### **Yunming Zhang's Blog**

### How to create your own layer in deep learning framework CAFFE

Posted on January 19, 2015

This is a post outlining the steps you need to take to create your own layer in CAFFE, a popular framework for writing convolutional neural networks. The post focuses on the latest version of CAFE as of Jan 2015.

#### NOTES:

■ I use \$CAFFEROOT and /caffe/ interchangeably, this is simply the root directory of the caffe installation.

A general and slightly outdated tutorial can be found here (<a href="https://github.com/BVLC/caffe/wiki/Development">https://github.com/BVLC/caffe/wiki/Development</a>).

- 1. Create the definition a new class in one of the .hpp files located in \$CAFFEROOT/include/caffe/. In my case, I am writing a customized convolution layer. As a result, I modify the vision\_layers.hpp to add a definition of myConvLayer.
- 2. Next we need to create a myConvLayer.cpp file in the following path \$CAFFEROOT/src/caffe /myConvLayer.cpp .
  - A. Implement the virtual methods required of the class. In my case, I needed to implement the required "LayerSetUp", "Reshape", "ForwardCPU" and "BackwardCPU".
- 3. Choose a name for your layer and write it in caffe/src/caffe/proto/caffe.proto
  - A. Find a message called LayerParameter
  - B. Find the latest unoccupied number, there should be a comment above the message declaration saying "the next available ID when you create....", use the smallestUnoccupiedNumber
  - C. Add your layer to the LayerType enum, for example, I add "MYCONVOLUTIONLAYER = 38"
- 4. If you have completed step 1 and step 2, you should be able to just compile the entire CAFFE directory fine. The next steps will require you to actually write a network and run it to get the protobuf set up right. To do this, I recommend simply use an existing network. I chose MNIST LENET and replaced the convolution layer in (/caffe/examples/mnist\_modified/lenet\_train\_test.protxt). To get it to run with my own set up, I
  - A. Created a modified directory(mint\_modified) that copies minist in examples.
  - B. Change the lent\_solver.prototxt to use my own file. CAFFE used hardcoded path, when it should have used relative path to find the training configuration set up.
    - i. TYPE = MYCONVOLUTION (depending on your declaration in the protobuf file in the previous step)
  - C. Once you are done getting your own layer running within an existing network, it will crash immediately because we haven't worked on getting
- 5. Dealing with protobuf and layer\_factory
  - A. Now you should see an error massage saying "unknown type 38" from a file called "layer\_factory.cpp". You can find the file at caffe/src/caffe/layer\_factory.cpp. The error message tells you that currently CAFFE cannot recognize the parameter specified in the protobuf file "myconvolution". To do this, you simply need to add a new case statement at the end of "layer\_factory.cpp" file.
  - B. case LayerParameter LayerType MYCONVOLUTION:return new MyConvolutionLayer<Dtype>(param);
  - C. This statement tells the system to use layer\_factory to create the class whenever that parameter is being read
- 6. The last part is to make sure that your own layers is using the parameters that you want.
  - A. An interesting note here is that the parameter has no fixed paring. The way it is specified in protobuf file for example, is nesting a parameter type within a layer type. As a result, you can switch the parameters among

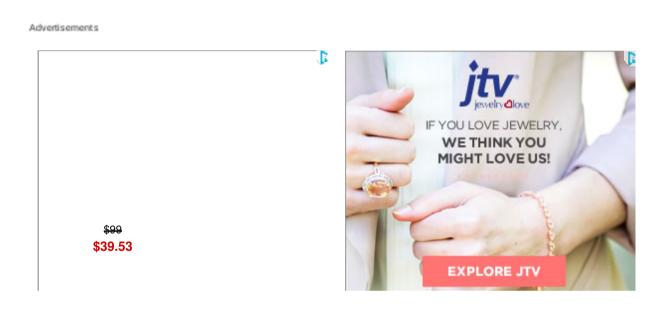
1 of 3 2017年07月12日 09:02

different layers. That is I could have used the original convolution\_parameter inside MyConvolution layer.

- B. To use my own set of parameter, I went back to caffe/src/caffe/proto/caffe.proto file
  - i. Define a new parameter type
    - a. "myconvolution\_param = 42"
  - ii. Define a new message class
    - a. MyConvolutionPoolingParameter { optional uint32 num\_output =1; .... }
  - iii. Then go back to the lenet\_train\_test.protxt file and set your layer to use the my convolution parameter in myconvolutionpooling layer.
    - a. An example layers { name:'myconv1´ type:MYCONVOLUTION; ..... myconvolution\_param { num\_output: 20 kernel\_size ....}}
  - iv. One last step is modifying the \$CAFFEROOT/src/caffe/myConvLayer.cpp to use the my
    convolution\_param. You can access the parameters by coding "ConvolutionPoolingParameter
    convpool\_param = this->layer\_param\_.convolutionpooling\_param();"

Now, you have fully functional customized layer with its own parameters! This tutorial applies not only to some modified convolution layer but can be used for any kind of new layer.

### Cheers!



#### Share this:



# Related

Meeting Notes with Aditya on Convolutional Neural Nets with CAFE, Dec 22nd In "Convoluted Neural Nets" How to instrument CAFFE to get timing on each layer

CAFFE notes: how to disable back propagation for certain layers in CAFFE In "Convoluted Neural Nets"

ral Nets" In "Convoluted Neural Nets"

This entry was posted in **Convoluted Neural Nets**. Bookmark the <u>permalink</u>.

## 3 Responses to How to create your own layer in deep learning framework CAFFE



Nathiyaa Sengodan says:

March 10, 2016 at 10:34 am

Very useful Post!

Reply

2 of 3 2017年07月12日 09:02



# Nathiyaa Sengodan says:

March 10, 2016 at 10:36 am

Good one!

Reply



**kusemanohar** says:

June 22, 2016 at 4:51 pm

Nice post...will try it...!

Reply

### **Yunming Zhang's Blog**

Create a free website or blog at WordPress.com.

3 of 3 2017年07月12日 09:02