

EÖTVÖS LORÁND TUDOMÁNYEGYETEM

TERMÉSZETTUDOMÁNYI KAR

ORVOSI KÉPFELDOLGOZÁS NEURÁLIS HÁLÓKKAL

SZAKDOLGOZAT

Börcs Dóra

Matematika BSc

Matematikai elemző szakirány

Témavezető: Lukács András

Számítógéptudományi Tanszék



ELTE

EÖTVÖS LORÁND
TUDOMÁNYEGYETEM

Budapest

2023

Köszönetnyilvánítás

Köszönöm témavezetőmnek, Lukács Andrásnak, hogy bevezetett a mély tanulás és a neurális hálók világába, valamint hogy fogyhatatlan türelemmel kísért végig az idáig vezető úton. Külön köszönöm Vas Bernadettnek, hogy bármikor kereshettem ha elakadtam és idejét nem sajnálva segített túllépni a felmerülő akadályokon.

Köszönöm a barátaimnak és családomnak, hogy tartották bennem a lelket és bátorítottak, amikor számomra lehetetlennek tűnt a szakdolgozatom elkészülte.

Tartalomjegyzék

1. Bevezetés	4
2. Neurális hálók	6
2.1. Konvolúciós hálók	6
2.2. Szegmentáció	9
3. U-Net	10
3.1. Architektúra és tanítás	10
3.2. Attention gate a U-Netben	12
4. Feladat	15
4.1. Adathalmaz, mérések	15
4.2. Tüdő szegmentáció	17
4.3. Fertőzés szegmentáció	19
4.4. Attention Gate	21
5. Összefoglalás	23

1. Bevezetés

Hazánkba 2020 első felében tört be a COVID-19 járvány, mely alapjaiban forgatta fel az életünket, bár dolgozatom írásakor csillapodni látszik a járvány, a járvány csúcsán regisztrált majd 300.000 aktív fertőzötthöz képest 2022.12.28-án csak 13.571-et tartanak számon és a kórházi ápolásra szorulóknak száma is drasztikusan, 12.553-ról 770-re csökkent. A tüdő röntgenfelvételek a járvány kitörésétől fogva fontos szerepet játszottak a diagnosztizálásban [8]. Egyrészt a tesztek pontossága miatt, ugyanis az RT-PCR típusú tesztek 2020-ig csak 71%-ban adtak valós eredményt [2]. Másrészt ezeket a teszteket sokszor csak a tüneteket (köhögés, láz, stb.) mutató betegeken végzik, ugyanakkor a tüdőben ezen tünetek jelenléte nélkül is megfigyelhető elváltozás. Illetve a betegség előrehaladottságának felismerésének egyik legfontosabb eszközei ezek a tüdőfelvételek, ami lehetővé teszi az orvosok számára a helyes kezelési mód elrendelését és a megbízható prognózis felállítását.

A mesterséges intelligencia (MI) a diagnosztizáció olyan területein is áttörést hozott, amire elsőre nem is gondolnánk.

Különösen egy ilyen fertőző betegségnél fontos, hogy az egészségügyi szakember a beteggel csak minimális időre kerüljön kontaktba, ezzel is csökkentve a megfertőződés valószínűségét. Egészen a közelmúltig egy mellkas CT vagy tüdőrontgen elkészítésekor elkerülhetetlen volt az orvosi személyzet jelenléte, ugyanis olyan paramétereket kell meghatározni a felvétel készítése előtt, amik nagyban befolyásolják annak minőségét, mint például hogy a páciens mellkasának pontosan mely részéről készüljön a röntgen ill. CT felvétel. Az MI segítségével a modern orvosi képalkotó műszerek ezen paraméterek túlnyomó többségét már maguktól állítják be, például adott testrészek, szervek felismerése után [8].

A fertőzés felismerése és kezelése szempontjából azonban mégis a különböző képalkotási módszerekkel készült felvételek klasszifikációja és szegmentációja bír óriási jelentőséggel. Ezen feladatok az orvosok számára sem egyszerűek, hiszen a fertőzés felismeréséhez tapasztalatra van szükség, ami a járvány kitörésekor nem állt rendelkezésre, ezen kívül nagy mennyiségű adatot kell rövid idő alatt feldolgozni. Egyetlen egy darab CT felvétel több száz rétegből áll, ezek vizsgálata rengeteg időt vesz igénybe. Ezekre a problémákra jelentettek megoldást a neurális hálók - köztük a dolgozatomban szereplő U-Net architektúra - , melyek viszonylag kevés tanító adat felhasználásával már elég jó pontossággal tudják végezni a klasszifikációt ill. szegmentációt. Ezek a hálók leggyakrabban röntgenképekkel

dolgoznak, bár a 2D vetítés és a bordák takarása miatt így nehezebb feladatot jelent a fertőzés felismerése gépi tanulási módszerekkel, egyszerűbb elkészítése miatt mégis ez az elterjedtebb képalkotási eljárás.

Dolgozatomban, a U-Net architektúráját felhasználva szeretnék elérni minél jobb eredményt a tüdő és fertőzés szegmentációjakor. Felteszem, hogy az olvasó tisztában van a gépi tanulási módszerek alapfogalmaival, ezeket csak megemlíten, részletesen nem írok róluk.

Dolgozatom második fejezetében bemutatom a konvolúciós neurális hálók általános felépítését illetve a szegmentációs feladatot. A harmadik fejezetben a U-Net architektúrájáról és ennek egy továbbfejlesztési lehetőségéről, az attention gate-ről lesz szó. Ezután a negyedik fejezetben prezentálom az ezen az architektúrán alapuló modellekkel elért elért eredményeimet . Az ötödik, egyben utolsó fejezetben pedig összefoglalom a szakdolgozat elkészülte és a hál tanítása során tapasztaltakat. A felhasznált kód a következő linken elérhető: <https://github.com/borcsdori/SzakdolgozatBorcsDora>

