

SQA Automation Setup

The project contains a number of automated testing and quality assurance measures which are explained in this document.

CI

The repository is setup to run an automated regression test suite whenever a change is pushed, or a pull request is merged. This functionality was implemented using Github actions. The CI workflow can be found in `./.github/workflows/run_tests.yml`. It checks out the latest version of the code, installs dependencies, compiles the code, and then runs all of the tests. Any failing tests get reported as a pipeline failure. An example CI run can be viewed at <https://github.com/sami-hatna66/NeuralNetwork-cpp/actions/runs/12014809253/job/33491450916>. The screenshots below demonstrate the CI pipeline functioning as expected:

```
build-and-test
succeeded on Nov 25, 2024 in 45s

Set up job

1 Current runner version: '2.321.0'
2 ▶ Operating System
6 ▶ Runner Image
11 ▶ Runner Image Provisioner
13 ▶ GITHUB_TOKEN Permissions
17 Secret source: Actions
18 Prepare workflow directory
19 Prepare all required actions
20 Getting action download info
21 Download action repository 'actions/checkout@v2' (SHA:ee0669bd1cc54295c223e0bb
22 Complete job name: build-and-test

Run actions/checkout@v2

1 ▶ Run actions/checkout@v2
12 Syncing repository: sami-hatna66/NeuralNetwork-cpp
13 ▶ Getting Git version info
17 Copying '/Users/runner/.gitconfig' to '/Users/runner/work/_temp/2b830982-8e19-
18 Temporarily overriding HOME='/Users/runner/work/_temp/2b830982-8e19-4653-92dd-
19 Adding repository directory to the temporary git global config as a safe direc
20 /opt/homebrew/bin/git config --global --add safe.directory /Users/runner/work/
21 Deleting the contents of '/Users/runner/work/NeuralNetwork-cpp/NeuralNetwork-c
22 ▶ Initializing the repository
36 ▶ Disabling automatic garbage collection
38 ▶ Setting up auth
44 ▶ Fetching the repository
147 ▶ Determining the checkout info
148 ▶ Checking out the ref
152 /opt/homebrew/bin/git log -1 --format='%H'
153 'e0e9482b03ae691e0d7bcef47704f4665b53ce8a'

Install gtest

1 ▶ Run brew install googletest
4 ==> Downloading https://ghcr.io/v2/homebrew/core/googletest/manifests/1.15.2
5 ==> Fetching googletest
6 ==> Downloading https://ghcr.io/v2/homebrew/core/googletest/blobs/sha256:d6ba5
7 ==> Pouring googletest--1.15.2.arm64_sonoma.bottle.tar.gz
8 🍺 /opt/homebrew/Cellar/googletest/1.15.2: 76 files, 2.3MB
```

```
Configure

1 ▶ Run mkdir build
6 -- The C compiler identification is AppleClang 15.0.0.15000309
7 -- The CXX compiler identification is AppleClang 15.0.0.15000309
8 -- Detecting C compiler ABI info
9 -- Detecting C compiler ABI info - done
10 -- Check for working C compiler: /Applications/Xcode_15.4.app/Contents/Develop
11 -- Detecting C compile features
12 -- Detecting C compile features - done
13 -- Detecting CXX compiler ABI info
14 -- Detecting CXX compiler ABI info - done
15 -- Check for working CXX compiler: /Applications/Xcode_15.4.app/Contents/Devel
16 -- Detecting CXX compile features
17 -- Detecting CXX compile features - done
