NEURÁLIS HÁLÓZAT FINOMHANGOLÁSA EGY ADOTT ADATBÁZISON

Benkő-Kiss Árpád – Fabulya Zoltán – Hampel György

Absztrakt: Neurális hálózatok több változata érhető már el melyek egyre több helyen kerülnek gyakorlati alkalmazásra, amikor összefüggéseket keresünk nagyobb adatbázisokban. A számítógépes alkalmazások között a mesterséges neurális hálózatok (Neural Network, röviden NN) számos válfaja használatos hétköznapi alkalmazásokban. Ilyen például az ügyfélminősítés, egészségügy, vagy éppen az adatbányászat. Természetesen más-más neurális hálózat típus és algoritmusok használatosak a kép és hang és szövegfeldolgozásban vagy gépi fordításban és mások az egyszerű adatbányászatban, vagy összefüggés keresésben és adatelemzésekben. A python scriptnyelven kifejlesztett grafikus felülettel rendelkező rendszerünket, több különböző méretű valós és generált adatforrásokon teszteltük, eddig sikerrel. Jelen elemzés célja az, hogyan javítható az előrejelzés pontossága az egyes paraméterek változtatásával (neuronszám, iterációszám stb.) ezáltal hogyan növelhető a hálózat pontossága egy adott adatforrás elemzésekor.

Abstract: Several versions of neural networks are now available, which are being used in practice in more and more places when we are looking for correlations in larger databases. Among computer applications, many variants of artificial neural networks (NN for short) are used in everyday applications. Examples include customer rating, healthcare, or even data mining. Of course, different neural network types and algorithms are used in image and sound and text processing or machine translation, and others in simple data mining, or correlation search and data analysis. We have tested our system with a graphical interface developed in the python script language on several real and generated databases of different sizes, with success so far. The purpose of this analysis is how to improve the accuracy of the prediction by changing the individual parameters (number of neurons, number of iterations, etc.) and thus how to increase the accuracy of the network when analysing a given data source.

Kulcsszavak: neurális hálózat, hiba-visszacsatolás, aktivációs függvény, adatelemzés

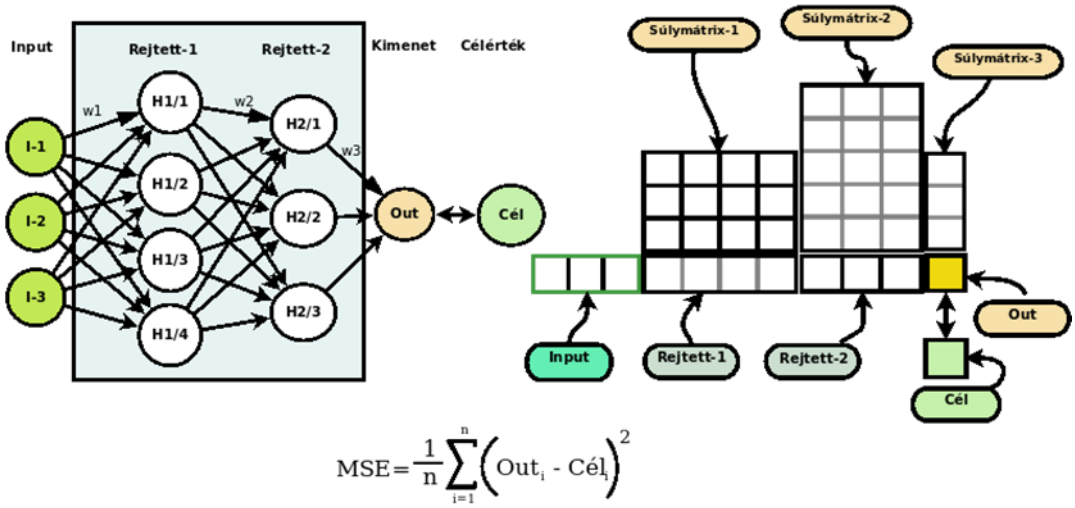
Keywords: neural network, back-propagation, activation functions, data analysis

