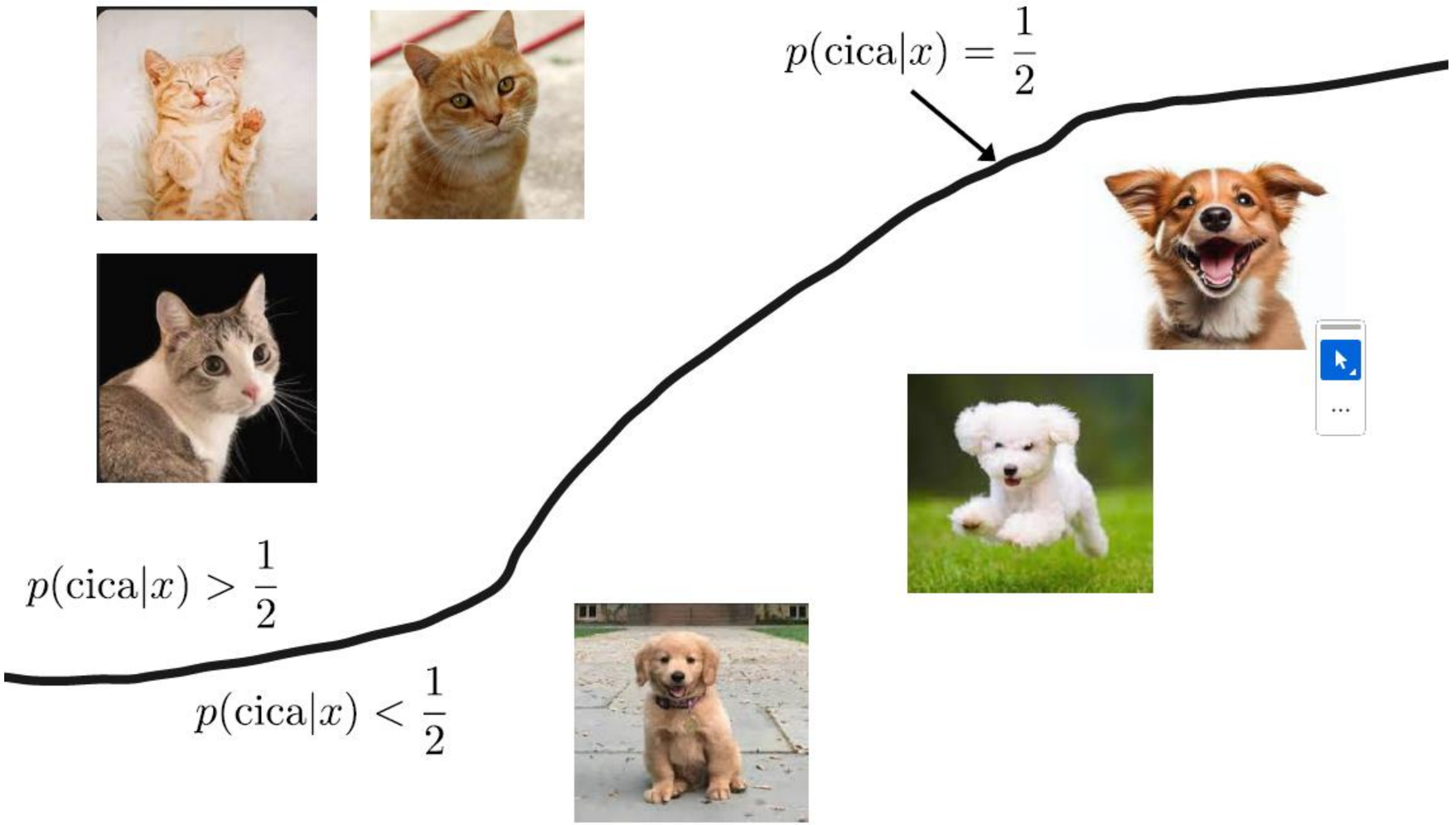
Neuronhálok serülékenysége és robosztusságának vizsgálatára használt eljárás fejlesztése

Bánhelyi Balázs, Zombori Dániel, Szász Attila, Jelasity Márk,
Csendes Tibor

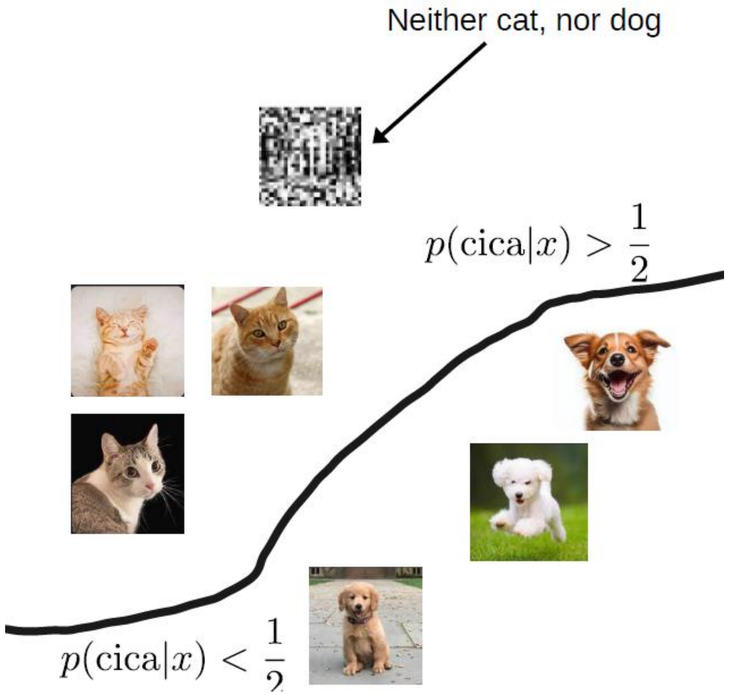
Osztályozó neuronhálók

- A leggyakoribb gépi tanulási feladat.
- Cél: objektum-példányok besorolása előre adott C_1,…, C_m osztályok valamelyikébe.
- Input: valamilyen mérési adatokból álló (fix méretű) vektor.
- Tanítópéldák halmaza: jellemzővektorokból és hozzájuk tartozó osztálycímkékből álló párosok egy halmaza.
- A tanulási feladat tehát: becslést adni az $(x_1, \dots, x_n) \rightarrow c_i$ függvényre a tanítópéldák alapján.

