Szegedi Tudományegyetem
Informatikai Intézet

Automatikus magpontkiválasztás régiónövelés-alapú agytumor-szegmentáláshoz MRI-felvételeken

Diplomamunka

Készítette:
Varga-Betlehem Ádám
programtervező informatikus MSc szakos hallgató

Témavezető:
Dr. Kardos Péter
adjunktus

Szeged
2025

Feladatkiírás

Feladatkiírás

Tartalmi összefoglaló

Tartalmi összefoglaló utólag

Tartalomjegyzék

Bevezetés

ÚJRAÍRNI!!!!!!!

Az agydaganatok diagnosztizálása egy létfontosságú kérdés a megfelelő kezelési stratégia megválasztásához és a beteg túlélési esélyeinek javításához. A leggyakrabban az agy vizsgálatára mágneses rezonancia képalkotást (MRI) alkalmazzák, mivel ez a képkészítési eljárás képes részletes és kontrasztgazdag képet adni a lágyszövetekről, de ennek ellenére egyéb képkészítő eljárást is használhatnak, mint például a röntgen alapú CT-t, ami hozzáférhetőbb és gyorsabb. Az így készült képek értelmezése függ az orvos tapasztalatától, szakértelmétől és időt igényel, így szükség lehet automatikus képfeldolgozó és szegmentáló eljárásokra, amik segítik az orvost munkája során.

Jelen esetben a szegmentálás célja az, hogy megkülönböztessük a tumort az agy többi részétől, ezzel gyorsítva és objektivizálva az orvos munkáját. Napjainkban a legnépszerűbbek a mélytanulásos módszerek, de nem szabad megfeledkezni a klasszikus szegmentáló módszerekről se. Ilyen módszer például a régiónövelés, aminek eredményességét nagyban befolyásolja a kezdőpontok (seed pontok) kiválasztása, ami sokszor manuális feladat. Maga a régiónövelés elve egyszerű, viszont a nem megfelelő seed pontok kiválasztása alul-, felülszegmentálást eredményezhet vagy akár hibás régiókat is. Ezért fontos egy olyan automatikus seed kijelölési eljárás, amivel minimalizálni tudjuk az emberi beavatkozást és növelni az eljárás reprodukálhatóságát. Ebben a diplomamunkában automatikus seed kiválasztó eljárásokat fogunk megnézni és megvalósítani, esetleg javítani.

1. Orvosi képfeldolgozás/képalkotás

Az orvosi képalkotás célja, hogy anatómiai vagy funkcionális információt szerezzünk a vizsgálandó területről anélkül, hogy egészségkárosító hatása lenne, vagy a diagnosztikus hasznával szemben elhanyagolható legyen az egészségkárosító hatás. Különböző képalkotó eljárások különböző információval szolgálnak.

A CT (Computed Tomogrhapy) röntgensugárral dolgozik, ezek áthaladnak a testen, amit különböző szervek, szövetek különböző mértékben nyelnek el, ezzel információt adva a test belső felépítéséről, elsősorban a csontrendszer felépítéséről. A CT tipikus skálája a Hounsfield-skála, amin a lágyszövetek nagy mértékben egybeesnek, ezért a CT vizsgálatoknál szokás még kontrasztanyagot is használni a vizsgált terület kiemelésére. A csontsérülések mellett használható érrendszeri betegségek, gyulladások, daganatok, belső vérzések, idegrendszeri betegségek diagnosztizálására is, emellett relatív gyorsan végezhető vizsgálat, viszont sugárterheléssel jár.

Funkcionális képeket készítenek például a PET (Positron Emission Tomography) technikával, aminek alapja, hogy bizonyos radioaktív anyagok bomlásakor pozitronok keletkeznek. Ezen pozitronok fokozatosan lelassulnak a közegben, majd elektronnal ütközve megsemmisülnek, ekkor két ellentétes irányba távozó gamma-foton keletkezik, aminek egyidejű érzékelése adja a vizsgálat alapját. Lényegében a pozitronkibocsátó izotópok testbeli eloszlását szeretnénk megállapítani, például felhalmozódik rákos sejtekben ez az anyag, vagy más energiaigényes sejtekben. Gyakran CT-vel együtt használják, úgynevezett PET-CT gépek, így anatómiai információt is kapunk a funkcionális mellé. A fő fókuszban az anyagcsere van, jellemzően rákdiagnosztika, Alzheimer-kór, szívizom, agyműködés vizsgálatára használják. Hasonló elven alapuló módszer a SPECT (Single-Photon Emission Computed Tomogrpahy), ahol a radiofarmakon eloszlását vizsgáljuk. A radiofarmakon radioaktív anyag, amely a szervezetben bomlása során gamma sugárzást bocsát ki, amit rögzítünk. A fő fókusza a véráramlás. Alkalmazzák a szívizom vérellátása, csontbetegségek, pajzsmirigy vizsgálatára, alkalmas lehet tumor kimutatására is, ha olyan radiofarmakont használunk, ami a tumorban dúsul, emellett az agyi funkciók vizsgálatára is alkalmas. A SPECT szélesebb körben elérhetőbb és olcsóbb, viszont alacsonyabb felbontásra és kontrasztra képes, mint a PET.

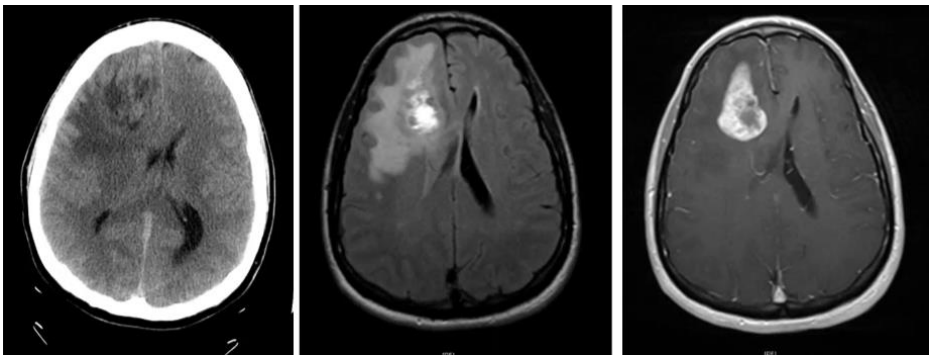
1.1. Az MRI

A mágneses rezonancia képalkotás (MRI), azon alapul, hogy egy atommag spinjének erős mágneses térben két állapota van. Párhuzamos, ami az alacsony energia szintű állapot, és az antipárhuzamos, ami a magas, és megfelelő hullámhosszúságú rádióhullámokkal az alacsonyból a magasabba vihető, ez a gerjesztési folyamat. Utána elindul a relaxációs folyamat, amiben a spinek visszatérnek nyugalmi, azaz az alacsony energiaszintű állapotukba, miközben rádiósugárzást bocsátanak ki. Ezt a folyamatot többször ismételjük T_R időnként a jel-zaj viszony javítása érdekében, végül ennek a sugárzásnak az erősségének és időbeni változásának mérésével kapjuk a képet. A T_1 a longitudinális, míg a T_2 a transzverzális relaxációs idő. Mindkettő a térrész fizikai és kémiai tulajdonságától függ, és a T_2 jóval rövidebb ideg tart, mint T_1. T_1 súlyozott képet kapunk akkor, ha az ismétlési idő (T_R) rövid, mivel nem telik el elég idő, hogy bizonyos szövetek longitudinális magnetizációja (T_1) visszaálljon, és a 90^o-os pulzus és maximális jel között eltelt idő (T_E) is kicsi vagy nincs, így T_2 különbségek nem érvényesülnek. T_2 súlyozott képet kapunk, ha az ismétlési idő hosszú, hiszen így T_1 relaxációban nincs jelentős különbség, és ha T_E hosszú, akkor a T_2 relaxációban lényegesebb különbségek keletkeznek. A FLAIR (Free Liquid Attenuated Inversion Recovery) kép inverziós szekvenciával készült T_2 kép, ahol a folyadék nem ad jelet, mert az első impulzus után addig várunk, amíg a folyadék mágnesezettsége nulla lesz és utána adjuk ki a következő impulzust.

1.2. Agytumor

Az agyban létrejövő kóros sejtszaporulatot agydaganatnak nevezzük. Egy agydaganat lehet elsődleges vagy másodlagos. Az elsődleges daganatok elsősorban az agyban képződnek vagy a közeli szövetekben. Az alapján csoportosítják őket, hogy honnan erednek, például idegsejtekből gliómának, agyhártyából meningeomának hívjuk. Az agy más területeire áttétet képezhetnek, de más testrészekre általában nem szoktak terjedni. A másodlagos vagy másnéven áttétes tumor, a test egy másik pontjáról például tüdő, mell, vese terjed át az agyra. Egy elsődleges tumor lehet jó- vagy rosszindulatú, míg másodlagos csak rosszindulatú lehet. A rosszindulatú agydaganat gyorsabban nő, mint a jóindulatú, és behatol a környező agyszövetekbe elpusztítva azt. Ezzel szemben a jóindulatú általában jól körülhatárolt, könnyebben eltávolítható lehet, és ritkábban újul ki, így a sebészi beavatkozás megoldást jelenthet. Általában először CT képet szokás készíteni, relatíve gyorsabb és elérhetőbb, mint MRI, emellett jól tud detektálni még vért, viszont a koponya a

röntgensugarak nagyrészét felfogja. Szinte mindig készül MRI kép is, mert jóval részletesebb képet ad lágyszövetekről, viszont sokkal lassabb és nem jár röntengsugárzással. Az orvosok különböző elváltozásokat figyelnek a képeken, például a középvonal eltolódását, sötétebb, világosabb foltokat. Az 1. ábrán jól látszik a CT és MRI minősége közötti különbség, és a bal oldalon levő CT kép a középvonal eltolódására jó példa, míg a középső a FLAIR kép, míg a jobboldalon levő egy kontrasztanyag használatával készült T_1 kép. Meglehet említeni, hogy a T1 képek kontrasztanyag használatával jól használhatóak új elváltozások, míg T2 és FLAIR képek pedig magas fokú gliómák és ödémák kimutatására. Mivel az adathalmazunk legfőképp gliómás daganatokat tartalmaz, ezért a FLAIR képein fogunk dolgozni.