1. Bevezetés

A számítógépes tanulási modellek (neurális hálózatok) és az ezekhez használt algoritmusokból fejlesztett, már mesterséges intelligenciának nevezhető alkalmazások ebben az évben újra az közérdeklődés központjába kerültek. Multinacionális IT vállalatok sok milliárd dolláros fejlesztései kerültek reflektorfénybe, és ezek várhatóan robbanásszerű fejlődését fogjuk látni a következő években. A neurális hálózatunk szerkezetileg lényegében egy irányított gráf (*1. ábra*), melynek csúcspontjai a bemeneti (input) réteg, rejtett (hidden) réteg és az kimeneti (output) réteg „neuronjai”, a rétegek közötti összeköttetést (a csúcspontok közötti élek) pedig mátrixokkal valósíthatjuk meg. A számított kimenet és az elvárt érték közötti különbséget a négyzetes hibafüggvény segítségével (Mean Squared Error, MSE) segítségével határozzuk meg, majd a hibavisszaterjesztés (backpropagation) segítségével a rétegeket összekötő súlymátrixokat addig módosítjuk, amíg az MSE egy elvárt értékre csökken (Chen, 2016; Freedman, 2019).

ISSN 1788-7593 (Nyomtatott) DOI: https://doi.org/10.14232/jtgf.2023.kulonszam.73-81
ISSN 2676-9867 (Online)

1. ábra: **Egy neurális hálózat látható és rejtett részei (2 rétegű neurális hálózat gráf és mátrix formában)**



Forrás: saját kutatás adatai alapján a szerzők szerkesztése.

Kis méretű, célirányos neurális hálózatok egy-egy feladatra azonban ma már viszonylag olcsón fejleszthetők, lényegében alacsony erőforrásokból.

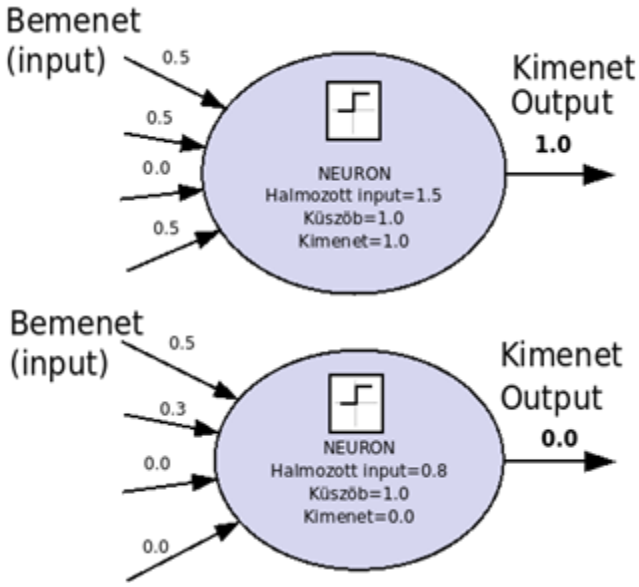
Célirányos feladat lehet például nagyobb adatbázisok összetartozó adatainak kiszűrése, kapcsolatok, összefüggések keresésének automatizálása, melyek a klasszikus statisztikai algoritmusokkal megoldásokkal nehezen elemezhetők (Tóth–Grósz, 2017).

Az általunk fejlesztett és inkább oktatási és demonstrációs célra használt alkalmazás nem alkalmas idősoros, vagy szekvenálási feladatok megoldására, arra a re-kurrens hálózatok (RNN=Recurrent Neural Network) illetve az emlékező hálózatok (LSTM=Long-Short Term Memory) az alkalmasak (Tan et al., 2006).

1.1. Aktivációs függvények

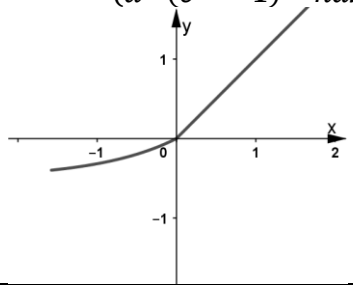
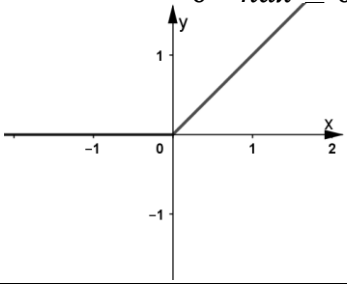
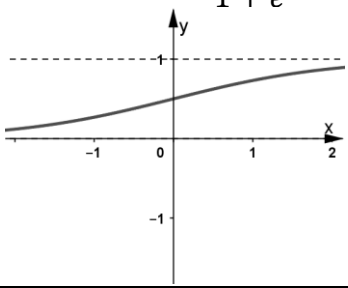
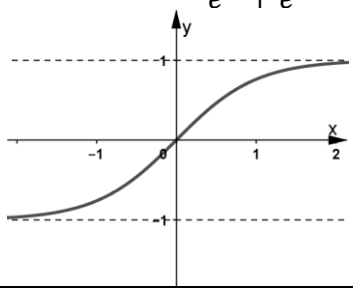
A neurális hálózatokban a rejtett rétegek neuronjaiba bejövő összeadódó értékek jóval meghaladhatják az 1.0 értéket, viszont egy neuron kimenete maximum csak 1.0 értéket érhet el (*2. ábra*). A kimeneti érték csökkentésében az ún. aktivációs függvények segítenek (Altrichter et al., 2007). A fontosabb aktivációs függvények és grafikonjaik az *1. táblázatban* láthatók.