Osztályozó neuronhálók



Problémák a neuronhálóval



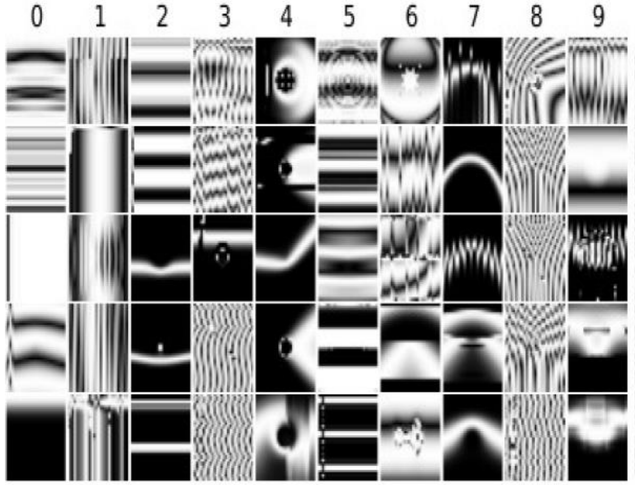


Figure 5. Indirectly encoded, thus regular, images that LeNet believes with 99.99% confidence are digits 0-9. The column and row descriptions are the same as for Fig. 4.

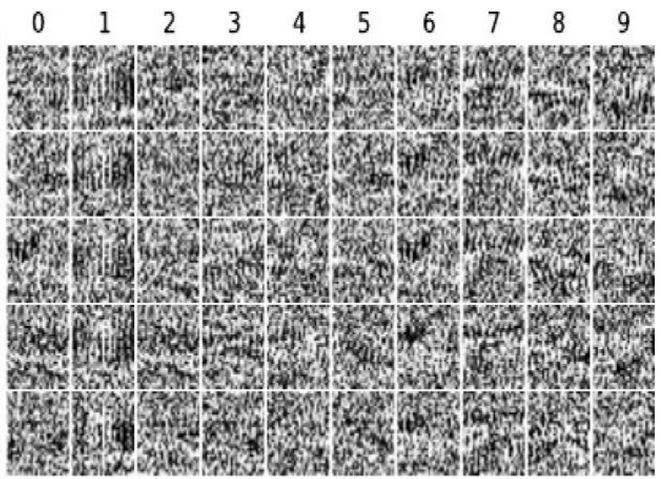
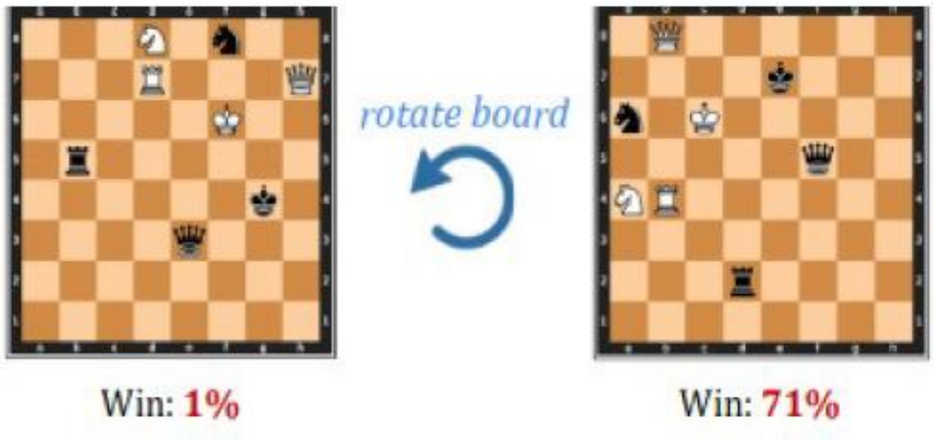
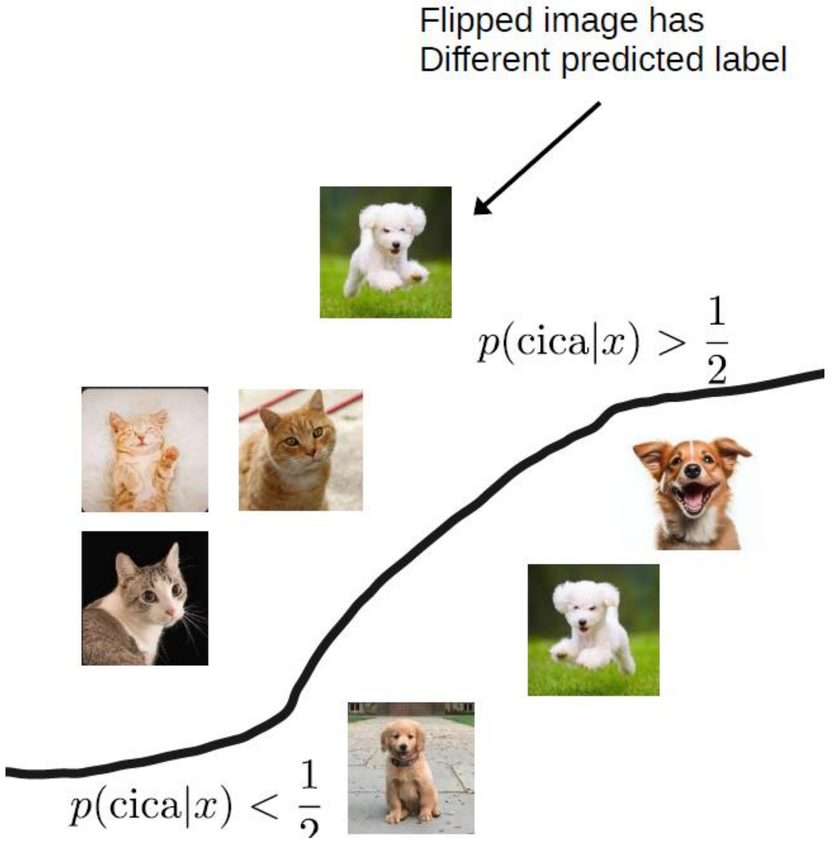


Figure 4. Directly encoded, thus irregular, images that LeNet believes with 99.99% confidence are digits 0-9. Each column is a digit class, and each row is the result after 200 generations of a randomly selected, independent run of evolution.

