Măsurarea performanţei instrumentelor specializate de verificare a reţelelor neuronale

Chira Andreea-Marina, andreea.chira00@e-uvt.ro
Hasna Diana-Mihaela, diana.hasna00@e-uvt.ro
Hasan Majd, majd.hasan93@e-uvt.ro
Chera Denis-Marian, denis.chera00@e-uvt.ro
Vanciu Catalin-Marian, catalin.vanciu00@e-uvt.ro

December 22, 2023

Abstract

Este bine cunoscut faptul că numărul de vehicule rutiere a crescut enorm datorită tehnologiilor dezvoltate din industria auto şi, mai precis, disponibilităţii tarifelor mici. Astfel, semnele de circulaţie contribuie atât la siguranţa şoferilor cât şi a pietonilor, având un rol esenţial în fluidizarea şi ordonarea traficului rutier.

În acest sens, lucrarea se concentrează pe tehnicile şi instrumentele disponibile pentru verificarea formală, astfel încat VNN-COMP-2023 (Concursul de verificare a reţelelor neuronale) a fost ales ca principal punct de plecare al acestui studiu. Concursul poate fi accesat folosind link-ul: `https://github.com/ChristopherBrix/vnncomp2023_benchmarks` şi propune un model de reţea neuronală în formatul ONNX împreună cu o specificaţie în formatul VNNLIB, aceste unelte stabilind dacă există o situaţie în care specificaţia se potriveşte cu modelul. Astfel de descoperiri facilitează verificarea unor proprietăţi diverse în diferite aplicaţii ale reţelelor neuronale.

Aşadar, din cea mai recentă ediţie, au fost selectate două instrumente care au obţinut cele mai bune performanţe, respectiv **Alpha-Beta Crown & Marabou** participând la cele mai multe criterii de referinţă. Prin urmare, obiectivul acestei lucrări este de a determina rezultatele obţinute în urma utilizării celor două tooluri asupra Benchmark-ului **Traffic Signs Recognition**, considerat set de date.

1 Introducere

În prezent, unul dintre cele mai utilizate sisteme de transport este reprezentat de transportul rutier, care generează un trafic semnificativ aflat într-o continuă creştere în întreaga lume, iar conform Organizaţiei Mondiale a Sănătăţii (OMS) "În fiecare an, 1,3 milioane de oameni mor şi 50 de milioane sunt răniţi". Un studiu realizat de către cei

de la National Highway Transportation Safety Administration (NHTSA) din SUA în anul 2016 subliniază faptul că, procentul accidentelor rutiere datorate erorilor umane reprezintă între 94% şi 96% din totalul accidentelor rutiere provocate de neglijenţa conducătorilor auto, ignorarea sau înţelegerea greşită a indicatoarelor rutiere, ceea ce atestă necesitatea şi importanţa existenţei unui sistem automat şi inteligent pentru a asista şoferul în sarcinile sale de conducere.

În prezent, numeroşi producători auto încorporează în vehiculele lor sisteme care oferă diferite grade de autonomie şi securitate. Printre aceste sisteme se numără şi sistemele de recunoaştere a semnelor de circulaţie (TSR), care ajută la minimizarea numărului de decizii eronate şi de accidente. [5]

În cazul sistemelor de tip TSR, un aspect crucial este reprezentat de reţelele neuronale care sunt folosite pentru a recunoaşte pattern-uri complexe şi pentru a face predicţii sau decizii pe baza datelor de intrare. Reţelele neuronale sunt adesea cutii negre şi este greu de asigurat că se comportă în condiţii de siguranţă şi predictibilitate în cazul unor intrări zgomotoase sau rău intenţionate (intrări maliţioase). De asemenea, cercetător-ul Huan Zhang a declarat că: "Verificarea reţelelor neuronale urmăreşte să ofere garanţii demonstrabile pentru proprietăţile dorite ale reţelelor neuronale, cum ar fi siguranţa şi robusteţea". [3] De asemenea, acestea pot fi antrenate să detecteze şi să clasifice diferite tipuri de semne de circulaţie din imagini sau clipuri video.

