

# Verificare Formală

Amza Rareș, Dienes Oliviu, Militaru Oana

Universitatea de Vest din Timișoara  
Facultatea de Matematică și Informatică, Inginerie Software

22 decembrie 2023



# Cuprins

## Descriere Dataset

Tipul de rețea

Rețea complet conectată (Fully Connected - FC)

Funcția de activare ReLU (Rectified Linear Unit)

Numărul de parametrii

Dimensiunea de intrare

Raritatea (Sparsity)

## Instalare Instrumente

## Bibliografie

Datasetul ACAS XU este conceput pentru a evalua performanța sistemului de evitare a coliziunilor integrate (ACAS Xu) și de detectare și evitare (DAA) în cazul sistemelor de aviație cu asistență la distanță (UAS). [1] Informații despre acest dataset:

# Tipul de rețea

Modelul ACAS XU este o rețea complet conectată (Fully Connected - FC). Funcția de activare folosită este ReLU. [2]

## Rețea complet conectată (Fully Connected - FC)

O rețea complet conectată este un tip de arhitectură neuronală în care fiecare neuron dintr-un strat este conectat la fiecare neuron din stratul următor. În cazul modelului ACAS XU, aceasta înseamnă că toți cei 5 neuroni de intrare sunt conectați la toți cei 13.000 de neuroni din stratul următor. Acest tip de arhitectură este utilizat într-o varietate de aplicații și permite modelului să învețe relații complexe între datele de intrare. [3] [4] [5]

# Funcția de activare ReLU (Rectified Linear Unit)

ReLU este o funcție de activare comună folosită în rețelele neuronale. Ea returnează 0 pentru toate valorile negative și returnează valoarea de intrare pentru valorile pozitive. Această funcție de activare adaugă non-linearitate modelului, ceea ce îi permite să învețe și să captureze relații complexe în date. [2] [6]

## Numărul de parametrii

Modelul ACAS XU are 13.000 de parametrii. Acești parametrii reprezintă ponderile și bias-urile asociate cu conexiunile dintre neuroni și permit modelului să învețe de la datele de antrenare. [4]

# Dimensiunea de intrare

Dimensiunea de intrare a modelului ACAS XU este 5, ceea ce înseamnă că modelul primește un vector de 5 elemente ca intrare pentru fiecare exemplu de date. [7]



# Raritatea (Sparsity)

Raritatea modelului ACAS XU variază între 0% și 20%. Acest lucru poate să se refere la proporția de conexiuni care au ponderi nenule față de totalul conexiunilor posibile.

Algoritmul folosește o rețea neuronală profundă (DNN) pentru comprimarea unei tabele numerice mari care conține scoruri asociate cu diferite manevre din milioane de stări discrete. Această rețea neurală profundă are 128 straturi, dintre care 5-128-64-32-16-5 și 5-256-128-64-32-16-5. Funcțiile activate folosite sunt funcții sigure standard (ReLU) sau funcții sigure saturate (Leaky ReLU). Numărul total de parametri ale rețelei neurali profundă este estimat la aproximativ 1.6 milioane. [8] [9]

# Instalare Instrumente

$\alpha,\beta$ -CROWN (alpha-beta-crown) este un vericator de rețele neuronale care a câștigat competiția VNN-COMP 2021, 2022 și 2023 având cel mai mare scor total și astfel depășind în performanță multe alte verificatoare de rețele neuronale pe o gamă largă de evaluări în decursul a 2 ani. [10] Acesta este un instrument care funcționează doar în sistemul de operare Linux.

Înainte de a începe configurarea instrumentului trebuie să verificăm dacă avem git și Miniconda instalate, iar în caz contrar să le instalăm urmând link-urile unde se găsesc fișierele pentru instalare:

<https://git-scm.com> și

<https://docs.conda.io/projects/miniconda/en/latest/>.

După setarea mediului de lucru vom accesa pagina unde se află proiectul

(<https://github.com/Verified-Intelligence/alpha-beta-CROWN>) și vom urma pașii indicați.

