網路，另外，為了保證的快速性，盡快實現功能，本文並沒有採用類等方法，用可讀性高的方法來寫，並且對卷積的輸...

蛋總的快樂...

發表於機器學習



C++卷積計算的實作（三種模式）

張一極

十大經典排序演算法 - 時間複雜度、空間複雜度與穩定性(附...

簡潔明了，直接上圖，文末為各排序演算法及改進傳送門。交換排序：AaronWang：C++交換排序：冒泡排序 (BubbleSort)AaronWang：C++交換排序

忘記...

登入即可查看超5億專業優質內容

超5 千萬創作者的優質提問、專業回答、深度文章和精彩影片盡在知乎。