18 CMake Warning (dev) at /opt/homebrew/share/cmake/Modules/FetchContent.cmake:13
19 The DOWNLOAD_EXTRACT_TIMESTAMP option was not given and policy CMP0135 is
20 not set. The policy's OLD behavior will be used. When using a URL
21 download, the timestamps of extracted files should preferably be that of
22 the time of extraction, otherwise code that depends on the extracted
23 contents might not be rebuilt if the URL changes. The OLD behavior
24 preserves the timestamps from the archive instead, but this is usually not
25 what you want. Update your project to the NEW behavior or specify the
26 DOWNLOAD_EXTRACT_TIMESTAMP option with a value of true to avoid this
27 robustness issue.
28 Call Stack (most recent call first):
29 test/CMakeLists.txt:2 (FetchContent_Declare)
30 This warning is for project developers. Use -Wno-dev to suppress it.
31
32 -- Performing Test CMAKE_HAVE_LIBC_PTHREAD
33 -- Performing Test CMAKE_HAVE_LIBC_PTHREAD - Success
34 -- Found Threads: TRUE
35 -- Configuring done (2.9s)
36 -- Generating done (0.0s)
37 -- Build files have been written to: /Users/runner/work/NeuralNetwork-cpp/Neur

Compile

1 ▶ Run cd build
5 [ 2%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Utils.cpp.o
6 [ 5%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Model.cpp.o
7 [ 8%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Activations.cpp.o
8 [ 11%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Layer.cpp.o
9 [ 13%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Loss.cpp.o
10 [ 16%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/LossActivation.cpp.o
```

```
10 [ 16%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/LossActivation.cpp.o
11 [ 19%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Optimizers.cpp.o
12 [ 22%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/Accuracy.cpp.o
13 [ 25%] Building CXX object nn-cpp/CMakeFiles/nn-cpp.dir/ModelLayer.cpp.o
14 [ 27%] Linking CXX static library libnn-cpp.a
15 [ 27%] Built target nn-cpp
16 [ 30%] Building CXX object _deps/googletest-build/googletest/CMakeFiles/gtest
17 [ 33%] Linking CXX static library ../lib/libgtest.a
18 [ 33%] Built target gtest
19 [ 36%] Building CXX object _deps/googletest-build/googletest/CMakeFiles/gtest
20 [ 38%] Linking CXX static library ../lib/libgtest_main.a
21 [ 38%] Built target gtest_main
22 [ 41%] Building CXX object test/CMakeFiles/basic_test.dir/basic_test.cpp.o
23 [ 44%] Linking CXX executable basic_test
24 [ 44%] Built target basic_test
25 [ 47%] Building CXX object test/CMakeFiles/utils_unit_tests.dir/utils_unit_te
26 [ 50%] Linking CXX executable utils_unit_tests
27 [ 50%] Built target utils_unit_tests
28 [ 52%] Building CXX object test/CMakeFiles/layer_unit_tests.dir/layer_unit_te
29 [ 55%] Linking CXX executable layer_unit_tests
30 [ 55%] Built target layer_unit_tests
31 [ 58%] Building CXX object test/CMakeFiles/utils_integration_tests.dir/utils_
32 [ 61%] Linking CXX executable utils_integration_tests
33 [ 61%] Built target utils_integration_tests
34 [ 63%] Building CXX object test/CMakeFiles/layer_integration_tests.dir/layer_