Prin urmare, verificarea reţelelor neuronale urmează adesea un ciclu de verificare continuă, care implică reantrenarea rețelelor neuronale având în vedere o anumită proprietate de verificare, după cum se poate observa şi în Figura 1. [4]

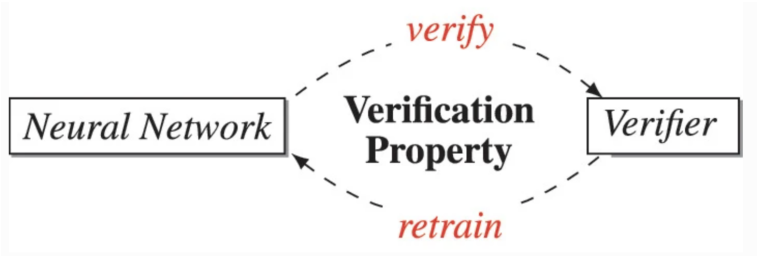

Figure 1: Ciclul de verificare a reţelelor neuronale

Astfel, în cadrul acestei lucrări, atenţia este orientată spre găsirea unei modalităţi optime de măsurare a eficienţei şi corectitudinii diferitelor instrumente de verificare a reţelelor neuronale în contextul recunoaşterii semnelor de circulaţie. Pentru a atinge acest obiectiv, vor fi detaliate, în secțiunile următoare, atât modul de funcționare al unui instrument de acest tip, cât și importanța unui benchmark și impactul pe care îl poate avea asupra performanței uneltelor selectate.

2 Benchmark

În contextul acestui experiment, s-a utilizat benchmark-ul **Traffic Sign Recognition** [1] constituit dintr-un ansamblu de fișiere ONNX (Open Neural Network Exchange) și VNNLIB, și care este folosit pentru a evalua performanța diferitelor metode și instrumente de verificare a rețelelor neuronale. Fișierele ONNX din benchmark-ul ales au la baza modelul unei rețele neuronale antrenate pe setul de date German Traffic

Sign Recognition Benchmark (GTRSB) existent pe site-ul Kaggle, care cuprinde peste 50000 de imagini ce sunt clasificate în 43 de clase, fiecare simbolizând un tip distinct de indicator rutier. Pe de altă parte, fișierele VNNLIB sunt reprezentate de problemele de verificare asociate cu acele modele, care includ diverse condiții ce pot fi legate de clasificare sau anumite proprietăți și caracteristici specifice modelelor respective.

Conţinutul fişierelor VNNLIB constă în:

1. **Declararea variabilelor de intrare**

   Fiecare valoare reprezintă o caracteristică (ex: pixel) a imaginii de intrare şi sunt declarate ca numere reale.

   Exemplu: (declare-const X_0 Real)

2. **Constrângerile de intrare (assert-uri)**

   Constrângerile sunt reprezentate de condiţiile impuse fiecărei variabile de intrare pentru a specifica intervalele valide pentru valorile pixelilor imaginii. Condițiile respective sunt exprimate cu ajutorul unor valori din intervalul [0, 255], care în contextul imaginilor se referă la scala intensităților de lumină sau culorilor.

   Exemplu:

   (assert(<= X_0 255.00000000))

   (assert(>= X_0 244.00000000))

3. **Constrângerile de ieşire**

   Constrângerile de ieşire sunt definite ca o disjuncţie (sau), în cadrul căreia fiecare clauză reprezintă o condiţie pentru variabilele de ieşire.

   Exemplu:

   (assert (or

   (and (>= Y_0 Y_3))

   (and (>= Y_1 Y_3))))

   Condiţiile impun ca valorile de ieşire să fie mai mari sau egale cu valoarea de ieşire a celui de-al treilea neuron.

4. **Valoarea Epsilon**

   Aceasta poate oscila, indicând nivelul permis de perturbație sau zgomot pentru variabilele de intrare în timpul verificării.