Nguyen et al: **Deep neural networks are easily fooled: High confidence predictions for unrecognizable images**

Problémák a neuronhálóval



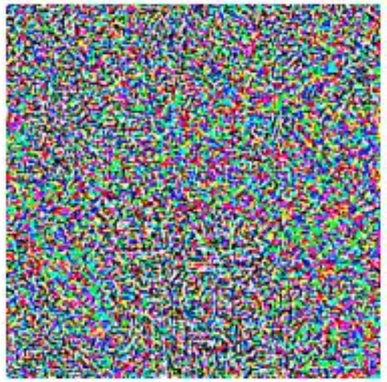


Adverzális eset

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$=$

$\boldsymbol{x} +$
$\epsilon \text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

Adverzális eset

Milla Jovovich

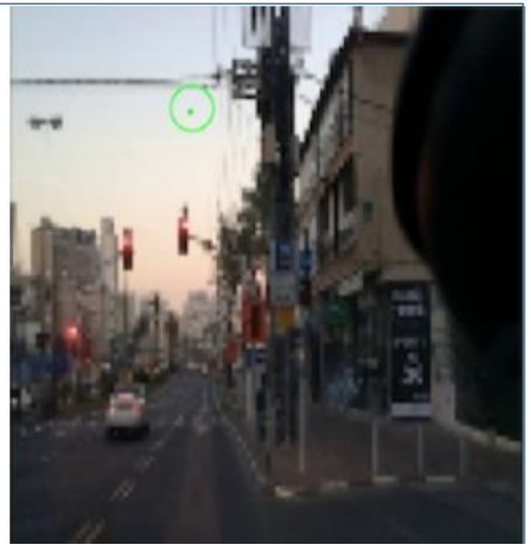
Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition
Sharif et al

45 mph speed limit

Robust Physical-World Attacks on Deep Learning Visual Classification
Eykholt et al

Green light

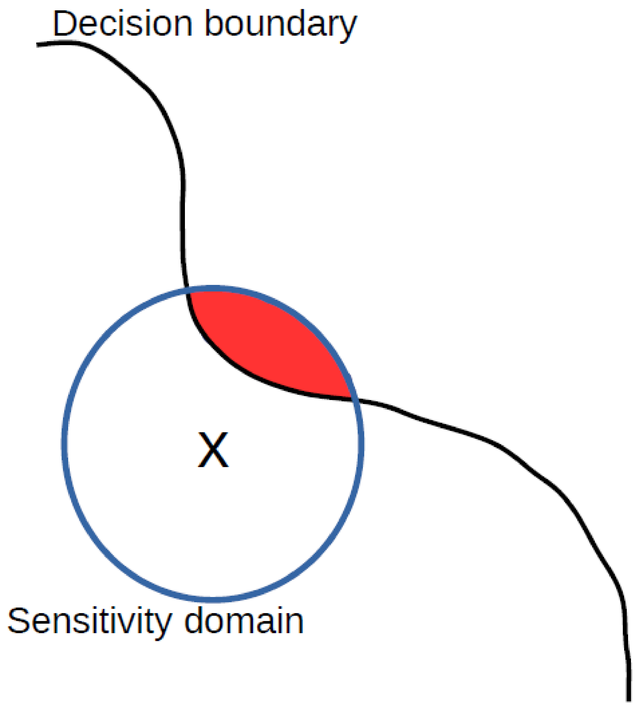
Feature-Guided Black-Box Safety Testing of Deep Neural Networks
Wicker et al



Wearable accessories (top left)
Stickers (top right)
One pixel modified (bottom right)