2. Neurális hálók

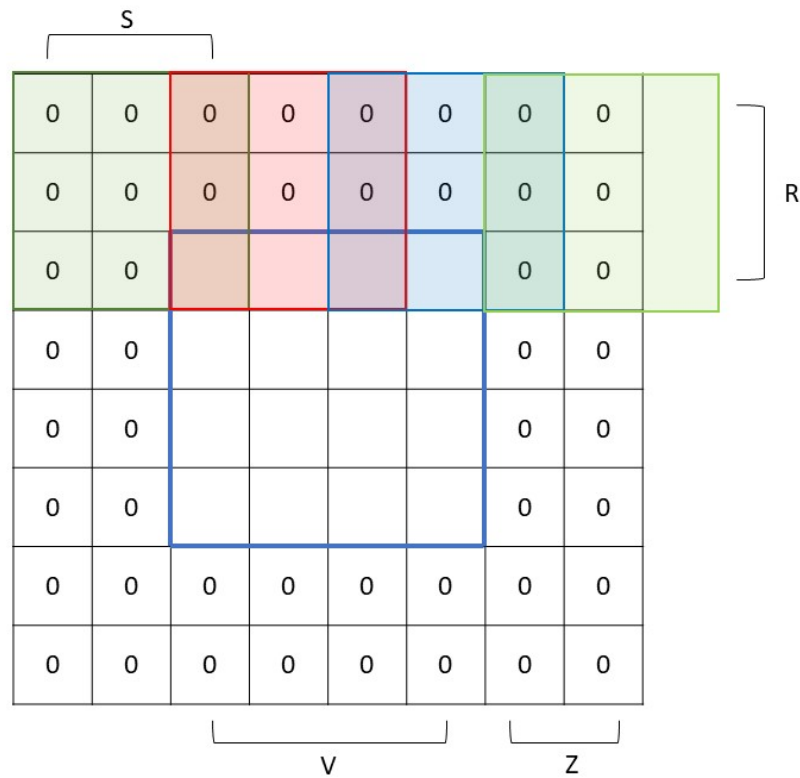
2.1. Konvolúciós hálók

A neurális hálók képfeldolgozásban legelterjedtebb változata a konvolúciós háló (CNN), amely bár sok mindenben hasonlít a sűrű neurális hálókhoz, mégis sokkal jobban teljesít a képek mintázatainak felismerésekor és kevesebb paraméter is elég ahhoz, hogy jól működjön. Ennek oka, hogy a konvolúciós neurális hálók architektúrájában elkódolhatunk képspecifikus tulajdonságokat, ezzel javítva a teljesítményt. A fejezetben [5] alapján szeretném beutatni a konvolúciós neurális hálókat.

Egy konvolúciós háló tipikusan háromféle rétegből állhat. Ezek a konvolúciós, összevonó (pooling) és sűrűn kapcsolt (fully connected vagy FC) rétegek. A konvolúciós réteg bemenete három dimenzióból áll, a magasságból, szélességből, illetve a harmadik dimenzió a kép csatornáinak száma. Ez egy színes képnél tipikusan három csatornát jelent, míg egy szürkeárnyalatosnál - az orvosi képek túlnyomó része, így a tüdő röntgen képe is ilyen - egy csatornát. A konvolúciós háló hatékonyságának legfőbb oka, hogy a rétegekben található neuronok csak az előző neuronok egy kis részéhez kapcsolódnak, ezzel csökkentve a szükséges számítások számát. Azt, hogy az adott rétegben ez mekkora környezetet jelent az úgynevezett szűrő (filter) mérete határozza meg. Ez a filter végighalad az előző neuronok kimenetén és egy aktivációs térképet (feature map) hoz létre a súlyozott összegekből. A konvolúciós réteg kimenetének mérete három hiperparamétertől függ:

- A kimeneti mélység határozza meg, hogy hány filtert fogunk a rétegen belül alkalmazni. Ezen filterek mindegyike egy-egy aktivációs térképet határoz meg, melyeket egymásra rakva megkapjuk a kimenet csatornáinak számát.
- A lépésköztől (stride) függ, hogy miközben a filter végighalad a bemeneten, mekkora lépéseket tesz. Ha például a stride 2-re van állítva, akkor minden alkalommal, mikor arrébb csúsztatjuk a filter, két pixelnyit fog haladni.
- A zero-paddinggel pedig bemenet széleire például 0 értékeket rakva befolyásolhatjuk a kimenet szélességét, illetve magasságát.

Fontos megjegyezni, hogy a lépésköz és a padding paraméterek nem függetlenek egymástól. Egy egyszerű példát tekintve:



1. ábra. Konvolúciós réteg hiperparamétere

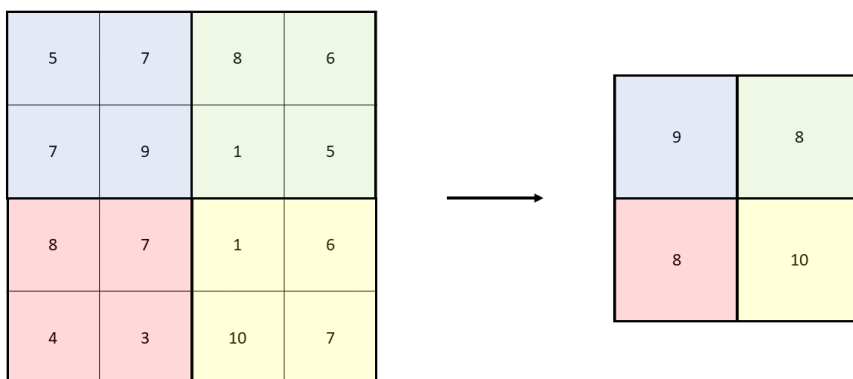
Az ábrán jól látható, hogy csatornánként 4x4-es bemenet esetén 3x3-as filtert, 2-es paddinget és 2-es lépésközt használva a szűrők egy része "nem fér rá" a bemenetre. A hiperparaméterek akkor vannak jól beállítva, ha az alábbi képlet eredménye egész szám.

$$\frac{(V - R) + 2Z}{S + 1}$$

Ahol V a bemenet mérete (magassága vagy szélessége), R a szűrő mérete, Z a padding, S pedig a lépésköz.

A pooling réteg segítségével csökkenthetjük a bemenet szélességét és magasságát, ezzel redukálva a szükséges paraméterek mennyiségét, míg a csatornák száma változatlan marad. Ebben a rétegben a konvolúciós réteghez hasonlóan végighaladunk az aktivációs térképen, de a tanult súlyozott összeg helyett itt egy fix függvény - leggyakrabban átlag vagy maximum - eredménye kerül a kimenetbe. Például 2x2-es filter és 2-es lépésköz esetén

így:



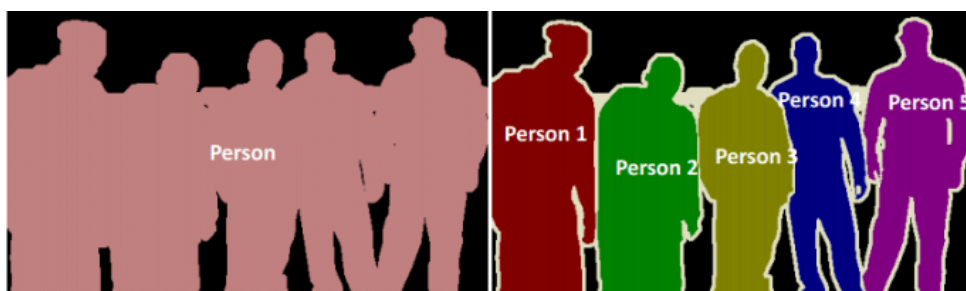
2. ábra. Maxpooling

A pooling hatékonyan csökkenti a bemenetének méretét, jellemzően 2x2-es szűrővel használjuk.

