卷積神經網路（七）建構CNN 網路結構

山與水你和我
體系結構小白，影像處理小白

前面

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

山與水你與我：卷積神經網路（六）Linear 線性層

分別完成了ReLU 層、MaxPool 層、Conv 卷積層、Linear 線性層的前向與反向傳播，現在開始建構CNN。

搭建

本人實現利用了C++ 的繼承，上面的網絡層都繼承自一個基礎的Layer 類，基本函數都有forward, backward，個別類如Conv、Linear 層需要更新參數有update_gradients 函數；每個層都有一個output變數儲存輸出，如下：

```
class Layer {
public:
    const std::string name;  // 这一层的名字
    std::vector<tensor> output;  // 输出的张量
public:
    Layer(std::string& _name) : name(std::move(_name)) {}

    virtual std::vector<tensor> forward(const std::vector<tensor>& input) = 0; // 前向
    virtual std::vector<tensor> backward(std::vector<tensor>& delta) = 0;      // 反向

    virtual void update_gradients(const data_type learning_rate=1e-4) {}       // 更新
```

```
    virtual std::vector<tensor> get_output() const { return this->get_output(); } // 
};
```

如此一來，在CNN 類別中，只要維護一個指標列表（暫時只考慮串列的CNN 結構），forward 時逐一傳遞，backward 從後往前傳遞，如下：

```
// 胡乱写的一个能跑的 CNN 网络结构，不是真的 AlexNet
class AlexNet {
private:
    std::list< std::shared_ptr<Layer> > layers_sequence;
};
```

（註：這不是真的AlexNet，只是個人隨便取的一個名字，跟AlexNet 沒啥關係，二者都很簡單）

初始化網路結構

```
AlexNet::AlexNet(const int num_classes=3)   {
    // batch_size X 3 X 224 X 224
    this->layers_sequence.emplace_back(new Conv2D("conv_layer_1", 3, 16, 3));
    this->layers_sequence.emplace_back(new ReLU("relu_layer_1"));
    // batch_size X 16 X 111 X 111
    this->layers_sequence.emplace_back(new MaxPool2D("max_pool_1", 2, 2));
    // batch_size X 16 X 55 X 55
    this->layers_sequence.emplace_back(new Conv2D("conv_layer_2", 16, 32, 3));
    this->layers_sequence.emplace_back(new ReLU("relu_layer_2"));
    // batch_size X 32 X 27 X 27
    this->layers_sequence.emplace_back(new Conv2D("conv_layer_3", 32, 64, 3));
    this->layers_sequence.emplace_back(new ReLU("relu_layer_3"));
    // batch_size X 64 X 13 X 13
    this->layers_sequence.emplace_back(new Conv2D("conv_layer_4", 64, 128, 3));
    this->layers_sequence.emplace_back(new ReLU("relu_layer_4"));
    // batch_size X 128 X 6 X 6
    this->layers_sequence.emplace_back(new LinearLayer("linear_1", 6 * 6 * 128, num_cl
    // batch_size X num_classes
}
```

有點類似Pytorch 的nn.Sequence，不過是低配版。

```
std::vector<tensor> AlexNet::forward(const std::vector<tensor>& input) {
    // 对输入的形状做检查
    assert(input.size() > 0);
    std::vector<tensor> output(input);
    for(const auto& layer : this->layers_sequence)
        output = layer->forward(output);
    return output;
}
```

