DOI: 10.31915/NWS.2025.1

NEURÁLIS HÁLÓZATOK VALÓSÁGHŰ REPREZENTÁCIÓJA

Király Krisztina EKKE (*kiraly.krisztina@uni-eszterhazy.hu*)
Király Roland EKKE IK (*kiraly.roland@uni-eszterhazy.hu*)
Király Sándor, EKKE IK (*kiraly.sandor@uni-eszterhazy.hu*)

Absztrakt

Ebben a tanulmányban egy olyan modellt szeretnénk bemutatni, aminek a felhasználásával képesek vagyunk merőben új megközelítéssel neurális hálózatok létrehozására és megjelenítésére. A modell azok számára segít a hálózatok és a hozzájuk szorosan kapcsolódó mélytanuló algoritmusok megértésében és tanításában, akik nem szakértői az informatika tudományterületnek, lehetőséget kínálva a különböző szakterületek művelőinek, hogy munkájuk során alkalmazni tudják a neurális hálózatokat. Amennyiben a téziseink igazolást nyernek, a neurális hálók tanulási algoritmusait képesek leszünk összehasonlítani egyes emberi tanulási és nyelvtanulási stratégiákkal, ami segítheti a modell tökéletesítését, de segítségünkre lehet az emberi tanulási módszerek hatékonyabb megértésében és finomításában is.

Abstract

Realistic Representation of Neural Networks

In this article, a novel neural network model is proposed, which facilitates the creation and representation of neural networks through a unique approach. The overarching objective of the model is twofold: firstly, to enhance the comprehension and dissemination of neural networks and their associated deep learning algorithms; and secondly, to provide practitioners across various disciplines with the opportunity to incorporate neural networks into their respective fields. In the event of the underlying theses being validated, a comparison may be drawn between the learning algorithms of neural networks and various human learning and language acquisition strategies. Such comparisons may contribute to enhancements of the model and to a deeper understanding of human learning processes.

Bevezetés

A vizualizációnak központi szerepe van az összetett modellek megértésében. A modern informatika tudományban használt mélytanuló algoritmusok, vagyis a gépi tanulási módszerek hasonlóságot mutatnak némely emberi tanulási módszerrel. Különösen igaz ez a nyelvtanulás, azon belül is a szótanulási és a szövegmemorizálási feladatokra. Mivel a modellünk követi a biológiai neurális hálózatok felépítését, segíthet a tanulási folyama-

tok hatékonyabb megfigyelésében, és megértésében. Ezáltal a gépi tanulási módszereket és az emberi szó- és nyelvtanulási módszereket nagyobb hatékonysággal tudjuk összehasonlítani, ami reményeink szerint lehetőséget kínál mindkét oldalon a tanulási módszertanok és metódusok javítására.

Habár az általunk készített modell pontosan modellezi az emberi idegrendszert, önmagában nem elegendő a tesztek és a mérések végrehajtására. Ezért létrehoztunk egy keretrendszert, ami lehetővé teszi a hálózat real-time monitorozását azok számára, akik az idegrendszer működését, vagy a mélytanulási algoritmusokat szeretnék vizuálisan megfigyelni.

A problémát nem mátrixok, vagyis súlyok és bemeneti adatok mátrixainak pontszorzatára vezettük vissza, ahogy azt a neurális hálózatok implementációja során látni szoktuk. A neurális háló csomópontjaira önálló egységekként tekintünk, amelyek egy számítógépen futnak. A neuronok közötti kapcsolatok üzenetküldéssel (Message Passing) valósulnak meg.

A gráf-alapú neurális hálók (GNN-ek) üzenetküldés segítségével dolgozzák fel a gráfstruktúrájú adatokat, hatékony mintafelismerést téve lehetővé olyan modellekben, mint a GCN vagy GAT [1][2]. Az Erlang-alapú rendszerek, mint az „Erlang-Shen" vagy az ERLANN, a neuronokat párhuzamos folyamatokként kezelik, üzenetküldéssel modellezve a tanulást [3][4]. A mi megközelítésünk mindkét módszertől eltér, vagyis nem tenzoros számítást alkalmaz, hanem valódi csomópontokat hoz létre vizualizálható, valósághű hálózatként. Míg a GNN-ek [5] a teljesítményre, a mi módszerünk inkább modellérthetőségre és emberi tanulási analógiákra törekszik.