1.1.1. ábra: Abnormális agyi CT, MRI FLAIR és T1 [hivatkozás]

2. Szegmentálás

A szegmentálás célja, hogy a képeket olyan részekre osszuk fel, amelyek megfelelnek a valós leképezett objektumoknak, területeknek. Általában az előfeldolgozás után következik és megelőzi a jellemző kinyerést, ha a gépi látás moduláris rendszerének modelljében próbáljuk elhelyezni. Egy nehéz feladatról van szó, ami egyáltalán nem egyértelmű. Orvosi képeken a sebésznek tudnia kell mit kell eltávolítania, a radiológusnak tudnia kell melyik részt bombázhatja sugarakkal, de azt is tudnia kell mely részeket nem érheti sugárzás. A szegmentálás bonyolultságából adódóan számos módszert kifejlesztettek, köztük küszöbölés, él-alapú, régió-alapú, illesztésen alapuló szegmentáló módszereket, hogy pár alap-módszert említsünk. Az (1) képletben a szegmentálás van leírva formálisan, ahol az *R* a teljes képet, *Ri* az *i*-edik régiót, a *P* a homogenitási kritériumot és *n* a régiók számát jelöli, emellett minden régió önmagával egybefüggő.

$$\bigcup_{i=1}^{n} R_i = R, \qquad R_i \cap R_j = \emptyset, \qquad P(R_i) = 1, \qquad P(R_i \cup R_j) = 0 \tag{1}$$

A formális leírás annyit jelent, hogy a régiók egyesítése visszaadja a teljes képet, minden régió diszkrét, minden régióra teljesül a homogenitási kritérium, és bármely 2 régióra már nem. A küszöbölésnek az alapgondolata az, hogy az adott régióra tudunk mondani egy intenzitás értéket, küszöböt, ami felett vagy alatt minden érték az adott szegmenshez tartozik. Sokféle küszöb választási stratégia létezik, talán a legismertebb példa erre az Otsu módszere, ami olyan küszöbértéket keres, amire minimális az osztályon belüli variancia, ezzel megpróbálva elválasztani a hátteret az előtértől. A régió-alapú szegmentálás az objektum által elfoglalt területet határozza meg. Ilyen módszer a régió növelés, ahol első lépésként kiválasztunk egy pontot vagy egy biztosan összetartozó pontokból álló kezdeti régiót. Minden lépésben megvizsgáljuk az aktuális régióval határos pontokat, majd hozzáadjuk azokat a régióhoz, amik kielégítik az adott hasonlósági kritériumot. Az algoritmus megáll, ha már nem tud több pontot hozzáadni a régióhoz. Ebből is látszik, hogy létfontosságú a magpontkiválasztás, ami nagyban feladatfüggő. Ebbe a kategóriába eső más módszer még a szétválasztás és egyesítés, ahol az egyesítés különböző régiókat von össze egy kritérium alapján, míg a szétválasztás esetén a régiók nem elégítik ki a feltételt. Rendszerint a képet, régiót négy egyenlő részre osztjuk szét, amit egy hierarchikus négyes fa adatstruktúrában tárolunk, ahol minden csúcspont a négy feldarabolt részére mutat.

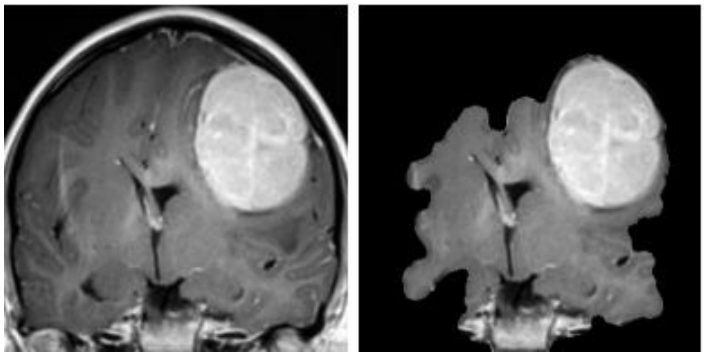
3. Blokk- és intenzitásalapú magpont-kijelölés

Ebben a cikkben FLAIR képekre javasoltak algoritmust, mivel ezeken a képeken lehet a leghatékonyabban felismerni a gliómákat, amik a rosszindulatú daganatos megbetegedések legalább 80%-t teszik ki felnőttek esetén. Továbbá a Brain Tumor Segmentation 2015 challenge (BraTS2015) közzétett adathalmazt használják, ami multimodális MR képeket tartalmaz gliómás daganatokról. Kihangsúlyozzák, hogy az előfeldolgozás, mint például a simítás, normalizálás és szükségtelen részek eltávolítása az nagyon fontos a nyers MRI képeken és a használt adathalmazon a nem kívánt részek már elvannak távolítva. Ajánlanak egy előfeldolgozási lépést a koponya eltávolítást, aminek a lényege, hogy mindent, ami nem agyi szövet eltávolítanak. Megemlítik, hogy ez egy nehéz, de szükséges feladat, amire számos megoldás létezik, ennek ellenére ajánlanak egy morfológián és küszöbölésen alapuló módszert.

```
func KoponyaEltávolítás(kép):
    küszöb = Otsu(kép)
    maszk  = küszöbölés(kép, küszöb)
    maszk  = Nyitás(maszk)
    maszk  = Dilatáció(maszk)
    maszk  = LegnagyobbObjektum(maszk)
    maszk  = Zárás(maszk)
    maszk  = Lyukfeltöltés(maszk)
    kép    = Maszkolás(kép, maszk)
    return kép
```

3.1 kódrészlet: Koponya eltávolítás pszeudokód

A 3.1 kódrészleten láthatóan, először elvégeznek egy Otsu küszöbölést, majd a kapott maszkon egy morfológiai nyitást és dilatációt. A nyitással megszabadulnak a kisebb leszakadozó részektől, majd a dilatációs lépés alkalmazásával megpróbálják elérni, hogy a ténylegesen összefüggő része összeérjenek. Ezután kiválasztják a legnagyobb összefüggő objektumot, ezzel feltételezve, hogy az az agy. Majd ezen a bináris objektumon elvégeznek egy morfológiai zárást, hogy a kisebb szétszakadásokat és lyukakat feltöltse, végül a nagy lyukakat is feltöltik. Az eredeti képből a maszk által jelült részt tartja meg.



3.1. ábra: Példa koponya eltávolításra. Eredeti kép [hivatkozás]

Érdemes megjegyezni, hogy ez a módszer nem feltétlenül csak az agy körüli részt tünteti el, hanem ha minimális a háttér és az agynak vannak sötétebb részei a perem mentén, akkor képes ezeket a részeket is háttérnek kezelni az Otsu módszerből adódóan, viszont ez a módszer szempontjából nem okoz problémát, hiszen a tumort megtartja és a mi adathalmazunkban a háttér elég nagy területet foglal el. Az ő általuk felhozott példában (3.1 ábra) a háttér minimális volt, ahol ez megtörtént, mert ott túl is lógott a kép szélein a koponya. A többi felhozott példájukban, ahol a háttér körbe ölelte a koponyát, ott megfelelően működött a módszer. A 3.2 és 3.3 ábrán minden lépése végig követhető a módszernek)

A következő lépésre továbbfejlesztett régió növelő (enhanced region growing) eljárásként hivatkoznak. A már előfeldolgozott képet 8x8-as blokkokra osztja fel az eljárás, majd kiválasztja a legjobb öt, olyan régiót, ahol a legmagasabb az átlagintenzitás, hiszen FLAIR képeken a tumor intenzitás nagyobb, mint a környező szöveteké. Majd ezen régiók középpontját választja meg magpontnak. A régiónövelés egy standard változatát használják, ahol az intenzitás küszöbértéke 0.1, mert állításuk szerint az ő adathalmazukban a legtöbb tumor homogénnek tűnik, kiegészítő műveletnek, ha nem volt homogén, akkor a lyukakat kitölti a maszkban. A különböző kiindulási pontokból öt darab maszkot kapunk, amit ROI-nak (region of interest) hívnak, ami az érdeklődési területet jelöli. Majd ezeket összeveti az adathalmazban található valós szegmentált tumorral különböző metrikák alapján, és a legjobbat fogja választani, mint szegmentált terület a kiértékeléshez.

Érződik is, hogy egy kezdőpont hibás lehet, nem véletlenül van szükség többre, hisz a tumor kicsi is lehet, és olyan módon foglalhatja el a 8x8-as blokkot, hogy a blokk középpont nem része, például a tumor határvonalakra esik, így az intenzitás átlag elmosódik és a magpont kívül eshet a tumoron. Az általam javasolt módszer az csúszó ablak (sliding window) technika, amivel lehetne orvosolni az előbb említett problémákat. Finomabb mintavételezést eredményezne, tehát a

legfényesebb régiók nagyobb valószínűséggel esnek a blokk középpontba, és csökken az esélye annak, hogy tumoron kívüli pontot választanánk magpontnak. Minden ablakban továbbra is számítható lenne az átlagintenzitás, vagy akár több különböző statisztika is, mint például intenzitás maximum vagy akár szórás. Mivel a szórás úgy van definiálva, hogy az átlagtól való átlagos eltérés, így első gondolatként a globális átlagtól való eltérését vizsgáltam, ami ugyanolyan tendenciát mutatott, mint az átlag szerinti pontozás, ezért új ötletként csak a lokálisablakon belüli szórást vizsgáltam. Ha a szórás az ablakban alacsony, akkor homogén területen vagyunk, ami önmagában nem elég viszont, súlyozhatjuk az ablak intenzitás átlagával, kiemelve a magasabb intenzitású homogénebb területeket. A következő gondolatként meglehet nézni mennyire folt vagy paca (blob) szerű, amit gyakran a Gauss szűrő második deriváltját használják fel (LoG: Laplacian of Gaussian). A foltok detektálására költséges, és a gyakorlatban az ablak jóval kisebb, mint maga a tumor. A LoG operátor önmagában egyszerű és konvolúcióval használható. A művelet duplán detektál éleket, ahogyan sötétebb területről világosabbra lépünk, először pozitív(világosabb), majd negatív(sötétebb) értékeket vesz fel, ahol nincs változás ott 0 marad. A képen nincs tökéletesen homogén fehér terület, és a tumor belsejében a gyakorlatban negatív és 0-hoz közeli értékek kerülnek, így a tumoron kívül a háttér és az agy határán ad hasonlóan alacsony értéket. Az agy többi részén, pedig magasabb értékeket kapunk, mint a tumornál és környékén. Tehát az ablakon belüli átlagot feltudjuk használni, viszont most is csak az ablak átlagintenzitásával súlyozva. A 3.4 ábra megmutatja a szórásból és LoG-ból számolt ponttérképet, ahol minden pixel az adott ablak középpontját jelenti, az eredeti agykép a 4.1 ábrán szerepel először.

