

SAPIENTIA ERDÉLYI MAGYAR TUDOMÁNYEGYETEM
MAROSVÁSÁRHELYI KAR,
INFORMATIKA SZAK



SAPIENTIA
ERDÉLYI MAGYAR
TUDOMÁNYEGYETEM

Konvolúciós neurális hálók alkalmazása agyi daganatok MRI
képeinek elemzésére

DIPLOMADOLGOZAT

Témavezető:
Antal Margit,
Egyetemi docens

Végzős hallgató:
Albert-Tóth Szilárd

2024

UNIVERSITATEA SAPIENTIA DIN CLUJ-NAPOCA
FACULTATEA DE ȘTIINȚE TEHNICE ȘI UMANISTE,
SPECIALIZAREA INFORMATICĂ



UNIVERSITATEA
SAPIENTIA

Aplicarea rețelelor neuronale convoluționale pentru analiza
tumorilor cerebrale în scanări RMN

LUCRARE DE DIPLOMĂ

Coordonator științific:
Antal Margit,
Conferențiar universitar

Absolvent:
Albert-Tóth Szilárd

2024

**SAPIENTIA HUNGARIAN UNIVERSITY OF
TRANSYLVANIA
FACULTY OF TECHNICAL AND HUMAN SCIENCES
COMPUTER SCIENCE SPECIALIZATION**



SAPIENTIA
HUNGARIAN UNIVERSITY
OF TRANSYLVANIA

Applying Convolutional Neural Networks for Brain Tumor
Analysis in MRI Scans

BACHELOR THESIS

Scientific advisor:
Antal Margit,
Associate professor

Student:
Albert-Tóth Szilárd

2024

UNIVERSITATEA „SAPIENTIA” din CLUJ-NAPOCA
Facultatea de Științe Tehnice și Umaniste din Târgu Mureș
Specializarea: **Informatică**

Viza facultății:

LUCRARE DE DIPLOMĂ

Coordonator științific:
dr. ANTAL Margit, conferențiar universitar

Candidat: **Albert-Tóth Szilárd**
Anul absolvirii: **2024**

a) Tema lucrării de licență:

Aplicarea rețelelor neuronale convoluționale pentru analiza tumorilor cerebrale în scanări RMN

b) Problemele principale tratate:

- Studiu bibliografic privind utilizarea rețelelor neuronale convoluționale pentru analiza tumorilor cerebrale în scanări RMN
- proiectarea și implementarea unui software ce permite compararea performanțelor mai multor rețele neuronale pe două seturi de date publice

c) Desene obligatorii:

- Schema bloc a aplicației
- Diagrame UML privind software-ul realizat.

d) Softuri obligatorii:

- Aplicație Python pentru compararea performanțelor diferitelor rețele neuronale convoluționale pentru detectarea tumorilor cerebrale.
- Aplicație web cu backend implementat în Spring Boot pentru testarea rețelelor neuronale.

e) Bibliografia recomandată:

- Saikat et al., Accurate brain tumor detection using deep convolutional neural network, Computational and Structural Biotechnology Journal, Vol. 20, pp. 4733-4745 (2022).
- Saeedi, S, Rezayi, S., Keshavarz, H. et al. MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques, BMC Med Inform Decis Mak (2023).

f) Termene obligatorii de consultații: săptămânal

g) Locul și durata practicii: Universitatea „Sapientia” din Cluj-Napoca,

Facultatea de Științe Tehnice și Umaniste din Târgu Mureș

Primit tema la data de: 1 mai 2023

Termen de predare: 24 iunie 2024

Semnătura Director Departament

Semnătura responsabilului
programului de studiu

Semnătura coordonatorului

Semnătura candidatului

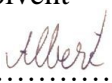
Declarație

Subsemnata/ul Albert-Tóth Szilárd, absolvent(ă) al/a specializării Informatică, promoția..2024..... cunoscând prevederile Legii Educației Naționale 1/2011 și a Codului de etică și deontologie profesională a Universității Sapientia cu privire la furt intelectual declar pe propria răspundere că prezenta lucrare de licență/proiect de diplomă/disertație se bazează pe activitatea personală, cercetarea/proiectarea este efectuată de mine, informațiile și datele preluate din literatura de specialitate sunt citate în mod corespunzător.

Localitatea, Târgu Mureș

Data: 16. Iunie 2024

Absolvent

Semnătura.....

Kivonat

A mélytanuláson alapuló konvolúciós neurális hálók az egyik legjobban teljesítő hálók közé tartoznak a gépi látás területén. Ezeknek a fejlődése segíthetik például a radiológusoknak agyi MRI képeken a tumor beazonosítását vagy a tumor típusának a megállapítását.

Dolgozatomban három, mások által kifejlesztett, különböző komplexitású konvolúciós neurális háló teljesítményét (VGG16, ComplexCNN, SimpleCNN) hasonlítom össze két adathalmazon. A méréseim azt mutatják, hogy a Kaggle adathalmazon a VGG16 és a ComplexCNN nagyon hasonlóan teljesítenek, 1 és 0.99 f1-score-t érnek el, míg a SimpleCNN-nek 0.90 az f1-score-ja. A Figshare adathalmazon a ComplexCNN és a SimpleCNN vannak le mérve, a SimpleCNN 0.92 f1-score-t ér el, míg a ComplexCNN 0.97-et. A méréseim ellentmondanak a ComplexCNN kifejlesztőinek, ők a tanulmányukban azt állítják, hogy a ComplexCNN jobban teljesít képosztályozási feladatokban, mint a VGG16 és az összes többi háló.

A kutatás mellett egy PoC (proof of concept) szoftver is elkészült, amely lehetővé teszi, hogy a böngészőben ki lehessen próbálni egy tetszőleges agyi MRI képre a mérések alapján kiválasztott hálót, illetve vizuálisan ábrázolja az osztályozásnak az eredményét.

Kulcsszavak: konvolúciós neurális háló, agyi daganat detektálás, agyi daganat osztályozás.

Rezumat

Rețelele neuronale convoluționale (CNN) bazate pe învățare profundă sunt printre cele mai performante rețele în domeniul vederii computerizate. Dezvoltarea lor poate asista, de exemplu, radiologii în identificarea tumorilor sau determinarea tipurilor de tumori în imaginile RMN ale creierului.

În teza mea, compar trei rețele neuronale convoluționale de complexitate variată pe două seturi de date, care au fost dezvoltate de alții, și investighez care dintre ele performează cel mai bine. Măsurătorile mele arată că pe setul de date Kaggle, VGG16 și ComplexCNN au performanțe foarte similare, atingând scoruri F1 de 1 și, respectiv, 0.99, în timp ce SimpleCNN are un scor F1 de 0.90. Pe setul de date Figshare, au fost măsurate ComplexCNN și SimpleCNN, SimpleCNN atingând un scor F1 de 0.92, iar ComplexCNN atingând 0.97. Măsurătorile mele contrazic dezvoltatorii ComplexCNN, care afirmă în studiul lor că ComplexCNN performează mai bine în sarcinile de clasificare a imaginilor decât VGG16 și toate celelalte rețele.

În plus față de cercetare, a fost dezvoltat și un software de tip proof of concept (POC). Acest software permite utilizatorilor să testeze rețeaua selectată într-un browser cu o imagine RMN cerebrală și ilustrează vizual rezultatul clasificării.

Cuvinte-cheie: Rețelele neuronale convoluționale, detecția tumorilor cerebrale, clasificarea tumorilor cerebrale.

Abstract

Deep learning-based convolutional neural networks (CNNs) are among the best-performing networks in the field of computer vision. Their development can assist, for example, radiologists in identifying tumors or determining tumor types in brain MRI images.

In my thesis, I compare three convolutional neural networks of varying complexity on two datasets which were developed by others, and I investigate which one performs the best. My measurements show that on the Kaggle dataset, VGG16 and ComplexCNN perform very similarly, achieving F1-scores of 1 and 0.99, respectively, while SimpleCNN has an F1-score of 0.90. On the Figshare dataset, ComplexCNN and SimpleCNN are measured, with SimpleCNN achieving an F1-score of 0.92 and ComplexCNN achieving 0.97. My measurements contradict the developers of ComplexCNN, who claim in their study that ComplexCNN performs better on image classification tasks than VGG16 and all other networks.

In addition to the research, a proof of concept (POC) software was also developed. This software allows users to test the selected network in a browser with a brain MRI image and it visually illustrates the classification result.

Keywords: convolutional neural network, brain tumor detection, brain tumor classification.

Tartalomjegyzék

1. Bevezető	10
1.1. Célkitűzések	11
2. Szakirodalmi tanulmány	12
3. Konvolúciós neurális hálózatok	14
3.1. A CNN előnye az elődeihez képest	15
3.2. Eltűnő gradiens	15
3.3. A konvolúciós neuronháló rétegei	15
3.3.1. Konvolúciós réteg	15
3.3.2. Pooling réteg	16
3.3.3. Sűrű réteg	18
3.3.4. Dropout réteg	18
3.3.5. Batch normalizálás	19
3.4. Háló tanításához kapcsolódó fogalmak	19
3.4.1. Aktivációs függvények	19
3.4.2. Veszteségfüggvények	19
3.4.3. Optimalizálók	19
4. Mérések	21
4.1. Hálók architektúrájának bemutatása	21
4.1.1. VGG16 konvolúciós neuronháló	21
4.1.2. ComplexCNN konvolúciós neuronháló	22
4.1.3. SimpleCNN konvolúciós neuronháló	23
4.2. Mérési protokoll	23
4.2.1. Adathalmazok	23
4.2.2. Adathalmaz előkészítése	24
4.2.3. A hálók tanítása	24
4.3. Mérési eredmények	28
4.3.1. A Kaggle adathalmaz mérési eredményei	28
4.3.2. Figshare adathalmaz 4.2.1 mérési eredményei	32
4.3.3. Mérési eredmények összehasonlítása	34
5. Alkalmazásfejlesztés	36
5.1. Szoftver áttekintése	36
5.1.1. Felhasználói követelmények	36
5.2. Rendszerkövetelmények	36

5.2.1.	Funkcionális követelmények	36
5.2.2.	Nem funkcionális követelmények	36
5.3.	Tervezés	38
5.3.1.	A rendszer architektúrája	38
5.3.2.	Szerveroldal kivitelezése	38
5.3.3.	Szerver elindítása és a weboldal hosztolása	39
5.3.4.	Kliensoldal kivitelezése	39
5.3.5.	Verziókövetés	39
6.	Összegzés	41
	Ábrák jegyzéke	43
	Irodalomjegyzék	45

1. fejezet

Bevezető

A daganat a szervezet sejtjeinek burjánzásából keletkező egyre gyorsuló szövetszaporulat. A daganatok lehetnek jóindulatúak és rosszindulatúak, ezekből több mint 120 különböző fajta létezik. A rosszindulatú agydaganat rendkívül súlyos betegség, amely kezelés nélkül halálhoz vezethet, emiatt kulcsfontosságú ezek korai felismerése és kezelése. Az agydaganatok változatossága és összetettsége nehézséget jelent a betegség diagnosztizálásában. Az elmúlt két évtizedben a radiológiai képalkotás egyre jobban elterjedt, ennek az oka a szükséglet egyre gyorsabb, nagyobb felbontású, költséghatékonyabb és kevésbé invazív kezelésekre. Ennek eredményeként a radiológusoknak egyre több és bonyolultabb képet kellett kiértékelniük, innen indult meg a szükséglet egy automatizált és intelligens rendszerre, ami támogatja a radiológusokat a képanalízisben, a diagnosztizálásban és a kép szegmentálásban. Erre a problémára próbálnak megoldást adni a különböző mesterséges intelligencia modellek modellek az utóbbi évtizedből.

A konvolúciós neuronhálók elterjedése új lehetőségeket hozott a gépi látás területén, felmerült a valós lehetősége annak, hogy szakorvosok, radiológusok munkáját segítsék tumorok, csonttörések, betegségek képekről való beazonosításával. Jelenleg a mágneses rezonancia képalkotás (MRI) a leggyakrabban használt módszer az agydaganatok diagnosztizálására, így a dolgozatom kutatási részében agyi MRI képeken daganatok diagnosztizálását vizsgálom konvolúciós neurális hálókkal.

Ezen a területen kihívást jelent a jó adathalmazok megtalálása, amelyek nagy mennyiségű és jó minőségű képeket tartalmaznak, emiatt nehéz úgy megtanítani egy neurális hálót, hogy az valós korhízi környezetben és független adathalmazokon is megfelelően teljesítsen. Emelett probléma az ilyen modellek által megállapított vizsgálati eredményekben a bizalom, főként mert ezeket "fekete doboz" modelleknek is hívják, azaz rendkívül nehéz végigkövetni, hogy egy osztályozást milyen okokból hoztak meg. Dolgozatomban is ugyanilyen "fekete doboz" modelleket hasonlítok össze, abban a reményben, hogy a méréseim eredményével egy pontosabb képet tudjak kapni ezen modellek teljesítményéről.