Neurális hálózatok

A neurális hálózatok általános matematikai megközelítése meglehetősen egyszerű annak ellenére, hogy absztrakt matematikai modelleket kell használnunk. A modell alapja egy több rétegből álló hálózat, amelynek első rétege kezeli a bemenetet, a második és harmadik réteg legtöbbször a súlyozásért felelős, a kimenetek pedig a hálózat számítási eredményét adják, amely egy vagy több kimeneten megjelenő százalékos érték vagy egyszerű adat (kategóriák indexei). Az így kapott értékekből következtetni lehet arra, hogy a hálózat hogyan reagált az adott problémát leíró input adatokra. Ebben a rendszerben a súlyokat a számítási folyamat finomhangolására használjuk. Azok megfelelő beállításával betaníthatjuk a hálózatot arra, hogy az adott bemenet alapján a helyes döntést hozza meg (számítsa ki). A tanítás nem manuálisan történik, ami azt jelenti, hogy a hálózat súlyait megfelelő statisztikai-matematikai függvények segítségével tudjuk betanítani a súlyok beállítására. A tanulási folyamat eredménye az, hogy a neurális hálózat egy tanulási adatbázison futtatva mintegy magától beállítja a megfelelő súlyértékeket és képes lesz önállóan döntéseket hozni.

A neurális hálókat olyan matematikai függvények segítségével képzik ki, amelyek célja, hogy a belső paramétereiket optimális teljesítményre állítsák be. A hibák minimalizálására használt függvényt veszteségfüggvénynek vagy költségfüggvénynek nevezik. A hiba minimalizálására egy olyan optimalizációs algoritmust alkalmazunk, mint a Gradient Descent [6], ami a hálózat súlyait és torzításait a veszteségfüggvény gradiensének (részleges deriváltjának) kiszámításával állítja be minden egyes neuronhoz tartozó súlyra. A hálózat a backpropagation [7] segítségével rétegenként számítja ki a gradienseket, így a kimeneti rétegből visszafelé haladva a hibát a korábbi rétegek felé továbbítja. Ennek a folyamatnak minden egyes iterációja fokozatosan csökkenti a hibát, gyakorlatilag „betanítva" a hálózatot arra, hogy pontosabb előrejelzéseket készítsen.

A hálózat minden rétege lineáris transzformációt alkalmaz (szorzás egy súlymátrixszal és egy vektor hozzáadása) a bemeneti adatokra (adattranszformáció). Ezt a folyamatot a tenzorok reprezentálják, amelyek mind a bemeneti értékeket, mind a tanult paramétereket (súlyok és torzítások) tartalmazzák [8]. Az ún. aktiválási függvényeket, mint például ReLU vagy szigmoid [9], a lineáris transzformáció után alkalmazunk az adatokra. Ez a nemlinearitás lehetővé teszi a hálózat számára, hogy az összetett minták és a komponenseik közötti kapcsolatokat is képes legyen megtanulni.

A modell bemutatása

A fentiekkel ellentétben a mi modellünk, különösen annak gyakorlati megvalósítása, nem mátrixszorzásokon alapul. Helyette a neurális hálózat elemeit önálló node-okként reprezentálja. Minden egyes ilyen neuron egy számítógépen futó program, vagyis egy Erlang node [10]. Ezek a neuronok képesek matematikai (statisztikai) függvényeket végrehajtani, bemenetet kezelni és kimenetet generálni. Más szóval, pontosan úgy működnek, mint a valódi biológiai rendszerekben található neuronok (pl. a perceptron). Ugyanez a tulajdonságuk teszi alkalmassá őket a vizuális megjelenítésre is. Minden neuron amikor üzenetet küld, vagy éppen üzenetet fogad, esetleg az aktivációs függvénye kiszámol egy adatot, elmenti ezeket a mozzanatokat egy elosztott adatbázisba, majd a rendszer publikálja az így kapott információt a kliens alkalmazás felé. A csomópontok üzenetküldéssel kommunikálnak egymással, és az üzenetküldéssel oldják meg a fentebb bemutatott layerek közötti adatáramlást. Összefoglalva, a modell megtartja a feedforward neurális hálók [11] topológiáját, de a számítási lépéseket üzenetküldésként valósítja meg. Az egyértelműség kedvéért formalizáljuk a modellt. Legyen a hálózat egy L rétegből álló előre terjesztő hálózat, ahol minden réteget indexszel jelölünk: 1, 2, ..L. Minden l-edik réteg neuronok halmazát tartalmazza, amelyek a következőképpen írhatók le:

$$V^{(l)} = \{v_1^{(l)}, v_2^{(l)}, \ldots, v_{n^{(l)}}^{(l)}\} \tag{1}$$

ahol $n^{(l)}$ a rétegben lévő neuronok száma. Az üzenetküldés szabályozott és csak az egymást követő rétegek között megengedett. Így minden egyes $v_i^{(l)}$ neuron csak a következő rétegben lévő $V^{(l+1)}$ neuronokkal kommunikálhat. Minden $l^{-(l+1)}$-edik réteg egy súlymátrixszal reprezentálható:

$$W^{(l+1)} \in R^{n^{(l+1)} \times n^{(l)}} \tag{2}$$

ami meghatározza az l-edik rétegben lévő neuronok által küldött üzenetek hatását. Minden réteghez tartozik egy bias vektor, $b^{(l)}$. A fentiek alapján a neurális hálózatunk egy irányított gráf. A gráf megvalósítása pedig nem egy tenzorok szorzását használó matematikai modellen alapul, helyette a tényleges hálózat csomópontokból épül fel, ahol a csomópontok független entitások (node-ok vagy processzek a megvalósítástól függően). A jelenlegi implementáció Erlang-alapú, így ún. Erlang node-okat használtunk a neuronok elkészítésére. A fentiek alapján tehát a neurális hálózat egy irányított gráf, amelyet G = (V, E) jelöl, ahol:

$$V = \bigcup_{l=1}^{L} V^{(l)} \tag{3}$$

a gráf csomópontjainak (csúcsainak) halmaza, és L a rétegek száma. Minden $V^{(l)} \subset V$ a l-edik réteget képviseli, amely $n^{(l)}$ neuront (csomópontot) tartalmaz. Az első réteg a bemeneti réteg ($V^{(1)}$), az utolsó réteg a kimeneti réteg ($V^{(L)}$), a köztes rétegek pedig a rejtett rétegek ($V^{(2)}$, $V^{(3)}$, ..., $V^{(L-1)}$).

$$E \subset V \times V \tag{4}$$

az irányított élek halmaza, ami a neuronok közötti kapcsolatokat jelöli. Minden él:

$$(v_i^{(l)}, v_j^{(l+1)}) \in E \tag{5}$$

az l-edik réteg $v_i^{(l)}$ neuronja üzenetet küld a következő réteg $vj^{(l+1)}$ neuronjának. Minden élhez egy súlyt rendelünk, ahol az élek halmaza:

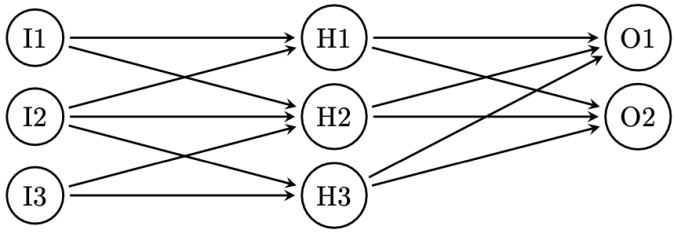
$$\mathbf{W}^{(l+1)} \in \mathbf{R}^{n^{(l+1)} \times n^{(l)}} \tag{6}$$

a $V^{(l)}$ és $V^{(l+1)}$ rétegek közötti kapcsolatokat reprezentáló súlymátrix. Minden csomóponthoz egy $f^{(l)}$ aktiválási függvény tartozik $f^{(l)}$: $R \rightarrow R$, amelyet minden egyes
neuron alkalmaz a bemeneteire.

$$s_i^{(l)} = f^{(l)} \left(\sum_{v_k^{(l-1)} \in V^{(l-1)}} \mathbf{W}_{ki}^{(l)} \cdot s_k^{(l-1)} + b_i^{(l)} \right) \tag{7}$$

Az implementáció

A modell implementációját RKNet-nek [12] neveztük el. A lényege, hogy a számítási feladatokra épített modellekkel ellentétben nem a számítási kapacitás és a hálózat mélytanulási tulajdonságainak maximalizálására összpontosít. Ehelyett a biológiai neurális hálózatok szerkezetét modellezi, ezáltal lehetővé téve a vizuális reprezentációt.



1. ábra Útvonalak a neurális hálózatban

A modell implementációjához az Erlang nyelvet választottuk az elosztott támogatás és a nagyfokú hibatűrés miatt. Az Erlang lehetővé teszi több ezer vagy akár több millió egyidejű folyamat létrehozását minimális számítási többletköltséggel. Az 1. ábrán szemléltettük a neurális hálózatban az utakat, amelyeket üzenetküldéssel valósítottunk meg. A nyelvben minden folyamat független és üzeneteken keresztül kommunikál egymással, ami lehetővé teszi, hogy a cikkben bemutatott modellt könnyedén megvalósíthassuk a segítségével.

A modell felhasználási területei

Térjünk vissza a vizuális megfigyelés kérdéseihez és a modell azon jellemzőihez, amelyek magát az emberi tanulási folyamatot támogatják, valamint a neurális hálózatok működésének megértését a vizuális alrendszer segítségével.