Deoarece am lucrat pe o mașină cu sistemul Windows toate comenzile ulterioare au fost efectuate cu ajutorul WSL și Ubuntu și am urmat următorii pași pentru a configura proiectul:

1. Am clonat proiectul pe dispozitivul nostru cu comanda git clone

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

2. Am creat mediul de lucru din Conda :

```
conda env create -f /complete_verifier  
/environment_py111.yaml --name alpha-beta-crown
```

3. După ce au fost instalate toate modulele necesare vom verifica că mediul de lucru a fost creat.

```
# conda environments:
#
base                /home/plyber/miniconda3
alpha-beta-crown    * /home/plyber/miniconda3/envs/alpha-beta-crown
```

Figura: Verificare mediu de lucru Conda

4. Pentru a putea folosi verificatorul trebuie să configurăm și submodulul auto\_LiRPA:

```
cd auto_LiRPA
python setup.py install
```

# Rulare Instrumente pentru Benchmark

Fiind cu două directoare deasupra complete\_verifier, am clonat repository-ul vnncomp2023\_benchmarks[11] astfel încât calea specificată în fișierul de tip yaml să coincidă cu calea necesară pentru funcționarea instrumentului  $\alpha, \beta$ -CROWN:

```
../../vnncomp2023_benchmarks/benchmarks/acasxu
```

Ne-am asigurat că am activat mediul de lucru ab-crown cu Miniconda3 folosind următoarea instrucțiune:

```
conda activate alpha-beta-crown
```

Apoi am rulat abcrown.py pe benchmarkul acasxu.yaml:

```
cd complete_verifier
```





```
python abcrown.py --config exp_configs/acasxu.yaml
```

Execuția instrumentului s-a finalizat cu succes și am primit următoarele rezultate:

```
##### Summary #####  
Final verified acc: 74.19354838709677% (total 186 examples)  
Problem instances count: 186 ,  
total verified (safe/unsat): 138 ,  
total falsified (unsafe/sat): 47 ,  
timeout: 1  
mean time for ALL instances (total 186):3.258072501031987,  
max time: 118.92911696434021
```

Astfel avem o acuratețe finală de 74,19% pentru 186 de instanțe: 138 instanțe fiind SAT și 47 UNSAT. De asemenea, una dintre instanțe a depășit timpul alocat, astfel că am avut și un timeout.

# Bibliografie I

-  Diego Manzananas López, “Acas xu github repository,” <https://github.com/mldiego/AcasXu>, 2023, accesat la data de 21 decembrie 2023.
-  DeepAI, “Rectified linear unit (relu) - deepai,” 2023, accesat la data de 21 decembrie 2023. [Online]. Available: <https://deepai.org/machine-learning-glossary-and-terms/relu>
-  S. Bak, “Neural network compression of acas xu early prototype is unsafe,” 2022, accesat la data de 22 decembrie 2023. [Online]. Available: <https://arxiv.org/pdf/2201.06626v3.pdf>
-  M. P. Owen, “Aiaa 38th digital avionics systems conference (dasc,” 2019, accesat la data de 19 decembrie 2023. [Online]. Available: <https://ieeexplore.ieee.org/document/9081758>



# Bibliografie II



G. Kutz, "Reluplex," 2017, accesat la data de 21 decembrie 2023. [Online]. Available: [https://link.springer.com/chapter/10.1007/978-3-319-63387-9\\_5](https://link.springer.com/chapter/10.1007/978-3-319-63387-9_5)



A. Vaswani and Shazeer, "Attention is all you need," 2018, accesat la data de 21 decembrie 2023. [Online]. Available: <https://arxiv.org/abs/1803.08375>





Jason T. Davies, "Comparative analysis of acas-xu and daidalus detect-and-avoid systems," <https://ntrs.nasa.gov/api/citations/20180001564/downloads/20180001564.pdf>, 2018, accesat la data de 19 decembrie 2023.



M. Jason T. Davies, "Nasa document," <https://ntrs.nasa.gov/api/citations/20180001564/downloads/20180001564.pdf>, 2018, accesat la data de 22 decembrie 2023.

# Bibliografie III

 K. D. Julian, "Deep neural network compression for aircraft collision avoidance systems," 2018, accesat la data de 22 decembrie 2023. [Online]. Available: <https://arxiv.org/pdf/1810.04240.pdf>

 H. Zhang, K. Xu, Z. Shi, S. Wang, L. Li, J. Chen, Z. Yang, and Y. Wang, "alpha-beta-crown," <https://github.com/Verified-Intelligence/alpha-beta-CROWN/tree/main>, 2021-2022.

 ChristopherBrix, "vnncomp2023<sub>benchmarks</sub>,"

, 2023.