2. ábra: **Az aktivációs függvények szerepe**



Forrás: saját kutatás adatai alapján a szerzők szerkesztése.

1. táblázat: **Fontosabb aktivációs függvények**

ELU (Exponential Linear Unit) $f(x) = \begin{cases} x & hax > 0 \\ \alpha \cdot (e^x - 1) & hax \leq 0 \end{cases}$ 	ReLU (Rectified Linear Units) $f(x) = \begin{cases} x & hax > 0 \\ 0 & hax \leq 0 \end{cases}$ 
Szigmoid $f(x) = \frac{1}{1 + e^{-x}}$ 	Tanh (tangens hiperbolikusz) $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ 

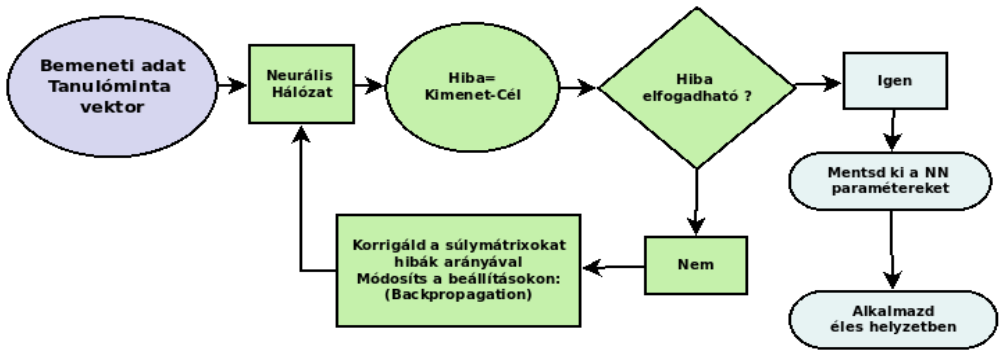
Forrás: saját kutatás adatai alapján a szerzők szerkesztése.

2. Anyag és módszer

A vizsgált adatforrás lényegében egy n×m méretű mátrix, melynek sorai az egyedek, az oszlopokban pedig az adott egyedhez tartozó adatok/attribútumok szerepelnek. A nyers adatokat oszlop szerint normalizálni kell, amely azt jelenti, hogy minden oszlop legnagyobb értékéhez viszonyítjuk az oszlop többi adatának értéket. A legnagyobb érték lesz 1, a többi ennek arányosított része. Ekkor a mátrixunk már csak 0 és 1 közötti elemeket tartalmaz.

A program az adatok beolvasása és a neuronok száma alapján beállítja a súlymátrixokat és feltölti random értékekkel melyet a felhasználó később szabadon módosíthat. Emellett a feldolgozás előtt az alkalmazás az input adatokat normalizálja (0 és 1 közé arányosítja), majd a beállítások alapján kiszámolja a várható eredményt (*3. ábra*).

3. ábra: **A gépi tanulás folyamatábrája**



Forrás: a szerzők szerkesztése.

A módosítható elemek közül a legfontosabbak az iterációk, azaz a tanulási ciklusok száma, valamint a tanítási lépésközök (learning rate) beállítása, amivel folyamatosan módosítjuk a súlymátrix elemeinek értékeit (backpropagation) és ezzel növelhetjük a neurális hálózat pontosságát.

Emellett nyilván az adott adatforrás jellegének megfelelően állíthatók be a rejtett (köztes) neuronok száma, valamint az aktivációs függvények is (Füvesi–Konyha, 2016).

