Traffic Signs Recognition

Verificare formala

Pinghireac Bogdan, Eduard Marian Neguriță,
Andreea Bîra-Negrut, Larisa Drăgănescu

[1]UNIVERSITATEA DE VEST DIN TIMIȘOARA

[2]FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

2023

Rolul retelelor neuronale profunde (DNNs) in rezolvarea problemelor complexe

- Capacitatea de înțelegere a datelor complexe
- Abilitatea de a face prognoze si decizii
- Invatare automata si adaptabilitate continua
- Aplicabilitate in diverse domenii
- Solutii eficiente in managementul datelor mari

Traffic Signs Recognition

- contine mii de imagini care reprezinta diferite semne de circulatie
- imaginile reflecta variatii semnificative in aspectul vizual al semnelor
- fiecare imagine este insotita de mai multe seturi de caracteristici pre-calculate

Tool-uri

- Alpha Beta Crown: instrument de verificare pentru a asigura functionarea corespunzatoare a retelelor neuronale
- Marabou: esential pentru testarea si validarea modelelor de invatare automata

Provocari in instalare

Alpha Beta Crown

- Unzip instalat, astfel apar erori
- Sistem de operare necesar:Linux prin WSL si anumite librarii instalate
- Activarea environment-ului pentru rulare
- Instalarea CUDA

Marabou

- CMake instalat si de asemenea pachetul build-essential
- Configurare PATH pentru cmake

Rezultate

```
(alpha-beta-crown) vf@DESKTOP-LA01G8M:~/VF/Marabou$ ./build/Marabou resources/nnet/acasxu/ACASXU_experimental_v2a_2_7.nnet resources/properties/acas_property_3.txt
Network: resources/nnet/acasxu/ACASXU_experimental_v2a_2_7.nnet
Number of layers: 8. Input layer size: 5. Output layer size: 5. Number of ReLUs: 300
Total number of variables: 610
Property: resources/properties/acas_property_3.txt

Engine::processInputQuery: Input query (before preprocessing): 309 equations, 610 variables
Engine::processInputQuery: Input query (after preprocessing): 609 equations, 838 variables

Input bounds:
        x0: [ -0.3035,  -0.2986]
        x1: [ -0.0095,   0.0095]
        x2: [  0.4934,   0.5000]
        x3: [  0.3000,   0.5000]
        x4: [  0.3000,   0.5000]

Branching heuristics set to LargestInterval
unsat
(alpha-beta-crown) vf@DESKTOP-LA01G8M:~/VF/Marabou$
```

Figure: Marabou

Rezultate

```
/opt/conda/conda-bld/pytorch_1646755888534/work/aten/src/ATen/native/cuda/IndexKernel.cu:91: operator(): block: [404,0,0], thread: [61,0,0] Assertion `index >= -sizes[i] && index < sizes[i] && "index out of bo
unds"` failed.
/opt/conda/conda-bld/pytorch_1646755888534/work/aten/src/ATen/native/cuda/IndexKernel.cu:91: operator(): block: [404,0,0], thread: [62,0,0] Assertion `index >= -sizes[i] && index < sizes[i] && "index out of bo
unds"` failed.
/opt/conda/conda-bld/pytorch_1646755888534/work/aten/src/ATen/native/cuda/IndexKernel.cu:91: operator(): block: [404,0,0], thread: [63,0,0] Assertion `index >= -sizes[i] && index < sizes[i] && "index out of bo
unds"` failed.
Traceback (most recent call last):
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/abcrown.py", line 612, in <module>
    abcrown.main()
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/abcrown.py", line 591, in main
    verified_status = self.complete_verifier(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/abcrown.py", line 398, in complete_verifier
    l, nodes, ret = self.bab(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/abcrown.py", line 241, in bab
    result = general_bab(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/bab.py", line 340, in general_bab
    global_lb = act_split_round(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/bab.py", line 165, in act_split_round
    split_domain(net, domains, d, batch, impl_params=impl_params,
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/bab.py", line 106, in split_domain
    ret = net.update_bounds(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/beta_CROWN_solver.py", line 329, in update_bounds
    tmp_ret = self.net.compute_bounds(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/auto_LiRPA/bound_general.py", line 1193, in compute_bounds
    ret1 = self._get_optimized_bounds(bound_side='lower', **kwargs)
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/auto_LiRPA/optimized_bounds.py", line 494, in _get_optimized_bounds
    full_ret, stop_criterion) = pruner.prune(
  File "/home/vf/VF/alpha-beta-CROWN/complete_verifier/auto_LiRPA/opt_pruner.py", line 63, in prune
    positive_domain_ratio = float(
RuntimeError: CUDA error: device-side assert triggered
CUDA kernel errors might be asynchronously reported at some other API call,so the stacktrace below might be incorrect.
For debugging consider passing CUDA_LAUNCH_BLOCKING=1.
Aborted
(alpha-beta-crown) vf@DESKTOP-LA01G8M:~/VF/alpha-beta-CROWN/complete_verifier$
```

Figure: Alpha Beta Crown

Concluzie

Cu toate acestea, este esential sa continuam dezvoltarea si imbunatatirea instrumentelor de verificare, abordand noile provocari care pot aparea odata cu evolutia tehnologiei. Astfel, ne asiguram ca sistemele de recunoastere a semnelor de circulatie devin tot mai precise, sigure si capabile sa contribuie semnificativ la imbunatatirea sigurantei si eficientei traficului rutier.

Multumim!