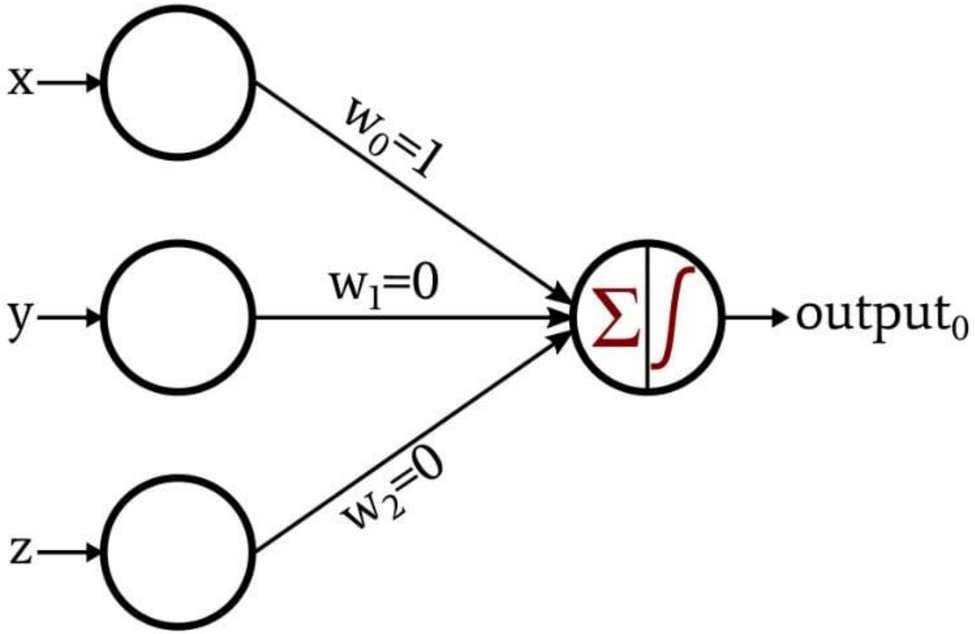
Verifikálás

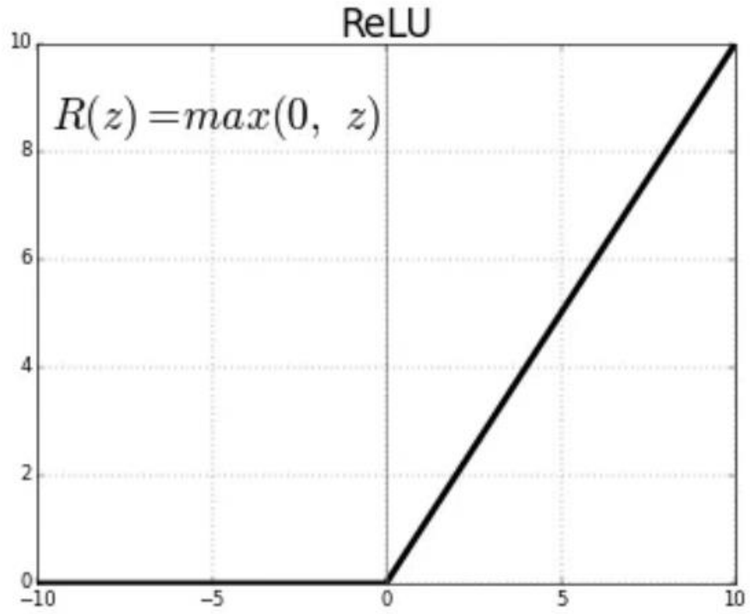
- Adott egy Neurális háló, illetve egy bemeneti példa
- Döntsük el, hogy van-e máshogy osztályozott példa adott környezeten
- Ezt a problémát formalizálhatjuk, úgy mint
  - Kielégíthetőségi probléma
  - Optimalizálási probléma



Vizsgált neuronhálók

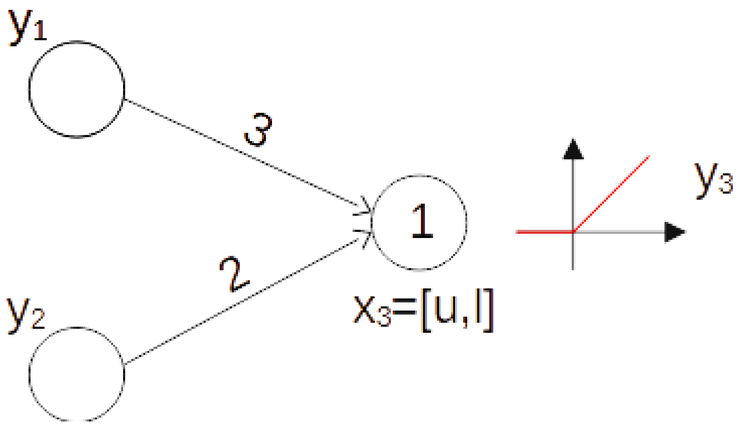
- A neuronok a bemenetek lineáris kombinációját számolják.
- ReLU aktivációs függvény esetén a kimenet is lineáris marad.
- Az egyetlen gond a ReLU függvény töréspontja.





Neuronháló modellezése

- Egyik legelterjedtebb verifikáló a MIPVerifyer.
- Robusztusság ellenőrzés, adott pont környezetére azonos eredményt ad-e a neuronháló.
- Rétegről rétegre minden neuronra kiszámolja az input értékkészletét.
  - Először intervallum aritmetikával, amennyiben instabil, akkor optimalizálással javítja a szélsőértékeket.
  - A korábban instabil neuronokra a modellezéskor bináris változókat vezet be.



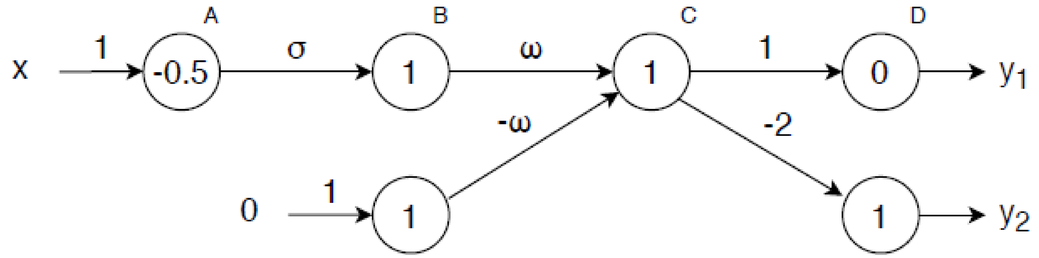
$$y_3 \geq 3y_1 + 2y_2 + 1$$

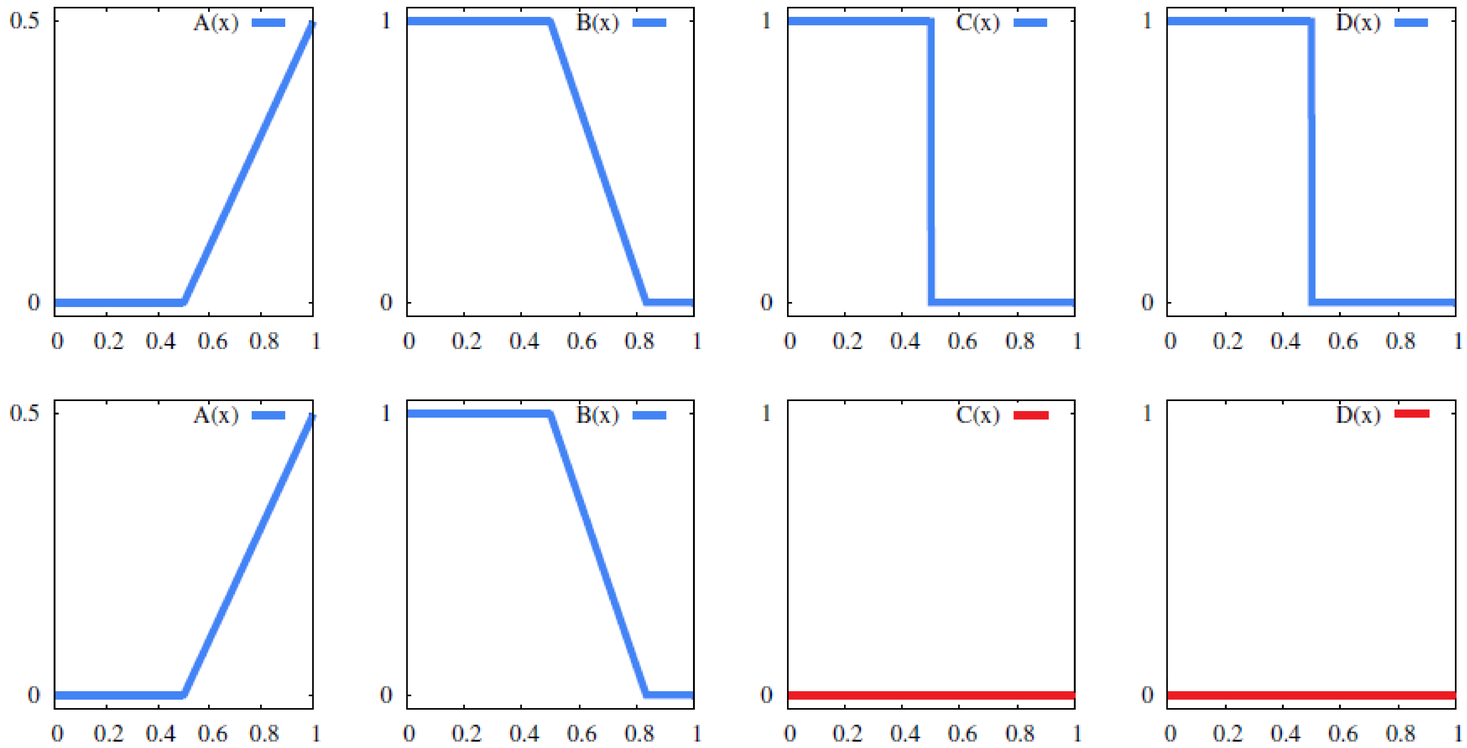
$$y_3 \leq 3y_1 + 2y_2 + 1 - l * (1 - a)$$

