# 模仿Caffe层的注册

Caffe中使用一个静态成员向量来保存层的名称，十分巧妙。在自己实现的CNN前向传播框架中对其进行了模仿，实现原理如下：

1. 定义一个LayerRegistry类，其中维护一个静态成员变量pVectorLayer，使用静态成员的好处是，无论定义多少个该类的实例化对象，这些对象贡献同一个静态成员。该类定义如下：

|  |
| --- |
| #ifndef LAYER\_REGISTRY\_H  #define LAYER\_REGISTRY\_H  #include <vector>  class LayerRegistry  {  public:  static std::vector<std::string> \*pVecLayer;  LayerRegistry(std::string type);  ~LayerRegistry();    };  #endif  #include "LayerRegistry.h"  LayerRegistry::LayerRegistry(std::string type)  {  if (pVecLayer == 0)  {  pVecLayer = new std::vector<std::string>(0);  }  pVecLayer->push\_back(type);  }  LayerRegistry::~LayerRegistry()  {  } |

在该类的构造函数中，将输入参数即类型，保存在pVecLayer中。

1. 定义一个注册宏，如下：

#define REGISTRY\_LAYER(type) LayerRegistry type##Reg(#type);

其中##表示连接两个参数，#表示字符串，所以如果有REGISTRY\_LAYER(Convolution)，则实际上LayerRegistry ConvolutionReg(Convolution);即定义了一个名为ConvolutionReg的LayerRegistry类型对象，并传入字符串参数Convolution，因此会调用LayerRegistry构造函数，将字符串Convolution保存到静态成员变量pVecLayer中。

1. 定义并初始化pVecLayer。这一步的原理是什么，我也不明白，为什么要单独对其进行定义。

vector<string>\* LayerRegistry::pVecLayer = 0;

1. 主函数：

|  |
| --- |
| #include "Net.h"  #include <string>  #include <opencv2\opencv.hpp>  #include "LayerRegistry.h"  #include <time.h>  using namespace std;  using namespace cv;  vector<string>\* LayerRegistry::pVecLayer = 0;  void main()  {  for (size\_t i = 0; i < LayerRegistry::pVecLayer->size(); i++)  {  cout << LayerRegistry::pVecLayer->at(i) << endl;  }  system("pause");  } |