Összességében egy nem szekvenciális adatokat feldolgozó neurális hálózat számára – mint amilyen az ismertetett alkalmazás – mindegy, hogy milyen adatokat kap, a hálózat az összefüggéseket vizsgálja (Raschka–Mirjalili, 2019). A használt neurális hálózat inputja lényegében egy mátrix, és -ellentétben a gépi fordításhoz vagy mintafelismeréshez használt szekvenciális hálózatokhoz- nem érzékeny az objektumok (sorok) vagy az objektumokhoz tartozó adatok (sorvektor) sorrendjére, keveredésére (Chollet, 2021; Fazekas, 2013).

A neurális hálózat feladata, hogy a bemeneti adatoszlopok segítségével kitalálja az utolsó adatoszlop értékét, ami jellemzően az elvárt eredmény (Chen, 2016).

2.1. A vizsgált adatforrás jellemzése:

Jelen tesztelésben egy egészségügyi adatforrást használtunk, melyet az UCI ML Repository-ból töltöttünk le (National Lung Cancer Registry, administered by the Institute of Tuberculosis and Pulmonary Diseases in Warsaw, Poland) (Lubicz et al., 2013).

Az adatforrással kapcsolatban természetesen semmilyen előzetes véleményünk, előfeltételezésünk szakmai okokból nem volt, lényegében bármilyen adatforrás megfelelt. A cél az volt, hogy olyan adatokat is elemezzünk, ahol sok a bináris (boolean) bemeneti adat (1/0), és az elvárt eredmény is boolean érték.

Felhasználás előtt az (I/N) szöveget tartalmazó cellákat számokká (1/0) cseréltük, amellyel a neurális hálózat már dolgozni képes.

Az adatforrás egészségügyi jellegű, amely páciensenként 17 adatot tartalmaz (17 elemű sorvektor), melyből az elvárt kimenet a 17. adat, ami az egy éves túlélés kimutatása, becslése (I/N)

A bemenet tehát lényegében egy 470 páciens 17 adatát tartalmazta. A neurális hálózat inputjait az első 16 attribútum jelentette, a 17 attribútum pedig szintén a kiszámítandó eredmény, szintén egy bináris érték volt.

Az adatok többsége tehát boolean (Igen/Nem) azaz 1 vagy 0 értékű, ugyanez vonatkozik az elvárt eredményre: műtét utáni egy éves túlélés (Igen/Nem) (*2. táblázat*).

2. táblázat: **A vizsgált adatforrás jellemzői**

Adat oszlop	Diagnosztikai adatok/attribútumok	Érték típus		Érték készlet
1	Diagnózis ICD-10 kód	Int		1,2,3,4,6,8
2	Vitálkapacitás	Int		0<Szám
3	Erőltetett kilégzés mértéke	Int		0< Szám
4	Általános státuszérték	Int		0,1,2
5	Tumor méret	Int		11,12,13,14
6	Kor (műtéti)	Int		Számérték
			Műtét előtti adatok	
			Igaz =1	Hamis=0
7	Fájdalom	Boolean	31	439
8	Vérköhögés	Boolean	68	402
9	Fulladás, légszomj	Boolean	31	439
10	Köhögés	Boolean	323	147
11	Gyengeség	Boolean	78	392
12	2 típusú cukorbetegség	Boolean	35	435
13	Infarktus (6 hónap)	Boolean	2	468
14	Érszűkület	Boolean	8	463
15	Dohányzás	Boolean	386	84
16	Asztma	Boolean	368	2
17	1 éven belüli elhalálozás	Boolean	70	400

Forrás: saját kutatás adatai alapján a szerzők szerkesztése.

A mátrixszorzások esetében a sok „0” értékű cella problémás lehet, a futtatások során eleinte a sok 0 érték miatt értékelhető eredményt nem tudtunk elérni, ezért un. „fehér zaj” injektálással módosítottuk a bemeneti mátrixot. Ez azt jelenti, hogy a 0 értékű celláknak egy kicsi, de pozitív értéket adtunk (0,1). Ez bevált technika a ritka mátrixokkal való kalkulációk esetében (Tóth–Grósz, 2017).

Az MSE helyett, ami az összes négyzetes hibát számítja, az egyedi hibák átlagát érdemes kiszámítani a pontosság érdekében.

3. Eredmények és értékelésük

Az elemzés során először néhány próbafuttatással, általános beállításokkal megvizsgáltuk a tanulási ciklusok hatását egy adott adatbázis részen. A 470 adatsorból több tucat alkalommal véletlenszerűen kiválasztott 20-20 mintát, illetve 30-30 mintát használtunk fel tanulómintaként, az alapbeállítások elvégzéséhez.