35 [ 66%] Linking CXX executable layer_integration_tests
36 [ 66%] Built target layer_integration_tests
37 [ 69%] Building CXX object test/CMakeFiles/test_all.dir/test_all.cpp.o
38 [ 72%] Linking CXX executable test_all
39 [ 72%] Built target test_all
40 [ 75%] Building CXX object _deps/googletest-build/googmock/CMakeFiles/gmock
41 [ 77%] Linking CXX static library ../lib/libgmock.a
42 [ 77%] Built target gmock
43 [ 80%] Building CXX object _deps/googletest-build/googmock/CMakeFiles/gmock
44 [ 83%] Linking CXX static library ../lib/libgmock_main.a
45 [ 83%] Built target gmock_main
46 [ 86%] Building CXX object bench/CMakeFiles/utils_bench.dir/utils_bench.cpp.o
47 [ 88%] Linking CXX executable utils_bench
48 [ 88%] Built target utils_bench
49 [ 91%] Building CXX object bench/CMakeFiles/layer_bench.dir/layer_bench.cpp.o
50 [ 94%] Linking CXX executable layer_bench
51 [ 94%] Built target layer_bench
52 [ 97%] Building CXX object bench/CMakeFiles/system_bench.dir/system_bench.cpp.o
53 [100%] Linking CXX executable system_bench
54 [100%] Built target system_bench
```

```
47 [ RUN      ] UtilsUnitTests/0.MeanSmall
48 [ OK       ] UtilsUnitTests/0.MeanSmall (0 ms)
49 [ RUN      ] UtilsUnitTests/0.MeanMed
50 [ OK       ] UtilsUnitTests/0.MeanMed (0 ms)
51 [ RUN      ] UtilsUnitTests/0.MeanBig
52 [ OK       ] UtilsUnitTests/0.MeanBig (0 ms)
53 [-----] 20 tests from UtilsUnitTests/0 (0 ms total)
54
55 [-----] 20 tests from UtilsUnitTests/1, where TypeParam = double
56 [ RUN      ] UtilsUnitTests/1.MatMulSmall
57 [ OK       ] UtilsUnitTests/1.MatMulSmall (0 ms)
58 [ RUN      ] UtilsUnitTests/1.MatMulMed
59 [ OK       ] UtilsUnitTests/1.MatMulMed (0 ms)
60 [ RUN      ] UtilsUnitTests/1.MatMulBig
61 [ OK       ] UtilsUnitTests/1.MatMulBig (0 ms)
62 [ RUN      ] UtilsUnitTests/1.ScalarMulSmall
63 [ OK       ] UtilsUnitTests/1.ScalarMulSmall (0 ms)
64 [ RUN      ] UtilsUnitTests/1.ScalarMulMed
65 [ OK       ] UtilsUnitTests/1.ScalarMulMed (0 ms)
66 [ RUN      ] UtilsUnitTests/1.ScalarMulBig
67 [ OK       ] UtilsUnitTests/1.ScalarMulBig (0 ms)
68 [ RUN      ] UtilsUnitTests/1.MatDivSmall
69 [ OK       ] UtilsUnitTests/1.MatDivSmall (0 ms)
70 [ RUN      ] UtilsUnitTests/1.MatDivMed
71 [ OK       ] UtilsUnitTests/1.MatDivMed (0 ms)
72 [ RUN      ] UtilsUnitTests/1.MatDivBig
73 [ OK       ] UtilsUnitTests/1.MatDivBig (0 ms)
74 [ RUN      ] UtilsUnitTests/1.PowerSmall
75 [ OK       ] UtilsUnitTests/1.PowerSmall (0 ms)
76 [ RUN      ] UtilsUnitTests/1.ScalarDivSmall
77 [ OK       ] UtilsUnitTests/1.ScalarDivSmall (0 ms)
78 [ RUN      ] UtilsUnitTests/1.ScalarSubSmall
79 [ OK       ] UtilsUnitTests/1.ScalarSubSmall (0 ms)
80 [ RUN      ] UtilsUnitTests/1.PowerMed
81 [ OK       ] UtilsUnitTests/1.PowerMed (0 ms)
82 [ RUN      ] UtilsUnitTests/1.PowerBig
83 [ OK       ] UtilsUnitTests/1.PowerBig (0 ms)
84 [ RUN      ] UtilsUnitTests/1.TransposeSmall
85 [ OK       ] UtilsUnitTests/1.TransposeSmall (0 ms)
86 [ RUN      ] UtilsUnitTests/1.TransposeMed
87 [ OK       ] UtilsUnitTests/1.TransposeMed (0 ms)
88 [ RUN      ] UtilsUnitTests/1.TransposeBig