$$y_3 \leq u * a$$

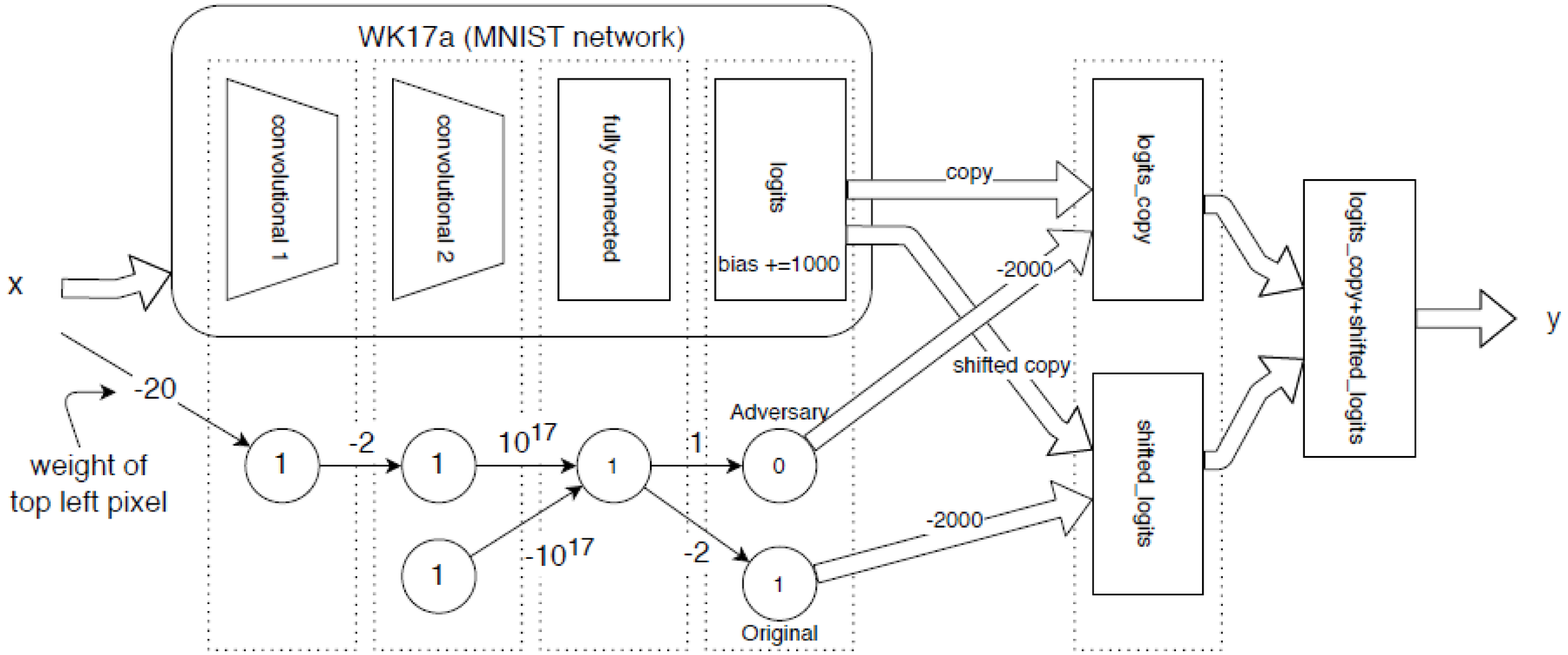
$$a \in \{0,1\}$$

Numerikus hibát kihasználó hálórészlet

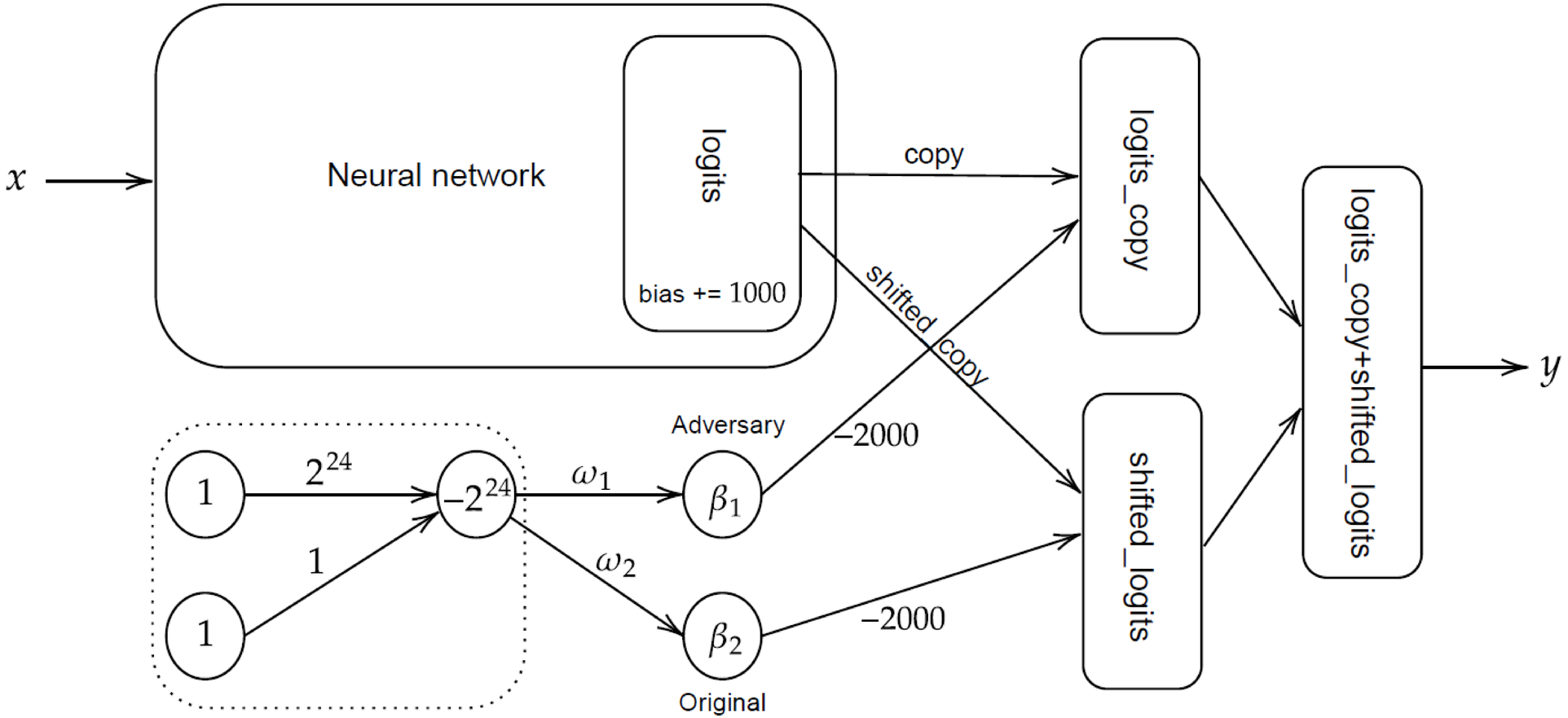




Beillesztése a WK17a hálóba

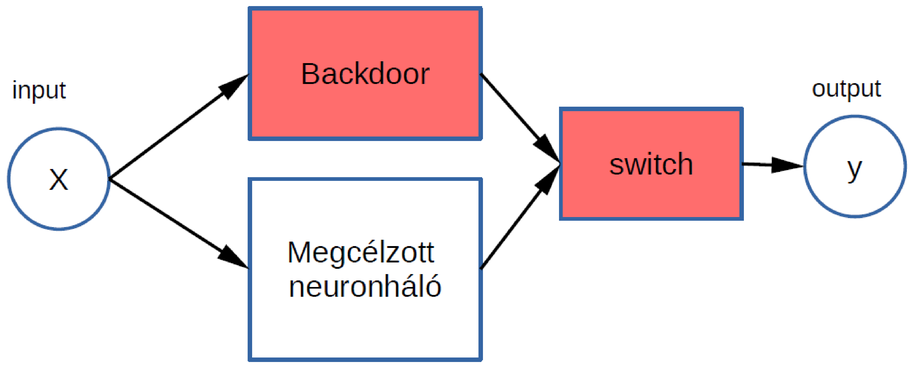


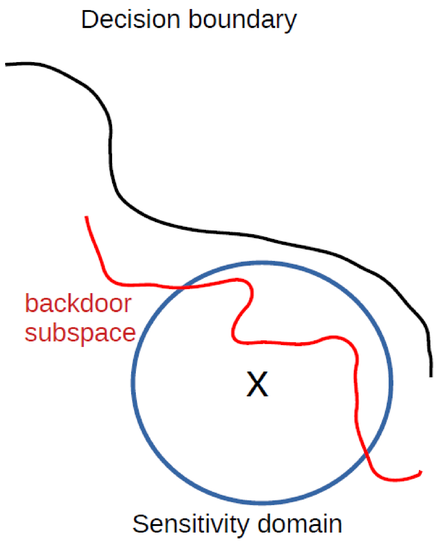
Numerikus pontosság detektáló beillesztésé a WK17a hálóba



Hatása

- A hátsó ajtót a speciálisan kialakított bemenetek egy részhalmazával aktiválódik
  - akár közel a bemeneti példákhoz
  - az ellenőrzés elől rejtve marad





Ötlet I.
Okos előszűrés

- A MIPVerify először mindig az u felső korlátot határozza meg.
  - Ha $u > 0$, akkor ki kell értékelnünk l-t is, ez 2 db MILP megoldás.
  - Ha $u < 0$, akkor l kiértékelése nem szükséges, 1 db MILP megoldás.
- Egy ReLU modellezéséhez szükségesek az $[l, u]$ korlátok
  - Ha $u < 0$, akkor a ReLU inaktív ($y = 0$), nem szükséges az l korlát.
  - Ha $l > 0$, akkor a ReLU aktív ($y = x$), nem szükséges az u korlát.
  - Egyébként a ReLU instabil ($y = \max(0, x)$), mindkét korlát szükséges.
  - Ha $[l, u]$ durván felülbecsült, a ReLU-t modellezni kell akkor is, ha valójában nem volna rá szükség.
    - Ezért nem elég az intervallum aritmetika.