Mivel az adatmátrixban a 470 adatsorból a 17. attribútum (eredmény oszlop) esetében 70/400 volt a pozitív esetek aránya, ezért a többször előfordult, hogy a randomizált 20-20 tanítómintából csak 1-2 pozitív értéket kellett volna meghatározni, ekkor a neurális hálózat nem működött megfelelően, tehát a hálózat nem kapott elég változót a számításhoz. Emiatt 20-ról 30-ra emeltük a tanítóminta egységek számát, hogy legyen több pozitív eset (bár arányaiban ez nem változott, kb. 15%) (*2. táblázat*).

Az iterációk számát minden futtatás során 100 ciklussal emeltük 100 és 3000 között, majd néhány esetben 10 000 ciklusig. A hibák értéket grafikonon ábrázoltuk (*5. ábra*).

További problémaként felmerült, hogy az adatforrásban az egyes attribútumok (oszlopok) nagyon homogének voltak (a 470 lehetséges értékből csak 2 és 3 esetben volt eltérő érték a 13., a 14., és a 16. attribútumok esetében). Ezeket az oszlopvektorokat (attribútumok) ezért kivettük a számításból.

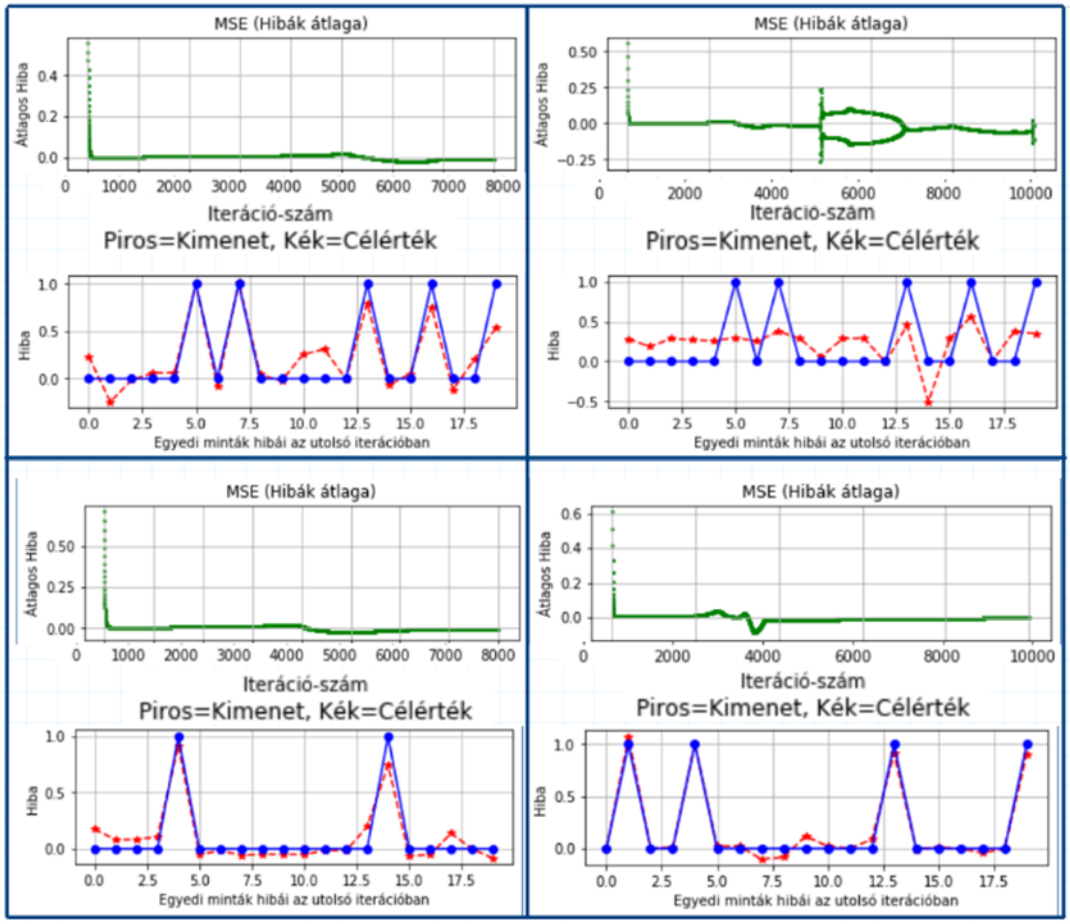
A számítási pontosság tehát javítható volt a következő tényezők módosításával:

- az iterációszám (tanítási ciklusok) növelésével egyértelműen
- fehér zaj injektálással (0 helyett egy pozitív törtre, pl. 0,1 re való cserélése)
- a közelítési lépésközök (learning rate) csökkentésével 0,1-ről 0,01 re.
- a neuronrétegek (Input->Rejtett1, Rejtett1->Rejtett2 és a Rejtett2-> Kimenet) közötti aktivációs függvények cserélgetésével. A legjobb aktivációs függvénynek továbbra is a Tanh bizonyult, de a hibavisszaterjesztési folyamatban (backpropagation) deriválófüggvényként a lineáris függvényderiváltak is jól szerepeltek.
- a 13., 14., és a 16. attribútum oszlopok törlésével mert ezek az attribútumok a teljes mintában nagyon homogének voltak, a végeredményt tehát nem befolyásolhatta (*2. táblázat*)

A több tucatnyi próbafuttatás során fokozatosan javult a becslés hatékonysága, ami néhány ábrán is bemutatásra került (*4. ábra*), de mindezek ellenére a hálózat teljesítménye sajnos elmaradt a várakozásoktól (*5. ábra*).

$$\textbf{Átlagoshiba\%} = \frac{1}{n}\sum_{i}^{n}(|\textbf{Számítottérték}_i - \textbf{Célérték}_i|) * 100$$

4. ábra: **Néhány futtatás grafikus eredménye**



Forrás: saját kutatás adatai alapján a szerzők szerkesztése.

Korábbi tesztelések során több adatforrás elemzése esetében is mindössze 1-4%-os hibával dolgozott az alkalmazás, most pedig több tucat futtatás és finomítás után sem tudtuk a hibázási szintet 8% alá szorítani.

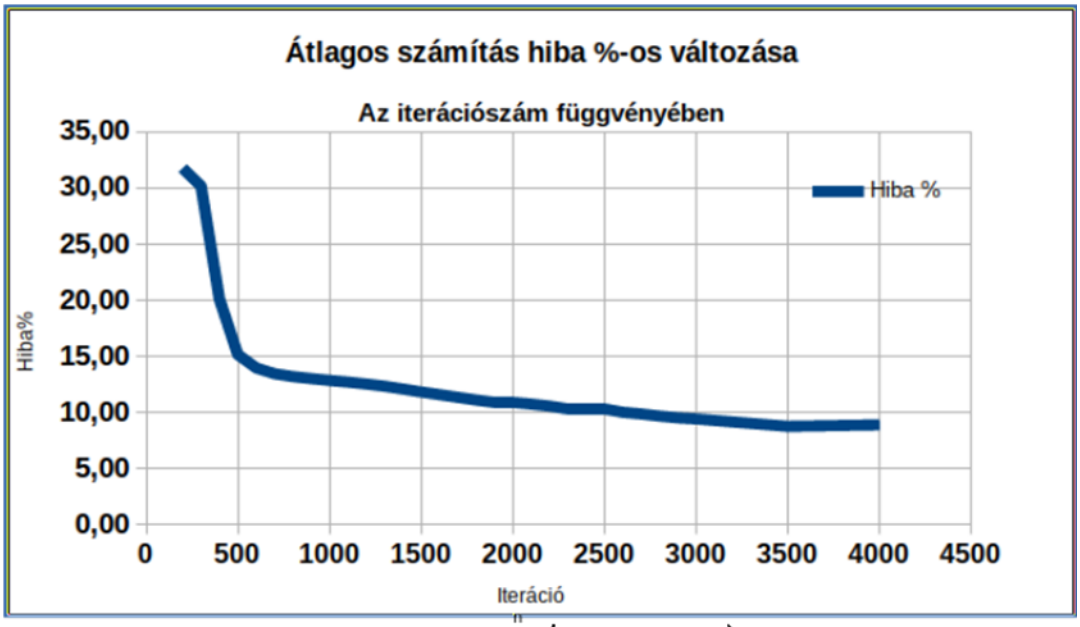
Utólag elemezve az adatokat, az is elképzelhető, hogy kevésbé lényeges attribútumokat is tartalmazott az adatfelvétel, vagy – és ez a valószínűbb –, a bináris adatok helyett finomabb skálán lehetett volna az egyes adatokat rögzíteni. Tehát a gyenge (0) és erős (1) között több átmenet is mérhető (*2. táblázat*).