3.4. ábra: BraTS20_Training_355_flair.nii 84-es rétegének szórás (bal) és LoG (jobb) ponttérképe

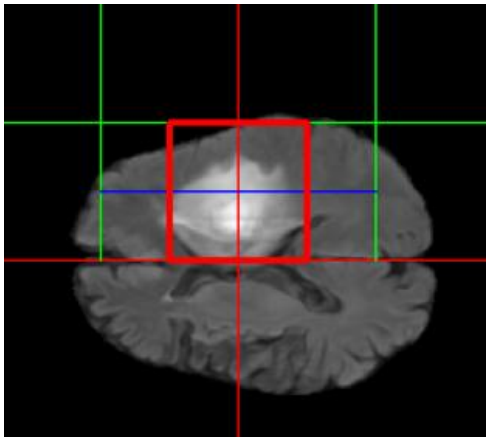
A blokkok középpontja helyett lehetne választani a blokk maximum intenzitású pixelét, mint magpont. Annak elkerülése, hogy közvetlenül egymás mellett levő ablakokat, vagy ugyanazon

magpontot válasszuk ki a túl közel lévő ablakokból a nem-maximális elnyomás (non-maximum suppression) technikával érem el.

4. Divergenciaalapú magpont-kiválasztás

Ebben a cikkben először a vizsgálni kívánt területet határozzák meg (ROI), majd ezen a területen kijelölik a magpontokat, és egy saját régiónövelést alkalmaznak. A felhasznált adathalmazról, annyi szó esik, hogy Diffúzió súlyozott MRI (Diffusion-Weighted MRI, DWI) képeket használtak fel. Nagyobb az elváltozott terület kontrasztja a hagyományos T_1, T_2 képekhez mérten. Ezzel a technikával azt mérik mennyire diffúz egy terület, ahol a magas diffúzió sötét (hypointense), míg az alacsony az világosabb (hyperintense) területként jelenik meg. A legérzékenyebb technikának tekintik, ha akut infarktust kell detektálni. Akut infarktus, vérzés (deoxihemoglobin), szolid tumor és tályog (fertőzés), amik világos hiperintenzív területként jelennek meg, míg krónikus infarktus és vérzés (oxihemoglobin) sötétebb hipointenzív területként jelenik meg. Az adathalmazukban előfeldolgozás nélküli képek szerepelnek, így először 0-1 közé normalizálják az intenzitásértékeket, majd eltávolítják a hátteret egyszerű küszöböléssel, amit tapasztalat útján állítottak be 0.023-ra. Ezután képesek végig követni az agy határvonalát és csak a benne levő részeket megtartani, ezzel hasonló eredményt kapnak, mint a mi adathalmazunkban szereplő FLAIR képek. Az általuk mellékelt hisztogram szerint alacsony az intenzitás a képen, ezért gamma korrekciót alkalmaznak.

A képjavító eljárások után következik a szétválasztás és egyesítés algoritmus, amely négyes-fa adatszerkezeten alapul. A hagyományosan vett értelemben a képből kiindulva azt 4 felé vágjuk, ha nem felel meg a homogenitási kritériumnak, majd a végére érve, a szomszédos régiók, ha együttesen megfelelnek egymásnak, akkor össze lesznek vonva. A 4.1 ábrán látható egy példa a szétválasztás és egyesítés algoritmusra, a mi adathalmazunkon.



4.1. ábra: BraTS20_Training_355_flair.nii 84-es rétegen végrehajtott szétválasztás egyesítés

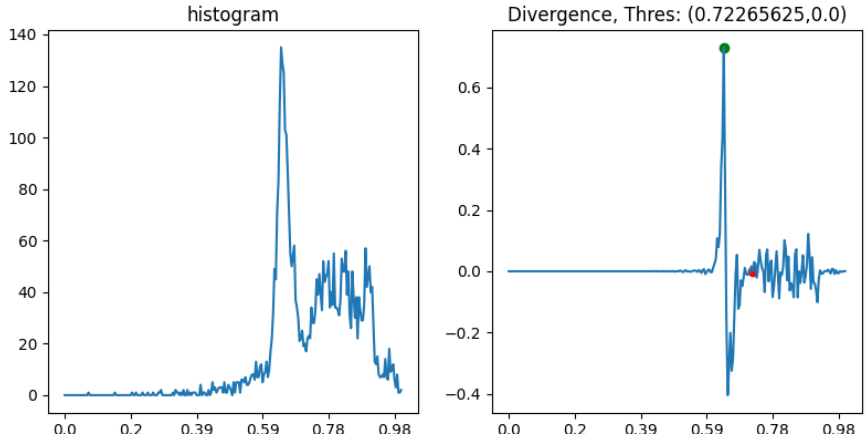
A cikkben javasolt módszer, abban tér el a hagyományos szétválasztás és egyesítés algoritmustól, hogy ha kell 4 részre vágja a képet, de legfeljebb csak a 3. szintig. A tényleges összevonás helyett pedig, kiválasztja azon régiókat, amelyek tovább bonthatóak lennének, mivel a homogenitási kritérium úgy van megfogalmazva, hogy szerepel-e elváltozott terület a képen. A kritériumot minden iterációban hisztogram alapján fogják meghatározni az adott régióban. Az egész adathalmazból empirikusan megállapítottak egy hiperintenzív (0.47-0.8) és hipointenzív (0.1-0.25) tartományt, amiben a tumor előfordul. Ezt a tartományt a hisztogramból kiragadva számol átlagot, pixelek darabszámát, entrópiát és szórást, majd tapasztalati úton úgy dönt, hogy csak az átlagot és a tartományba eső pixelek darabszámát használja fel. A hiperintenzív tartomány küszöbértékét 0.495-re választja meg, tehát ami ennél magasabb az elváltozott területnek tekinti, ha ennél alacsonyabb, csak akkor fogja hozzátartozónak tekinteni, ha a tartományba eső pixelek száma meghaladja a 30-at. Természetesen az ellenkező irányban 0.2 átlagnál alacsonyabb, de nem 0 értékek vagy legalább 100 db tartományba eső pixel. Ezen küszöbértékeket az adathalmazból szintén az adathalmazból fogja meghatározni.

A kapott területen fognak magpontot kijelölni úgy, hogy elkészítik a terület hisztogramját. Aminek kiszámolják a divergenciáját ((2)-es képlet), és a maximális érték után eső, első 0-hoz legközelebbi értéket választja ki optimális küszöbértéknek, tehát az annál nagyobb intenzitásértékek magpontnak lesznek kiválasztva. A (2)-es képletben a P(i) jelöli a hisztogramot, míg az előtte lévő hányados a deriváltját. Látszik, hogy a képlet nem egyértelmű és a cikkben se tárgyalják részletesebben. Ebből a képletből következhet, hogy csak a hisztogram deriváltját nézik, vagy annak szorzatát a hisztogrammal, vagy a derivált szorzatát a normalizált hisztogrammal hiszen „P” betűvel jelölik.

$$div(i) = \frac{dy}{dx} P(i) \tag{2}$$

A hisztogram az intenzitáseloszlásokat mutatja, aminek a végén egy kisebb csúcs jelenik meg, mert a hiperintenzív terület sokkal világosabb. Ha a hisztogram deriváltját nézzük, akkor megkapjuk milyen gyorsan változik az előfordulási gyakoriság az intenzitás mentén. A leggyakoribb intenzitásnál a legmagasabb hisztogram értéke, miután lecseng a hisztogram és elérnénk a hiperintenzív régiót, újra nőni fog az érték, tehát egy lokális minimum helyen is áthaladtunk. Tehát a hisztogram deriváltjának maximuma a legmeredekebb változást mutatja, ahol a domináns szövetosztályból átlépünk egy másikba, és az azutáni 0 érték jelzi, hogy a változás kiegyenlítődik, határhoz értünk. Ezt a deriváltat megszorozzák a normalizált hisztogrammal, tehát az intenzitások

előfordulási valószínűségével. Ezzel megpróbálják figyelembe venni azt is, hogy mennyire jelentős az intenzitástartomány a képen belül. Tehát ha kicsi, akkor az egész tag gyenge, így képes lehet elnyomni a zajokat, ha nagy, akkor a változás megerősödik, ezzel kiemeli a lényeges struktúrákat. Végül minden magpont lesz, ami a küszöb felett helyezkedik el. Érződik, hogy ha túl pici a ROI, amin túlnyomó részt tumor szerepel, akkor nem fog tudni feltétlenül magpontot kiválasztani. Ha túl nagy a terület, amin pici elváltozás szerepel, akkor túl alacsony intenzitású pontokat fog választani, amik nem részei a tumornak.



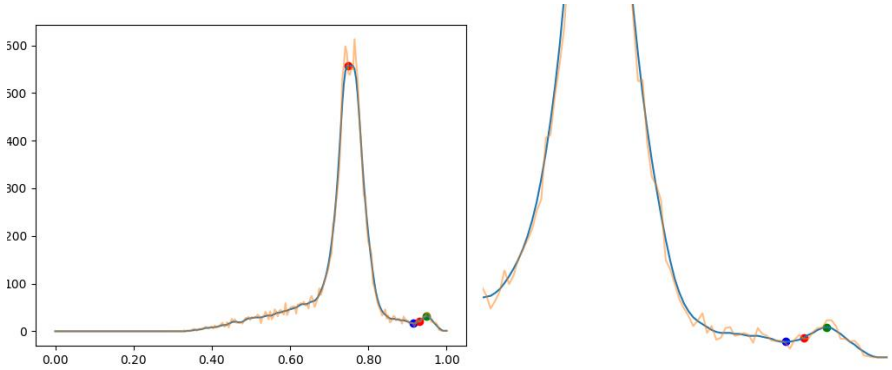
4.2. ábra: A 4.1 ábrán jelölt terület hisztogramja és divergenciája

A 4.2 ábrán lehet látni a hisztogramot és a hozzátartozó divergencia képét, amit a 4.1 ábrán pirossal jelölt területen számoltunk. Az ábrán a zöld pont a maximumot jelöli, majd utána a piros pedig a küszöbértéknek választott intenzitást, ami fölött mindent magpontnak választunk.