A sűrűn kapcsolt (fully connected) réteg neuronjai pedig össze vannak kötve a két szomszédos réteg összes neuronjával. Leggyakrabban a CNN utolsó rétege(i)ként használunk sűrű rétegeket, az osztályozáshoz.

2.2. Szegmentáció

A konvolúciós neurális hálókat leggyakrabban képek klasszifikációjára használják. Sok esetben azonban nem csak azt szeretnénk tudni, hogy mit látunk az adott képen, hanem azt is, hogy a keresett az objektum(ok) a kép melyik részén található(ak). Ennek a feladatnak alkalmazási területe például az orvosi képelemzés, önvezető autók, vagy a videós megfigyelőrendszerek. A képek szegmentációját úgy is felfoghatjuk, mint az őket alkotó pixelek klasszifikációját. Ezzel a módszerrel a képet partíciókra vagy objektumcsoportokra bontjuk, és ezekhez a csoportokhoz rendelünk címkéket. A szegmentáció két fajtáját különböztetjük meg abból a szempontból, hogy az objektumcsoportokat tovább osztjuk-e objektumokra vagy sem. Semantic segmentation-ről beszélünk, ha az egyes objektumokat nem különböztetjük meg egymástól, ahogyan a 3. ábra bal oldalán látható. A szegmentálásakor kapott maszk megkülönbözteti a képen látható embereket a háttértől, de egymástól nem. Instance segmentation használatakor az partíciókon belül az egyes objektumok is különböző címkét kapnak, ahogyan a 3. ábra jobb oldalán is látszik.

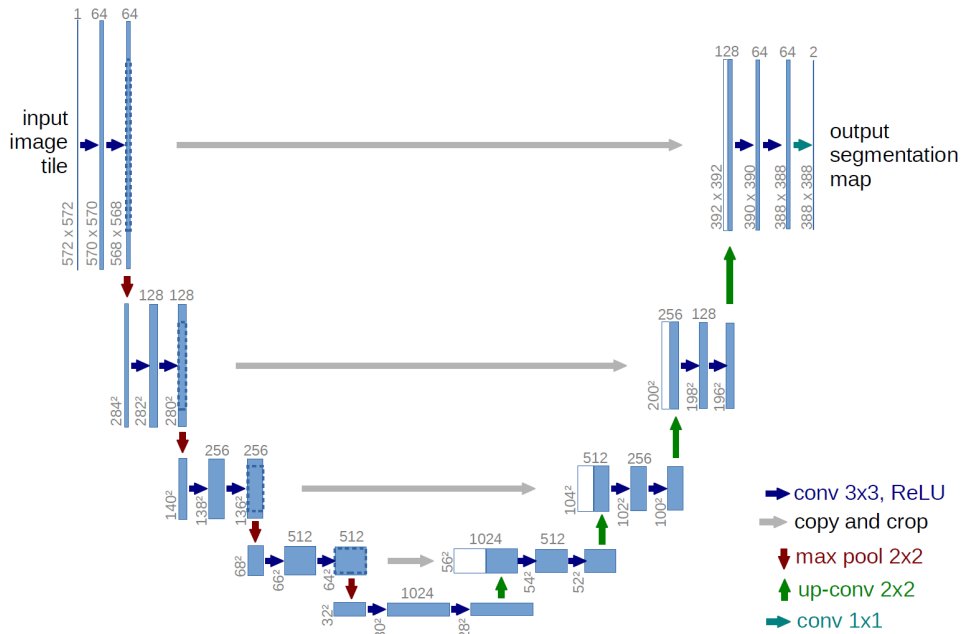


3. ábra. Szegmentáció [9]

Az első szegmentációra használt konvolúciós neurális hálók az orvosi képek elemzésének kapcsán kezdtek el kialakulni, fejlődni. Az elsők között volt Dan Cireşan et al. architektúrája [4], mely 2012-ben az ISBI versenyén messze a mezőny fölött teljesített, a képek pixelein egy keretet végigcsúsztatva, a képeket a pixelek körüli régiókra felosztva. Ez a módszer egyrészt lassú volt, mert minden egyes ilyen régiónak egyenként kellett végigfutni a hálón, másrészt pedig nem lehetett igazán jól optimalizálni a régió méretét. Nagy méretű régiók esetén több pooling réteget kellett használni, ami a lokalizációs pontosságon rontott, kisebb méretű régiók esetén viszont nem rendelkezett információval a kép többi részéről.

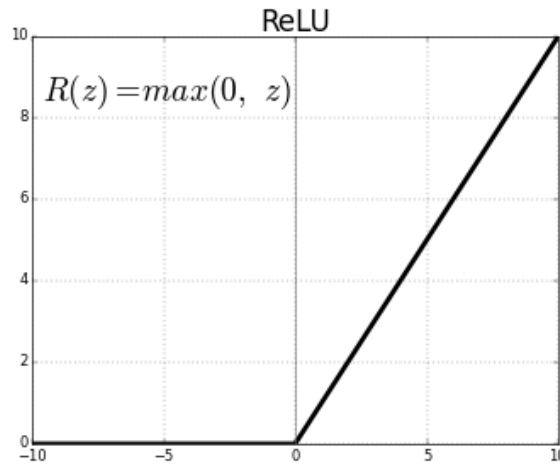
3. U-Net

3.1. Architektúra és tanítás



4. ábra. U-Net Architektúra [7]

Az addigi modellek lassúságára és rossz optimalizálhatóságára mutattak megoldást a [7] cikk írói a U-Net formájában. A háló architektúrája két részből - contracting és expansive path - épül fel. A 4. ábra bal oldalán megfigyelhető contracting path-ben minden szinten két konvolúciós réteget használunk 3x3-as kernelmérettel, 1-es lépésközzel, padding nélkül. A konvolúciós rétegeket minden esetben az 5. ábrán látható ReLU aktívációs függvény követi. Az első konvolúciós rétegben - az első szint kivételével - duplájára növeljük a csatornák számát.