Dolgozatomban több mások által elkészített változó komplexitású konvolúciós neuronhálót hasonlítok össze adathalmazokon, hogy megtudjam melyik teljesíti a legjobban és mekkora az eltérés köztük. Ezen mérésekkel kiválasztom a legjobban teljesítő modellt és készítek egy webalkalmazást, ahol ki lehet próbálni képek feltöltésével, hogyan tud daganatokat diagnosztizálni a kiválasztott háló.

1.1. Célkitűzések

Az egyik célkitűzésem ezzel a dolgozattal az, hogy összehasonlítsak mások által megtervezett neuronhálókat több adathalmazon, azonos mérési metrikákat használva. Megakartam vizsgálni a Saikat et al. [15] tanulmányában elhangzottakat, miszerint az általuk kifejlesztett neuronháló jobb eredményeket tud elérni, mint az eredeti VGG16 [18] és az összes többi létező neurális háló. Ezenkívül látni akartam, hogy egy komplex neuronháló mennyivel teljesít jobban az agyi MRI képek osztályozásában, mint egy sokkal egyszerűbb architektúrájú háló, ezt Mohamed et al. github repositoryjából [5] vettem.

A dolgozatom másik célja egy szoftver fejlesztése, ami gyakorlatba ülteti a dolgozatomban legjobban teljesítő modellt és bárki számára kipróbálhatóvá teszi azt.

2. fejezet

Szakirodalmi tanulmány

Az új technológiák kifejlődése, különösen a gépi tanulás és mélytanulás jelentősen befolyásolta az orvostudomány területét az elmúlt évtizedben. Ezek a technológiák lehetőséget adtak olyan támogató szoftverek kifejlesztéséhez az orvostudomány különböző ágazataiban, amelyek segítenek a szakembereknek egy másodvéleményt adni, például agyi MRI képekről megállapítani, hogy azok egészségesek, vagy tumor van rajtuk, ezzel segítve a radiológusok munkáját.

Saikat et al. [15] tanulmánya betekintést nyújt az agydaganatok MRI-képek alapján történő felismerésébe mélytanulási technikák alkalmazásával. A kutatás az Figshare [1] és a Harvard Medical [2] agyi MRI-képek adathalmazát használja, amelyen augmentálást végeznek, hogy csökkenjen az overfitting esélye, és hogy potenciálisan jobban tudjon általánosítani ismeretlen képekre a háló. Egy új 23 rétegű CNN architektúrát javasolnak, illetve egy módosított és kibővített VGG16-on alapuló hálót, amelyet a hiperparaméterek finomhangolásával optimalizálnak. A tanulmányban arra a következtetésre jutottak, hogy a javasolt 23 rétegű CNN és a kibővített VGG16 alapú hálók az összes azelőtti CNN-nél jobb eredményeket érnek el az agyi tumor osztályozás területén és jelentősen javítják az agydaganatok felismerésének pontosságát.

Saeedi et al. tanulmányában [17] egy 3264 MRI agyi képet tartalmazó adathalmazt [3] használtak, amelyek között glioma, meningioma, pituitary típusú daganatok és egészséges agyi képek szerepelnek, ezeken preprocesszálást és adat augmentálást végeztek. Egy új architektúrájú hálót fejlesztenek ki: egy 2D Konvolúciós neurális hálót, amelyhez specifikus hiperparamétereket választanak ki. A háló nyolc konvolúciós és négy pooling réteggel rendelkezik, illetve mindenik konvolúciós réteg után egy batch normalizációs réteget raktak. Hat mások által tervezett neurális hálót is összehasonlítottak az agydaganatok osztályozására a sajátjukkal. A javasolt 2D CNN tanítási pontossága 96,47%. A nem mély tanuláson alapuló hálók közül a legmagasabb pontosságot a KNN (K-Nearest Neighbors) érte el 86%-kal, míg a legalacsonyabbat az MLP (Multilayer Perceptron) 28%-kal. Statisztikai tesztek jelentős különbségeket mutattak ki a kifejlesztett módszerek és más gépi tanulási módszerek között. A tanulmányban arra a következtetésre jutottak, hogy a 2D CNN optimális pontosságot ért el az agydaganatok osztályozásában, így alkalmas a klinikai használatra.

Milica et al. tanulmányában [12] a Figshare adathalmazt [1] használják, amely 3064 T1 súlyozott képet tartalmaz. A T1 súlyozott képekben a zsírszövetek kiemeltebben jelennek meg, míg a víz gyengébben van megjelenítve. A képeken preprocesszálást végez-

nek, normalizálják és újraméretezik őket. Augmentációs módszereket is alkalmaznak az adathalmazon, így a végső adathalmaz, amin a háló tanítva és mérve van 9192 képet tartalmaz. Egy új 22 rétegű konvolúciós neuronhálót terveznek meg, amely a bemeneti rétegből, kétfajta konvolúciós blokkból és egy osztályozási blokkból áll. Az első konvolúciós blokknak a sajátossága hogy a kimeneti kép mérete feleakkora mint a bemeneté, egy konvolúciós rétegből, egy ReLu aktivációs rétegből, egy dropout és egy max pooling rétegből áll. A második konvolúciós blokk annyiban különbözik az elsőtől, hogy a kimeneti kép mérete ugyanakkora marad, mint a bemeneté. Az klasszifikációs blokk két darab teljesen kapcsolt rétegből és egy softmax rétegből áll. A háló összességében a bemeneti rétegből, két első típusú konvolúciós blokkból, két második típusú konvolúciós blokkból és egy klasszifikációs blokkból áll. A háló teljesítménye a 10-fold cross-validation módszerrel lett kiértékelve, és augmentált képek voltak használva teszt adatnak. A végső tanítási pontosság 96.56% volt a Figshare adathalmazon. A tanulmányban arra a következtetésre jutottak, hogy a hálónak jó az általánosítási képessége, jól teljesít új képeken és gyors is, emiatt megkönnyítheti a munkáját egy radiológusnak.

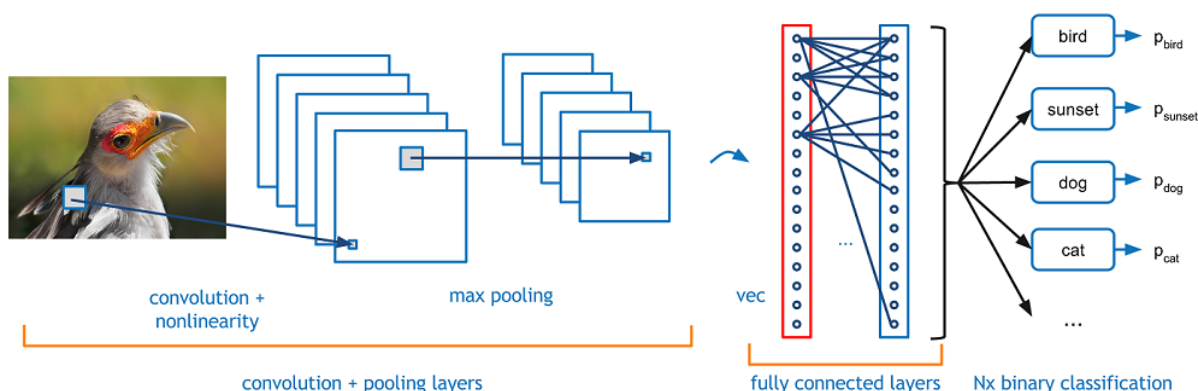
Véleményem szerint a Saikat et al. tanulmányában van egy hiba, a validációs halmazt használják teszt halmaznak minden mérés esetében, ami torzíthatja a kapott eredményeket, jobb teljesítményt eredményezhet a betanult adathalmazon, mint amilyen az valójában. Emiatt nincs jól letesztelve az, hogy a modell hogyan teljesít teljesen ismeretlen képeken annak ellenére, hogy ez a legfontosabb egy orvosi környezetben.

3. fejezet

Konvolúciós neurális hálózatok

A konvolúciós neuronháló (Convolutional Neural Network -CNN) egyfajta mély tanulási modell az adatok feldolgozására, amelynek struktúrája rács mintázatú, mint például a képek. A majmok idegrendszeréről mintázták [14] és arra tervezték, hogy automatikusan és adaptívan tanulja a jellemzők térbeli hierarchiáit, az alacsony szintű mintáktól a magas szintű mintákig. Az egyes rétegek bemenete lokális, azaz a kép kis részleteire koncentrál. A magasabb rétegek egyre nagyobb képrészeket fednek le, de ezzel egyidejűleg egyre rosszabb felbontásúak (a finom részletek egyre kevésbé számítanak).

Egy tipikus CNN háromféle rétegből áll: konvolúciós, pooling és sűrű rétegekből. A konvolúciós és a pooling rétegek jellemző kinyerést végeznek, míg a sűrű réteg a kinyert jellemzőket a végső kimenetre képezi le, például osztályozásra.



3.1. ábra. Egy tipikus konvolúciós neurális háló architektúrája [8]

A konvolúciós réteg fontos szerepet játszik a CNN-nél, amely több matematikai műveletből áll, mint például a konvolúció, amely egy lineáris operáció, amivel a súlyozott összeget számolhatjuk ki két függvénynek.

A digitális képekben a pixelértékek egy kétdimenziós rács formájában vannak struktúrázva, ezeken egy kernel nevű paraméterrács van végigcsúsztatva, ez egy optimalizálható jellemző kinyerő. A paraméterek, mint például a kernel, optimalizálásának folyamatát tanításnak hívjuk, amely úgy hajtodik végre, hogy a háló kimenete és az alap igazság közti különbség minél kisebb legyen egy optimalizáló algoritmus használatával, mint a backpropagation algoritmus és a gradiens descent.

3.1. A CNN előnye az elődeihez képest

A CNN előnyei az egyszerűbb neurális hálókhoz, illetve az SVM vagy Random Forest gépi tanulási algoritmusokhoz képest:

1. **Hierarchikus reprezentációk:** A konvolúciós neuronhálók speciális rétegekből állnak, és minden réteg saját absztrakciós szintet képvisel. Az alsóbb rétegek az alapvető jellemzőket és mintázatokat tanulják meg, míg a magasabb rétegek ezeket az alacsonyabb szintű reprezentációkat kombinálják, és magasabb szintű jellemzőket kódolnak.

2. **Automatikus jellemzőtanulás:** A konvolúciós neuronhálók automatikusan tanulják meg a jellemzőket az adatokból anélkül, hogy expliciten meghatároznák azokat. Ebben az értelemben képesek a magasabb szintű absztrakciók automatikus kinyerésére, mivel az algoritmus magától felismeri a fontos jellemzőket a bemeneti adatokban.

3. **Általánosabb reprezentációk:** A mély neurális hálók általában általánosabb reprezentációkat hoznak létre, ami azt jelenti, hogy a jellemzőkinyerési réteg általában átültethető más, hasonló jellegű problémákra is. Ez lehetővé teszi az általánosabb tanulást és a modell újrafelhasználhatóságát különböző alkalmazásokban.

4. **Nagyobb kapacitás:** A mély neurális hálók általában nagyobb paraméterszámmal rendelkeznek, ami lehetővé teszi számukra, hogy nagyobb mennyiségű információt tároljanak és tanuljanak meg. Ennek eredményeként jobban alkalmazkodnak a komplex adathalmazokhoz.

5. **Invariancia:** A konvolúciós neuronhálók invarianciát képesek kialakítani bizonyos transzformációkra, például eltolásra vagy kis mértékű elforgatásra. Ez növeli a hálózat robusztusságát a különböző képi változásokkal szemben.

3.2. Eltűnő gradiens

A mély neurális hálók tanítása kihívást jelent a hagyományos hálókhoz képest, mivel a backpropagation algoritmus során a hiba visszaterjesztése a kimeneti rétegtől indul és halad visszafelé. Ahogy az információ halad a rétegek mélyére, a gradiens "eltűnésének" esélye nő, emiatt a mélyebb rétegek nehezebben tanulnak.