A régiónövelést az egyik magpontból kezdjük el. Amint befejeződik egy olyan magpontot fogunk választani, ami még nem tartozik egy szegmentált területhez sem, így a végeredményül egy maszkot kapunk, amin akár több elkülönült régió is előfordulhat. A régiónövelés kritériumához felhasználják az előbb meghatározott optimális küszöbértéket. Ha az aktuális intenzitás különbsége a régió átlagával nagyobb, mint a régió átlagának különbsége az optimális küszöbértékkel, akkor megáll a régiónövelés.

A hiperintenzív intervallum egy fix küszöbérték, amit a cikkben az adathalmaz alapján határozták meg, ami pár kép volt. Érdekes lehet adaptívvá tenni ezt a küszöbértéket és a hozzátartozó átlagot, ezzel is rugalmasabbá téve az algoritmust. Ha az agyi képen kellően nagy a kontraszt az agy és a tumor között, akkor a hisztogramon elkülönül a hiperintenzív régió a leggyakrabban előforduló intenzitástól. Ebben az esetben a globális maximum után lokális maximumot kell keresni. Ha találtam ilyet az azt jelenti, hogy van még egy gyakran előforduló

elem, ami eltér az agyszövettől. Akkor azt tudom mondani, hogy megtaláltam a közepét a hiperintenzív tartománynak. A 4.3. ábrán jobb oldalon a zöld pont jelöli a megtalált maximumot, és a narancssárga vonal jelöli az eredeti, míg a kék a simított hisztogramot.



4.3. ábra: BraTS20_Training_355.flair.nii 78-as réteg Hisztogramja(bal) és nagyított változata(jobb)

A következő lépés a tartomány szélének a megtalálása. Azt úgy érem el, hogy a globális és lokális maximum között keresem a legkisebb lokális minimum értéket. Feltéve, hogy találtam lokális maximumot, illetve minimumot (4.3. ábra kék pont). Ha veszem az átlagintenzitást ezen tartományon belül, az közel lesz a lokális maximumhoz, viszont az egy kicsit túl szigorú a homogenitási kritériumként, ezért kicsit megengedőbb az a megközelítés, ha a hiperintenzív tartomány alsókorlátjának és a tartományon belüli átlagnak az átlagát választam, azaz a kettő intenzitástól egyforma távolságra levő értéket (4.3.ábra jobb oldali piros pont). Természetesen ez nem mindig oldható meg, olyankor az alapértelmezett értékeket használom. Másik probléma, hogy ha túl nagy az algoritmus által kiválasztott terület, amin keresi a magpontokat, akkor túlságosan sokat fog találni, ami a tumor szélein lehet, vagy akár a tumoron kívül is az elég világos apró területeket. Utólag a magpontokból fogok eltávolítani elemeket.

4.4. ábra: BraTS20_Training_355_flair.nii 79-es réteg élkép(bal), megtartott élek dilatáltja(közép), eldobbott ill. megtartott magpontok(jobb)

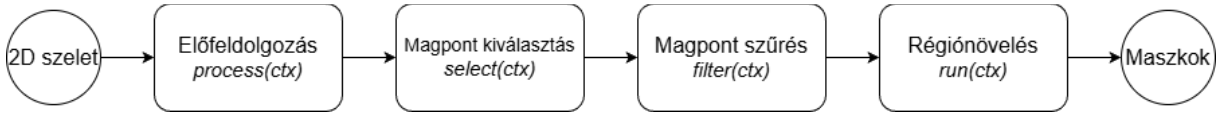
Sokszor zajra, apró foltokra hasonlítanak a tumoron kívül jelölt elemek, illetve nem szerencsés a tumor szélein is magpontokat jelölni, ezért veszem az agyról készült élképet (4.4. ábra), amin az

erősebb éleket megtartom és binarizálom. Ezen bináris képen elvégzek egy dilatációt (4.4 ábra középső), mivel szeretném, hogy az tumoron kívül választott magpontok teljesen eltűnjenek, illetve a tumoron belül se legyen az élen és közvetlen közelében magpont. Ezért a magpontokból készítek egy bináris képet, és azon pontokat tartom meg rajta, ahol az élképen nincsen él. A 4.4 ábra jobboldali képén pirossal jelölve a törölt és zölddel jelölve a megtartott képpontok.

5. Implementációk

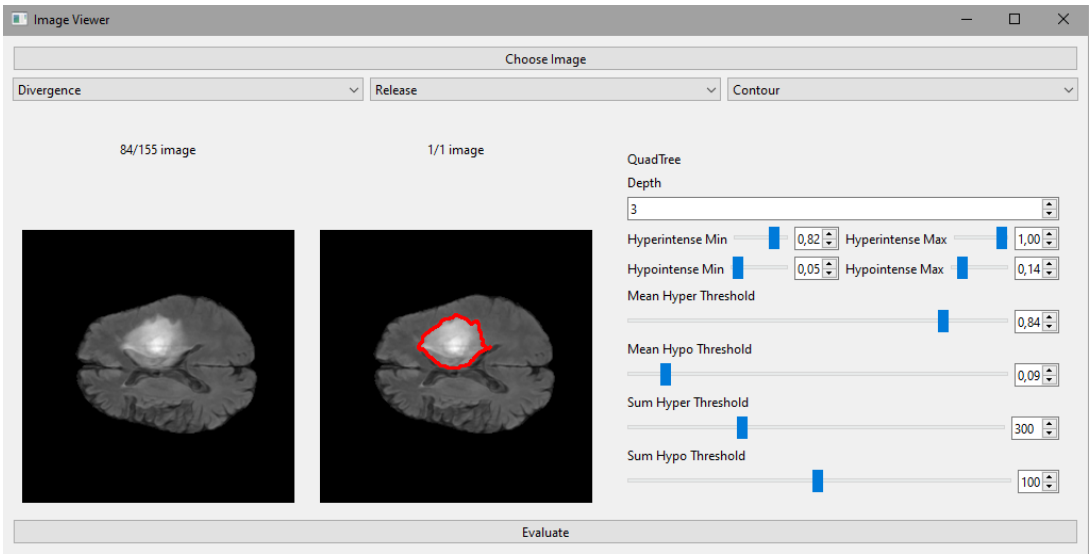
5.1. A Program megvalósítása

Az alkalmazás Python nyelven íródott. A felhasználói felület Pysied6 és matplotlib könyvtárral lett megvalósítva. A felületen töltjük be a képeket, választjuk ki a futtatni kívánt algoritmust, és állítjuk be a paramétereket. A számításokhoz numpy, opencv és egyéb könyvtárakat használtam fel. A program egy MR képet fog szegmentálni régió növelési algoritmussal, így lehetséges felbontani a feldolgozást diszkrét, moduláris lépésekre. Ennek a megvalósítása a feldolgozási lánc vagy csővezeték (pipeline) elv szerint történik, vagyis egymás után végrehajtott lépések sorozatával (5.1 ábra).



5.1. ábra: A feldolgozás lépései

Először az előfeldolgozási lépések futnak le, majd a magpont-kiválasztás, végül pedig a régiónövesztés. Minden lépés ugyanazon a `PipelineContext` objektumon keresztük kommunikál egymással, amiben tároljuk az eredeti képet, a különböző lépések eredményét és a felhasználó által beállított paramétereket. Mivel ezek a lépések különböző megvalósításokat követhetnek, így a megvalósításukra a stratégia mintát (strategy pattern) használtam fel. Azaz minden lépés egy azonos interfészt valósít meg, így tetszőlegesen cserélhetők, anélkül, hogy a program bármelyik másik részéhez hozzá kelljen nyúlni. Az előfeldolgozási lépések a `process(ctx)`, a magpont-kiválasztás a `select(ctx)`, míg a régiónövelés a `run(ctx)` metódust hívja, így a lépések nem egy interfészből származnak, így jól elkülönülve egymástól. A különböző algoritmusokat a gyártó minta (factory pattern) segítségével hoztam létre. A `PipelineFactory` osztály minden módszerhez, egy előre definiált csővezetéket épít fel, amit statikus függvényként lehet meghívni, így a felhasználói felületen elég kiválasztani melyik módszert szeretnénk futtatni. Minden csővezeték deklarálhatja, milyen vezérlőelemekre van szüksége ahhoz, hogy a lépések paramétereit a felhasználó beállíthassa. Ezen felsorolt elemek egy `ControlPanel` ősosztályból származnak, és csúszkákat, számbeíró mezőket valósítanak meg. Ezen paneleket a `ControlWindow` osztály kezeli és jeleníti meg a felhasználó számára, amikor megváltoztatja az adott pipeline-t. Ez a megközelítés a kompozit (composite) és építő (builder) mintákat ötvözi.



5.5.1. ábra: A felhasználói felület

A felhasználói felületen a Pyside6 függvénykönyvtárral készült, amit az 5.2 ábra mutat be. A felületen lehetőség van kép kiválasztására a „Choose Image” gombbal. Elsősorban „NifTi” (Neuroimaging Informatics Technology Initiative) fájlformátum feltöltésére van lehetőség, ami egy olyan nyílt formátum, amit leggyakrabban agyi és orvosi képalkotásban használnak, viszont hagyományos kép típust is fellehet tölteni. Lényegében a „NifTi” formátumú fájl egy képsorozat, jelen esetben az agyról, így a megjelenítésben egyszerre csak egy darab képet jelenít meg a program. A különböző rétegek között az egér görgőjével lehet váltani, ha a kurzor a kép felett van, ezen állítás az eredmény maszkokra is igaz, ha a módszer több darabot adna vissza. A képek felett van megjelenítve melyik réteg látszódik mennyiből. A bal oldali képen lehetőség van magpontok elhelyezésére és eltávolítására a bal, illetve a jobb egérgombbal, viszont ez csak a manuális módszer használja fel. Az 5.2 ábrán a képkiválasztó gomb alatt látható 3 legördülő listából balra az elsőn lehet kiválasztani a futtatni kívánt módszert. A középsőn lehet választani a „Release” és „Debug” mód között, ahol az előbbi a normál futtatást, míg az utóbbi a köztes lépések megjelenítését jelenti új ablakban, mint a 4.1 ábra. Végezetül a jobb oldalin a „Mask” és „Contour” opciók között lehet választani, ami az eredményt egy fekete alapon fehér maszkként, vagy az ábrán látható módon jeleníti meg. A megjelenített képek mellett jobb oldal található az aktuális módszerhez beállított vezérlőpanel, ahol az előre beégetett értékeket lehet változtatni. Végezetül az „Evaluate” gomb megnyomásával lehet futtatni a kiválasztott algoritmust az aktív rétegen a beállított paraméterekkel.