5. ábra. ReLU aktivációs függvény

A szintek között downsampling-et végzünk egy 2x2-es maxpooling réteg használatával, így a bemenet mérete - a szélessége és a magassága - a felére csökken. Az expansive path szintjeiben egy up-sampling után egy konvolúciós réteggel duplázzuk meg a bemenet méretét és felezzük csatornák számát, majd a contracting path azonos szintjének kimenetét a megfelelő átméretezés után hozzáfűzzük (skip connection). A konkatenációt szintenként két konvolúciós réteg követi 3x3-as kernelmérettel. A konvolúciós rétegeket az expansive path-ben is ReLU nemlinearitások követik. Az utolsó szintet egy 1x1-es konvolúciós réteg követi, ami az architektúra klasszifikáló rétege. A U-Net sztochasztikus gradiens módszert használ tanításkor. A veszteségfüggvény két részből épül fel. Először az alábbi softmax függvényt futtatjuk végig a kimeneten:

$$p_k(x) = \frac{\exp a_k(x)}{\sum_{k'=1}^K \exp a_{k'}(x)}$$

ahol $a_k(x)$ az x helyen lévő pixelt jelenti a k -adik aktivációs csatornán ($x \in \Omega$; $\Omega \subseteq \mathbb{Z}^2$), K az osztályok száma, $p_k(x)$ pedig a becült maximum függvény. Ezután a következő cross entropy függvénnyel számoljuk ki a becült és a tényleges osztály eltérését:

$$E = \sum_{x \in \Omega} w(x) \log(p_{l(x)}(x))$$

ahol $l : \Omega \rightarrow \{1, \dots, K\}$ a pixelek tényleges osztálya és $w : \Omega \rightarrow \mathbb{R}$ a súly térkép (weight

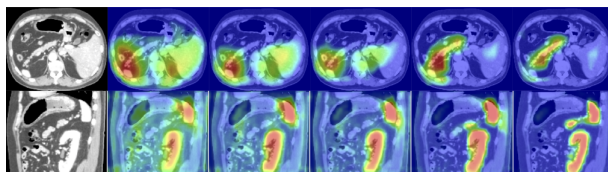
map), ami az egyes pixelekhez nagyobb jelentőséget társít, mint a többihez. Erre azért van szükség, hogy a hálót rákényszerítsük az élek, azaz a különböző osztályba sorolt, de térben egymás közelében lévő pixelek osztályának felismerésére. A weight map értékeinek inicializálása nagyon fontos, ugyanis ha rosszul inicializálunk, fontos részek elveszhetnek a tanulás során és helyettük kevésbé fontos jellemzők kerülnek előtérbe. Egy ilyen struktúrájú háléhoz a legjobbnak az a megoldás bizonyult K. He cikke [3] alapján, hogy a súlyokat $\sqrt{2/N}$ átlagú standard normális eloszlásból vesszük, ahol N az adott neuronba beérkező élek száma.

3.2. Attention gate a U-Netben

Az eddigi modell eredményeinek további fejlesztéséhez felhasználhatunk egy másik módszert a kép fontos részeinek kiemelésére. Az attention gate (AG) modell megkeresi a képen azokat a pixeleket, amik a szegmentálás szempontjából fontosak és ezeknek nagyobb jelentőséget tulajdonít, azaz nagyobb súlyt kapnak. Egy tüdőrontgenen a fertőzött terület szegmentálása szempontjából például sokkal nagyobb hangsúlyt fordít magára a tüdőre, hiszen a fertőzést ezen belül kell megtalálnunk. Ezzel a módszerrel bizonyos esetekben kiválthatjuk, hogy egy már előre feltanított modellel először szegmentáljuk a tüdőt, enélkül is hasonlóan jó eredményeket érhetünk el felesleges számítások nélkül.

Az AG-knek két típusát különböztethetjük meg:

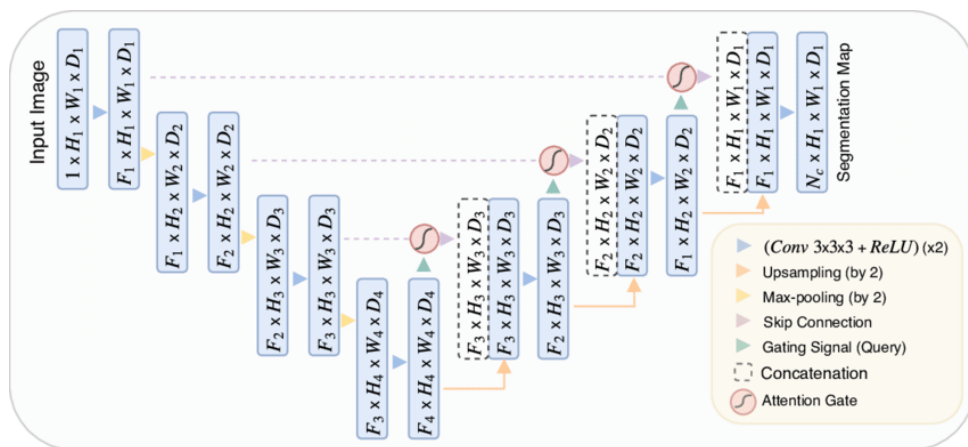
- Hard attention használatakor kivágjuk a képek azon részeit, amik a feladat - a mi esetünkben a tüdő- vagy fertőzésszegmentáció - szempontjából fontosak és csak ezeket vesszük figyelembe a konvolúciós háló további tanítása során. Ezt a változatot használva az egyes pixelek csak kétféle címkét kaphatnak: vagy fontos területnek számítanak, vagy nem. Nincsen köztes állapot. A kép bizonyos részeinek kivágása nem differenciálható művelet, ezért ezzel a módszerrel kizárjuk a backpropagation használatának a lehetőségét.
- Ezzel szemben ha soft attentiont használunk, a pixeleket a bináris címkék helyett súlyozzuk $\alpha_i \in [0, 1]$ súlyokkal. A feladat szempontjából fontos részek nagyobb, míg a kevésbé fontos részek kisebb, nullához közeli súlyokat kapnak. Ez a művelet már differenciálható, tehát használhatjuk a backpropagationt, így az attention gate súlyai is tanulnak a konvolúciós háló tanítása közben.



6. ábra. Az AG által kiemelt területek 3, 6, 10, 60 ill. 150 epoch után [6]

Ahogy a 6. ábrán láthatjuk, egyre több epoch lefutása után egyre jobban körvonalazódik a kiemelt terület, ami ebben a példában egy hasi CT képen a hasnyálmirigy, a vese és a lép kiemelését jelenti.

Az AG-eket sokféle feladathoz, sokféle modellben felhasználhatjuk, dolgozatomban a U-Net-beli implementációt szeretném bemutatni Ozan Oktay et al. cikke [6] alapján.