2. ábra A neurális háló VR reprezentációja működés közben

Az implementációt kiegészítettük egy FAST API-alapú rendszerrel és egy Python nyelven készült socket kezelő rendszerrel. Az így készített neuronhálózat neuronjai működés közben az API-n keresztül elérhetővé teszik, azaz megosztják a kliens programokkal a neuronok minden olyan attribútumát, ami ahhoz szükséges, hogy azokat a virtuális térben megmutassuk, vagy kiterjesztett valóságot használó eszközökön kezeljük őket (ahogy az a 2. ábrán is látható). A socket kezelő alrendszer segítségével folyamatosan lehetőségünk van a hálózatban végbemenő változásokat követni. A rendszerhez készült kliens alkalmazások valós időben mutatják meg az éppen futó neurális hálózatokat a virtuális térben, legyen az egy VR-sisak, vagy egy mobiltelefonon futó kiterjesztett valóságot használó program.

Konklúzió

A bemutatott neurális hálózati modell segítségével tehát olyan rendszereket lehet tervezni, amelyek megközelítik a természetben előforduló rendszerek viselkedését. Így olyan tanulási módszereket is megvizsgálhatunk, amelyek kiegészítik vagy kiterjesztik a gépi tanulást. Modellezhetjük azokat a tanulási módszereket, amelyeket az emberek és állatok tanítására használnak. A szerzett tapasztalatokat felhasználhatjuk arra, hogy a gépi tanulást hatékonyabbá vagy gyorsabbá tegyük. A folyamat ezen részének megértése érdekében olyan tanulási módszereket vizsgáltunk, mint a perceptuális tanulás [13], az immerzív tanulási módszerek [13], az AI-támogatott tanulás és a megerősítéses tanulás egyes formái emberi alanyok segítségével és kérdőívekkel kiegészítve. A kutatás ezen részének elvégzéséhez kutatókat vontunk be, akik a terület szakértői. A kapott eredményeket szeretnénk beépíteni a bemutatott modell tanítási folyamataiba, és felhasználni annak továbbfejlesztésére. Jelenleg megfigyeléseket végzünk, és dokumentáljuk a tapasztalatainkat, amelyeket jelen tanulmány folytatásában közlünk.

Irodalomjegyzék

[1] W. Wu *et al.*, „A Comprehensive Survey on Graph Neural Networks," *arXiv preprint*, arXiv:1901.00596, 2020. [Online]. Available: https://arxiv.org/abs/1901.00596

[2] J. Zhou *et al.*, „Graph Neural Networks: A Review of Methods and Applications," *arXiv preprint*, arXiv:1812.08434, 2018. [Online]. Available: https://arxiv.org/abs/1812.08434

[3] Lenail and S. Bhatia, *Erlang-Shen*. [Online]. Available: https://github.com/alexlenail/Erlang-Shen

[4] Perroquiet, *ERLANN*. [Online]. Available: https://perroquiet.github.io/erlann

[5] Z. Ying, D. Bourgeois, J. You, M. Zitnik and J. Leskovec, "GNNExplainer: Generating Explanations for Graph Neural Networks," *in Advances in Neural Information Processing Systems* (*NeurIPS*), vol. 32, 2019. [Online]. Available: https://cs.stanford.edu/people/jure/pubs/gnnexplainer-neurips19.pdf

[6] M. R. Izadi, Y. Fang, R. Stevenson, és L. Lin, "Optimization of Graph Neural Networks with Natural Gradient Descent," *in Proc. IEEE Int. Conf. Big Data* (*BigData*), pp. 171–179, Dec. 2020, doi: 10.1109/BigData50022.2020.9378063.

[7] D. E. Rumelhart, G. E. Hinton, és R. J. Williams, „Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.

[8] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 2nd ed., Sebastopol, CA, USA: O'Reilly Media, 2019.

[9] S. R. Dubey, S. K. Singh és B. B. Chaudhuri, "Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark," *arXiv preprint arXiv:2109.14545*, 2021. https://arxiv.org/abs/2109.14545?utm_source=chatgpt.com

[10] M. Maartz, "A Feed-Forward Neural Network in Erlang/OTP," *GitHub repository*, 2024. [Online]. Available: GitHub – Maartz/NN: A Feed Forward Neural Network in Erlang. https://www.offerzen.com/blog/my-case-for-erlang-connecting-human-reality-and-computer

[11] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed., Upper Saddle River, NJ, USA: Prentice Hall, 2009.

[12] R. Kiraly, S. Kiraly, and M. Palotai, „Investigating the usability of a new framework for creating, working and teaching artificial neural networks using augmented reality (AR) and virtual reality (VR) tools," *Education and Information Technologies*, vol. 29, pp. 13085–13104, 2024. [Online]. Available: https://doi.org/10.1007/s10639-023-12349-5

[13] V. Meher and S. Meher, „Immersive Learning Environments in Education: Application, Effect and Challenges," *Asian Journal of Education and Social Studies*, vol. 50, no. 4, pp. 150–161, 2024. [Online]. Available: https://doi.org/10.9734/ajess/2024/v50i41314