89 [ OK       ] UtilsUnitTests/1.TransposeBig (0 ms)
90 [ RUN      ] UtilsUnitTests/1.MeanSmall
91 [ OK       ] UtilsUnitTests/1.MeanSmall (0 ms)
92 [ RUN      ] UtilsUnitTests/1.MeanMed
```

```
Test
1 ▶ Run cd build
5 [=====] Running 83 tests from 9 test suites.
6 [-----] Global test environment set-up.
7 [-----] 1 test from HelloTest
8 [ RUN      ] HelloTest.BasicAssertions
9 [ OK       ] HelloTest.BasicAssertions (0 ms)
10 [-----] 1 test from HelloTest (0 ms total)
11
12 [-----] 20 tests from UtilsUnitTests/0, where TypeParam = float
13 [ RUN      ] UtilsUnitTests/0.MatMulSmall
14 [ OK       ] UtilsUnitTests/0.MatMulSmall (0 ms)
15 [ RUN      ] UtilsUnitTests/0.MatMulMed
16 [ OK       ] UtilsUnitTests/0.MatMulMed (0 ms)
17 [ RUN      ] UtilsUnitTests/0.MatMulBig
18 [ OK       ] UtilsUnitTests/0.MatMulBig (0 ms)
19 [ RUN      ] UtilsUnitTests/0.ScalarMulSmall
20 [ OK       ] UtilsUnitTests/0.ScalarMulSmall (0 ms)
21 [ RUN      ] UtilsUnitTests/0.ScalarMulMed
22 [ OK       ] UtilsUnitTests/0.ScalarMulMed (0 ms)
23 [ RUN      ] UtilsUnitTests/0.ScalarMulBig
24 [ OK       ] UtilsUnitTests/0.ScalarMulBig (0 ms)
25 [ RUN      ] UtilsUnitTests/0.MatDivSmall
26 [ OK       ] UtilsUnitTests/0.MatDivSmall (0 ms)
27 [ RUN      ] UtilsUnitTests/0.MatDivMed
28 [ OK       ] UtilsUnitTests/0.MatDivMed (0 ms)
29 [ RUN      ] UtilsUnitTests/0.MatDivBig
30 [ OK       ] UtilsUnitTests/0.MatDivBig (0 ms)
31 [ RUN      ] UtilsUnitTests/0.PowerSmall
32 [ OK       ] UtilsUnitTests/0.PowerSmall (0 ms)
33 [ RUN      ] UtilsUnitTests/0.ScalarDivSmall
34 [ OK       ] UtilsUnitTests/0.ScalarDivSmall (0 ms)
35 [ RUN      ] UtilsUnitTests/0.ScalarSubSmall
36 [ OK       ] UtilsUnitTests/0.ScalarSubSmall (0 ms)
37 [ RUN      ] UtilsUnitTests/0.PowerMed
38 [ OK       ] UtilsUnitTests/0.PowerMed (0 ms)
39 [ RUN      ] UtilsUnitTests/0.PowerBig
40 [ OK       ] UtilsUnitTests/0.PowerBig (0 ms)
41 [ RUN      ] UtilsUnitTests/0.TransposeSmall
42 [ OK       ] UtilsUnitTests/0.TransposeSmall (0 ms)
43 [ RUN      ] UtilsUnitTests/0.TransposeMed
44 [ OK       ] UtilsUnitTests/0.TransposeMed (0 ms)
45 [ RUN      ] UtilsUnitTests/0.TransposeBig
46 [ OK       ] UtilsUnitTests/0.TransposeBig (0 ms)
```

```
93 [ OK       ] UtilsUnitTests/1.MeanMed (0 ms)
94 [ RUN      ] UtilsUnitTests/1.MeanBig
95 [ OK       ] UtilsUnitTests/1.MeanBig (0 ms)
96 [-----] 20 tests from UtilsUnitTests/1 (0 ms total)
97
98 [-----] 10 tests from LayerUnitTests/0, where TypeParam = float
99 [ RUN      ] LayerUnitTests/0.DenseLayerInit
100 [ OK       ] LayerUnitTests/0.DenseLayerInit (1 ms)
101 [ RUN      ] LayerUnitTests/0.DenseLayerCompute
102 [ OK       ] LayerUnitTests/0.DenseLayerCompute (0 ms)
103 [ RUN      ] LayerUnitTests/0.DenseLayerBackpropNoReg
104 [ OK       ] LayerUnitTests/0.DenseLayerBackpropNoReg (0 ms)
105 [ RUN      ] LayerUnitTests/0.DenseLayerBackpropL1Reg
106 [ OK       ] LayerUnitTests/0.DenseLayerBackpropL1Reg (0 ms)
107 [ RUN      ] LayerUnitTests/0.DenseLayerBackpropL2Reg
108 [ OK       ] LayerUnitTests/0.DenseLayerBackpropL2Reg (0 ms)