3.3. A konvolúciós neuronhálók rétegei

3.3.1. Konvolúciós réteg

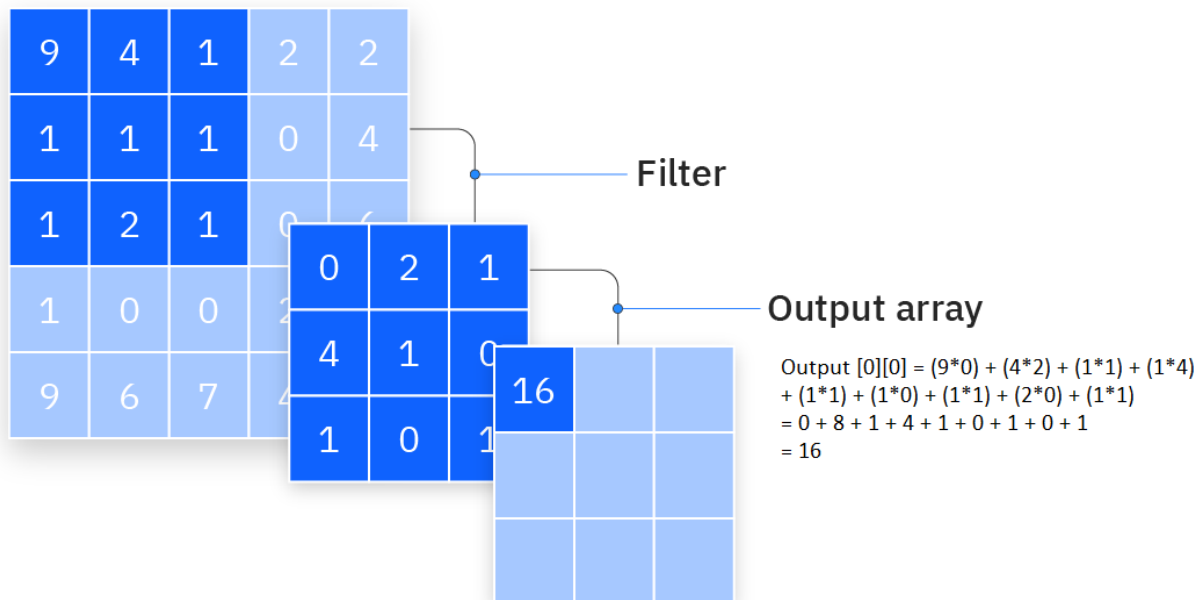
A konvolúciós neuronok hasonlóak a hagyományos neuronokhoz, de egy pár dologban különböznek ezektől:

- **Lokálitás:** a neuronok a bemeneti képnek egyszerre csak egy kis, lokális részletét dolgozzák fel

- **Konvolúció:** a konvolúciós réteg működése során a bemeneti adaton egy konvolúciós kernel (szűrő) mozgatása történik. A kernel ezután a bemenet kis részleteinél alkalmazódik, és a súlyok súlyozott összegzését hozza létre. Ez a művelet sorban a teljes bemeneti térképen végrehajtódik, és így létrejön egy új térkép, amely a kernel megtanult jellemzőit tartalmazza. A "weight sharing" révén ugyanaz a kernel alkalmazódik különböző helyeken, lehetővé téve a lokális jellemzők hatékony kinyerését. Az eredményül kapott térképek a réteg kimenetét képezik, és további rétegeken keresztül haladnak a hálózaton.

Ezen folyamat segít a hierarchikus jellemzők tanulásában és a komplex vizuális feladatok megoldásában, például objektumfelismerésben.

Input image



3.2. ábra. Konvolúciós réteg működése [9]

A konvolúciós réteg működése a következő lépésekből áll:

1. Konvolúció: A bemeneti adatot és a konvolúciós kernelt alkalmazzák. A kernel mozog a bemeneti térképen, és a súlyozott értékeket számolja ki a kis részleteken.

2. Súlyozott értékek összegezése: A súlyozott értékeket összegezik, létrehozva a konvolúciós réteg kimenetét.

3. Aktivációs függvény alkalmazása: Az összegzett értékekre egy aktivációs függvényt alkalmaznak, gyakran a ReLU-t (Rectified Linear Unit), amely egy nemlineáris függvény, amely növeli a hálózat reprezentációs képességeit.

Ezen lépések révén a konvolúciós réteg kinyeri és kódolja a bemeneti térkép lokális jellemzőit. A kernel súlyai a tanulás során módosulnak, így a hálózat alkalmazkodik a feladathoz. A konvolúciós réteg hierarchikus jellemzőket tanul, amelyek segítenek különböző szintű absztrakciókat kifejleszteni a vizuális adatokban.

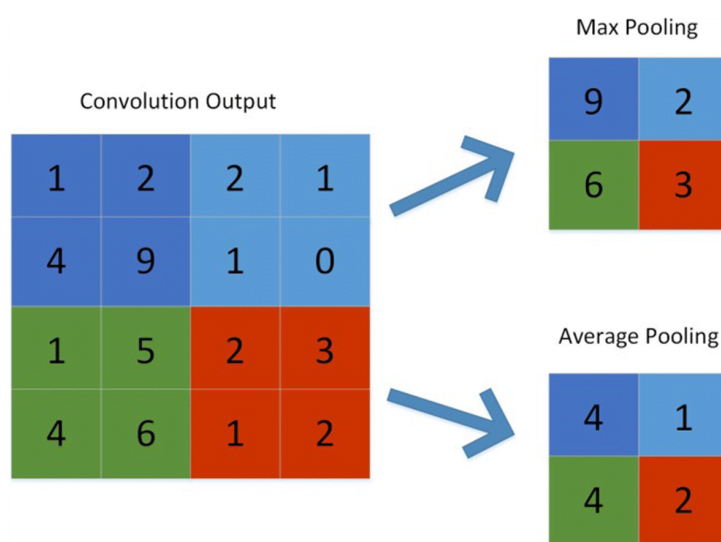
3.3.2. Pooling réteg

A pooling réteg célja a térbeli dimenziók csökkentése, például a magasság és szélesség, megtartva a fontos jellemzőket, és hogy egyre magasabb szinten egyre kevésbé maradjanak

meg finom részletek a képből. A leggyakrabban használt pooling művelet a max pooling, de average pooling is alkalmazható.

A pooling réteg működése:

- **Bemenet:** A pooling réteg kap egy térbeli térfogatot (általában a konvolúciós réteg kimenetét) bemenetként.
- **Szűrő mozgatása:** A pooling réteg egy kis ablakot vagy szűrőt mozgat a bemeneten. A szűrő lépéseket tesz a térbeli dimenziókon, a strázsza paraméter határozza meg, hogy hány pixelt mozog lépésenként.
- **Pooling:** A szűrő által lefedett területen a max pooling esetében legnagyobb érték az average pooling-nál az értékek átlaga kiválasztásra kerül, és ezt az értéket tartalmazza az új, csökkentett dimenziójú térkép.
- **Csökkentett Dimenzió:** A max pooling réteg műveletének eredményeként a térbeli dimenziók csökkentésre kerülnek. Például, ha 2x2-es max pooling-ot alkalmazunk 2-es strázsával, a magasság és a szélesség mindkét dimenziója a felére csökken.



3.3. ábra. Pooling réteg működése [10]

A pooling réteg előnyei a következők:

- Csökkenti a számításigényt, mivel csökkenti a térbeli dimenziókat.
- Növeli a hálózat általánosítási képességeit, mivel megtartja a fontos jellemzőket, miközben elhagyja az apróbb részleteket.
- Segít kezelni az invarianciát bizonyos transzformációkkal szemben, mivel a max pooling a legnagyobb értéket választja ki, függetlenül attól, hol található ez a jellemző a szűrő ablakban.

3.3.3. Sűrű réteg

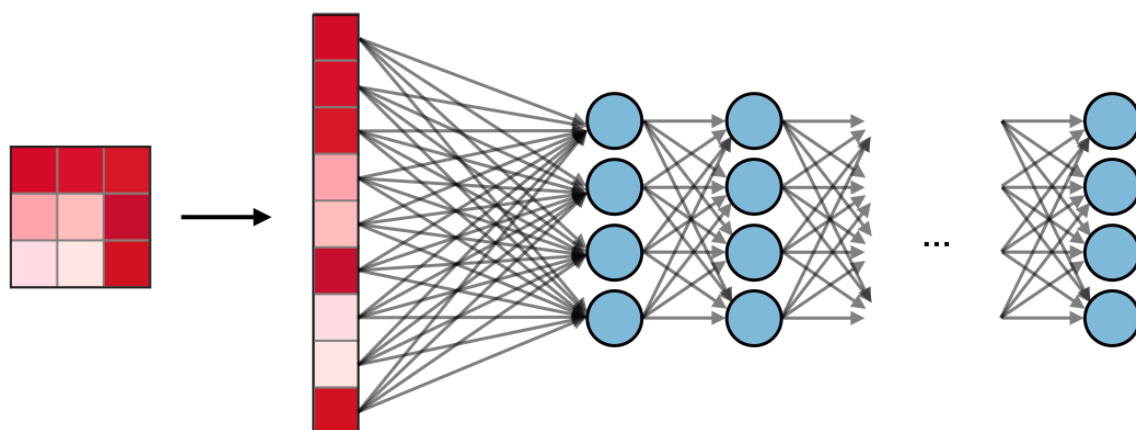
A sűrű réteg (dense layer) működése a konvolúciós neuronhálózatokban a következő lépésekből áll:

Kilapítás: A konvolúciós rétegek előállítják a magasabb szintű jellemzőket a bemeneti képről. Ezek a jellemzők egydimenziós vektorba alakítódnak át, amely a sűrű réteg bemeneteként szolgál. Az egydimenziós vektor bemenet miatt ez a réteg elveszíti a konvolúciós rétegek által megőrzött térbeli kapcsolatokat, és egy globális perspektívát nyújt.

Súlyozott összegzés: A sűrű réteg minden neuronjához tartozik egy súly és egy eltolás (bias). A réteg a bemeneti lapos vektor elemeit súlyozza és összegzi, majd hozzáadja a neuronhoz tartozó eltolást. Ez a lépés egy súlyozott összegzést eredményez, amely kifejezi a bemeneti jellemzők fontosságát a neuron szempontjából.

Aktivációs függvény alkalmazása: A súlyozott összegzést egy aktivációs függvény követi, például a ReLU (Rectified Linear Unit) vagy a szigmoid függvény. Ez a lépés nemlinearitást ad hozzá a rendszerhez, és segít a hálózatnak komplexebb összefüggéseket tanulni a bemeneti adatokban.

Ezáltal a sűrű réteg összekapcsolja és kombinálja a konvolúciós rétegek által megtalált jellemzőket, előállítva a végleges kimenetet, például osztályvalószínűségeket egy osztályozási feladatban.



3.4. ábra. Sűrű réteg [11]

3.3.4. Dropout réteg

A Dropout réteg használata általában a túltanulás csökkentésére és a modell általánosító képességének javítására irányul, különösen nagyobb méretű és komplexebb hálózatok esetén. A réteg egy adott valószínűséggel kikapcsolja (nullára állítja) a neuronokat a réteg bemenetén a tanítás során. Ez azt jelenti, hogy a kimeneti értékeik nem kerülnek továbbításra a hálózatban az adott iteráció során. A neuronok kikapcsolása véletlenszerűen történik minden tanítási iteráció során. Ez azt eredményezi, hogy a hálózat nem támaszkodik túlzottan egyes neuronokra, és így különböző alhálózatok tanulhatnak különböző jellemzőket.

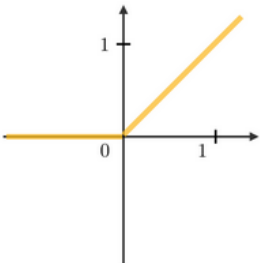
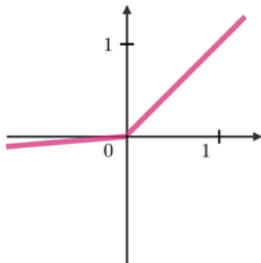
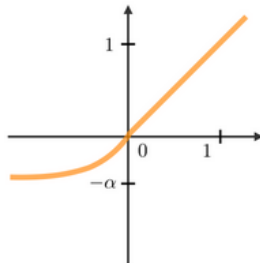
3.3.5. Batch normalizálás

Ez az eljárás a bemeneti adatokat úgy alakítja át, hogy azoknak átlaga nulla legyen, és szórása egységnyi. Ez segít a hálózatnak stabilabban tanulni, kiegyenlítve a rétegek közötti aktivációkat, és elősegítve a gyorsabb és hatékonyabb konvergenciát a tanítási folyamat során, illetve lehetővé teszi hogy a rétegek egymástól függetlenebbül tanuljanak.

3.4. Háló tanításához kapcsolódó fogalmak

3.4.1. Aktivációs függvények

Az aktivációs függvények szerepe a neurális hálózatokban az, hogy bevezessék a nemlinearitást a hálózatba, lehetővé téve számára, hogy bonyolultabb összefüggéseket tanuljon meg az adatok között. Ezek a függvények döntenek arról, hogy egy neuron aktiválódjon vagy sem, a bemeneti értékek súlyozott összege és az eltolás (bias) alapján. A leggyakrabban használt nemlineáris aktivációs függvény a ReLU a Leaky ReLU és az ELU (Exponential Linear Unit). A konvolúciós neurnálóknál az aktivációs függvények általában minden konvolúciós és sűrű réteg után használva vannak.

ReLU	Leaky ReLU	ELU
$g(z) = \max(0, z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$	$g(z) = \max(\alpha(e^z - 1), z)$ with $\alpha \ll 1$
		
<ul style="list-style-type: none">• Non-linearity complexities biologically interpretable	<ul style="list-style-type: none">• Addresses dying ReLU issue for negative values	<ul style="list-style-type: none">• Differentiable everywhere

3.5. ábra. Gyakran használt aktivációs függvények [11]

3.4.2. Veszteségfüggvények

A veszteségfüggvény másnéven költségfüggvény a háló kimeneti predikciói és az igazság közti eltérést méri. A gyakran használt veszteségfüggvények közé tartozik a többosztályos feladatoknál a kereszt-entrópia, a regressziós problémáknál az átlagos négyzetes eltérés (mean squared error).