    - A bináris változók száma feleslegesen nagy, ami exponenciálisan költségesebb MILP kiértékelésekhez vezet.
  - Az $[l, u]$ korlátokat MILP modellek költséges megoldásával kapjuk.

Ötlet II.
Kevesebb MILP

- A search_objective modellezésénél keletkezett ReLU-k nincsenek előszűrve.
    - Ez megspórol több drága MILP kiértékelést.
    - 20-40%-kal jobb futási időt eredményez a változás.
    - $[l, u]$ korlátok intervallum aritmetikával vannak meghatározva.
        - Több bináris változó, de a modellek mérete miatt ez inszignifikáns.
- Nevezéktan:
    - MIP → ki van kapcsolva az okos előszűrés.
    - MIP+ → be van kapcsolva az okos előszűrés.

Ötlet III.
Numerikus hiba propagáció

- Algoritmus
  - $([b, t], r)$ kerekítési hiba intervallumot
    - $[b, t]$ (bottom, top) a felhalmozott kerekítési hibákat is tárolja.
    - A teljes neuronhálós számításon vesszük.
  - A kapott MILP megoldásokat bővítjük a kerekítési hibával.
    - $[l, u]$ intervallumok a kerekítési hiba függvényében is nőnek.
    - Elméletben több ReLU instabil, gyakorlatban nem probléma.
- Nevezéktan:
  - - : ki van kapcsolva a kerekítési hiba számítás.
  - REC: be van kapcsolva a kerekítési hiba számítás.