109 [ RUN      ] LayerUnitTests/0.DropoutLayerComputeTrain
110 [ OK       ] LayerUnitTests/0.DropoutLayerComputeTrain (0 ms)
111 [ RUN      ] LayerUnitTests/0.DropoutLayerComputeEval
112 [ OK       ] LayerUnitTests/0.DropoutLayerComputeEval (0 ms)
113 [ RUN      ] LayerUnitTests/0.DropoutLayerBackprop
114 [ OK       ] LayerUnitTests/0.DropoutLayerBackprop (0 ms)
115 [ RUN      ] LayerUnitTests/0.InputLayerCompute
116 [ OK       ] LayerUnitTests/0.InputLayerCompute (0 ms)
117 [ RUN      ] LayerUnitTests/0.InputLayerBackprop
118 [ OK       ] LayerUnitTests/0.InputLayerBackprop (0 ms)
119 [-----] 10 tests from LayerUnitTests/0 (2 ms total)
120
121 [-----] 10 tests from LayerUnitTests/1, where TypeParam = double
122 [ RUN      ] LayerUnitTests/1.DenseLayerInit
123 [ OK       ] LayerUnitTests/1.DenseLayerInit (2 ms)
124 [ RUN      ] LayerUnitTests/1.DenseLayerCompute
125 [ OK       ] LayerUnitTests/1.DenseLayerCompute (0 ms)
126 [ RUN      ] LayerUnitTests/1.DenseLayerBackpropNoReg
127 [ OK       ] LayerUnitTests/1.DenseLayerBackpropNoReg (0 ms)
128 [ RUN      ] LayerUnitTests/1.DenseLayerBackpropL1Reg
129 [ OK       ] LayerUnitTests/1.DenseLayerBackpropL1Reg (0 ms)
130 [ RUN      ] LayerUnitTests/1.DenseLayerBackpropL2Reg
131 [ OK       ] LayerUnitTests/1.DenseLayerBackpropL2Reg (0 ms)
132 [ RUN      ] LayerUnitTests/1.DropoutLayerComputeTrain
133 [ OK       ] LayerUnitTests/1.DropoutLayerComputeTrain (0 ms)
134 [ RUN      ] LayerUnitTests/1.DropoutLayerComputeEval
135 [ OK       ] LayerUnitTests/1.DropoutLayerComputeEval (0 ms)
136 [ RUN      ] LayerUnitTests/1.DropoutLayerBackprop
137 [ OK       ] LayerUnitTests/1.DropoutLayerBackprop (0 ms)
138 [ RUN      ] LayerUnitTests/1.InputLayerCompute
```

```
139 [ OK ] LayerUnitTests/1.InputLayerCompute (0 ms)
140 [ RUN ] LayerUnitTests/1.InputLayerBackprop
141 [ OK ] LayerUnitTests/1.InputLayerBackprop (0 ms)
142 [-----] 10 tests from LayerUnitTests/1 (3 ms total)
143
144 [-----] 7 tests from UtilsIntegrationTests/0, where TypeParam = float
145 [ RUN ] UtilsIntegrationTests/0.MulMulPow
146 [ OK ] UtilsIntegrationTests/0.MulMulPow (0 ms)
147 [ RUN ] UtilsIntegrationTests/0.MulDivMean
148 [ OK ] UtilsIntegrationTests/0.MulDivMean (0 ms)
149 [ RUN ] UtilsIntegrationTests/0.MulAddRoot
150 [ OK ] UtilsIntegrationTests/0.MulAddRoot (0 ms)
151 [ RUN ] UtilsIntegrationTests/0.MulSubExp
152 [ OK ] UtilsIntegrationTests/0.MulSubExp (0 ms)
153 [ RUN ] UtilsIntegrationTests/0.DivAddLog
154 [ OK ] UtilsIntegrationTests/0.DivAddLog (0 ms)
155 [ RUN ] UtilsIntegrationTests/0.DivSubAbs
156 [ OK ] UtilsIntegrationTests/0.DivSubAbs (0 ms)
157 [ RUN ] UtilsIntegrationTests/0.AddSubTranspose
158 [ OK ] UtilsIntegrationTests/0.AddSubTranspose (0 ms)
159 [-----] 7 tests from UtilsIntegrationTests/0 (0 ms total)
160
161 [-----] 7 tests from UtilsIntegrationTests/1, where TypeParam = double
162 [ RUN ] UtilsIntegrationTests/1.MulMulPow
163 [ OK ] UtilsIntegrationTests/1.MulMulPow (0 ms)
164 [ RUN ] UtilsIntegrationTests/1.MulDivMean
165 [ OK ] UtilsIntegrationTests/1.MulDivMean (0 ms)
166 [ RUN ] UtilsIntegrationTests/1.MulAddRoot