5.2. Blokk- és intenzitásalapú magpont-kijelölés megvalósítása

A programban való elhelyezéshez, el kell tudni helyezni a pipeline-ban (5.1. ábra). Az első lépés a koponya eltávolító függvény implementálása, ami egy előfeldolgozó lépés lesz, majd jön a maga a magpont kiválasztás és végül a standard régiónövelés. A koponya eltávolítás a `SkullStripping` névre hallgató osztályban készült el, aminek belépési pontja a `process` nevű függvény. A cikkben mellékelt pszeudokódban matlab beépített függvényhívásokat használnak. A legtöbb függvénynek van opencv könyvtárbéli megfelelője.

```
class SkullStripping:
 def process(ctx):
  //…
  cv2.threshold(img, 0, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)
  //…
  se = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (2*r+1,2*r+1))
  cv2.morphologyEx(bw, cv2.MORPH_OPEN, se)
  //…
  return ctx
```

5.1 kódrészlet: Részlet a koponya eltávolítás

Az 5.1 kódrészlet példát hoz az opencv metódsuok használatára, követve a 3.1 kódrészlet pszeudokódját. Az első lépés az Otsu küszöbölés volt, amit `cv2.threshold` nevű metódus megfelelő paraméterezésével érünk el, ami után a morfológiai nyitást, és az összes morfológiai műveletet a `cv2.morphologyEx` függvény és annak megfelelő paraméterezésével érünk el, jelen esetben a `cv2.MORPH_OPEN` paraméterrel érünk el, és egy lemezszerű strukturáló elemmel, amit a `cv2.getStructuringElement` nevű metódussal tudunk elkészíteni. Következő lépés a dilatáció, amit a `cv2.MORPH_DILATE` paraméterrel valósítható meg. A következő lépés már nem valósítható meg egy darab függvény hívással, ami a legnagyobb bináris objektum kiválasztása. Ebben az esetben egy egyszerű bináris címkézést alkalmazok, amit `cv2.connectedComponentsWithStats` nevű függvénnyel fogunk elvégezni, ami visszaadja a címkézett képet és minden komponens statisztikáját, például a területet, ami alapján választjuk ki az objektumot. A legnagyobb objektumon elvégzett zárást egyértelműen a `cv2.MORPH_CLOSE` paraméterrel fogjuk elvégezni, majd következik a lyukfeltöltés, ami nincs megvalósítva opencv függvénykönyvtárban (5.2 kódrészlet). Az adott bináris képen egy darab objektumunk van, amiben lehetnek lyukak, így elvégzünk egy egyszerű régiónövelést a `cv2.floodFill` metódussal, ahol a

magpont a háttér egy pontja lesz, például (0,0) koordináta. Először létre kell hozni egy üres maszkot, amiben majd írni fog ez a metódus, és a bejövő bináris maszkról készíteni egy másolatot, mivel módosítja a függvény, és az eredetijét még használjuk.

```
def fillHoles(self, bw):
    mask = np.zeros((bw.shape[0] + 2, bw.shape[1] + 2), dtype=np.uint8)
    inputMask = bw.copy().astype(np.uint8)
    cv2.floodFill(inputMask, mask, (0,0), 255)
    inputMaskInv = cv2.bitwise_not(inputMask)
    outImg = cv2.bitwise_or(bw.astype(np.uint8), inputMaskInv)
    return outImg
```

5.2 kódrészlet: Lyukfeltöltése egy objektumos bináris képen

A kapott maszkot invertáljuk a `cv2.bitwise_not` függvénnyel, így a háttér és az objektum fekete lesz és az objektumban lévő lyukak fehérek. Utolsó lépésként bitenkénti vagyolást végzünk el a `cv2.bitwise_or` operációval az eredeti egy objektumos maszkon és az invertálás után kapott maszkon, tehát csak az objektum és azon belüli terület lesz fehér. Végül az eredeti képből csak a maszk által jelölt részt választjuk ki, és bele írjuk a `PipelineContext` objektumba, hogy a következő lépés használni tudja.

A következő lépés a magpont kijelölés, ami a `BlockBasedSeedSelector` nevű osztályba került, és a `select(ctx)` metódus a belépési pontja. Az adathalmazban 240x240 méretű képek vannak, míg a cikk által használtban 256x256 méretűek voltak. Mindkettő 8-cal osztható így nincs probléma, de azért elvégzünk egy kép újra méretezést, hogy biztosan 8-cal osztható legyen, ha nem lett volna. Mivel a képek kis méretűek és minimális a számolás, így nem szükséges felbontani őket 8x8-as blokkokra, bőven elég két egymásba ágyazott ciklus, ami 8-asával lépeget. Minden iterációban az adott blokk középpontját és átlagát elmentjük egy listába, majd a ciklus után kiválasztjuk az 5 legnagyobb átlagintenzitású blokk középpontját, beállítjuk őket a kontextusban a magpontoknak. A régiónövelést az `OpenCVRegionGrowing` osztály `run(ctx)` metódusa végzi. Kiolvassa a kontextusból a magpontokat, majd egy ciklusban mindegyikre elvégez egy régiónövelést a `cv2.floodFill` metódussal. Mivel csak egész számokon értelmezett ez az algoritmus így a futtatás előtt a képet 0-255 közé normalizáljuk, és az intenzitás határ alapértelmezett értékét 75-re állítjuk be, amit a felhasználó, ha akar megtud változtatni. Az elkészült maszkokat egy listába gyűjti és elmenti a kontextusba. Ezután a felhasználói felület megjeleníti mind az 5 maszkot, amik között lehet lapozni.

Az általam javasolt módszer a csúszó ablak technika, ahol nem 8x8-as méretű blokkokra osztom fel a képet, hanem egy fix méretű ablakot léptetek végig a képen és pontozom az összes ablakot, majd választom ki az N darab, alapból 5-nek beállított legjobb ablakot. Az ablakok létrehozását és sorba rendezését a `SlidingWindow` osztály valósítja meg, míg a magpont választást a `WindowSeedSelector`. Először egy generátor függvénnyel végig léptetjük az ablakot a képen. Minden egyes ablakra kiszámoljuk a beállított módszer szerinti pontszámot (5.3. kódrészlet), alapértelmezetten az átlagintenzitást, majd a képpel megegyező méretű üres képre kiírjuk az ablakközéppontjának koordinátáiba, ezzel készítve egy ponttérképet (3.4 ábra).

```
def scoreWindow(self, window, mode):
    //...
    elif mode==”std”:
        score = -np.std(values) * 1-(np.mean(values))
    elif mode==”blob”:
      LoG = gaussian_laplace(window, sigma)
      score = -(LoG.mean()) * np.mean(values)
    return score
```

5.3 kódrészlet: Részlet az ablak pontszámításból

Az átlagintenzitás és maximum intenzitás megtalálása, csak egy `np.mean` és `np.max` függvényhívás, viszont a másik két statisztika megtalálása trükkösebb. Az 5.3 kódrészlet kiemeli a szórás és LoG alapú pontozást. A ponttérképből a maximális értékek vannak felhasználva és egyszerre mindig csak a vizsgált ablakot lehet látni, ezzel ellentétesen a szórásból a kisebb érték a jobb, ezért veszem mínusz egyszeresét, majd súlyozom az átlaggal, pontosabban az (1-átlag)-gal, mert a 0-hoz közelebbi értékeknél nem szeretnénk, hogy nőne. A másik kiemelt számolásnál is hasonló probléma merül fel. Először a `gaussian_laplace` elvégzi a konvolúciót, aminek vesszük az átlagát negatív előjellel, így a 0-hoz közeli negatív értékek pozitívra fordulnak az eredetileg világosabb helyeken, így elég az ablak átlagával súlyozni.

```
def nonMaxSupression(self, scoreMap, size=3, th=0):
    localMax = maximum_filter(scoreMap, size=size) == scoreMap
    maxima = np.argwhere(localMax & (scoreMap>th))
    return maxima
```

5.4 kódrészlet: Nem-maximális elnyomás

A kész ponttérképen nem-maximális elnyomást (NMS) alkalmazunk a scipy.ndimage csomag `maximum_filter` metódusával az 5.4 kódrészletben, ami a paraméterben megadott méretben tartja

meg a maximumokat, pontosabban egy olyan mátrixot ad vissza, ahol az adott ablakban minden pozíción a megtalált maximum szerepel. Ezt összevetjük az eredeti ponttérképpel és csak az adott mezőn megegyező értékeket tartjuk meg, ami után egy küszöböt használva kidobjuk a túl alacsony értékeket, például a 0-t. A megmaradt ablakok sorba rendezése és legjobb N db megtartása, így már kivitelezhető. A megtartott ablakokból alapértelmezetten a középpontja lesz kiválasztva, mint magpont, de a felhasználónak opciója van beállítani az ablakban megtalált maximum értéket, mint magpont, ami előnyösebb lehet kevésbé homogén tumorok esetén.