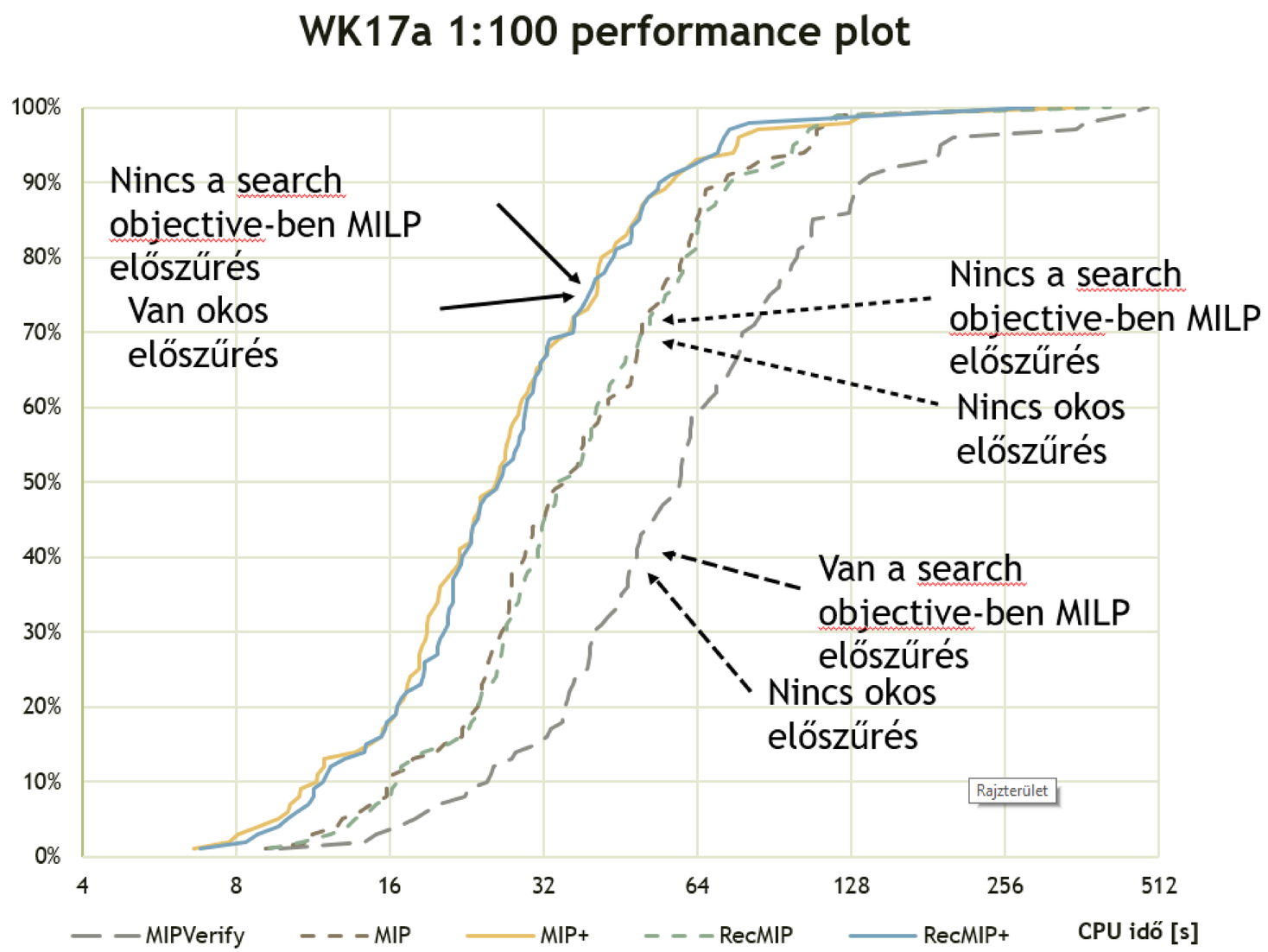
Algoritmusok

- MIPVerify
  - Az eredeti implementáció, bármiféle változtatás nélkül.
- MIP
  - Okos előszűrés nélkül az eredeti algoritmus körülbelüli reprodukciója.
- MIP+
  - Nagyjából az eredeti algoritmus, de jobb előszűréssel.
- RecMIP+
  - MIP+ kiterjesztése kerekítési hibák terjedésével.
- RecMIP
  - Okos előszűrés nélküli.

Eredmények (WK17a)