backward

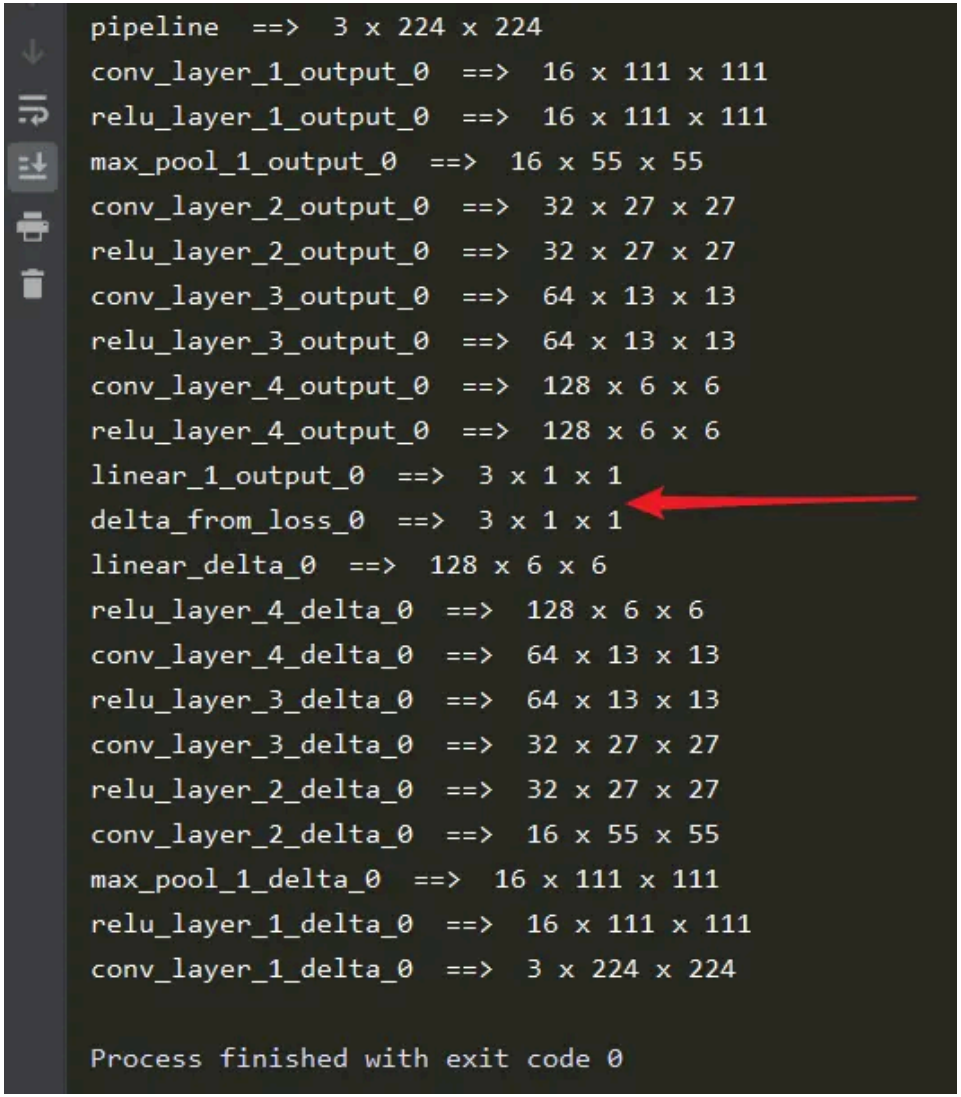
```
void AlexNet::backward(std::vector<tensor>& delta_start) {
    for(auto layer = layers_sequence.rbegin(); layer != layers_sequence.rend(); ++laye
        delta_start = (*layer)->backward(delta_start);
}
```

十分簡單。

驗證

給batch_size = 8，每張圖片尺寸224 * 224，輸出3 分類的結果，

```
using namespace architectures;
// 网络
AlexNet network(3, false);
network.print_info = true;
// 定义输入
std::vector<tensor> input;
const int batch_size = 8;
for(int i = 0;i < batch_size; ++i)
    input.emplace_back(new Tensor3D(3, 224, 224));
// forward
auto output = network.forward(input);
// 定义梯度
std::vector<tensor> delta;
for(int i = 0;i < batch_size; ++i)
    delta.emplace_back(new Tensor3D(3, 1, 1));
// backward
```



紅箭頭以上是forward, 以下是backward，張量的變化過程。

程序正常退出，說明沒有出現越界訪問等問題，可行性較高。

下一步，開始跑整個流程，設計損失函數、最佳化器、模型儲存等操作。

卷積神經網路（CNN） C++ 影像分類

寫下你的評論...

4 則評論 預設 最新

狡詐
等你的訓練程序哦，博主😄
2022-02-19 號 回覆 號 喜歡

狡詐 ▸ **山與水你和我**
好的，謝謝啦博主，寫的很好，支持你🤭
2022-02-21 號 回覆 號 喜歡

山與水你和我 作者
訓練過程已經上線😏
2022-02-19 號 回覆 號 喜歡

展開其他1 則回復號 》

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

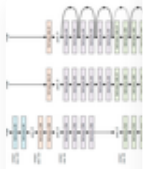
推薦閱讀

CNN網路的層級結構及其相關作用

CNN（卷積神經網路）-優化指

CNN經典結構解析（2）- ResNet，DenseNet

前言結構解析1中著重分析了VGG和Inception網路的設計思想，這兩個網路都主要在解決同一個問題——應該選擇多大的捲積核。如上一篇



【論文解讀+ 積神經網路-...