3.4.3. Optimalizálók

Az optimalizáló fontos szerepet játszik a mély neurális hálók tanításában, célja a hálózat paramétereinek olyan értékek felé változtatása, amelyek minimalizálják a költségfüggvényt és hogy minimalizálják az eltűnő gradiens problémáját.

Az optimalizálók feladatai:

- **Dinamikus tanulási ráta:** Az optimalizálók lehetővé teszik a tanulási ráta automatikus szabályozását, vagyis azt, hogy mennyire jelentősek a súlyok és eltolások frissítései az egyes iterációk során. Ez fontos annak érdekében, hogy a tanítás során ne kerüljön sor túl gyors vagy túl lassú konvergenciára.
- **Költségminimalizáció:** Az optimalizáló feladata a hálózat paramétereinek frissítése oly módon, hogy a tanító adathalmazon való teljesítményt javítsa. Ezáltal a költségfüggvény értéke minimalizálódik, és a modell képes lesz jobban illeszkedni a tanító adatokhoz.
- **Stabilitás:** Az optimalizálók számos technikát alkalmaznak a gradiensok kiegyensúlyozására mint például a gradiens skálázás, ami megakadályozza hogy a gradiens túl nagy vagy túl kicsi legyen, ezáltal segítve a hálózat stabilitását a tanulás során és a numerikus stabilitást

A leggyakrabban használt optimalizálók közé tartozik az Adam (Adaptive Moment Estimation) és az RMSProp (Root Mean Square Propagation).

4. fejezet

Mérések

4.1. Hálók architektúrájának bemutatása

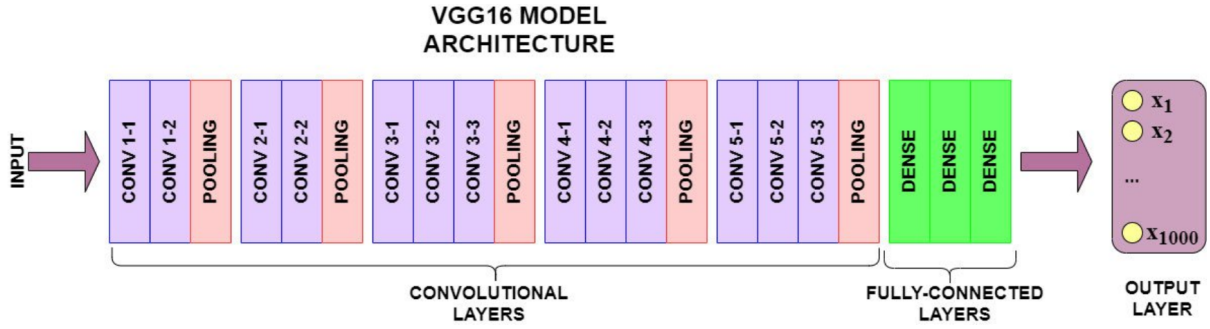
Azért választottam a konvolúciós neurális hálókat a kijelölt feladat megoldásához, mert jó eredményekre képesek kép osztályozási problémákban. A célom az volt hogy kiválasszak több, mások által megtervezett konvolúciós neurális hálót és összehasonlítsam őket különböző adathalmazokon végzett mérések esetében. A választásom a következő három neuronhálóra esett:

- egy egyszerű háromrétegű neuronháló, amelyet **SimpleCNN**-nek fogok hívni,
- a **VGG16** [18] neuronháló, amelynek tizenhat rétege van, és nagyon jó eredményeket ért el komplex képosztályozási feladatokban
- egy komplex huszonhárom rétegű konvolúciós neurális háló, ezt **ComplexCNN**-nek fogom nevezni.

4.1.1. VGG16 konvolúciós neuronháló

A VGG16 egy népszerű CNN, amely jó eredményeket tud elérni komplex több osztályos osztályozási problémákban. A hálót az Oxfordi egyetemen a Visual Geometry Group fejlesztette ki 2014-ben [18]. A VGG16 második helyezést ért el az ImageNet ILSVRC-2014 versenyén a klasszifikációs és objektum lokalizációs problémáknál.

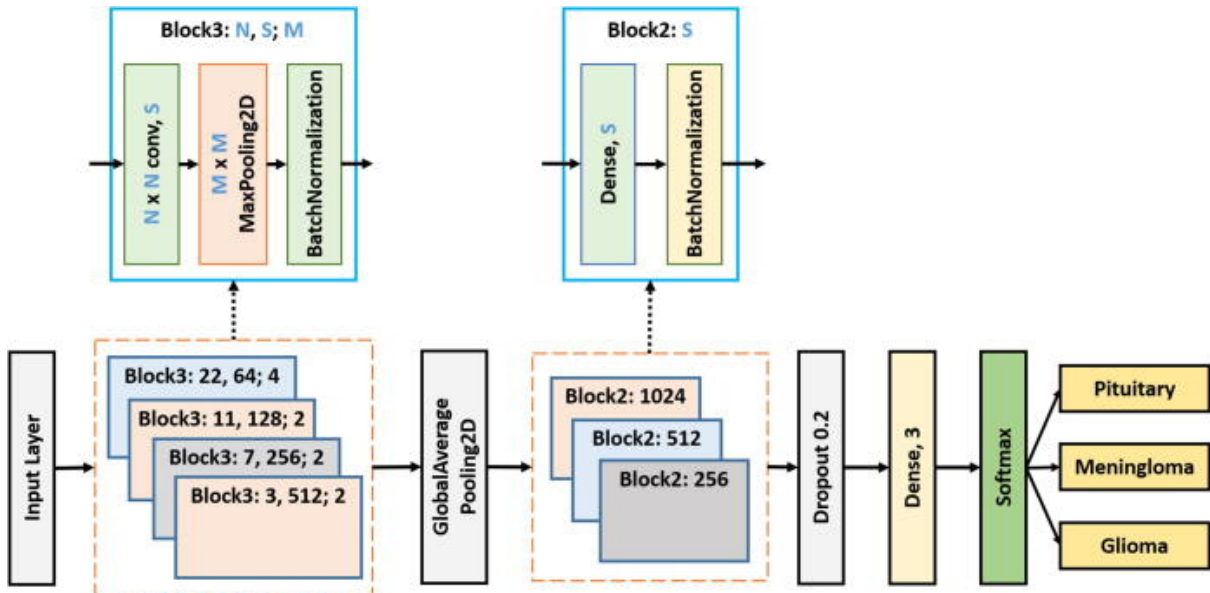
A bemeneti kép egy adag konvolúciós rétegen keresztül kerül feldolgozásra, ezeknek a rétegeknek a kernelje 3x3-as méretű, azért kicsi, mert ez a legkisebb méret amin értelmezhetők a: fel, le, baljra, jobbra, középen pixel kapcsolatok, a strázsa 1 pixel. Mindegyik konvolúciós rétegnél ReLu aktivációs függvény van használva, hogy bevezetődjön a nem-linearitás és csökkenjen az eltűnő gradiens problémája. A háló öt max-pooling réteget tartalmaz, minden kettes vagy hármas csoport konvolúciós réteget követ egy max-pooling réteg. A max-pooling szűrő mérete 2x2 és a strázsa 2 pixel. A konvolúciós hálókat három sűrű réteg követi, az első kettőnek 4096 kimeneti neuronja van, a legutolsónak 1000 neuronja, így tudja klasszifikálni az ImageNet ILSVRC adathalmaz 1000 kategóriájába a képeket. A legutolsó réteg az egy softmax réteg.



4.1. ábra. VGG16 neurális háló architektúrája [7]

4.1.2. ComplexCNN konvolúciós neuronháló

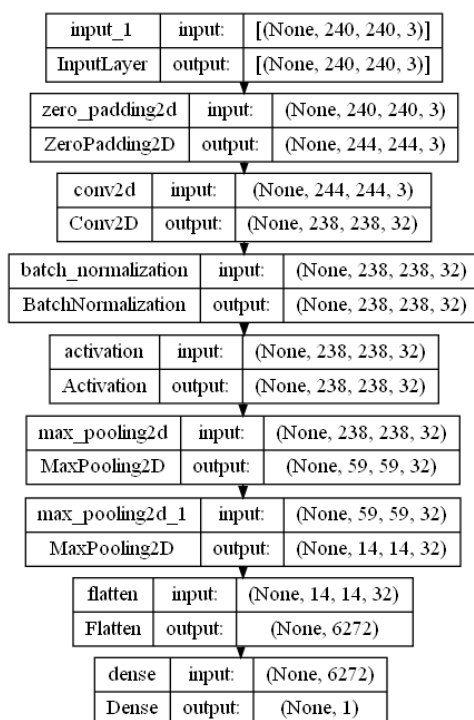
Ezt a neuronháló architektúrát a Saikat et al. [15] publikációjából vettem át, ahol azzal érvelt, hogy jobb eredményeket tud elérni, mint az összes többi létező modell. Az egyik célom a dolgozatomban, hogy összehasonlítsam ezt a hálót másokkal a Kaggle, illetve a Figshare adathalmazon. A háló architektúrája 2 típusú konvolúciós blokkból épül fel, az első típusú blokk egy konvolúciós rétegből, egy max pooling rétegből és egy batch normalizációs rétegből áll; a második típusú blokk egy sűrű rétegből és egy batch normalizációs rétegből áll. Összességében a hálónak van négy darab első típusú blokkja, amit egy average pooling layer követ, ezután pedig három második típusú blokk következik, ezután egy dropout réteg van és egy sűrű réteg, amiben a neuronok számát mindig az adott osztályozási feladat osztályainak a számához igazítom. A háló architektúráját a 4.2 ábra mutatja.



4.2. ábra. A ComplexCNN neurális háló architektúrája [15]

4.1.3. SimpleCNN konvolúciós neuronháló

Ez egy egyszerű neuronháló három tanítható réteggel (lásd 4.3 ábra). Azért választottam egy ilyen egyszerűbb hálót is a mérésekehez, hogy megvizsgáljam, milyen eredményeket ér el a komplexebb neurális hálózatokhoz képest a kiválasztott adathalmazokon. A háló első rétege egy zero padding réteg, 2 pixel padding értékkel minden irányba. Ezt egy konvolúciós réteg követi 32 x 32 pixel feature map paraméterrel, 7 x 7 pixel méretű kernellel, 1 pixel strázsa paraméterrel. Ezt követi egy batch normalization réteg és egy ReLu aktivációs függvény. Ezeket két max pooling réteg követi 4x4 méretű kernellel és egy flatten réteg. Az utolsó egy sűrű réteg sigmoid aktivációs függvénnyel, ennek a rétegnek a kimeneti dimenzionalítását és aktivációs függvényét fogom változtatni az adathalmaz kategória számának függvényében.



4.3. ábra. SimpleCNN neurális háló architektúrája

4.2. Mérési protokoll

Ebben a fejezetben bemutatom a tanító, validációs és tesztelő adathalmazok kiválasztásának módját, ezeknek az előkészítését a tanításhoz. Ezenkívül bemutatom még a hálók tanításának folyamatát az adathalmazokon és ezen hálók teljesítményét. A hálók tanítását és adathalmazok előkészítését és a mérések eredményének ábrázolását Mohamed et al. [5] és Saikat et al. [6] munkái ötvöztetésével végeztem.

4.2.1. Adathalmazok

A méréseket a következő adathalmazokon végeztem:

- Kaggle augmentált adathalmaz [4] (**Kaggle**)
- Figshare adathalmaz [1] (**Figshare**)

Kaggle augmentált adathalmaz

Az eredeti adathalmaz 253 darab jpg formátumú grayscale képből áll, azaz minden pixelnél mindhárom színcsatornában ugyanaz az érték van. Az adathalmaz képei két osztályba sorolódnak, tumorral rendelkező agyi képek és tumor nélküli agyi képek. Az első osztályból 155 kép van, a másodikból 98. Az adathalmazon augmentálás van elvégezve, azért hogy a komplex modell tanítása esetén csökkenjen az overfitting esélye, illetve hogy az osztályok közti kiegyensúlyozatlanság megoldódjon.

Az augmentált adathalmazt használtam a méréseknél, ez 2065 képből áll 1085 pozitív, agyi daganatos képből és 980 negatív, egészséges agyi képből. A végső adathalmaz az eredeti adathalmaz képeit is tartalmazza.

Figshare adathalmaz

Ez az adathalmaz 3064 darab .mat formátumú képből áll, amelyet Cheng J. et al [13] gyűjtöttek össze 2005 és 2010 között a Nanfang kórháztól és a Tianjing orvosi egyetemről, Kínából. Csak daganatos agyi képeket tartalmaz. A képek három osztályra vannak bontva: glioma, meningioma és pituitary. A glioma osztályból 1426, a meningioma-ból 708, a pituitary-ből 930 darab kép van. A képek három nézőpontból vannak készítve: szagittális, axiális és koronális.