167 [ OK ] UtilsIntegrationTests/1.MulAddRoot (0 ms)
168 [ RUN ] UtilsIntegrationTests/1.MulSubExp
169 [ OK ] UtilsIntegrationTests/1.MulSubExp (0 ms)
170 [ RUN ] UtilsIntegrationTests/1.DivAddLog
171 [ OK ] UtilsIntegrationTests/1.DivAddLog (0 ms)
172 [ RUN ] UtilsIntegrationTests/1.DivSubAbs
173 [ OK ] UtilsIntegrationTests/1.DivSubAbs (0 ms)
174 [ RUN ] UtilsIntegrationTests/1.AddSubTranspose
175 [ OK ] UtilsIntegrationTests/1.AddSubTranspose (0 ms)
176 [-----] 7 tests from UtilsIntegrationTests/1 (0 ms total)
177
178 [-----] 4 tests from LayerIntegrationTests/0, where TypeParam = float
179 [ RUN ] LayerIntegrationTests/0.DenseReluDenseCCE
180 [ OK ] LayerIntegrationTests/0.DenseReluDenseCCE (0 ms)
181 [ RUN ] LayerIntegrationTests/0.DenseSoftmaxDenseBCE
182 [ OK ] LayerIntegrationTests/0.DenseSoftmaxDenseBCE (0 ms)
183 [ RUN ] LayerIntegrationTests/0.DenseSigmoidDenseMSE
```


```
184 [ OK ] LayerIntegrationTests/0.DenseSigmoidDenseMSE (0 ms)
185 [ RUN ] LayerIntegrationTests/0.DenseLinearDenseMAE
186 [ OK ] LayerIntegrationTests/0.DenseLinearDenseMAE (0 ms)
187 [-----] 4 tests from LayerIntegrationTests/0 (0 ms total)
188
189 [-----] 4 tests from LayerIntegrationTests/1, where TypeParam = double
190 [ RUN ] LayerIntegrationTests/1.DenseReluDenseCCE
191 [ OK ] LayerIntegrationTests/1.DenseReluDenseCCE (0 ms)
192 [ RUN ] LayerIntegrationTests/1.DenseSoftmaxDenseBCE
193 [ OK ] LayerIntegrationTests/1.DenseSoftmaxDenseBCE (0 ms)
194 [ RUN ] LayerIntegrationTests/1.DenseSigmoidDenseMSE
195 [ OK ] LayerIntegrationTests/1.DenseSigmoidDenseMSE (0 ms)
196 [ RUN ] LayerIntegrationTests/1.DenseLinearDenseMAE
197 [ OK ] LayerIntegrationTests/1.DenseLinearDenseMAE (0 ms)
198 [-----] 4 tests from LayerIntegrationTests/1 (0 ms total)
199
200 [-----] Global test environment tear-down
201 [-----] 83 tests from 9 test suites ran. (8 ms total)
202 [ PASSED ] 83 tests.

Post Run actions/checkout@v2

1 Post job cleanup.
2 /opt/homebrew/bin/git version
3 git version 2.47.0
4 Copying '/Users/runner/.gitconfig' to '/Users/runner/work/_temp/8a7e8f99-b5d6-462e-a210-7d5fc6290233/.gitconfig'
5 Temporarily overriding HOME='/Users/runner/work/_temp/8a7e8f99-b5d6-462e-a210-7d5fc6290233' before making global c
6 Adding repository directory to the temporary git global config as a safe directory
7 /opt/homebrew/bin/git config --global --add safe.directory /Users/runner/work/NeuralNetwork-cpp/NeuralNetwork-cpp
8 /opt/homebrew/bin/git config --local --name-only --get-regexp core.sshCommand
9 /opt/homebrew/bin/git submodule foreach --recursive sh -c 'git config --local --name-only --get-regexp 'core.sshC
10 /opt/homebrew/bin/git config --local --name-only --get-regexp http.https://github.com/.extraheader
11 http.https://github.com/.extraheader
12 /opt/homebrew/bin/git config --local --unset-all http.https://github.com/.extraheader
13 /opt/homebrew/bin/git submodule foreach --recursive sh -c 'git config --local --name-only --get-regexp 'http.https://github.com/.extraheader' || :'


Complete job

1 Cleaning up orphan processes
```