5.3. Divergenciaalapú magpont-kiválasztás megvalósítása

Azon előfeldolgozó lépések, amelyek egy egyszerű függvény hívással megoldhatóak, mint a 0-1 közé normalizálás vagy a gamma korrekció, létrehoztam egy `PreprocessingStep` nevű osztályt, ami paraméterben egy függvényt vár és annak paramétereit, majd meghívja ezt a `process(ctx)` metódusával, hogy beilleszkedjen az előző fejezetekben tárgyalt csővezetékbe (5.1. ábra). A szétválasztás és egyesítés algoritmus a `SplitMerge` nevű osztállyal lett megvalósítva. A klasszikus megvalósítás négyfa adatstruktúrát használ, és ameddig homogenitási kritérium nem teljesül, addig rekurzívan a képet 4 részre bontja. Utána lentről felfele azok a szomszédos régiók, amelyek kielégítik a homogenitási kritériumot össze lesznek vonva. Ha szegmentálásra használják fel, akkor általában minden csúcsa a fának egy régiót fog jelölni. A cikkben tárgyalt homogenitási kritériumnál először kiszámoljuk a hisztogramot az adott régióra az `np.histogram` függvénnyel, ami visszaad 2 listát, ahol az egyik a megszámolt értékek, a másik pedig az intenzitás értékek határai, hogy milyen intenzitások kerültek egy osztályba a hisztogramon, majd ezt a két értéket fogjuk felhasználni a `getMeanSumIntensity` függvényben az 5.5 kódrészletben `hist` és `bin_edges` név alatt.

```
def getMeanSumIntensity(self, hist, bin_edges, intensityRange):
    low, high = intensityRange
    mask = (bin_edges[:-1] >= low) & (bin_edges[1:] >=high)
    bin_centers = (bin_edges[:-1] + bin_edges[1:]) / 2
    sum = np.sum(bin_centers[mask] * hist[mask])
    N = np.sum(hist[mask])
    avgHyper = 0
    if N != 0:
        avgHyper = sum / N
    return avgHyper, N
```

5.5 kódrészlet: Átlagintenzitás és pixel darabszám meghatározása a homogenitási kritériumhoz

Meghatározzuk a maszkot a küszöbértékek és az előbbi intenzitás határok alapján, és ezen belül összeszámoljuk hány darab intenzitásérték van. A cikk emellett még az intenzitás átlagot is számolja, bár a megadott képlet csak az előfordulási mennyiségeket átlagolja, de a kapott eredményeik az intenzitás átlagot használják fel, amit úgy lehet elérni, hogy az előfordulási mennyiségeket nem csak összeadni kell, hanem összeszorozni a hozzájuk tartozó intenzitásértékkel, így kapva használható információt. Ezeket az intenzitásértékeket a `bin_centers` változó jelöli, mivel a `bin_edges` változó az osztályok határait jelöli, amikből több van, így a felső és alsó határ átlagát véve számolom ki az intervallum közepét. A választott homogenitási kritérium azt mondja meg, hogy előfordul-e elváltozott, hiperintenzív terület a vizsgált régióban. A szétválasztás `split` nevű függvény valósítja meg, ami rekurzívan 4 részre bontja a képet, ha nem homogén vagy nem érte el a 3. szintet. Ezután következne az összevonás, ami a szomszédos területeket összevonná, de erre nincs szükség, hiszen ugyanez a feltétele a végén a terület kiválasztásának, tehát az összevont területeket biztosan nem választanánk ki a végén. Így elég a 3. szinten megvizsgálni a szétbontási fázisban, hogy tovább bontható-e az adott régió, ha igen felvesszük egy listába. Az összevonás rész csak formális, amit a `merge` függvény valósít meg, ami a listában levő régiókból elkészít egy maszkot, és kivágja az eredeti képből ezt a részt, így megkapva a ROI-t.

A megkapott területből a `DivergenceSeedSelector` osztály fogja kiválasztani a magpontokat a `select(ctx)` metódusával.

```
def select(self, ctx):
    //…
    P = hist / np.sum(hist)
    div = np.gradient(hist) * P
    idx = np.argmax(div)
    tail = div[idx + 1:]
    nearest_zero = np.argin(np.abs(tail))
    optimal_threshold_idx = nearest_zero + 1 + idx
    optimal_threshold = bin_edges[optimal_threshold]
    //…
    return ctx
```

5.6 kódrészlet: Magpont kijelölés részlet

Az előző lépésben megkapott régiónak vesszem a hisztogramját, majd az `np.gradient` függvénnyel a deriváltját, amit beszorzok a normalizált hisztogrammal (5-6. kódrészlet). A kapott divergenciából `np.argmax` metódussal megkeresem a maximumhelyet, majd az utána következő elemek abszolútértékén az első minimumhelyet az `argmin`, és `abs` függvényekkel, hogy megkapjuk a maximumhely után az első 0-hoz legközelebbi helyet, amit kiválasztunk optimális küszöbértéknek. Végül pedig készítünk egy üres maszkot, amin 1-re állítjuk azon pixeleket, ahol a vizsgált régión nagyobb az intenzitás érték, mint az optimális küszöb. Ezen koordinátákat kinyerjük a maszkból az `np.argwhere` függvény segítségével, ami a sort, majd oszlopot adja vissza, ezért mi azt megfordítjuk, hogy oszlop és sor legyen, azaz (x,y) koordinátákkal tudjunk dolgozni. Végül pedig elmentjük a kontextusba a magpontokat és az optimális küszöbértéket, amit később felhasználunk.

A régiónövelést azzal kezdjük, hogy a magpontokon iterálunk. Ha az adott pont már eleme az összesített maszknak, akkor kihagyjuk, egyébként elvégzünk az adott pontra egy régiónövelést, majd az összesített maszkot frissítjük a régiónövelés által adott maszkkal. Egy magpontra a `regionGrowing` függvény végzi el a műveletet, ami egy sor adatstruktúrával dolgozik, amibe a kezdőpontot tesszük (5.7. kódrészlet).

```
def regionGrowing(self, image, seed, optimal_threshold):
    //…
    queue = deque([tuple(firstSeed)])
    while len(queue) > 0:
      x, y = queue.popleft()
      //…
      if np.abs(intensity – regionMean) >= np.abs(regionMean-optimal_threshold):
         continue
     segmented[y+dy. x+dx] = 1
    //…
    return segmented
```

5.7 kódrészlet: Régiónövelés részlet

Addig iterálunk amíg nem lesz üres a sor. Az első művelet a sorból a legrégebben rajta levő pont levétele, utána minden érvényes szomszédjára, ami nem lóg le a képről és még nem vizsgáltuk, arra megnézzük, hogy teljesül-e a homogenitási feltétel, azaz a régió átlagintenzitásától kisebb mértékben tér el a vizsgált pixel, mint az optimális küszöbérték és a régióátlag eltérése. Ha teljesül a feltétel hozzávesszük a maszkhoz a pontot és felírjük a sorba. Miután egy magpontra befejeződik

a régiónövelés, egy összesített maszkban 1-re állítom azon értékeket, ahol a visszaadott maszknak is volt értéke.

Ide kéne a javítások implementálása: adaptív hiperintenzív range és küszöb – aztán seed filter, esetleg a leszakadó blokkok elhagyása.

6. Eredmények

Az én általam használt adathalmaz a 2020-ban szervezett agytumor szegmentálás kihíváshoz használt adathalmaz, röviden BraTS2020 lesz, ami 369 páciensről készült MRI vizsgálat eredményeit tartalmazza. T_1, súlyozott T_1, T_2, FLAIR alapú képeket tartalmaz, ahol mindegyik kép 155 db axiális szeletből áll, és minden szelethez tartozik már a szakorvosok által szegmentált maszk a tumorról, tehát tudjuk ellenőrizni majd a szegmentálás eredményét. Megjelenésben, formában és szövettan tekintetében lényegében homogén daganatok szerepelnek a képeken, nevezetesen gliómák, amik FLAIR típusú képeken jól elkülönülnek. Az adatok már előfeldolgozottak, például ugyanarra a felbontásra vannak interpolálva, a koponya eltávolítás is megtörtént.

6.1. Használt metrikák

Magát a magpont kiválasztást, nem igazán lehet értékelni, a különböző cikkek se teszik, viszont magát a szegmentálás végeredményét lehet. Ehhez kikell számolni a konfúziós mátrixot, ami a valódi pozitív (TP), valódi negatív (TN), hamis pozitív (FP) és hamis negatív (FN) értékekből áll. A TP és TN jelzi, azt a pixelszámot, amit mi tumornak találtunk az valóban tumorhoz tartozik, amit nem rendeltünk hozzá az valóban nem tartozik hozzá. A FP és FN értékek pedig a hibásan besorolt értékeket jelzik, hogy amit tumorhoz soroltunk az valójában nem hozzá tartozik és fordítva, amit nem soroltunk hozzá az valójában hozzá tartozik.

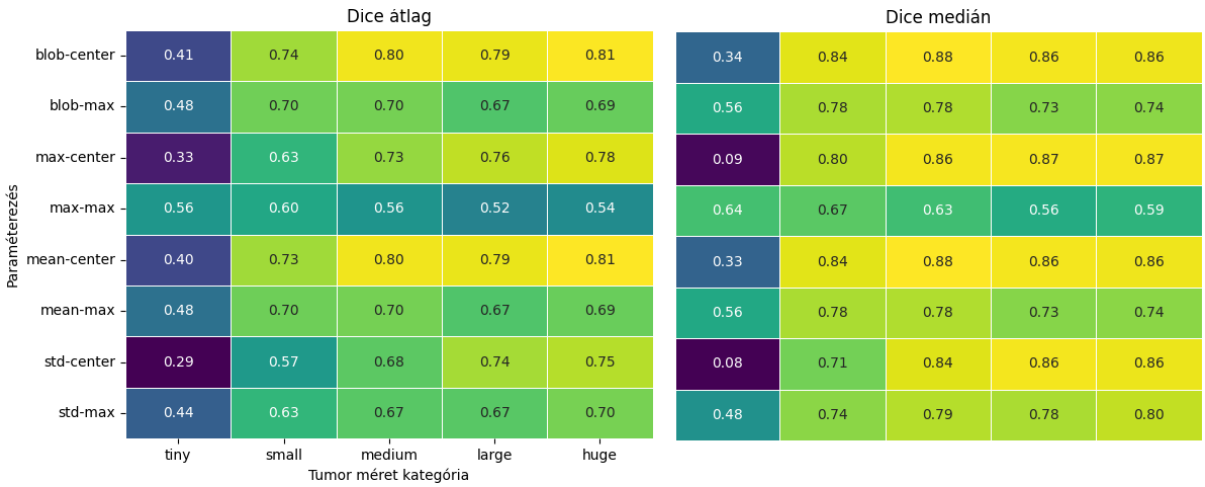
Ezen számokból számos metrikát elő lehet állítani, amit a feldolgozott cikkek is tárgyalnak. Elsősorban a pontosság (accuracy), a helyesen osztályozott pixelek arányát mutatja, ami félrevezető lehet kiegyensúlyozatlan adatoknál. Jelen esetben, ha túl kicsi tumor, például 5%-a a képnek és a maradék háttér, és mi nem találjuk el a tumort, hanem mondjuk a háttérből pár pixelt jelölünk ki szegmentált területnek, még akkor is 90% felett lenne a pontosság. A Jaccard-index vagy másnéven metszet-unió arány (Intersection over Union, IoU) a szegmentált területek átfedését méri, kihagyja a hátteret az összehasonlításból az előző metrikához képest. Használják még objektum detektálásnál a befoglaló téglalapok összehasonlítására. A következő metrika a Dice-együttható, ami két halmaz közötti hasonlóságot méri hasonlóan az előző IoU-hoz, viszont a

ténylegesen eltalált értékek duplán számítanak, így kicsit megbocsátóbb pontszám. A specificitás (Sp) értéke magas, ha a nem tumorhoz tartozó területeket jól jelöljük és minimális a hamis pozitív értékek száma, ennek a párja a szenzitivitás, amit a cikk átfedési hányadnak hív (overlap fraction: OF) ami magas, ha a helyesen tumorhoz soroltak száma magas és minimális a hamis negatívak száma. A többlethányadot (extra fraction, EF), ami azon pixeleket jelöli, amiket hamisan tumorhoz sorolunk be, tehát minél kisebb annál jobb, feltéve, hogy kevés a fals negatív. A legutoljára említett EF és OF helyett érdemesebb lehet Fals pozitív rátát (FPR) és Fals negatív rátát (FNR) nézni, ami azt mondja meg, hogy ha FPR magas akkor túlszegmentálás történt, hiszen hamisan tumorhoz soroltuk, viszont háttér, míg a FNR esetén az ellenkezője igaz, vagyis alulszegmentálás történt.