4.2.2. Adathalmaz előkészítése

A képek sorrendjét felosztás előtt összekeverem egy függvénnyel, hogy random legyen a kategóriák eloszlása. Mindenik adathalmaznak 70 százalékát vettem tanítóhalmaznak, ami a Kaggle adathalmaz esetében 1444 képet jelent, a Figshare adathalmaz esetében 2134 képet. A tanításkor 32-es batch méretet használtam és 40 epoch-on keresztül tanítottam mindenik esetben. A validációs adathalmaz a képek 15 százalékából áll, ami a Kaggle adathalmaz esetében 310 képet jelent és a Figshare-nél 458-at. Teszt adathalmazt azért csináltam, hogy tudjam lemérni a háló teljesítményét olyan képekre, amelyet még nem látott. Azért fontos egy külön teszt adathalmaz használata a validációs adathalmazon kívül, mert a validációs adathalmazt indirekt módon megtanulja a háló, mert ez a hiperparaméterek optimalizálására van használva tanításkor és csak független teszt adathalmazzal lehet lemérni megfelelően egy háló valós teljesítményét. A teszt adathalmaz minden esetben a képek 15 százalékából áll.

4.2.3. A hálók tanítása

Tanítás során csak a legnagyobb validációs pontosságú modell súlyait mentettem el. A tanítás minden esetben 40 epoch-ra volt beállítva. A tanításnál az Adam [16] optimalizátort használtam, adaptív tanulási együtthatót is használtam de volt olyan háló, ami esetében az eredmények romlottak ennek hatására, így ott a nem adaptív tanulási együtthatós eredményt vettem alapul, ilyen esetekben látszik, hogy a tanulási görbék nem si-

Tanítási idők

	ComplexCNN	VGG16	SimpleCNN
Figshare adathalmaz	4 perc 46 mp	-	8 perc 43 mp
Kaggle adathalmaz	6 perc 57 mp	3 perc 15 mp	1 perc 15 mp

4.4. ábra. Tanítási idők

mák. Az Adam egy gyakran használt optimalizáló a mély tanulásban, előnyei, hogy minden paraméternek külön állítja a tanítási együtthatóját és általában gyorsabban konvergál, mint az SGD. Veszteség függvénynek a Kaggle adathalmaz esetében Binary Crossentropy függvényt használtam, a Figshare-nél pedig Sparse Categorical Crossentropy függvényt. A VGG16 háló az Imagenet adathalmazon tanított súlyokkal van inicializálva és a benne lévő konvolúciós rétegek tanítása ki lett kapcsolva, ezáltal lényegében transfer learning történik a háló esetében, mivel az Imagenet képei három színcsatornásak és a Figshare adathalmazban a képek egy színcsatornásak ezért ezeknél a csatornákat megháromszoroztam és átkonvertáltam három színcsatornás képekbe.

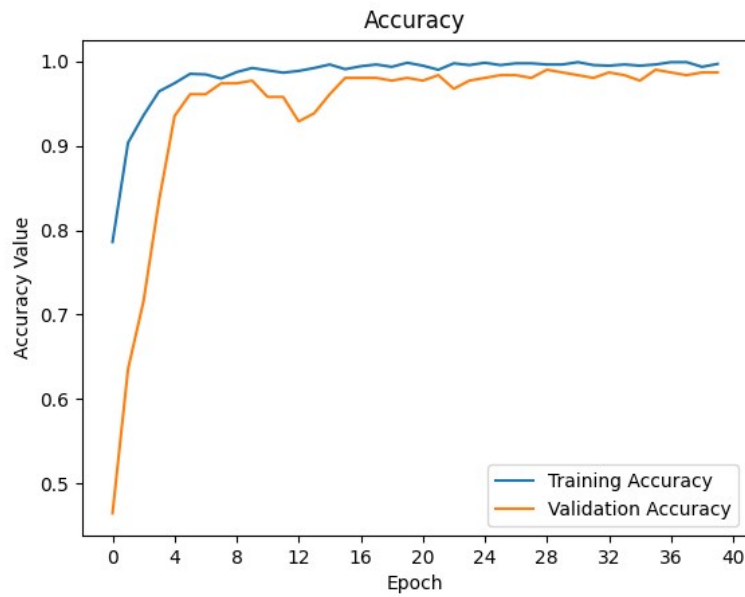
A tanítást egy AMD RX 6700XT videokártyával és 16 GB memóriával rendelkező számítógéppel végeztem, a tensorflow-t egy Directml nevű library-val tudtam működtetni az AMD videokártyán Windows 10-en. A modellek a tanítási ideje az adathalmazokon a 4.4 táblázatban láthatók.

A 4.5 ábrán látszik, hogy a ComplexCNN eléri a legnagyobb validációs pontosságot a 16-ik korszaktól és onnan kezdve nem ingadozik lényegesen. A tanulási görbe sima az adaptív tanulási együttható miatt. A legjobb validációs pontosság a tanítás során 99%.

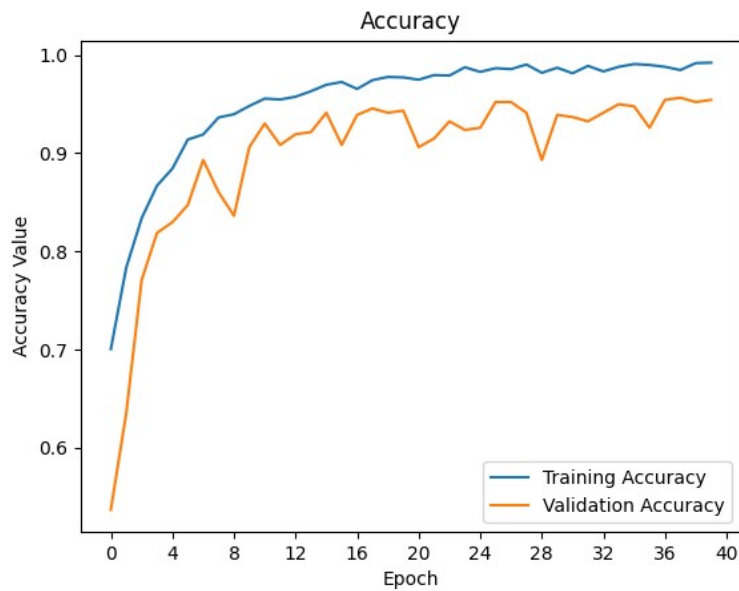
A 4.6 ábrán látszik, hogy a ComplexCNN eléri a közel a legnagyobb validációs pontosságát már a 14-ik korszaktól. A legjobb validációs pontosság a tanítás során 95.6%.

A 4.7 ábrán látszik, hogy a VGG16 a 4. korszaktól eléri a maximális validációs pontosságát, ami 99% és ettől kezdve semennyit nem ingadozik. Adaptív tanulási együtthatóval jobb eredményt ért el, mint anélkül mérve.

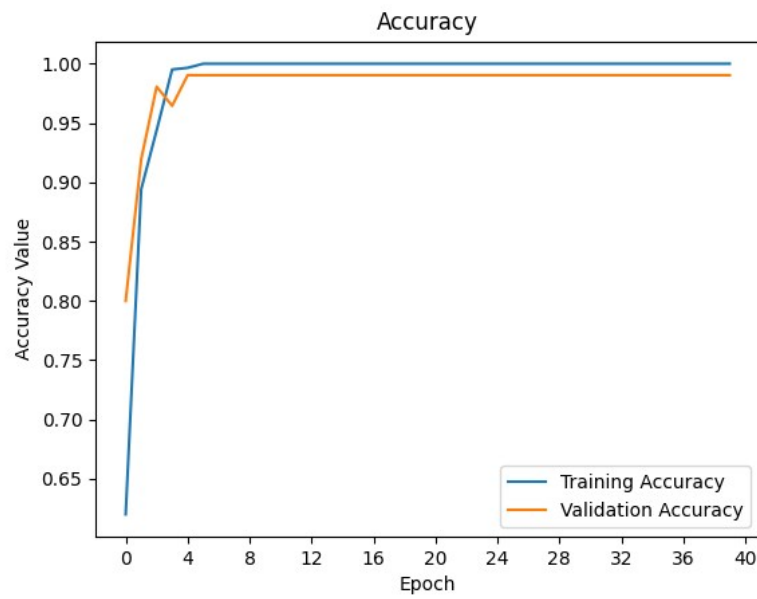
A 4.8 diagramon látszik a SimpleCNN validációs pontossági görbéje nem sima, nem egyenletesen tanult a háló, próbáltam kisebb, $1e-4$ tanítási együtthatóval de az alapértelmezett $1e-3$ értékkel jobb validációs pontosságot ért el a modell. A modell 28-ik korszakra éri el a legmagasabb a validációs pontosságot, 91.3%-ot. Fontos megjegyezni, hogy próbáltam adaptív tanulási együtthatóval sokkal egyenletesebb a tanulási görbe, viszont lényegesen rosszabb eredményt ér el akkor a háló.



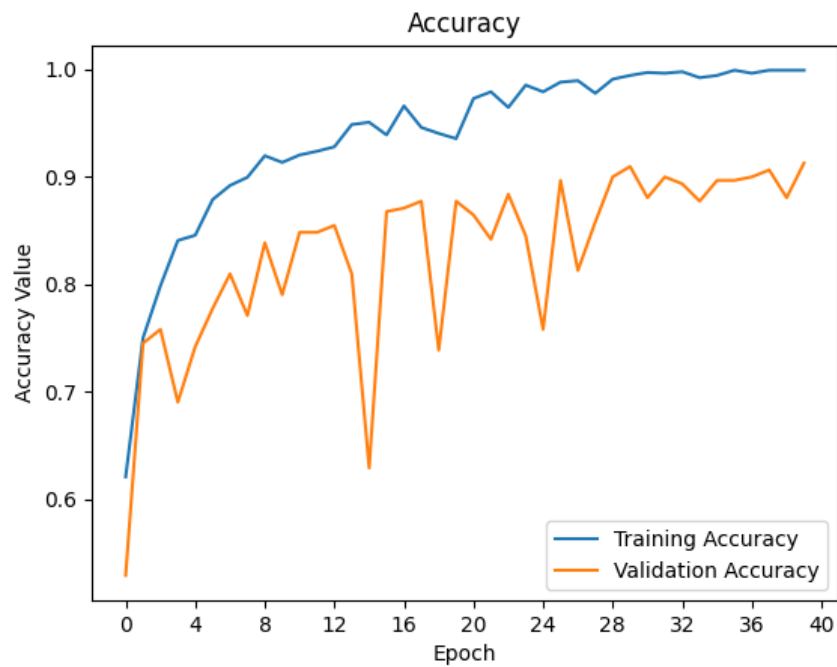
4.5. ábra. ComplexCNN tanítási és validációs pontossága a Kaggle adathalmazon



4.6. ábra. ComplexCNN tanítási és validációs pontossága a Figshare adathalmazon

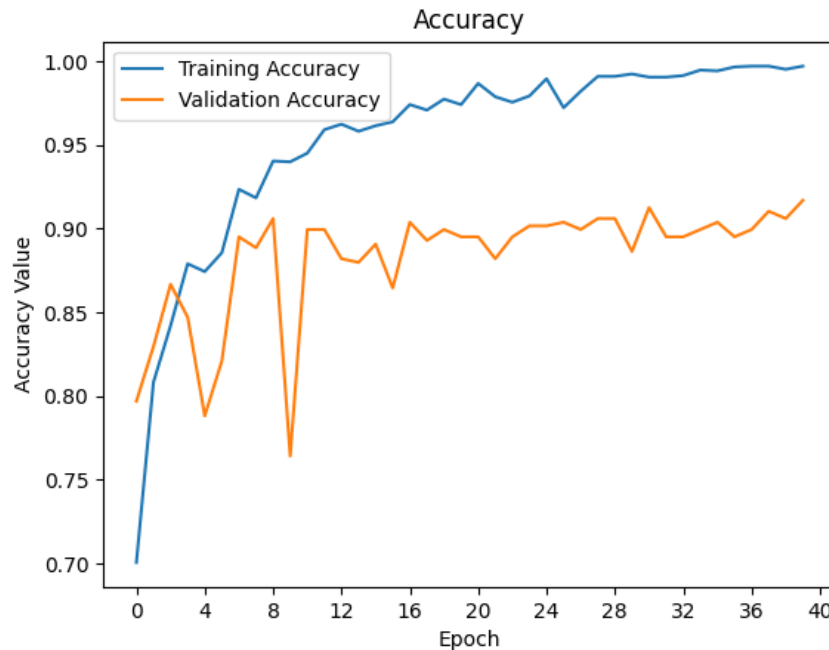


4.7. ábra. VGG16 tanítási és validációs pontossága a Kaggle adathalmazon



4.8. ábra. SimpleCNN tanítási és validációs pontossága a Kaggle adathalmazon

A 4.9 diagramon a megfigyelhető, hogy a SimpleCNN A legjobb validációs pontossága 91.7%. A 6-ik korszaktól a validációs pontosság nem nő lényegesen. A pontossági görbék nem simák, a modell nem egyenletesen tanult, próbáltam kisebb tanítási együtthatóval és adaptív tanulási együtthatóval is, ezek simább görbét eredményeztek de a validációs pontosság kisebb lett, így visszatértem az alapértelmezett $1e-3$ értékhez. Figyelemre méltó, hogy ez az egyszerű háló 91.7%-os validációs pontosságot ért el, ami csak 3.9%-al kevesebb, mint a sokkal komplexebb ComplexCNN hálóa.



