Comparison between caffe-windows and caffe-crfasrnn-gpu-master

* include
  + caffe
    - layers
      * **RORPH: crfasrnn\_multi\_stage\_meanfield.hpp**
      * **RORPH: crfasrnn\_multi\_stage\_meanfield.hpp\_bk**
    - util
      * **RORPH: crfasrnn\_hash\_table.hpp**
      * **RORPH: crfasrnn\_modified\_permutohedral.hpp**
      * **RORPH: crfasrnn\_modified\_permutohedral.hpp\_bk**
      * **RORPH: crfasrnn\_tvg\_common\_utils.hpp**
      * **RORPH: crfasrnn\_tvg\_util.hpp**
      * DIFF: cudnn.hpp
      * DIFF: io.hpp
    - DIFF: common.hpp
    - DIFF: layer\_factory.hpp
    - DIFF: solver\_factory.hpp
* **RORPH: examples**
* src
  + caffe
    - layers
      * DIFF: bnll\_layer.cu
      * **RORPH: crfasrnn\_meanfield\_iteration.cpp**
      * **RORPH: crfasrnn\_meanfield\_iteration.cu**
      * **RORPH: crfasrnn\_multi\_stage\_meanfield.cpp**
      * **RORPH: crfasrnn\_multi\_stage\_meanfield.cpp\_bk**
      * **RORPH: crfasrnn\_multi\_stage\_meanfield.cu**
    - proto
      * DIFF: caffe.proto
    - test
      * DIFF: <all files>
    - util
      * DIFF: db\_lmdb.cpp
      * DIFF: io.cpp
      * DIFF: signal\_handler.cpp
      * **RORPH: crfasrnn\_hash\_helper.cu**
      * **RORPH: crfasrnn\_modified\_permutohedral.cpp**
      * **RORPH: crfasrnn\_modified\_permutohedral.cpp\_bk**
      * **RORPH: crfasrnn\_modified\_permutohedral.cu**
    - DIFF: CMakeLists.txt
    - DIFF: common.cpp
    - DIFF: layer\_factory.cpp
    - LORPH: solver\_factory.cpp

Steps to create caffe-crfasrnn-windows

==Modify Caffe source code==

1. Copy caffe-windows, name it caffe-crfasrnn-windows
2. Copy the following files from caffe-crfasrnn-gpu-master into caffe-crfasrnn-windows
   1. include/caffe/layers [these are the layers’ class declaration files]
      1. crfasrnn\_multi\_stage\_meanfield.hpp
      2. crfasrnn\_multi\_stage\_meanfield.hpp\_bk
   2. include/caffe/util [these are the layers’ utility function declaration files]
      1. crfasrnn\_hash\_table.hpp
      2. crfasrnn\_modified\_permutohedral.hpp
      3. crfasrnn\_modified\_permutohedral.hpp\_bk
      4. crfasrnn\_tvg\_common\_utils.hpp
      5. crfasrnn\_tvg\_util.hpp
   3. examples\_crfasrnn/\* [these are CRFasRNN-specific example scripts and data]
   4. src/caffe/layers [these are the layers’ implementation files]
      1. crfasrnn\_meanfield\_iteration.cpp
      2. crfasrnn\_meanfield\_iteration.cu
      3. crfasrnn\_multi\_stage\_meanfield.cpp
      4. crfasrnn\_multi\_stage\_meanfield.cpp\_bk
      5. crfasrnn\_multi\_stage\_meanfield.cu
   5. src/caffe/util [these are the layers’ utility function implementation files]
      1. crfasrnn\_hash\_helper.cu
      2. crfasrnn\_modified\_permutohedral.cpp
      3. crfasrnn\_modified\_permutohedral.cpp\_bk
      4. crfasrnn\_modified\_permutohedral.cu
3. Copy the following folder from crfasrnn into caffe-crfasrnn-windows
   1. python-scripts
4. Modify the following file in caffe-crfasrnn-windows with that from caffe-crfasrnn-gpu-master
   1. src/caffe/proto [this holds the parameter declaration file]
      1. caffe.proto
         1. Immediately below this line

|  |
| --- |
| optional SolverType solver\_type = 30 [default = SGD]; |

Add these lines

|  |
| --- |
| // Overlap compute and communication for data parallel training  optional bool layer\_wise\_reduce = 41 [default = true]; |