- másik cikk AO, FPR, FNR, MA, MAPE, Rerr

6.2. Összehasonlítás

Az eredeti módszert hasonlítom össze annak továbbfejlesztésével. Mivel továbbfejlesztésnek a csúszóablak-módszert ajánlottam, annak is 4 különböző ablak rangsorolási stratégiát, majd az ablakokból 2 magpont kiválasztási startégiát, így először ezt a 8 különböző lehetőséget hasonlítom össze egymással. A 6.1 fejezetben említett metrikákat számoltam és mentettem el csv fájlokba, majd dolgoztam fel őket. Az említett metrikák közül a pontosság végig magas. Az IoU és a Dice-együttható ugyanazt mérik csak az utóbbi megengedőbb, én is ezt fogom használni a legfőbb mérőszámnak. Mivel többszázezer szeletnyi adat áll rendelkezésre, így lehet őket különféleképpen csoportosítani, például tumor mérete szerint.

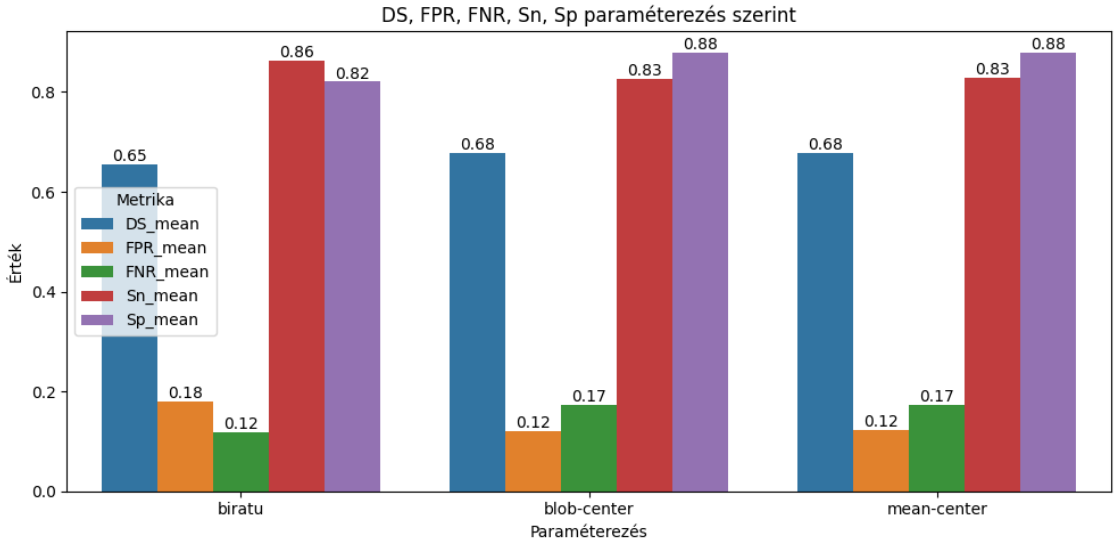


6.1. ábra: Dice-együttható átlag és medián hőtérkép, módszer és tumorméret szerint

A 6.1. ábrán az előbb említett 8 paraméterezésnek lehet látni a Dice-együtthatójának átlagát, illetve mediánját a tumor mérete szerint csoportosítva. A mediánt nézve azt lehet látni, amikor az ablak a

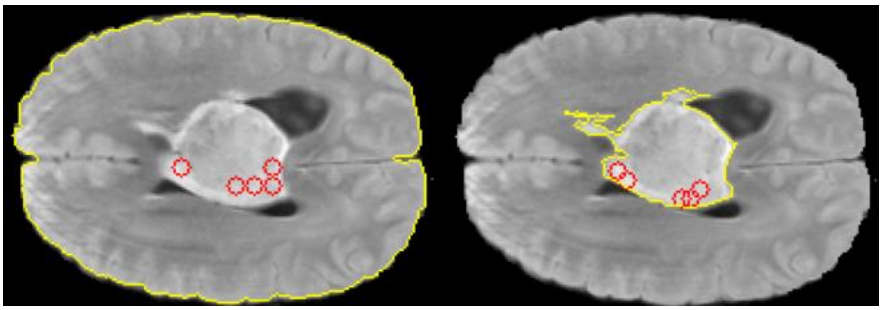
maximum intenzitás szerint van pontozva, és az ablak közepe van választva magpontnak, akkor a lehető legmagasabb az elért pontszám, kivéve a pár pixel méretű tumorok esetén, viszont ugyanennek a módszernek az átlagát nézve 87% helyett csak 78%-ot ér el a legjobb esetben. Amikor az ablak az átlagintenzitást kapja pontszámnak, vagy a folt szerűséget nézi a LoG-val, továbbra is az ablak közepét választva magpontnak, akkor hasonlóan magas értékeket ad a mediánra, mint az előző esetben, kb. 86%-ot, azzal a különbséggel, hogy a legjobb esetben is 79-81%-ra esik vissza a teljesítmény, ami így jóval stabilabbnak tűnik. Azt érdemes lehet megjegyezni, hogy amikor a maximum intenzitású pixel van magpontnak választva az ablakból, az minden esetben jobban teljesít, a legkisebb méretű tumorokon, mint a párja. A metrikák között említett többlethányad (EF) alacsony, amikor magasabb a Dice-együttható, és kiugróan magas, a pici tumorok (6.1. ábra első oszlop) esetén. Az előbb említett legjobb Dice pontszámot elérő paraméterezéseknek alacsony a fals pozitív rátája, viszont annál kicsivel magasabba fals negatív rátája, ami azt mutatja, hogy a hiba az alul szegmentálásból fakad.

Az eredeti cikk elért átlagos Dice-együtthatója a közepes méretű tumorokon ugyanolyan átlagot ér el, mint az előbb említett továbbfejlesztés legjobb paraméterezései (6.2 ábra). A közepesnél nagyobb tumorokon 1-2%-kal jobb átlagos eredményt ér el, míg a közepesnél kisebb tumorokon ugyanennyivel rosszabbat. Az összes szeletre nézve az átlagokat a 6.3 ábrán, azt látni, hogy a javított módszer átlagosan jobb eredményt ér el, 2%-kal a Dice-együtthatót nézve, illetve kisebb a fals pozitív rátája 6%-kal, ami azt jelenti kevésbé szegmentál túl adott képeken, viszont jobban alul szegmentál, amit az FNR érték mutat és 5%-kal nagyobb. Az érzékenység (Sn) 3%-kal csökkent, ami azt mondja meg hogy átlagosan kevesebb tumorterületet találunk, míg a specificitás (Sp) 6%-kal nőtt, ami a helyesen eltalált háttér pixeleket jelöli, így az Sn és Sp nem ütközik a FPR és FNR által mutató értékekkel.



6.3. ábra: Az átlag pontszámok minden szeletre az eredeti és továbbfejlesztett módszer 2 változata szerint

Érdekes még, hogy a csúszóablak-módszer milyen képeken teljesít jobban az eredeti módszerhez képest. Az eredeti módszer kidolgozásából fakadóan nagyon rosszul teljesít, sőt egyáltalán nem talál el semmit, abban az esetben, amikor az agyképen egy olyan alacsony rétegben kell dolgoznia, amiben az agy része is kicsi, mert a morfológia alapú koponya eltávolítás adott esetben képes eltávolítani azokat a minimális részeket, amiken a tumor található. Hasonló eset amikor pici és jól határolt a tumor és 2 ablak közé esik, ezzel nem belelógva az ablak középpontjára. A régiónövelés hibájából adódóan rosszul teljesít, amikor a tumor és az agy közötti kontraszt minimális, mivel tumoron belüli magpontokat talál, viszont a régiónövelés túl megy a tumor határain (6.3. ábra). Ezeket a csúszóablak-módszer orvosolja, hiszen nem használja a morfológiai koponyaeltávolítást, és a módszerből adódóan finomabban mintavételez, így magasabb intenzitásokat választva és ezzel elkerülve a régiónövelésből adódó hibát, ami viszont olyan képeken fog megjelenni, amikor a tumor nem feltétlenül elég homogén így túl magas intenzitásokat választ magpontnak és nem tud lejjebb menni (6.4. ábra).



6.4. ábra: BraTS20_Training_340_flair 86-ös rétegen magpontkijelölés eredeti (bal) csúszóablak-módszer (jobb)

Aztán Cikk2 Bir21 Itt az eredeti és a másik lesz összehasonlítva, (az eredet is rosszabb mint 50% Dice, :/ borzasztó magas FNR)

Irodalomjegyzék

Biratu, E. S., Schwenker, F., Debelee, T. G., Kebede, S. R., Negera, W. G., & Molla, H. T. (2021). Enhanced Region Growing for Brain Tumor MR.

Saad, N. M., Bakar, S. A., Muda, A. S., Mokji, M. M., & Abdullah, A. R. (2012). Automated Region Growing for Segmentation of Brain Lesion in Diffusion-weighted MRI.

Saad, N. M., Bakar, S. A., Muda, S., & Mokji, M. (2010). Automated segmentation of brain lesion based on diffusion-weighted MRI using a split and merge approach.