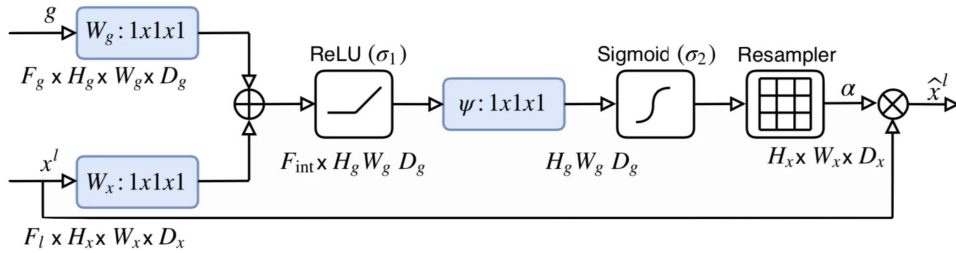
7. ábra. Az AG beépítése a U-Net modellbe [6]

Az attention gate-eket a U-Netben a 7. ábrán látható helyen, a skip connection után alkalmazzuk minden szinten. Minden egyes AG két bemenetet kap, egyrészt a contracting path azonos szintjének kimenetét - a 8. ábrán g -vel jelölve -, másrészt az expanding path előző szintjének kimenetét - az ábrán x -szel jelölve -. Ezen két bemenet mérete a szintek különbözősége miatt nem egyezik, ezért a ReLU aktivációs függvény előtt szükség van egy-egy átméretező konvolúciós rétegre. A g bemenet esetében ez 1x1x1-es konvolúciót jelent 1x1-es lépésközzel, míg az x bemenet esetében szintén 1x1x1-es konvolúciót, de 2x2-es lépésközzel. A konvolúciós rétegek után már megegyeznek a bemenetek dimenziói, ezért elemenként összeadhatjuk őket. Bár additív attentionnel számolni több erőforrást igényel, jobb accuracy-t lehet elérni vele, ezért ezt használjuk a multiplikatív attentionnel szemben. Az összeadást a következő képlettel írhatjuk le:

$$q_{att}^l = \psi^T(\sigma_1(W_x^T x_i^l + W_g^T g_i + b_g)) + b_\psi$$

$$\alpha_i^l = \sigma_2(q_{att}^l(x_i^l, g_i, \Theta_{att}))$$

ahol $\sigma_2(x_{i,c}) = \frac{1}{1 + \exp(-x_{i,c})}$ sigmoid aktivációs függvény, Θ_{att} pedig a következő paraméterek halmaza: $W_x \in \mathbb{R}^{F_{int} \times F_{int}}$, $W_g \in \mathbb{R}^{F_g \times F_{int}}$, $\psi \in \mathbb{R}^{F_{int} \times F_1}$ lineáris transzformációk, $b_\psi \in \mathbb{R}$, $b_g \in \mathbb{R}^{F_{int}}$ eltolásvektorok. A sigmoid függvénnyel normalizáljuk a súlyokat, majd upsampling után elemenként összeszorozzuk x -szel, végül pedig ezt a kimenetet kapja meg a következő réteg bemenetként.



8. ábra. Attention blokk [6]

4. Feladat

4.1. Adathalmaz, mérések



(a) COVID-19 fertőzés



(b) Egyéb fertőzés



(c) Egészséges tüdő

9. ábra. Példa az adathalmaz egyes osztályaira [1]

A feladathoz felhasznált adatathalmaz forrása a Kaggle COVID-QU-Ex Dataset adatbázisa [1], amely több adatbázisból összegyűjtött mellkas röntgenképekből áll. A felvételek szürkeárnyalatosak, mindegyikük 256x256 pixelből áll. Az adatbázis tartalmaz rossz minőségű képeket is, sokszor látszanak például orvosi műszerek, vezetékek, ez nehezíti a szegmentációs feladatot. A felvételek a tanító-, teszt- és validációs adat besoroláson kívül címkékkel is el vannak látva aszerint, hogy a páciens milyen betegségben szenved. A címkék a következők, a 9. ábrán mindből látható egy-egy példa:

- COVID-19 fertőzés
- Egyéb fertőzés
- Egészséges

Az adathalmaz két részre oszlik, a nagyobb, 33920 tüdőfelvételt tartalmazó részadatbázis 21742 tanítóadatból, 6788 teszt adatból és 5417 validációs adatból áll. Itt a tüdőfelvételekhez csak a tüdő maszk található meg, a fertőzés szegmentációhoz közvetlenül nem voltak felhasználhatók ezek a képek.

COVID-QU-Ex Dataset adatok összetétele - Tüdő szegmentáció			
	COVID-19	Non-COVID	Normal
Tanító adat	7658	7208	6849
Teszt adat	2395	2253	2140
Validációs adat	1903	1802	1712

A többi felvételhez már tüdő és fertőzés maszk is található az adatbázisban, eloszlásuk a következő: 3726 tanító adat, 1166 teszt adat és 1032 validációs adat.

COVID-QU-Ex Dataset adatok összetétele - Fertőzés szegmentáció			
	COVID-19	Non-COVID	Normal
Tanító adat	1862	932	932
Teszt adat	583	292	291
Validációs adat	466	233	233

A kísérleteket az ELTE AI Research Group GPU szerverein végeztem, a Python Pytorch könyvtárát felhasználva, mely éppen az ilyen gépi tanulási feladatokhoz lett kifejlesztve. A mérések pontosságának kiértékeléséhez a Sørensen–Dice együtthatót használtam, mely a helyesen tüdőként vagy fertőzésként klasszifikált pixelek és az összes pixel arányát vizsgálja az alábbi formulával:

$$\frac{2|X \cap Y|}{|X| + |Y|}$$

Bináris értékekkel ezt a következőképpen fogalmazhatjuk meg: Tekintsük az alábbi metrikákat.

- true positive (tp) - Helyesen tüdőként vagy fertőzésként klasszifikálva
- true negative (tn) - Helyesen nem tüdőként vagy fertőzésként klasszifikálva
- false positive (fp) - Helytelenül tüdőként vagy fertőzésként klasszifikálva
- false negative (fn) - Helytelenül nem tüdőként vagy fertőzésként klasszifikálva

Ezekkel a metrikákkal az alábbi módon írható fel az együttható képlete:

$$\frac{2 * tp}{2 * tp + fp + fn}$$

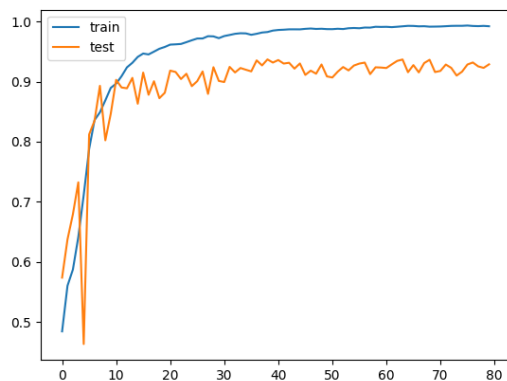
A Sørensen–Dice együtthatót használtam fel a hibafüggvényemben, a DiceLoss-ban is. Az együttható értéke 0 és 1 között változhat, minél több pixelt klasszifikál helyesen a háló,

annál közelebb van 1-hez. Mivel a tanítás során a hibafüggvény értékeinek minimalizálása a cél, ezen értékeket a következőképpen kapjuk meg:

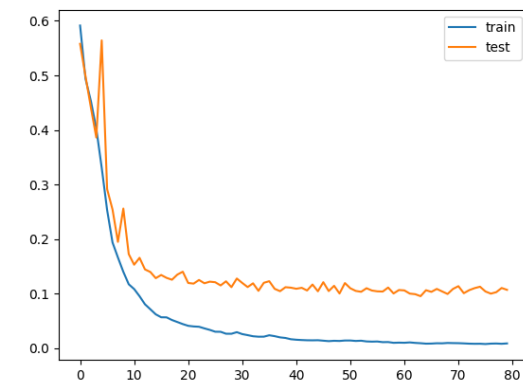
$$1 - \frac{2 * tp + 1}{2 * tp + fp + fn + 1}$$

4.2. Tüdő szegmentáció

A kísérleteket a tüdő szegmentációjával kezdtem. Hogy az eredmények különbsége valóban a hiperparaméterek változását és a tüdő- illetve fertőzés szegmentáció nehézségének különbségét tükrözze, és ne a tanító- és tesztadatok mennyiségét, feladat ezen részénél is csak azokat a felvételeket használtam, melyekhez tüdő és fertőzés maszkok is megtalálhatóak voltak az adathalmazban. A kísérleteknél a U-Net struktúrájából indultam ki, a contracting path-ben same padding-et használva. Ahogyan [7]-ben olvasható, a tanító adathalmaz az ő kísérletüknél 30 képből állt, így először én is ennyi felvétellel próbáltam ki a hálót, az első kísérlet alkalmával adataugmentáció nélkül.



(a) Dice-index

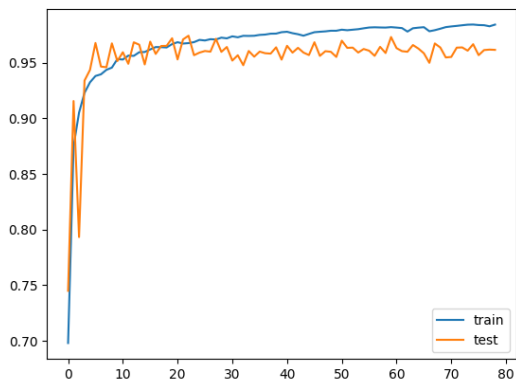


(b) Hibafüggvény

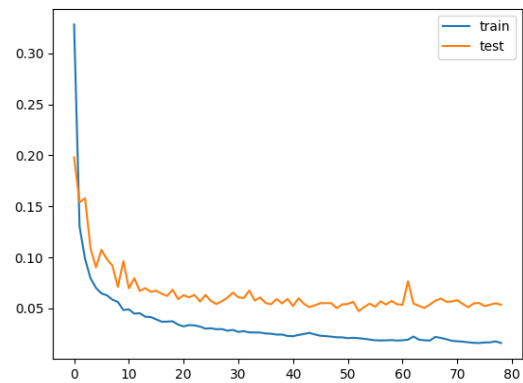
10. ábra. Tüdőszegmentáció adataugmentáció és dropout réteg nélkül

Ahogyan a 10. ábrán is látszik, a teszt adatokon mért Dice index már az első 20 epochban elérte a nagyjából 94%-ot, azonban nem javult tovább. A hibafüggvényen megfigyelhető, hogy a 10. epoch után a tanító és tesztadaton mért függvényértékek távolodni kezdenek egymástól, ebből arra lehet következtetni, hogy a háló túlságosan rátanul a tanító adatokra, nem tud általánosítani, így a teszt adatokon nem szerepel olyan jól, mint

a tanító adatokon. Ezt megelőzve különböző hiperparaméter beállítások megváltoztatásával próbáltam javítani a teljesítményen. Ezek közül a képek mennyiségének növelése vezetett célra, a tanító és teszt adatok mennyiségét is - az adataugmentáció megtartása mellett - 30-ról 300-ra növeltem illetve a túltanulás megelőzése céljából a kimeneti réteg elé beszúrta egy dropout réteget, ami 6%-os valószínűséggel hagyta ki neuronokat, így sikerült 96%-os pontosságot elérnem, illetve a hibafüggvény értékét 0,05 körülire redukálnom, ahogy ez a 11. ábrán megfigyelhető.



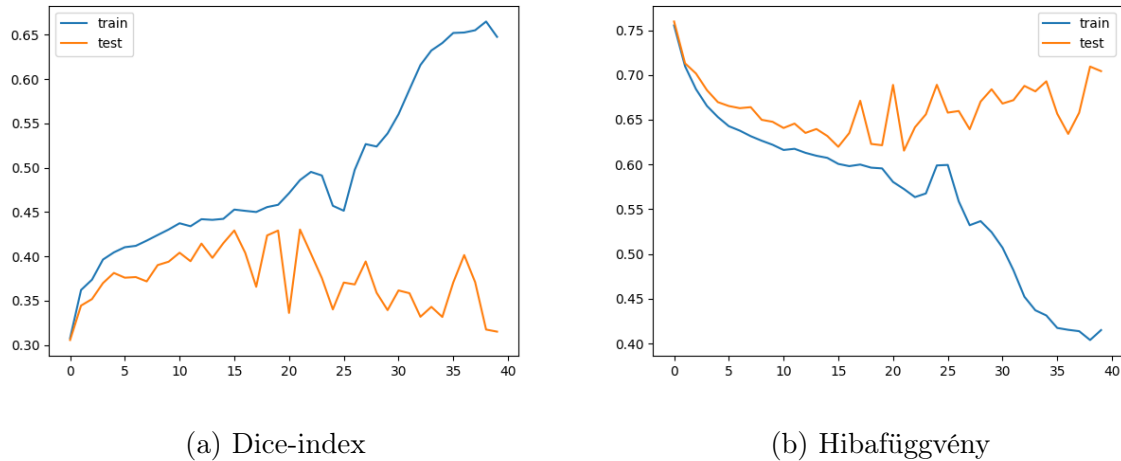
(a) Dice-index



(b) Hibafüggvény

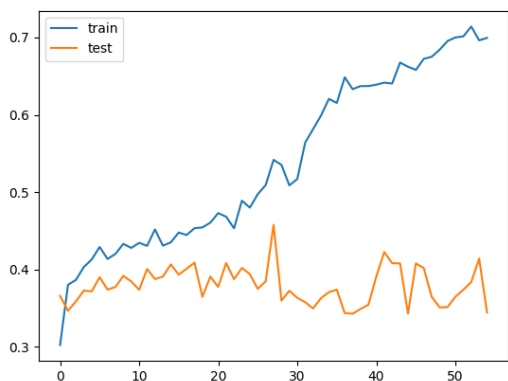
11. ábra. Tüdőszegmentáció adataugmentáció és dropout réteg beszúrása után