   Exemplu: 10.00000

3 Tool

3.1 Alpha-Beta Crown

3.2 Descriere

α , β -CROWN (alpha-beta-CROWN) este un verificator de rețele neuronale care se bazează pe un mecanism eficient de propagare a limitelor liniare și pe o metodă de

"branch and bound" (Bab). Procesul poate fi accelerat în mod eficient pe GPU și se poate adapta la rețele convoluționale relativ mari (de exemplu, milioane de parametri). De asemenea, prin intermdiul bibliotecii versatile auto_LiRPA acesta acceptă o gamă largă de arhitecturi de rețele neuronale, cum ar fi: CNN, ResNet și diverse funcții de activare. Totodată, tool-ul α , β -CROWN poate oferi garanții demonstrabile de robustețe împotriva atacurilor adversarilor și poate verifica, de asemenea, alte proprietăți generale ale rețelelor neuronale. [2]

Descrierea detaliată a tool-ului:

- CROWN: reprezintă un algoritm eficient de verificare bazat pe propagarea limitelor. Acesta propagă o inegalitate liniară în sens invers prin rețea și utilizează limite liniare pentru a relaxa funcțiile de activare.
- α-CROWN: este un verificator de rețele neuronale optimizat, utilizat în verificatorul Fast-and-Complete, care optimizează în comun limitele stratului intermediar și limitele stratului final în CROWN prin intermediul variabilei α. De obicei, are o putere mai mare decât LP (Linear Programming), deoarece LP nu poate restrânge în mod necostisitor limitele stratului intermediar.
- β-CROWN: introduce un parametru β optimizabil pentru a încorpora constrângerile de divizare în branch and bound (BaB) în procesul de propagare a limitelor în CROWN. Un verificator de rețele neuronale robust și scalabil este creat prin combinarea branch and bound cu o propagare eficientă și accelerată de GPU a limitelor.

3.3 Instalare

Paşii urmaţi pentru configurarea tool-ului α , β -CROWN:

1. **Instalare Miniconda pentru Sistemul de Operare Linux:**

   Pentru realizarea acestui proces s-au folosit comezile ce pot fi regăsite în Figura 2.

```
mkdir -p ~/miniconda3
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh -O ~/miniconda3/miniconda.sh
bash ~/miniconda3/miniconda.sh -b -u -p ~/miniconda3
rm -rf ~/miniconda3/miniconda.sh
```

Figure 2: Instalare Miniconda

2. **Clonarea proiectlui α , β -CROWN**

   Pentru realizarea acestui proces s-a folosit comanda ce poate fi regăsită în Figura 3.

```
git clone --recursive https://github.com/Verified-Intelligence/alpha-beta-CROWN.git
```

Figure 3: Clonarea proiectului

3. **Configurarea şi activarea environment-ului α , β -CROWN:**

   Pentru parcurgerea acestei etape a fost necesară utilizarea fișierul environment.yaml aflat în folderul complete_verifier al proiectului și s-au folosit comezile ce pot fi regăsite în Figura 4.

```
# Remove the old environment, if necessary.
conda deactivate; conda env remove --name alpha-beta-crown
# install all dependents into the alpha-beta-crown environment
conda env create -f complete_verifier/environment.yaml --name alpha-beta-crown
# activate the environment
conda activate alpha-beta-crown
```

Figure 4: Environment-ul α , β -CROWN

4. **Testare**

   După activarea environment-ului a urmat procesul de testare realizat prin intermediul comenzilor ce pot fi regăsite în Figura 5.

```
cd complete_verifier
python abcrown.py --config exp_configs/cifar_resnet_2b.yaml
```

Figure 5: Testare

În urma testării tool-ului α , β -CROWN, rezulatate pot fi observate în Figura 6:

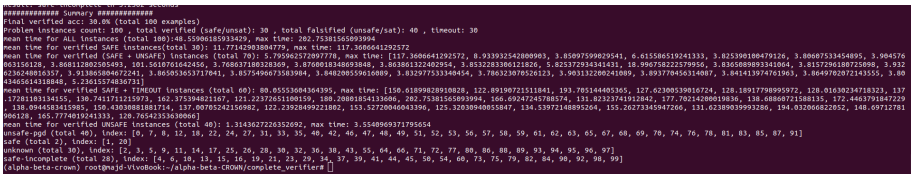

Figure 6: Rezulatele obţinute

3.4 Rularea α-β Crown pentru benchmark-ul ales

Acest proces a debutat prin clonarea proiectului VNN-COMP2023 cu scopul de a avea acces la benchmark-ul descris mai sus în cadrul acestei lucrări, și anume Traffic Sign Recognition ce are rolul de a permite evaluarea instrumentului de verificare folosit. După o căutare meticuloasă, s-a descoperit faptul că, configuraţiile aferente acestei ultime ediții ale competiţiei se regăsesc în path-ul complete_verifier exp_configs în folderul numit vnncomp23, făcând parte din proiectul α-β Crown. Astfel, în continuare s-a identificat fișierul gtrsb.yaml, care a fost configurat corespunzător pentru a stabili conexiunea necesară între tool și benchmark pentru a putea iniția, mai apoi, acțiunea de rulare propriu-zisă.

În consecință, după pregătirea configurației, pentru rulare s-a folosit comanda ce poate fi vizualizată în Figura 7, comandă cu ajutorul căreia s-a realizat testarea tool-ului adaptată astfel încat rezultatele din urma rulării să fie salvate în acest caz într-un fişier (output1.txt) pentru a putea fi vizualizate și analizate ulterior.

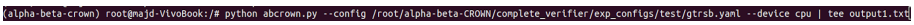

Figure 7: Rulare α , β -CROWN

Output-ul obţinut în urma rulării benchmark-ului prin intermediul tool-ului poate fi observat în Figura 8 de mai jos.

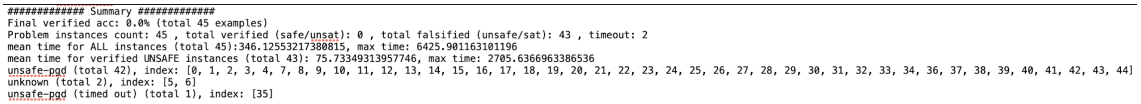

Figure 8: Output

Rezultatele furnizate indică faptul că, din cele 45 de instanțe (fiecare reprezentând un fișier vnnlib) niciuna nu a fost verificată ca fiind sigură, majoritatea (43) au fost clasificate ca nesigure, iar două dintre cazurile nesigure au atins timeout-ul. În contextul actual, "unsafe/sat" se referă la instanțele nesigure care sunt satisfiabile, adică există cel puțin o configurație a rețelei neuronale pentru care nu a fost posibil să se demonstreze că rețeaua răspunde într-un mod dorit sau sigur.

De asemenea, timpul mediu și maxim se calculează atât pentru toate instanțele, cât și în mod particular pentru cele nesigure verificate.

References

[1] *Traffic Signs Recognition - Benchmark, Publicat: 05/07/2023*
https://github.com/ChristopherBrix/vnncomp2023_results/tree/main/alpha_beta_crown/2023_traffic_signs_recognition

[2] α , β -CROWN, *Publicat: 10/10/2023*
https://github.com/Verified-Intelligence/alpha-beta-CROWN/blob/main/README.md

[3] Neural Networks, *Publicat: 15/09/2021*
https://www.cs.cmu.edu/news/2021/zhang_vnn_comp2021

[4] Network Robustness as a Verification Property: A Principled Case Study, *Publicat: 2022*
https://link.springer.com/chapter/10.1007/978-3-031-13185-1_11

[5] Traffic sign recognition and analysis for intelligent vehicles, *Publicat: 01/03/2003*
https://www.sciencedirect.com/science/article/abs/pii/S0262885602001567

Link GitHub: https://github.com/MajdHHH/Team-6-SC