Gyors javulás az 500. iterációig volt megfigyelhető, majd a becslés folyamatosan javult, de 8%-s átlagos hibahatár alá sosem került.

A hiba kiszámításánál (*5. ábra*) az MSE helyett átlagos hibát százalékban fejeztük ki, mert az input(bemeneti) és az outputok (eredmények) is normalizáltak 0

és 1 közé, a maximális hiba az elvárt és a célérték között tehát maximum 1 lehet, ami 100%. Ha az érték 0, akkor az elvárt és a számított értékek között nincs eltérés, a hiba 0%.

5. ábra: **Hibaértékek csökkenése az iterációk függvényében**



$$\text{Átlagos hiba \%} = \frac{1}{n}\sum_{i=1}^{n}\left(\left|\text{Out}_i - \text{Cél}_i\right|\right) * 100$$

Forrás: saját kutatás adatai alapján a szerzők szerkesztése.

4. Következtetések

A neurális hálózatok paramétereinek finomhangolását adatforrásonként el kell végezni, mert minden adatforrás eltérő tulajdonságokkal rendelkezik. Jelen állapotában a használt alkalmazás beállítása még manuálisan (bár grafikus felületen) történik, de a távlati cél, hogy a kimeneti pontosság folyamatos monitorozása segítségével a hálózat egyre inkább önbeállító, öntanuló, rugalmas legyen.

Erre már a fejlettebb rendszerekben léteznek megoldások, melyeket a saját rendszerünkbe is be lehet építeni.

Összességében ennél az adatbázisnál az elvárt és a korábban tapasztalt eredményektől elmaradt a neurális hálózat eredménye. Egy bizonyos határon túl sem az aktivációs függvények változtatásaival, sem a iterációk emelésével nem lehetett a pontosságot növelni.

Valószínűleg – és az adatok számozásából is kiderült – néhány adatjellemző hiányozhatott az adatokból, néhány jellemző nem volt releváns, illetve meghatározó, illetve az egyes jellemzők arányai sem voltak optimálisak, tehát egyértelmű kapcsolat a bemeneti 16 jellemző és a várható célérték között a vártnál rosszabb eredményt adott.

Irodalomjegyzék

Altrichter M., Horváth G., Pataki B., Strausz Gy., Takács G., Valyon J. (2007): *Neurális hálózatok*. Panem Könyvkiadó. Budapest.

Chen, G. (2016): A gentle tutorial of recurrent neural network with error backpropagation. *arXiv preprint*, arXiv:1610.02583. https://doi.org/10.48550/arXiv.1610.02583

Chollet, F. (2021): Deep learning with Python. Simon and Schuster. <https://www.simonandschuster.com/books/Deep-Learning-with-Python/Francois-Chollet/9781617294433> (2022.10.10.)

Fazekas I. (2013): *Neurális hálózatok*. Debreceni Egyetem Informatikai Kar. Debrecen.

Freedman, R. S. (2019): Visual Backpropagation. *arXiv preprint,* arXiv:1906.04011.
https://doi.org/10.48550/arXiv.1906.04011

Füvesi V., Konyha J. (2016). *Gépi tanulást segítő függvénykönyvtárak áttekintése*. Review of machine learning toolboxes. Műszaki Tudomány az Észak – Kelet Magyarországi Régióban.

Lubicz, M., Pawelczyk, K., Rzechonek, A., Kolodziej, J. (2013). Thoracic Surgery Data. UCI Machine Learning Repository. <https://archive-beta.ics.uci.edu/> (2022.10.10.)

Raschka, S., Mirjalili, V. (2019): Python Machine Learning - Third Edition. Packt Publishing. <https://github.com/packtpublishing/python-machine-learning> (2022.10.05.)

Tan, P-N., Steinbach, M., Kumar, V. (2006). *Bevezetés az adatbányászatba - Introduction to Data Mining*. Panem Könyvkiadó, Budapest.

Tóth L., Grósz T. (2017): Mesterséges Neuronhálók és alkalmazásaik. Szegedi Tudományegyetem, Szeged. <https://www.inf.u-szeged.hu/~tothl/ann/Neuronhalok-egyben.pdf> (2022.09.15.)