4.3. Fertőzés szegmentáció

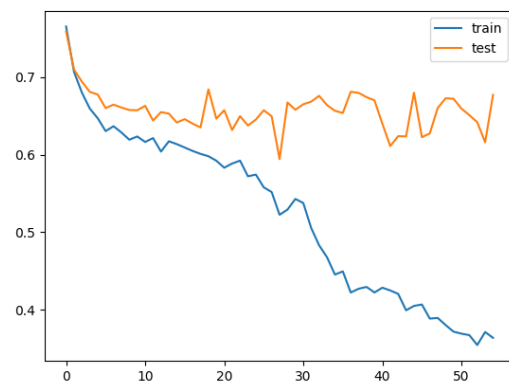


12. ábra. Az első kísérlet mérési eredményei

A fertőzés szegmentációnál hasonlóan kezdtem neki a szegmentációs modell beállításának. Az első kísérlet az eredeti U-Net architektúrával, augmentáció nélkül készült. Ahogyan a 12. ábrán látható, a futtatás elején tanult a háló, de körülbelül a 25. epochnál a tanító- és tesztadaton mért hibafüggvények grafikonjai a tüdőszegmentációhoz hasonlóan távolodni kezdenek egymástól. A 12.a ábrán is jól látszik, hogy a tanító adaton mért Dice-index egy néhány epochon át tartó visszaesést kivételével folyamatosan javul, de a teszt adaton ugyanez a metrika nem mutat növekedést, sőt csökkenni kezd. Az overfitting megelőzése érdekében megpróbálkoztam az adatok augmentálásával. A képeket 50%-os valószínűséggel horizontálisan tükröztem, majd 20%-os valószínűséggel megváltoztattam a kontrasztot és a fényességet. Ezek után a felvételeket 50%-os valószínűséggel 0° - 10° közötti szögben véletlenszerűen elforgattam.



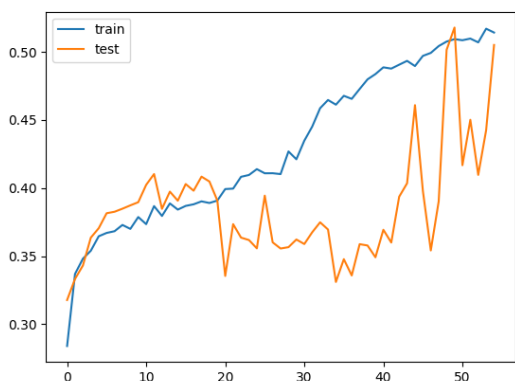
(a) Dice-index



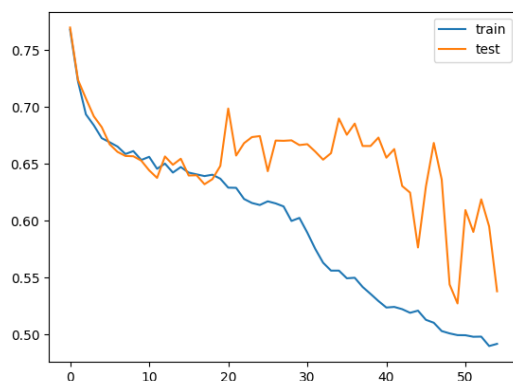
(b) Hibafüggvény

13. ábra. A második kísérlet mérési eredményei

Látható a 13. ábrán, hogy a mérési eredmények ugyan nem javultak, de a Dice-index romlása - ha nem is állt meg teljesen - minimálisra csökkent. Tisztán látszik, hogy a hibafüggvény értékek is javultak néhány százalékkal. Ennek további javítása érdekében a kimeneti aktivációs réteg elé betettem egy dropout réteget, mellyel először 2, majd 4 százalékos valószínűséggel hagytam ki neuronokat a tanításból.

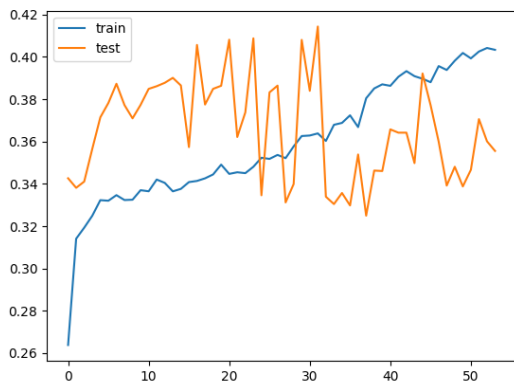


(a) Dice-index

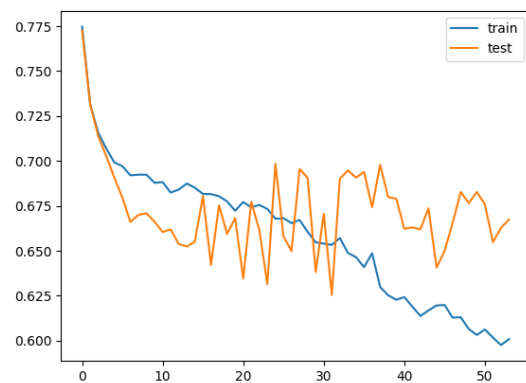


(b) Hibafüggvény

14. ábra. A harmadik kísérlet mérési eredményei



(a) Dice-index



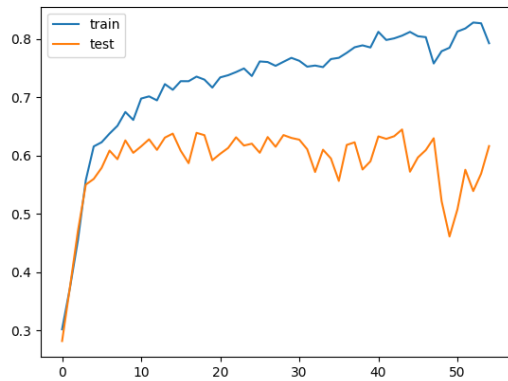
(b) Hibafüggvény

15. ábra. A negyedik kísérlet mérési eredményei

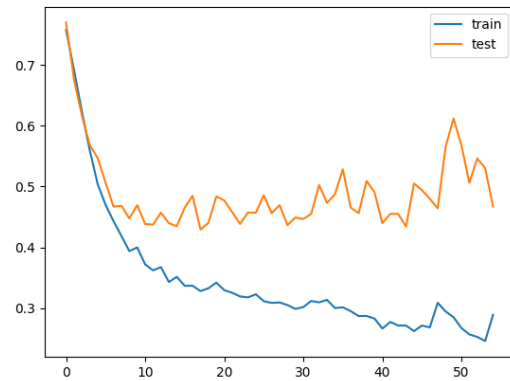
A 14. és 15. ábrákon látható, hogy a 20%-os dropout réteg beépítésének lett eredménye, a háló már nem tanult túl, jobban tudott általánosítani. A hibafüggvényen is megfigyelhető, hogy nincs akkora különbség a tanító- és tesztadat függvényértékei között és a tesztadatok grafikonja együtt csökken a tanító adatokéval. A fertőzés szegmentációjánál az adathalmaz növelése, amin a tanítást végeztem nem változtatott szignifikánsan a mérések eredményén.

