테스터

2015년 10월 26일에 한국어로 옮겨짐

원문: https://github.com/torch/torch7/blob/master/doc/tester.md

목차

이 클래스는 시험 프레임워크를 제공합니다. 이 클래스는 이미 nn 패키지에서 클래스들의 정확함을 검증하기 위해 사용되고 있습니다.

이 프레임워크는 일반적으로 다음과 같이 사용됩니다.

```
mytest = {}

tester = torch.Tester()

function mytest.TestA()
    local a = 10
    local b = 10
    tester:asserteq(a,b,'a == b')
    tester:assertne(a,b,'a ~= b')
end

function mytest.TestB()
    local a = 10
    local b = 9
    tester:assertlt(a,b,'a < b')
    tester:assertgt(a,b,'a > b')
end

tester:add(mytest)
tester:run()
```

이 코드의 실행은 두 개의 시험 함수에서 두 개의 에러를 보고할 것입니다. 매우 관련된 시험 케이스들이 존재하지 않는 이상, 일반적으로 각 시험 함수에 하나의 시험 케이스만 넣는 것이 더 좋습니다. 에러 리포트에는 메시지와 에러가 몇 번째 줄에서 생겼는지를 포함됩니다.

```
Running 2 tests
**  ==> Done

Completed 2 tests with 2 errors
```

```
--------------------------------------------------------------------------------
TestB
a < b
 LT(<) violation   val=10, condition=9
    ...y/usr.t7/local.master/share/lua/5.1/torch/Tester.lua:23: in function 'assertlt'
    [string "function mytest.TestB()..."]:4: in function 'f'

--------------------------------------------------------------------------------
TestA
a ~= b
 NE(~=) violation   val=10, condition=10
    ...y/usr.t7/local.master/share/lua/5.1/torch/Tester.lua:38: in function 'assertne'
    [string "function mytest.TestA()..."]:5: in function 'f'

--------------------------------------------------------------------------------
```

torch.Tester()

`torch.Tester` 클래스의 한 새로운 인스턴스를 리턴합니다.

add(f, 'name')

이름이 `name`인 한 새로운 시험 함수를 추가합니다. 그 시험 함수는 `f`에 저장됩니다. 그 함수는 어떤 함수도 없이 어떤 값도 리턴하지 않고 실행될 것으로 가정됩니다.

add(ftable)

재귀적으로 테이블 `ftable`의 모든 함수 엔트리들을 시험들로 추가합니다. 이 테이블은 오직 함수들 또는 함수들의 nested 테이블만 가질 수 있습니다.

assert(condition [, message])

만약 조건이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertlt(val, condition [, message])

만약 `val < condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertgt(val, condition [, message])

만약 `val > condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertle(val, condition [, message])

만약 `val <= condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertge(val, condition [, message])

만약 `val >= condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

asserteq(val, condition [, message])

만약 `val == condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertne(val, condition [, message])

만약 `val ~= condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertTensorEq(ta, tb, condition [, message])

만약 `max(abs(ta-tb)) < condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertTensorNe(ta, tb, condition [, message])

만약 `max(abs(ta-tb)) >= condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertTableEq(ta, tb, condition [, message])

만약 `max(abs(ta-tb)) < condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertTableNe(ta, tb, condition [, message])

만약 `max(abs(ta-tb)) >= condition`이 true가 아니면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

assertError(f [, message])

만약 함수 f() 호출이 에러를 리턴하지 않으면, 에러를 선택적으로 입력된 메시지와 함께 저장합니다.

run()

add() 함수를 사용하여 저장된 모든 시험 함수들을 실행합니다. 실행하는 동안 이 함수는 진행 과정을 보고하고 끝나면 모든 에러들의 요약을 보여줍니다.

목차