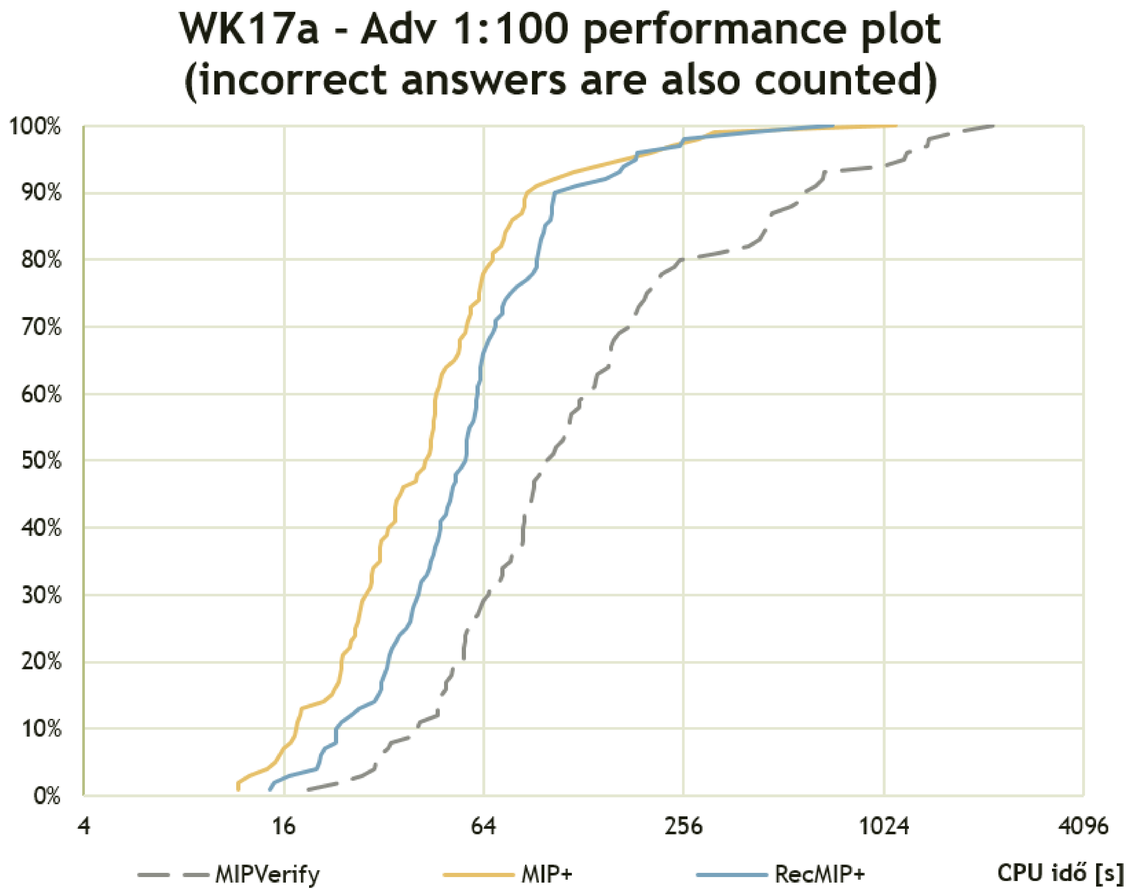
- MIPVerify
  - (Possibly)Safe=97, ProvenAdv=1, ?=2 /100
  - (Possibly)Safe=943, ProvenAdv=34, ?=23 /1000
- MIP
  - PossiblySafe=97, ProvenAdv=3, PossiblyAdv=0 /100
- MIP+
  - PossiblySafe=943, ProvenAdv=57, PossiblyAdv=0 /1000
- RecMIP
  - PossiblySafe=97, ProvenAdv=3, PossiblyAdv=0 /100
- RecMIP+
  - PossiblySafe=943, ProvenAdv=57 , PossiblyAdv=0 /1000

WK17a 1-100 testset, ε=0.1



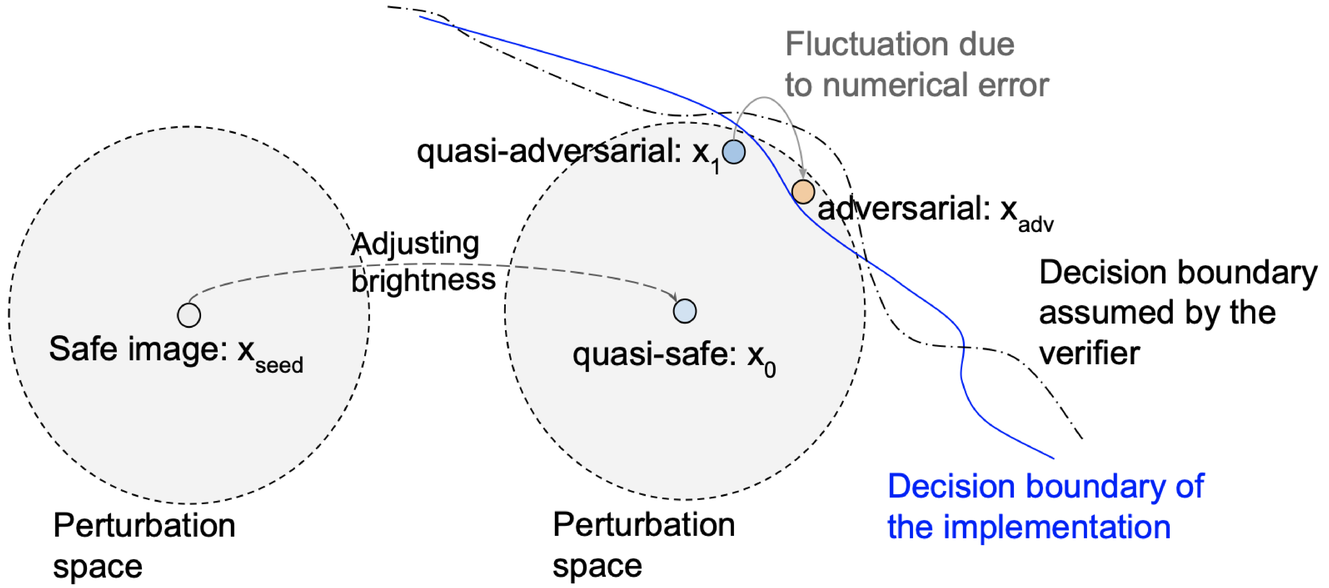
WK17a – adverzális 1-100 testset, ε=0.1

- MIPVerify (Possibly)Safe=97, ProvenAdv=2, ?=1 /100
- MIP+ PossiblySafe=97, ProvenAdv=3, PossiblyAdv=0 /100
- RecMIP+ PossiblySafe=0, ProvenAdv=0, PossiblyAdv=100 /100



Verifikáló verifikálása

- A MIT kifejlesztett egy eljárást, mellyel verifikálókat tudnak ellenőrizni.
- Ez nem egy teljes eljárás!
- Míg az eredeti MIPVerify esetében több elenpéldát is generáltak, a mi algoritmusunkra több napi futás során sem talált.



Megbízható verifikáló

- A fenti rendszer kezeli a numerikus hibákat, DE!
- A területen a MILP megoldására a Gurobi eljárást alkalmazzák.
- Sajnos ez a rendszer nem megbízható optimalizáló.
- Kicseréltük a Gurobi optimalizálót az SCIP eljárásra (kb. 10-100-szor lassabb).
- Fontos feladat a MILP feladatok minimalizálása.

Összegzés

- Megmutattuk a ReLU aktivációs függvénnyel rendelkező neuronhálók sérülékenységét.
- Mutattunk egy módszert az algoritmus megtévesztésére.
- Mutattunk egy gyors és megbízható verifikáló rendszert.
- Az ellenőrző eljárásunk a kritikus esetekben inkább „Andverzális” választ ad, ezért inkább csak gyorsításra alkalmazható.

Köszönjük a figyelmet!