* + - 1. Immediately below this line

|  |
| --- |
| optional WindowDataParameter window\_data\_param = 129; |

Add this line

|  |
| --- |
| optional MultiStageMeanfieldParameter multi\_stage\_meanfield\_param = 147; |

* + - 1. Immediately below these lines

|  |
| --- |
| optional bool normalize = 2;  } |

Add these lines

|  |
| --- |
| // Message that stores parameters used by MultiStageMeanfieldLayer  message MultiStageMeanfieldParameter {  enum Mode {  POTTS = 0;  }  optional Mode compatibility\_mode = 1 [default = POTTS];  optional float threshold = 2;  required float theta\_alpha = 3 [default = 10.];  required float theta\_beta = 4 [default = 10.];  required float theta\_gamma = 5 [default = 10.];  required uint32 num\_iterations = 6 [default = 1];  optional float spatial\_filter\_weight = 7 [default = 1];  optional float bilateral\_filter\_weight = 8 [default = 1];  optional float forced\_spatial\_filter\_weight = 9;  optional float forced\_bilateral\_filter\_weight = 10;  } |

* + - 1. Immediately below these lines

|  |
| --- |
| message InfogainLossParameter {  // Specify the infogain matrix source.  optional string source = 1; |

Add this line

|  |
| --- |
| optional int32 axis = 2 [default = 1]; // axis of prob |

* + - 1. Remove the lines

|  |
| --- |
| optional NormalizeParameter norm\_param = 149; |
| optional RandCatConvParameter rand\_cat\_conv\_param = 147;  optional RandCatParameter rand\_cat\_param = 148; |
| // Message that stores parameters used by NormalizeLayer  message NormalizeParameter {  optional bool across\_spatial = 1 [default = true];  // Initial value of scale. Default is 1.0 for all  optional FillerParameter scale\_filler = 2;  // Whether or not scale parameters are shared across channels.  optional bool channel\_shared = 3 [default = true];  // Whether we learn the scale parameter or fix it  optional bool fix\_scale = 4 [default = true];  // Epsilon for not dividing by zero while normalizing variance  optional float eps = 5 [default = 1e-10];  } |
| // added by ab --  message RandCatParameter {  // for rand\_cat\_param --  // Specify if we should randomly sample or not  optional bool rand\_selection = 1 [default = true];  // Specify no. of outputs randomy sampled from the data --  optional uint32 num\_output = 2 [default = 1000];  }  message RandCatConvParameter {  // for rand\_cat\_param --  // Specify if we should randomly sample or not  optional bool rand\_selection = 1 [default = true];  // Specify no. of outputs randomy sampled from the data --  optional uint32 num\_output = 2 [default = 1000];  //  repeated int32 pooling\_factor = 3;  // specify the amount of padding done during the test time  // this is to avoid redundant computation --  optional uint32 pad\_factor = 4 [default = 0];  } |
| NORMALIZE = 42; |
| RAND\_CAT\_CONV = 41;  RAND\_CAT = 40; |
| optional NormalizeParameter norm\_param = 45; |
| optional RandCatConvParameter rand\_cat\_conv\_param = 44;  optional RandCatParameter rand\_cat\_param = 43; |



==Modify CRFasRNN source code==

1. In the file caffe-crfasrnn-windows/include/caffe/util/crfasrnn\_tvg\_util.hpp, make the following changes:

|  |  |
| --- | --- |
| template void PrintBlob(const Blob<float>\* blob, bool print\_diff = false, const char\* info = 0);  template void PrintBlob(const Blob<double>\* blob, bool print\_diff = false, const char\* info = 0); | template void PrintBlob(const Blob<float>\* blob, bool print\_diff, const char\* info);  template void PrintBlob(const Blob<double>\* blob, bool print\_diff, const char\* info); |
| template void FillWithMax(Blob<float>\* const blob, float max\_value = 1);  template void FillWithMax(Blob<double>\* const blob, float max\_value = 1); | template void FillWithMax(Blob<float>\* const blob, float max\_value);  template void FillWithMax(Blob<double>\* const blob, float max\_value); |