4.4. Attention Gate

Kísérleteimet a U-Net-be beépített AG-vel fejeztem be. Ahogyan korábban már említettem, a háló minden szintjén, a skip connection után került a háló architektúrájába az attention gate. A felépítés módosításától azt vártam, hogy a modell könnyebben megtanulja majd a fertőzés szegmentáció szempontjából fontos tulajdonságokat, így gyorsabban és nagyobb pontossággal tud predikciót adni a fertőzött területre. A várakozásom beigazolódott, a 16. ábrán látható, hogy az attention gate beépítése után a modell már a 10. epoch előtt elérte a 60% fölötti Dice-indexet. A 17. ábra a predikciók változását mutatja be az első 20 epochban.

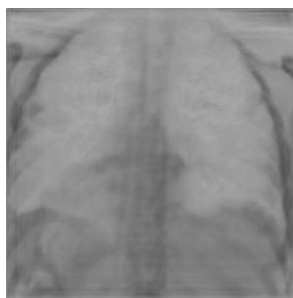


(a) Dice-index

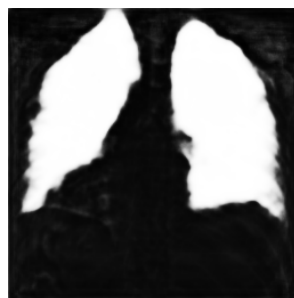


(b) Hibafüggvény

16. ábra. A fertőzés szegmentációs modell teljesítménye AG beépítése után



(a) 1. epoch



(b) 5. epoch



(c) 20. epoch

17. ábra. Predikciók az első, ötödik valamint huszadik epoch után az attention gate-tel kiegészített fertőzés szegmentációs modellben

5. Összefoglalás

A dolgozat célja az U-Net modell [7] bemutatása és kipróbálása volt egy aktuális orvosi szegmentációs feladathoz kapcsolódó adathalmazon. A kísérletek teljesítették a várakozásaimat, a tüdőszegmentációs modell eredeti formájában is jó pontosságot tudott elérni, amit adataugmentációval és az architektúra minimális változtatásával tovább tudtam javítani. A fertőzés szegmentáció már nem volt ennyire egyszerű. A képek nagyon sokélék, több kórház különböző fajta műszereivel készültek, illetve sokszor észrevehetőek rajtuk egyéb orvosi eszközök és más, a fertőzés szempontjából lényegtelen dolgok. Ennek ellenére sikerült csökkentenem az adathalmazon való túltanulást, ezzel együtt tovább tanítani a modellt. Az attention gate beépítése után pedig tovább javult a háló teljesítménye. Erőforrások és idő hiányában nem tudtam több kísérletet végezni, de különféle adatugmentációk hozzáadása illetve kicserélése vagy a dropout réteg valószínűségi rátájának változtatása nagy eséllyel további javulást eredményezne.

A lehető legjobb pontosság elérése után a modell nem csak a szegmentációs feladatok megoldására felhasználható, de klasszifikáló architektúrák teljesítményét is hatékonyan növelheti, ha azoknak csak már az előre kijelölt területet kell figyelembe venniük a predikciók előállításakor.

Azt hiszem elmondhatom, hogy a dolgozatom elkészítése közben nagyon sokat tanultam, hiszen egy éve ilyenkor még szinte semmilyen ismererettel nem rendelkeztam a neurális hálókkal és a mély tanulással kapcsolatban. Bár a matematikának ez az ága rengeteget fejlődött az elmúlt években, a téma még mindig kimeríthetetlen. A modernebb orvostechikai eszközök már manapság is kihasználják a mesterséges intelligencia adta lehetőségeket, a jövőben pedig mégjobban megkönnyíthetik a szakemberek feladatát, a tudomány további fejlődésével pedig egyre több ember felépüléséhez járulhat hozzá.

Hivatkozások

- [1] „COVID-QU-Ex Dataset”. (2021). DOI: <https://doi.org/10.34740/kaggle/dsv/3122958>.
- [2] Yicheng Fang és tsai. „Sensitivity of chest CT for COVID-19: comparison to RT-PCR”. *Radiology* (2020).
- [3] Kaiming He és tsai. „Delving deep into rectifiers: Surpassing human-level performance on imagenet classification”. *Proceedings of the IEEE international conference on computer vision*. 2015, 1026–1034. old.
- [4] Shervin Minaee és tsai. „Image segmentation using deep learning: A survey”. *IEEE transactions on pattern analysis and machine intelligence* (2021).
- [5] Keiron O’Shea és Ryan Nash. „An introduction to convolutional neural networks”. *arXiv preprint arXiv:1511.08458* (2015).
- [6] Ozan Oktay és tsai. „Attention U-Net: Learning Where to Look for the Pancreas”. *arXiv preprint arXiv:1804.03999* (2018).
- [7] Olaf Ronneberger, Philipp Fischer és Thomas Brox. „U-Net: Convolutional networks for biomedical image segmentation”. *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, 234–241. old.
- [8] Feng Shi és tsai. „Review of artificial intelligence techniques in imaging data acquisition, segmentation, and diagnosis for COVID-19”. *IEEE reviews in biomedical engineering* 14 (2020), 4–15. old.
- [9] Vinorth Varatharasan és tsai. „Improving learning effectiveness for object detection and classification in cluttered backgrounds”. *2019 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED UAS)*. IEEE. 2019, 78–85. old.

NYILATKOZAT

Név: Börcs Dóra

ELTE Természettudományi Kar, szak: Matematika


NEPTUN azonosító: LCHR6K

Szakedolgozat címe:

Orvosi képfeldolgozás neurális hálókkal

A **szakedolgozat** szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló szellemi alkotásom, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.

Budapest, 2023.01.02.



a hallgató aláírása