4.9. ábra. SimpleCNN tanítási és validációs pontossága a Figshare adathalmazon

4.3. Mérési eredmények

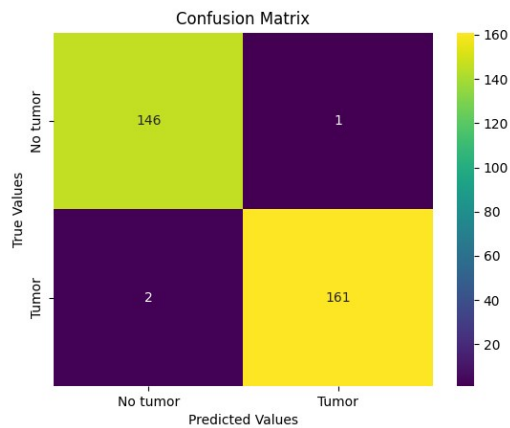
Ebben a fejezetben a mérések eredményét mutatom be. Az eredményeket konfúziós mátrixsal, ROC-görbével, f1-score-al, precision és recall metrikákkal fogom szemléltetni. A célom többek között az is volt, hogy kiválasszak több, mások által megtervezett konvolúciós neuronhálót, betanítsam ezeket több adathalmazon és összehasonlítsam ezeket.

A bináris osztályozásnál a tumorokat tartalmazó képeket a pozitív osztályba soroltam, a tumor nélküli képeket pedig a negatív osztályba. A többosztályos osztályozásnál, három osztály használtam, amennyi az Figshare adathalmaznak megfelel.

4.3.1. A Kaggle adathalmaz mérési eredményei

Ezen az adathalmazon a mindhárom modell jó eredményeket ért el.

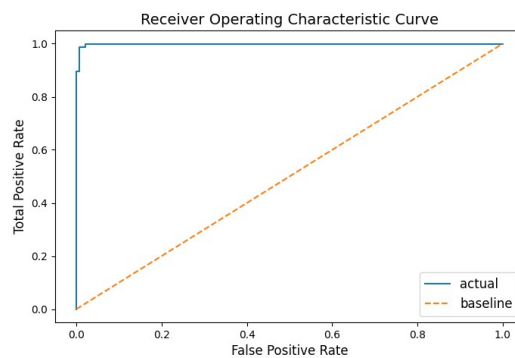
- A ComplexCNN klasszifikációs riportja az 4.11 táblázatban látható, 0.99 weighted average f1-score-t ér el. A konfúziós mátrixot a 4.10 ábra mutatja, itt látható hogy a 310 képből álló tesztadatra 2 darab hamis negatív és 1 darab hamis pozitív osztályozás történt. A ROC-görbe a 4.12 ábrán látható.



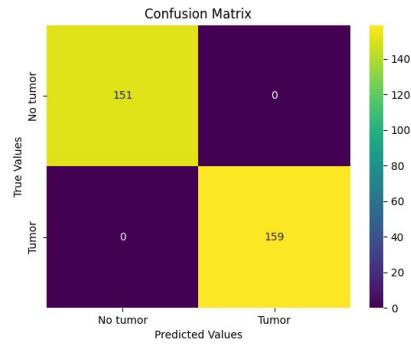
4.10. ábra. ComplexCNN háló konfúziós mátrixa a Kaggle adathalmazon

Classification Report				
	precision	recall	f1-score	support
No tumor	0.99	0.99	0.99	147
Tumor	0.99	0.99	0.99	163
accuracy			0.99	310
macro avg	0.99	0.99	0.99	310
weighted avg	0.99	0.99	0.99	310

4.11. ábra. ComplexCNN eredményei a Kaggle adathalmazon



4.12. ábra. ComplexCNN ROC görbéje a Kaggle adathalmazon



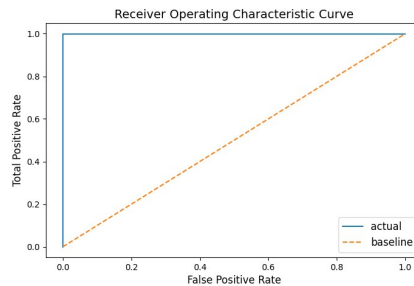
4.13. ábra. VGG16 háló konfúziós mátrixa a Kaggle adathalmazon

Classification Report				
	precision	recall	f1-score	support
No tumor	1.00	1.00	1.00	151
Tumor	1.00	1.00	1.00	159
accuracy			1.00	310
macro avg	1.00	1.00	1.00	310
weighted avg	1.00	1.00	1.00	310

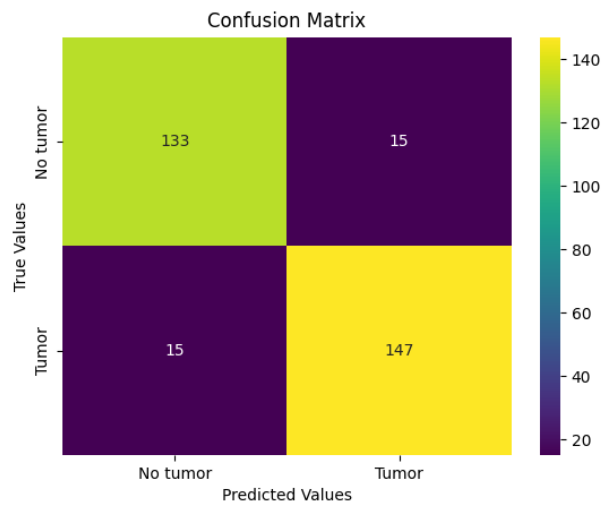
4.14. ábra. VGG16 eredményei a Kaggle adathalmazon

- A VGG16 klasszifikációs riportja a 4.14 táblázatban látható, a maximális 1 weighted average f1-score-t ér el. A konfúziós mátrix a 4.13 ábrán látható, mindenik képet jól osztályozza a teszhalmazból, ez figyelemre méltó. A ROC-görbe a 4.15 ábrán látható.

- A SimpleCNN classification report-ja a 4.17 táblázatban látható, 0.9 weighted average f1-score-t ér el. A ROC-görbe a 4.18 ábrán látható, a konfúziós mátrix a 4.16 ábrán, a modell 15 hamis negatív és 15 hamis pozitív osztályozást csinált.



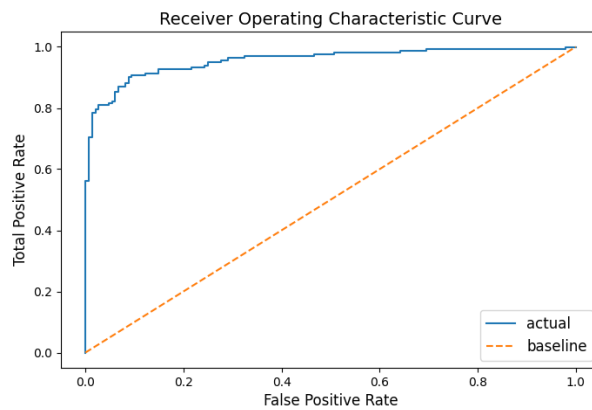
4.15. ábra. VGG16 ROC görbéje a Kaggle adathalmazon



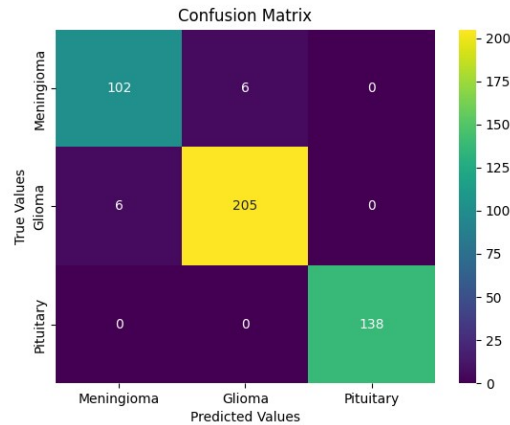
4.16. ábra. A SimpleCNN háló konfúziós mátrixa a Kaggle adathalmazon

Classification Report				
	precision	recall	f1-score	support
No tumor	0.90	0.90	0.90	148
Tumor	0.91	0.91	0.91	162
accuracy			0.90	310
macro avg	0.90	0.90	0.90	310
weighted avg	0.90	0.90	0.90	310

4.17. ábra. A SimpleCNN a Kaggle adathalmazon



4.18. ábra. A SimpleCNN ROC görbéje a Kaggle adathalmazon



4.19. ábra. ComplexCNN háló konfúziós mátrixa a Figshare adathalmazon

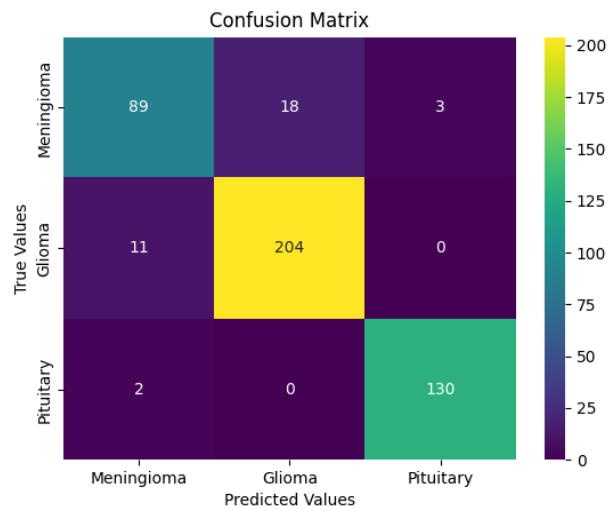
Classification Report				
	precision	recall	f1-score	support
Meningioma	0.94	0.94	0.94	108
Glioma	0.97	0.97	0.97	211
Pituitary	1.00	1.00	1.00	138
accuracy			0.97	457
macro avg	0.97	0.97	0.97	457
weighted avg	0.97	0.97	0.97	457

4.20. ábra. ComplexCNN eredményei a Figshare adathalmazon

4.3.2. Figshare adathalmaz 4.2.1 mérési eredményei

- A ComplexCNN háló classification report-ja a 4.20 táblázatban látható, a teszt halmazban a Meningioma típusú tumorokból lényegesen kevesebb volt, mint a másik két kategóriából, ez a support oszlopban látszik. A weighted average f1-score 0.97. A konfúziós mátrix a 4.19 ábrán látható, a 457 képből álló tesztadaton 12 képet osztályozott rosszul.

- A SimpleCNN classification report-ját a 4.22 táblázat mutatja, 0.92 modell a weighted average f1-score-ja. A konfúziós mátrix a 4.21 ábrán látható, összesen 34 képet azonosított be rosszul a modell.



4.21. ábra. A SimpleCNN háló konfúziós mátrixa a Figshare adathalmazon

Classification Report				
	precision	recall	f1-score	support
Meningioma	0.87	0.81	0.84	110
Glioma	0.92	0.95	0.93	215
Pituitary	0.98	0.98	0.98	132
accuracy			0.93	457
macro avg	0.92	0.91	0.92	457
weighted avg	0.92	0.93	0.92	457

4.22. ábra. A SimpleCNN a Figshare adathalmazon

Mérési eredmények a Kaggle adathalmazon

	precision	recall	f1-score
VGG16	1	1	1
ComplexCNN	0.99	0.99	0.99
SimpleCNN	0.90	0.90	0.90

4.23. ábra. Mérés összehasonlító táblázat a Kaggle adathalmazon

Mérési eredmények a Figshare adathalmazon

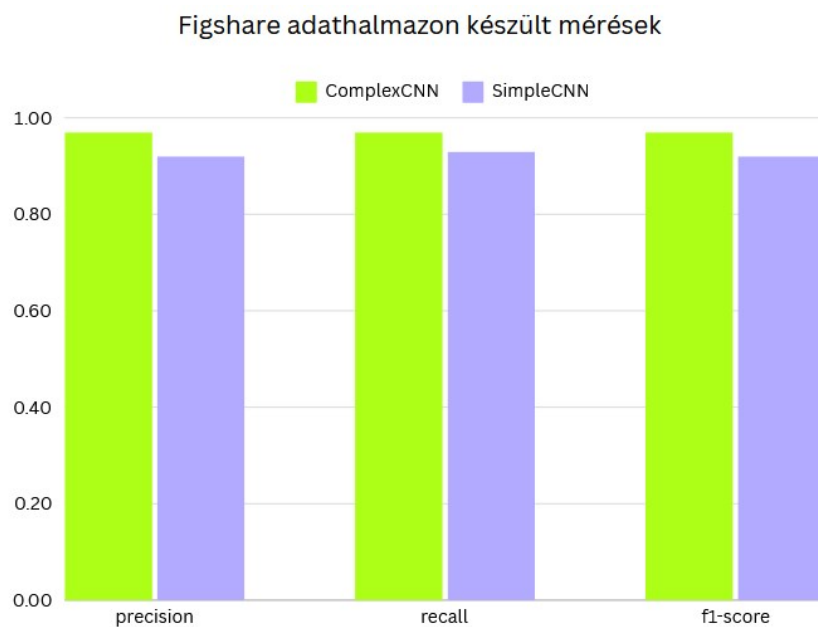
	precision	recall	f1-score
ComplexCNN	0.97	0.97	0.97
SimpleCNN	0.92	0.93	0.92