Triggered via push 2 months ago	Status	Total duration
 sami-hatna66 pushed -> e0e9482 main	Success	56s

run_tests.yml

on: push

 build-and-test

45s

Evaluation

Currently, the tests only run on a MacOS server, as this was the platform the library was developed on. If I had more time, I would have extended the pipeline to different operating systems and compilers. This can be done in Github actions using a matrix strategy which creates multiple pipeline configurations based on combinations of environment variables. The current pipeline only runs the unit and integration tests. It doesn't run the system tests because they would exceed the runtime limit on the free tier of Github actions. If I had the budget, I would rent a more capable server and add the system tests to the pipeline. With a higher budget I could also add performance regression testing to the pipeline, possibly feeding the outputs into a Grafana dashboard to produce nicely presented daily performance reports.

PR Templates

A simple automation measure I included to improve software quality is adding an automatic checklist to every pull request opened against the repo. The checklist contains a number of guidelines for C++ programming best practices, thus encouraging anyone contributing to the project to think about the quality of the code they are submitting before trying to get it merged in. Below is an example PR with the checklist automatically added:

Testing #1

Merged sami-hatna66 merged 25 commits into main from testing on Nov 23, 2024

Conversation 0 Commits 25 Checks 2 Files changed 19

sami-hatna66 commented on Nov 15, 2024 • edited

PR Checklist

- Project compiles locally and on CI without error
- Project passes all regression tests
- Unit and integration tests added for new features
- New tests are integrated into the project's GTest framework and are called by test_all
- New files added to CMake
- Every header has include guards
- Code adheres to project naming conventions
- Every class with a destructor has a copy/move constructor (may be default or deleted)
- Every class with a virtual function has a virtual destructor
- References are used where possible to avoid unnecessary copies