Még a Review-s pár pdf

https://www.inf.u-szeged.hu/~mate/medproc/OKF.pdf MÁtéEörs jegyzete

https://hu.wikipedia.org/wiki/Agydaganat ---> források részt kiszedni

https://learnopencv.com/filling-holes-in-an-image-using-opencv-python-c/

https://www.youtube.com/watch?v=NZkI4O8u6Mc&t=548s

https://www.kaggle.com/datasets/sudipde25/mri-dataset-for-detection-and-analysis?select=BrainTumorDataPublic -----> ezekből a kért forrásokat betenni

Átnézni ezek közül melyiket használtam:

https://www.yalemedicine.org/conditions/primary-brain-tumors

https://www.digirad.com/spect-vs-pet/

https://www.apollohospitals.com/hu/diagnostics-investigations/pet-ct-scan

https://hu.uniproyecta.com/k%C3%BCl%C3%B6nbs%C3%A9gek-a-kis%C3%A1llat-%C3%A9s-a-szem%C3%BCveg-k%C3%B6z%C3%B6tt/

https://ujbudamedicalcenter.hu/mit-kell-tudni-a-kontrasztanyagos-ct-vizsgalatrol/

https://ujbudamedicalcenter.hu/minden-amit-tudni-kell-a-koponya-ct-vizsgalatrol/

https://www.digirad.com/spect-vs-pet/

https://bhc.hu/betegsegek/agydaganat

https://www.medmastery.com/guides/brain-ct-clinical-guide/recognizing-intra-axial-tumors-brain-computed-tomography-ct-and?srsltid=AfmBOoodTXiqX10utBRMAMiHfn3vK8Odh7TYwpb12ZWkX6AVxj39t0Ss

https://www.ajnr.org/content/27/3/475

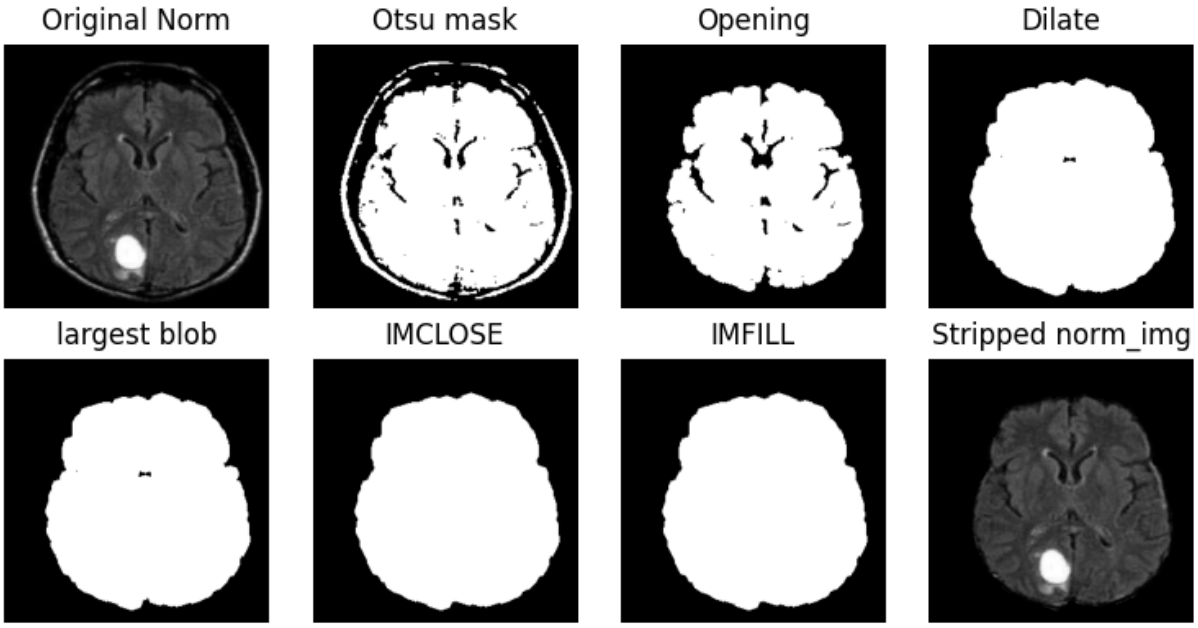
Nyilatkozat

Alulírott ……..…… szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Intézet …….. Tanszékén készítettem, …….……..…… diploma megszerzése érdekében.

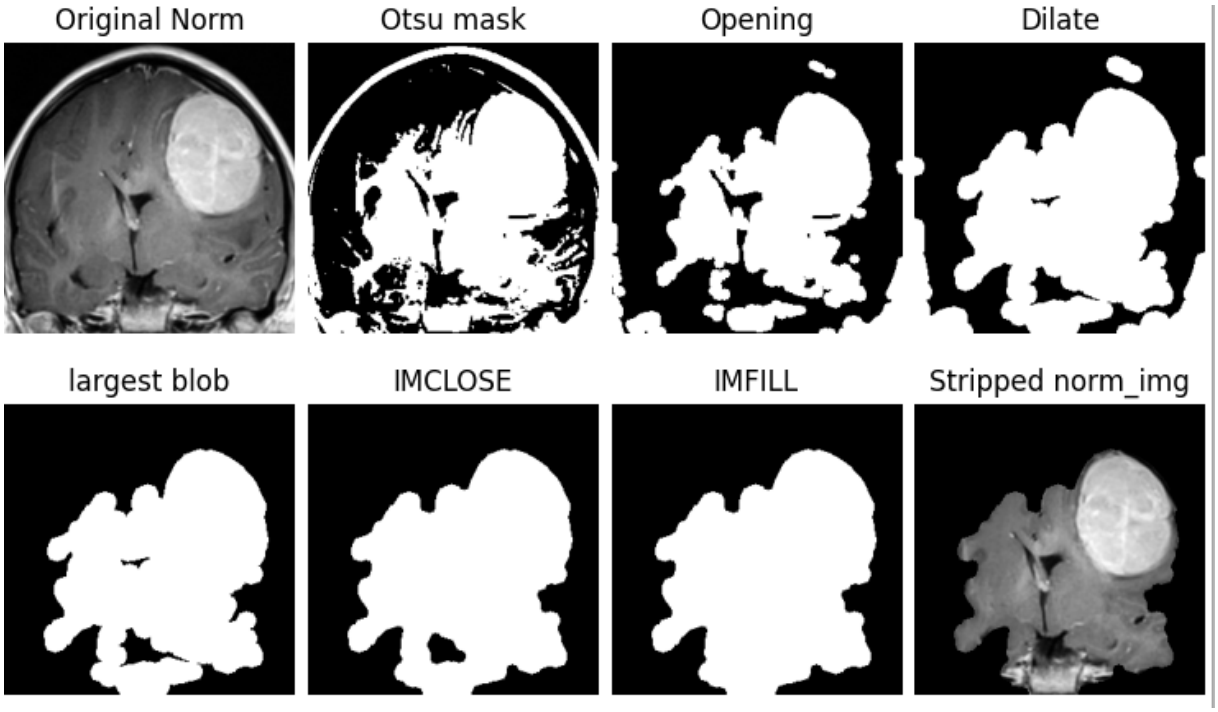
Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel. Tudomásul veszem, hogy szakdolgozatomat / diplomamunkámat a Szegedi Tudományegyetem Diplomamunka Repozitóriumában tárolja.

Melléklet

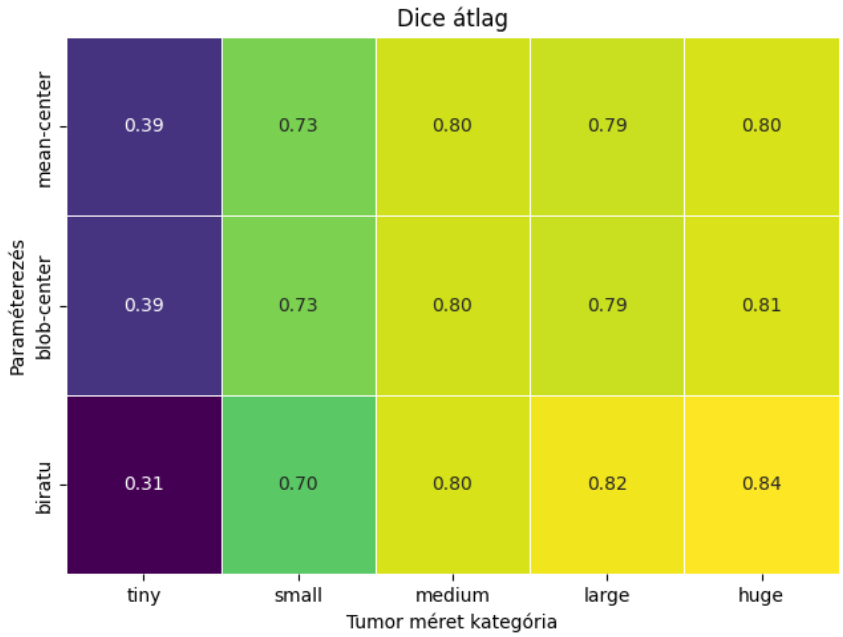
1. 3.2 ábra: Koponya eltávolítás lépései



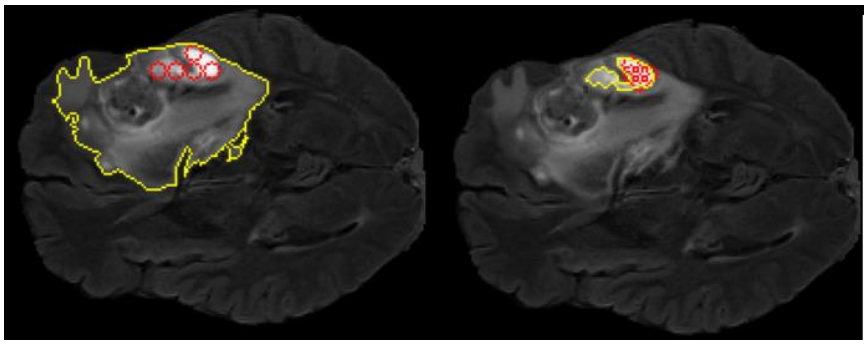
2. 3.3 ábra: Koponya eltávolítás lépései



3. 6.2. ábra: Dice-együttható átlagosan tumorméret szerint, az eredeti(biratu) és továbbfejlesztett módszer hőtérképe



4. 6.5. ábra: BraTS20_Training_26_flair 62-es rétegen magpontkijelölés eredeti (bal) csúszóablak-módszer (jobb)



5. a
6. a
7. a
8. a