4.24. ábra. Mérés összehasonlító táblázat a Figshare adathalmazon

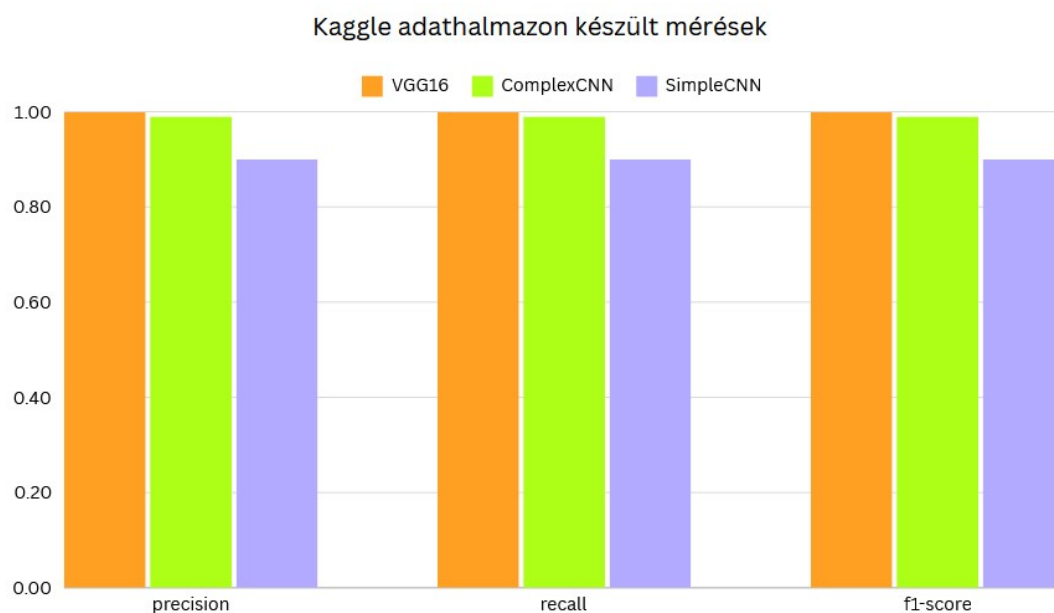
4.3.3. Mérési eredmények összehasonlítása

A 4.23 táblázatban összesítve látszanak a Kaggle adathalmazon végzett mérések eredményei. A VGG16 és a ComplexCNN mérési eredményei közt nincs nagy eltérés. A SimpleCNN eredményei gyengébbek, 0.90-et ér el mindhárom metrikában. A 4.26 ábrán egy összehasonlító oszlopdiagram látható ezekről a mérésekről.

A 4.24 táblázatban a Figshare adathalmazon végzett mérések láthatók, itt az a meglepő, hogy a sokkal egyszerűbb SimpleCNN nem marad le sokkal a komplexebb ComplexCNN modellhez képest. A 4.25 ábrán egy összehasonlító oszlopdiagram látható ezekről a mérésekről.



4.25. ábra. Oszlopdiagram a Figshare adathalmazos mérésekről



4.26. ábra. Oszlopdiagram a Kaggle adathalmazos mérésekről

5. fejezet

Alkalmazásfejlesztés

5.1. Szoftver áttekintése

Egy olyan proof of concept web alkalmazást készítettem, ami lehetővé teszi az agyi MRI képek feltöltését és ezek osztályozását a daganatos és nem daganatos osztályokba, ezzel potenciálisan segítve a jövőben egy orvos munkáját.

5.1.1. Felhasználói követelmények

A felhasználó feltölt egy agyi MRI képet, ezután le tudja olvasni a valószínűségi ábrán megjelenített értéket, ami jelöli, hogy mekkora valószínűséggel daganatos a feltöltött képen lévő agy. A [5.1](#) ábrán látható a rendszer eset diagramja, illetve a [5.2](#) ábrán található a rendszerhez tartozó szekvencia diagram.

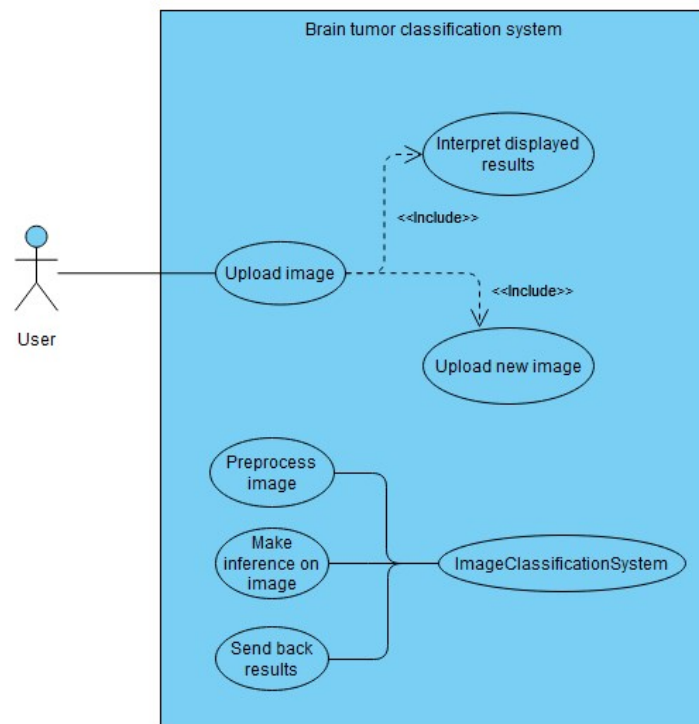
5.2. Rendszerkövetelmények

5.2.1. Funkcionális követelmények

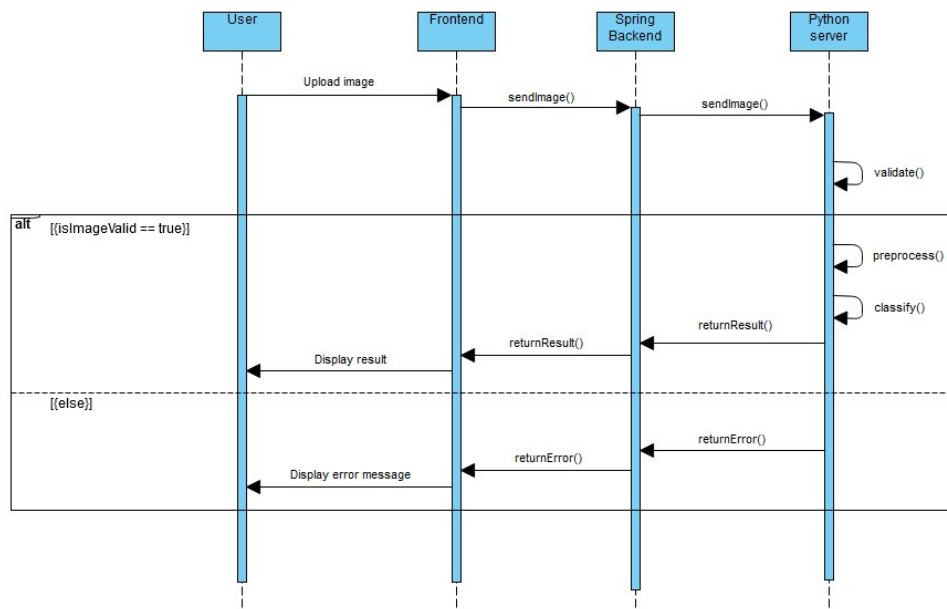
A felhasználó a képernyő közepén lévő négyzetbe behúzza egy .jpg formátumú agyi MRI képet, ezután a szoftver egy valószínűségi ábrán jeleníti meg, hogy 0-tól 100 százalékig mennyire biztos abban, hogy a feltöltött képen az agy daganatos.

5.2.2. Nem funkcionális követelmények

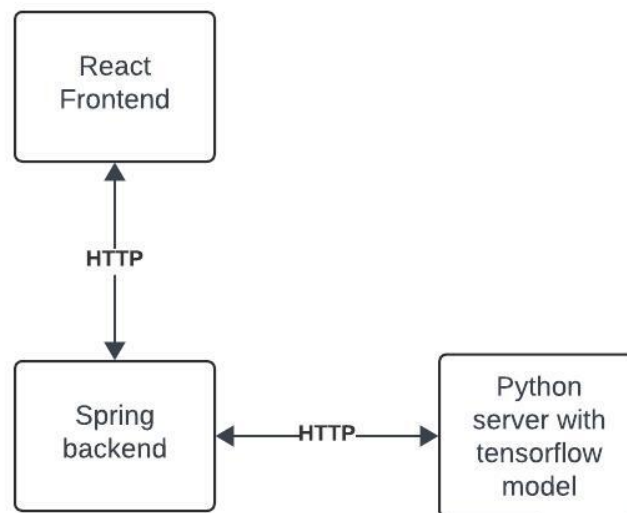
- A szoftver legyen elindítható bárki számára aki egy docker-t feltelepített számítógéppel rendelkezik.
- A szoftver biztosítson megbízható osztályozást amennyiben a képen ténylegesen egy agyi MRI felvétel található
- A szoftver működjön megbízhatóan: a szoftver fejlesztése során a hibalehetőségeket lekezeltem, ha a felhasználónak a felöltött képe valamiért nem megfelelő, például túl nagy a mérete vagy túl kicsi a felbontása, akkor a szoftver visszajelzést ad arról, hogy pontosan mi a baj, ezzel segítve a felhasználót a probléma megoldásában.



5.1. ábra. Használati eset diagram



5.2. ábra. A rendszer szekvencia diagramja



5.3. ábra. A rendszer architektúrája

5.3. Tervezés

A szoftver célja a kutatási rész alatt összehasonlított modellek közül legjobban teljesítő modell kipróbálhatóvá tétele.

5.3.1. A rendszer architektúrája

A szoftver egy React frontend részből, egy Spring backend részből és egy python szerverből áll. A frontend a Next.js keretrendszert használja. A Spring backend jó alapot nyújt a jövőbeli fejlesztéseknek, mint az autentikáció, adatok lementése adatbázisba. A python szerver a tensorflow modellt teszi elérhetővé és a Flask keretrendszert használja. A szoftver architektúra diagramja a 5.3 ábrán látható.

5.3.2. Szerveroldal kivitelezése

A backend részt eredetileg Spring-ben akartam kizárólag de a DJL libraryt és a Tensorflow Java libraryt használva nem sikerült úgy betölteni a modellt, hogy megfelelően osztályozzon. Így lett az a végső terv, hogy készül egy központi backend szerver Spring-ben, ami a jövőben stabil alapot biztosít a fejlesztéseknek (például autentikáció vagy adatbázis), ez a központi szerver kommunikál egy Python szerverrel, amelybe be van töltve konkrétan a Tensorflow modell. Ez a Python szerver a Flask framework segítségével kezeli le a HTTP kéréseket, és az openCV library segítségével végzi el a képeken a preprocesszálást. A Spring backend a Controller-Service-Repository mintát követi, viszont, mivel jelenleg nincs adatbázis használva, ezért csak controller meg service osztály található benne. A **ClassificationController** osztály tartalmazza a `classify()` végpontot, amely egy `MultipartFile` példányt kap meg paraméterként, ez az osztályozandó kép. Ez a végpont hívja meg a **ClassificationService** példányban a `classify()` függvényt, amely készít egy HTTP kérést benne a képpel és elküldi a python szervernek. Ha a kérés sikerte-

len volt, akkor visszaküldött hibakódot lekezeli és visszaküldi válaszként a frontend-nek. A python szervernek van egy `getPrediction()` végpontja, amely validálja a payload-ban jött file-nak a kiterjesztését és méretét. A következő lépés a kép preprocessálása, a kép az agy kontúrja mentén kivágódik, a megfelelő felbontásra méreteződik (amit a modell bemenete elvár) és normalizálódik, azaz a pixelek értékei elosztódnak 255-el és az értékük 0 és 1 között lesz. Ezeket a preprocessáló függvényeket a Mohammed et al. munkájából [5] vettem. A preprocessálás után a képet megkapja a tensorflow modell és osztályozza azt, ennek az eredménye visszatérítődik a Spring backendnek.