Code Coverage Reporting

The final instance of SQA automation I added was code coverage report generation using a tool called gcovr. gcovr is a utility for the code coverage tool gcov which generates appealing HTML reports from gcov's opaque text-based outputs. Automated code coverage reporting is worked into the project's build system by adding a CMake option ENABLE_GCOV (see project README for more detailed instructions). After running the tests, one need only build the 'coverage' target to generate the reports. Below is a screenshot of the code coverage report. The full coverage report can be viewed by opening `./coverage/coverage.html` in your browser. Detailed analysis can be found in the "Test and Benchmark Results" document.

File	Lines			Functions		Branches	
bench/layer_bench.cpp	<div><div></div></div>	0.0%	0 / 69	0.0%	0 / 25	0.0%	0 / 86
bench/system_bench.cpp	<div><div></div></div>	0.0%	0 / 38	0.0%	0 / 5	0.0%	0 / 52
bench/utils_bench.cpp	<div><div></div></div>	0.0%	0 / 66	0.0%	0 / 22	0.0%	0 / 98
nn-cpp/Accuracy.cpp	<div><div></div></div>	0.0%	0 / 43	0.0%	0 / 14	0.0%	0 / 84
nn-cpp/Activations.cpp	<div><div></div></div>	38.2%	39 / 102	33.3%	8 / 24	24.0%	46 / 192
nn-cpp/Layer.cpp	<div><div></div></div>	81.9%	127 / 155	67.9%	38 / 56	63.2%	172 / 272
nn-cpp/Loss.cpp	<div><div></div></div>	42.1%	82 / 195	38.5%	10 / 26	31.9%	106 / 332
nn-cpp/LossActivation.cpp	<div><div></div></div>	0.0%	0 / 31	0.0%	0 / 10	0.0%	0 / 44
nn-cpp/Model.cpp	<div><div></div></div>	0.0%	0 / 156	0.0%	0 / 1152	0.0%	0 / 13696
nn-cpp/ModelLayer.cpp	<div><div></div></div>	22.2%	2 / 9	40.0%	4 / 10	-%	0 / 0
nn-cpp/Optimizers.cpp	<div><div></div></div>	0.0%	0 / 80	0.0%	0 / 24	0.0%	0 / 424
nn-cpp/utils.cpp	<div><div></div></div>	100.0%	180 / 180	100.0%	40 / 40	74.7%	236 / 316
sample/MnistSample.cpp	<div><div></div></div>	0.0%	0 / 89	0.0%	0 / 3	0.0%	0 / 284
test/basic_test.cpp	<div><div></div></div>	100.0%	4 / 4	100.0%	3 / 3	30.0%	12 / 40
test/layer_integration_tests.cpp	<div><div></div></div>	100.0%	76 / 76	100.0%	28 / 28	39.4%	168 / 426
test/layer_unit_tests.cpp	<div><div></div></div>	100.0%	304 / 304	100.0%	64 / 64	35.1%	960 / 2734
test/test_all.cpp	<div><div></div></div>	100.0%	3 / 3	100.0%	1 / 1	-%	0 / 0
test/utils_integration_tests.cpp	<div><div></div></div>	100.0%	56 / 56	100.0%	46 / 46	34.9%	352 / 1008
test/utils_unit_tests.cpp	<div><div></div></div>	100.0%	690 / 690	100.0%	124 / 124	40.1%	1728 / 4310

Evaluation

gcovr currently crashes if you try to generate reports after running the system tests. Some preliminary debugging seems to suggest it is a problem with running gcov on OpenCV dependencies which are used for image handling in the system tests. I didn't have enough time to debug this, and it is likely a problem on gcovr's side, so outside the scope of what I can resolve. The outputs pictured above are from running the unit and integration tests; code coverage would be higher if the system tests were accounted for.