1. In the file caffe-crfasrnn-windows/src/caffe/layers/crfasrnn\_multi\_stage\_meanfield.cpp, make the following changes:

|  |  |
| --- | --- |
| float spatial\_kernel[2 \* num\_pixels\_]; | float \*spatial\_kernel = new float[2 \* num\_pixels\_]; |

1. In the file caffe-crfasrnn-windows/src/caffe/util/crfasrnn\_modified\_permutohedral.cpp, make the following changes:

|  |  |
| --- | --- |
| const float \* f = (feature + k \* num\_dimensions); | const float \* f = (features + k \* num\_dimensions); |

1. For only CPU support, comment out all references to Forward\_gpu and Backward\_gpu in caffe-crfasrnn-windows/src/caffe/layers/crfasrnn\_meanfield\_iteration.cpp, caffe-crfasrnn-windows/src/caffe/layers/crfasrnn\_multi\_stage\_meanfield.cpp, then delete crfasrnn\_meanfield\_iteration.cu and crfasrnn\_multi\_stage\_meanfield.cu
2. For GPU support, in the file caffe-crfasrnn-windows/cmake/Cuda.cmake, immediately after the line

|  |
| --- |
| find\_package(CUDA 5.5 QUIET) |

Add the line

|  |
| --- |
| set(CUDA\_CUBLAS\_LIBRARIES "${CUDA\_TOOLKIT\_ROOT\_DIR}/lib/x64/cublas.lib") |

1. For GPU support (since CUDA version is greater than or equal to 7.5), in the file C:\Users\chanlynd\.caffe\dependencies\libraries\_v140\_x64\_py35\_1.1.0\libraries\include\boost-1\_61\boost\config\compiler\nvcc.hpp, comment out lines 22-24

==Modify Caffe compilation code==

1. Add the following lines to the top of the file in caffe-crfasrnn-windows
   1. scripts/build\_win.cmd

|  |
| --- |
| rmdir /s /q build  set CMAKE\_C\_COMPILER=C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\bin\cl.exe  set CMAKE\_CXX\_COMPILER=C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\bin\cl.exe  set CUDNN\_ROOT=C:\Users\Public\cudnn\cudnn-9.0-windows10-x64-v7  set WITH\_NINJA=0  set WITH\_CUDA=1  set BUILD\_MATLAB=0  set BUILD\_PYTHON=1  set PYTHON\_VERSION=3  set CPU\_ONLY=0  set CMAKE\_CONFIG=Release  set USE\_CUDNN=1 |

1. In the file caffe-crfasrnn-windows/cmake/Cuda.cmake, immediately above the line

|  |
| --- |
| find\_package(CUDA 5.5 QUIET) |

Add the lines

|  |
| --- |
| set(CUDA\_TOOLKIT\_ROOT\_DIR "C:/Users/Public/CUDA/v8.0")  set(CUDA\_NVCC\_EXECUTABLE "${CUDA\_TOOLKIT\_ROOT\_DIR}/bin/nvcc.exe") |

1. Run the following lines in CMD

|  |
| --- |
| cd caffe-crfasrnn-windows  scripts\build\_win.cmd |

==Test==

1. Copy the folder caffe-crfasrnn-windows/python/caffe into C:\User\_Program\_Files\Python35\Lib\site-packages\caffe
2. Convert crfasrnn\_demo.py into Python 3.5 with the following steps:
   1. Replace “for j in xrange(0, n):” with “for j in range(n):”
3. Edit the demo script caffe-crfasrnn-windows/python-scripts/crfasrnn\_demo.py, by making the following changes:

|  |  |
| --- | --- |
| gpu\_device = -1 # Use -1 to run only on the CPU, use 0-3[7] to run on the GPU | gpu\_device = 0 # Use -1 to run only on the CPU, use 0-3[7] to run on the GPU |
| print("Time taken to run the network: {:.4f} seconds".format(end\_time - start\_time)) | print("Time taken to run the network: {:0.4f} seconds".format(end\_time - start\_time)) |

1. Run the following lines in CMD

|  |
| --- |
| cd caffe-crfasrnn-windows\python-scripts  download\_trained\_model.ps1  python crfasrnn\_demo.py |