5.3.3. Szerver elindítása és a weboldal hosztolása

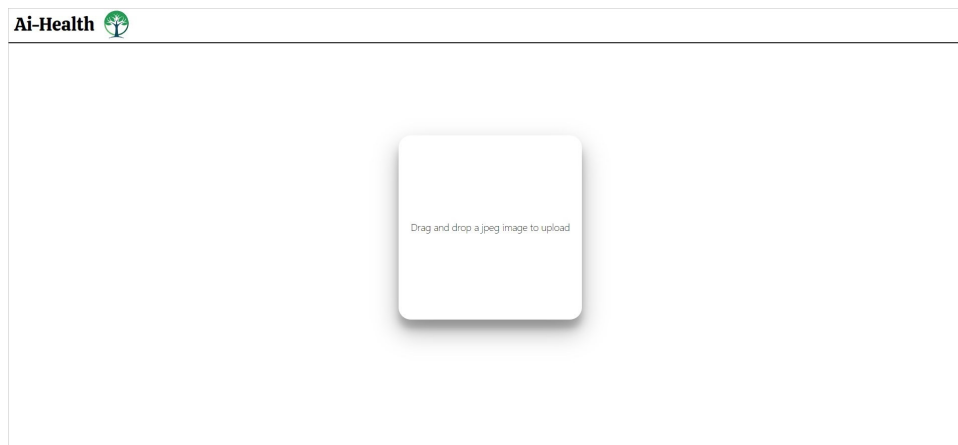
Az egész alkalmazást a kezdetektől fogva konténerizálni akartam, hogy könnyen lehessen hosztolni a felhőben. Ez docker-el van megoldva, illetve a konténerek közti hálózat beállítását docker compose segítségével oldottam meg. Jelenleg a program lokálisan futtatható a ***docker compose up*** paranccsal, a jövőbeli tervem, hogy egy ingyenes kategóriájú Amazon AWS EC2 instance-ben legyen hosztolva.

5.3.4. Kliensoldal kivitelezése

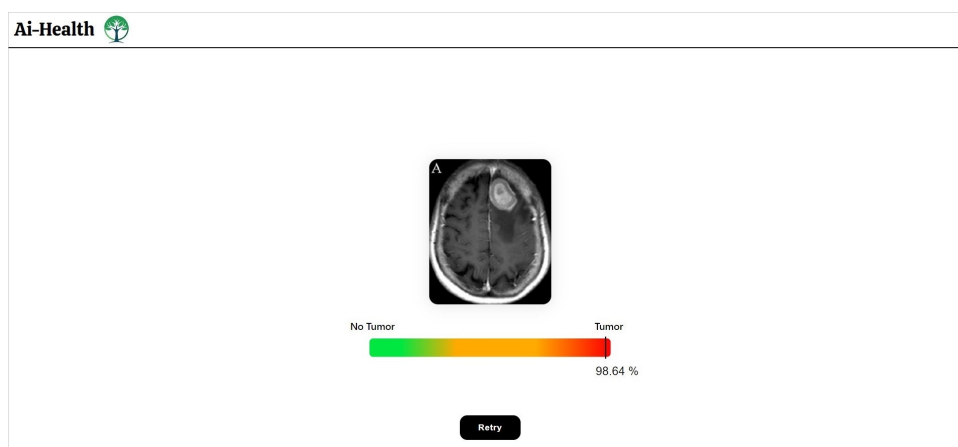
A frontendet a NextJs React framework-ben készítettem, ez is jó alapot biztosít a jövőbeli potenciális bővítésekhez. A funkciót, hogy behúzásra fel lehessen tölteni a html onDrop esemény lekezelésével és a File API segítségével valósítottam meg, így töltődik be a memóriába a kép, ezt egy multipart/form-data kéréssel és a Fetch Api segítségével küldődik el a backendnek. A projektben mindenik főbb komponens saját fileban van és külön funkcionális komponensre van bontva, ilyenek a: FileDragField, az ImagePreview és a LinearGauge funkcionális komponensek. A FileDragField komponens felel azért, hogy a kép a megfelelő négyzetbe való behúzáskor elküldődjön a backendnek. A LinearGague komponens jeleníti meg egy diagramon a klasszifikáció eredményét, ha sikeres volt a HTTP kérés a backendhez. Az ImagePreview komponens, pedig megjeleníti a feltöltött képet, ezáltal is visszajelzést adva a felhasználónak hogy biztosan jó képet töltött fel. A 5.4 képen a webalkalmazás látható, amiután a felhasználó megnyitja azt. A 5.5 képen a webalkalmazás látható, amiután a felhasználó feltölt egy képet és arra megjelenik egy osztályzás. A 5.6 képen a webalkalmazás látható, amiután a felhasználó feltölt egy rossz formátumú képet.

5.3.5. Verziókövetés

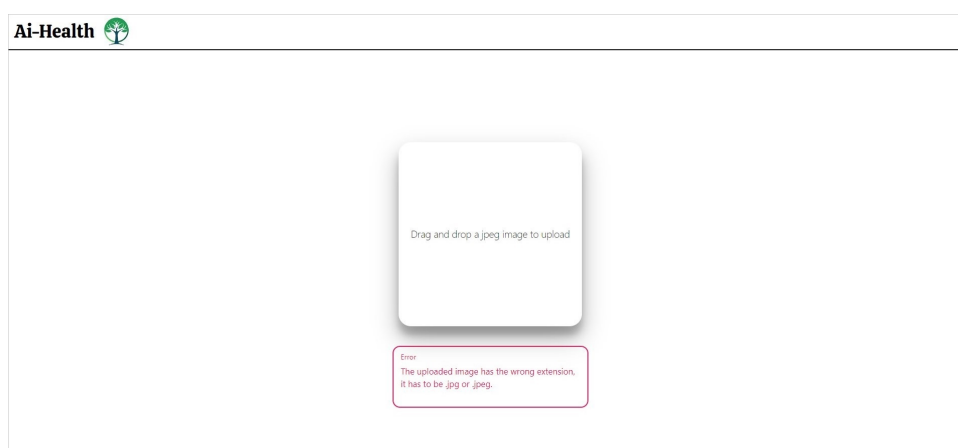
A verziókövetést mind a három szoftverrész esetében a git-el valósítottam meg és Github repository-t készítettem nekük.



5.4. ábra. A felhasználót fogadó nézet.



5.5. ábra. Az osztályzás eredményét ábrázoló nézet.



5.6. ábra. Egy hibát követően megjelenő hibaüzenet

6. fejezet

Összegzés

Dolgozatomban összehasonlítottam a ComplexCNN konvolúciós neurális hálót az eredeti VGG16 hálóval, illetve a sokkal egyszerűbb SimpleCNN hálóval. Elkészítettem egy webalkalmazást, ami gyakorlatba ülteti a betanított modellekből a legjobbat és lehetővé teszi, hogy olyan is tudja használni, aki nem ért a programozáshoz, például egy orvos. A szoftver a feltöltött agyi MRI kép osztályozásáról kapott eredményt vizuálisan ábrázolja.

A kutatási részből következtetésként arra jutottam, hogy a Kaggle adathalmazon nincs szignifikáns eltérés a VGG16 és a ComplexCNN között de jobban teljesítenek, mint a SimpleCNN. A Figshare adathalmazon csak a ComplexCNN és a SimpleCNN lett tesztelve memória problémák miatt, ahol a ComplexCNN jobb eredményeket ér el. A méréseim ellentmondanak Saikat et al. tanulmányában kapott eredményekkel, megmutatják, hogy a ComplexCNN nem jobb az agyi tumorok osztályozásában, mint az összes többi létező háló, sőt a transfer learning alapon betanított VGG16 a Kaggle adathalmaz esetében kicsit jobban is teljesít nála.

Továbbfejlesztési lehetőségek közé tartozik, hogy az alkalmazást ellássam autentikációval, adatbázissal a régebb feltöltött képek eltárolására és újra megjelenítéséhez. Ez az alkalmazás csak szemléltető jellegű, nem orvosoknak készült, mert nem kértem ki szakorvosnak a véleményét a felhasznált adathalmazokról, illetve szerintem ennél jobban kellene tesztelni ismeretlen, nem látott képekre, hogy erre potenciálisan alkalmas legyen, jelenleg a teszt halmazban vannak olyan képek, amik augmentált változatai az eredetieknek, ez torzíthatja kicsit az eredményeket. Ezeknek a teljesítése is egy jövőbeli célkitűzés lehet.

A munkámhoz használt github repository-k a következő linkeken érhetők el:

<https://github.com/aszilard99/Ai-Health-CNN-models>

<https://github.com/aszilard99/ai-health-frontend>

<https://github.com/aszilard99/ai-health-backend>

https://github.com/aszilard99/py_pred_server

Ábrák jegyzéke

3.1. Egy tipikus konvolúciós neurális háló architektúrája [8]	14
3.2. Konvolúciós réteg működése [9]	16
3.3. Pooling réteg működése [10]	17
3.4. Sűrű réteg [11]	18
3.5. Gyakran használt aktivációs függvények [11]	19
4.1. VGG16 neurális háló architektúrája [7]	22
4.2. A ComplexCNN neurális háló architektúrája [15]	22
4.3. SimpleCNN neurális háló architektúrája	23
4.4. Tanítási idők	25
4.5. ComplexCNN tanítási és validációs pontossága a Kaggle adathalmazon	26
4.6. ComplexCNN tanítási és validációs pontossága a Figshare adathalmazon	26
4.7. VGG16 tanítási és validációs pontossága a Kaggle adathalmazon	27
4.8. SimpleCNN tanítási és validációs pontossága a Kaggle adathalmazon	27
4.9. SimpleCNN tanítási és validációs pontossága a Figshare adathalmazon	28
4.10. ComplexCNN háló konfúziós mátrixa a Kaggle adathalmazon	29
4.11. ComplexCNN eredményei a Kaggle adathalmazon	29
4.12. ComplexCNN ROC görbéje a Kaggle adathalmazon	29
4.13. VGG16 háló konfúziós mátrixa a Kaggle adathalmazon	30
4.14. VGG16 eredményei a Kaggle adathalmazon	30
4.15. VGG16 ROC görbéje a Kaggle adathalmazon	30
4.16. A SimpleCNN háló konfúziós mátrixa a Kaggle adathalmazon	31
4.17. A SimpleCNN a Kaggle adathalmazon	31
4.18. A SimpleCNN ROC görbéje a Kaggle adathalmazon	31
4.19. ComplexCNN háló konfúziós mátrixa a Figshare adathalmazon	32
4.20. ComplexCNN eredményei a Figshare adathalmazon	32
4.21. A SimpleCNN háló konfúziós mátrixa a Figshare adathalmazon	33
4.22. A SimpleCNN a Figshare adathalmazon	33
4.23. Mérés összehasonlító táblázat a Kaggle adathalmazon	34
4.24. Mérés összehasonlító táblázat a Figshare adathalmazon	34
4.25. Oszlopdiagram a Figshare adathalmazos mérésekről	35
4.26. Oszlopdiagram a Kaggle adathalmazos mérésekről	35
5.1. Használati eset diagram	37
5.2. A rendszer szekvencia diagramja	37
5.3. A rendszer architektúrája	38
5.4. A felhasználót fogadó nézet.	40

5.5. Az osztályzás eredményét ábrázoló nézet.	40
5.6. Egy hibát követően megjelenő hibaüzenet	40

Irodalomjegyzék

- [1] Figshare adathalmaz. https://figshare.com/articles/dataset/brain_tumor_dataset/1512427.
- [2] Harvard medical adathalmaz. <https://www.med.harvard.edu/AANLIB>.
- [3] Kaggle 3264 agyi mri képeket tartalmazó adathalmaz. <https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri>.
- [4] Kaggle adathalmaz. <https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection>.
- [5] Mohammed et al. brain-tumor-detector github. <https://github.com/MohamedAliHabib/Brain-Tumor-Detection>. (elérés dátuma: 2024. marc. 17.).
- [6] Saikat et al. brain-tumor-detection github. <https://github.com/saikat15010/Brain-Tumor-Detection>. (elérés dátuma: 2024. marc. 17.).
- [7] Vgg16 architektúra. <https://www.learndatasci.com/tutorials/hands-on-transfer-learning-keras/>. (elérés dátuma: 2024. apr. 06.).
- [8] Konvolúciós neurális háló architektúra. <https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/>. (elérés dátuma: 2024. jan. 14.).
- [9] Konvolúciós réteg működése. <https://www.ibm.com/topics/convolutional-neural-networks>. (elérés dátuma: 2024. jan. 14.).
- [10] Pooling réteg működése. <https://medium.com/aiguys/pooling-layers-in-neural-nets-and-their-variants-f6129fc4628b>. (elérés dátuma: 2024. jan. 14.).
- [11] Convolutional neural networks cheatsheet by afshine amidi and shervine amidi. <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks>. (elérés dátuma: 2024. jan. 14.).
- [12] BADŽA, M. M., AND BARJAKTAROVIĆ, M. Classification of brain tumors from mri images using a convolutional neural network. *Applied Sciences* 10 (2020).
- [13] CHENG, J., HUANG, W., ET AL. Enhanced performance of brain tumor classification via tumor region augmentation and partition. *PLoS ONE* 10 (2015), 861–874.

- [14] HUBEL, D. H., AND WIESEL, T. N. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology* 195 (1968), 215–243.
- [15] KHAN, M. S. I., ET AL. Accurate brain tumor detection using deep convolutional neural network. *Computational and Structural Biotechnology Journal* 20 (2022), 4733–4745.
- [16] KINGMA, D. P., AND BA, J. Adam: A method for stochastic optimization. *arXiv* (2017).
- [17] SAEEDI, S., REZAYI, S., KESHAVARZ, H., ET AL. Mri-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques. *BMC Medical Informatics and Decision Making* 23 (2023).
- [18] SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. <https://doi.org/10.48550/arXiv.1409.1556>, 2015.