Primjena neuronskih mreža za segmentaciju tkiva u biomedicinskim snimkama pacijenata

Delovski, Boris

Master's thesis / Diplomski rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture / Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:235:481058

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-12-18**

Repository / Repozitorij:

Repository of Faculty of Mechanical Engineering and Naval Architecture University of Zagreb

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Boris Delovski

Zagreb, godina 2019

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

DIPLOMSKI RAD

Primjena neuronskih mreža za segmentaciju tkiva u biomedicinskim snimkama pacijenata

Mentori:

Prof. dr. sc. Bojan Jerbić, dipl. ing.
Dr. sc. Filip Šuligoj, dipl. ing.

Student:

Boris Delovski

Zagreb, godina 2019.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se svom mentoru, profesoru Bojanu Jerbiću te asistentu Filipu Šuligoju koji su mi svojim stručnim savjetima pomogli oblikovati i realizirati ovaj rad. Zahvalio bih se na svom znanju koje sam stekao za vrijeme izrade ovog rada, te na pruženoj prilici da svoju ideju sprovedem od početka do kraja.

Također bih se želio zahvaliti svojoj obitelji i prijateljima na pruženoj podršci i razumijevanju.

Boris Delovski

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za diplomske radove studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment,
inženjerstvo materijala te mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum:	Prilog:
Klasa:	
Ur. broj:	

DIPLOMSKI ZADATAK

Student: **BORIS DELOVSKI** Mat. br.: 0035193300

Naslov rada na hrvatskom jeziku: **PRIMJENA NEURONSKIH MREŽA ZA SEGMENTACIJU TKIVA U BIOMEDICINSKIM SNIMKAMA PACIJENATA**

Naslov rada na engleskom jeziku: **APPLICATION OF NEURAL NETWORKS FOR TISSUE SEGMENTATION IN BIOMEDICAL PATIENT IMAGES**

Opis zadatka:

U medicini se roboti mogu primijeniti kao dio složenih i računalno potpomognutih sustava za dijagnozu, predoperativno planiranje, operaciju, poslijeoperacijsku njegu pacijenata i bolničku logistiku. Kirurški robotski sustavi mogu poboljšati postojeće operativne postupke u pogledu veće učinkovitosti, točnosti i veće pouzdanosti. U sklopu unapređenja primjene robota u neurokirurškim operacijama cilj rada je istražiti mogućnosti primjene neuronskih mreža za automatsku segmentaciju tkiva (npr. tumora, ventrikularne šupljine ili žila) prema intenzitetu „voksela" na tomografskim snimkama pacijenta, a kako bi se unaprijedila i ubrzala dijagnostika i automatsko planiranje kirurškog zahvata. Segmentacija slike je proces dijeljenja digitalne slike na više podskupova piksela ili voksela prema zadanim ili naučenim karakteristikama. Segmentirana slika može ponuditi kvalitetniju razlučivost varijacije tkiva od uobičajene optičke analize digitalne slike pacijenta.

Svi razvijeni softverski moduli i algoritmi trebali bi biti testirani na medicinskom skupu podataka u okviru projekata RONNA i NERO.

U radu se predlaže korištenje c++ ili Python knjižnice (Tensor flow, VTK i OpenCV) koje imaju ugrađene funkcije za primjenu algoritama umjetne inteligencije, za uvoz 3D volumetrijskih medicinskih slika i za procesuiranje slika.

U radu je potrebno navesti korištene izvore i literaturu.

Zadatak zadan:
26. rujna 2019.

Rok predaje rada:
28. studenog 2019.

Predviđeni datum obrane:
04. prosinca 2019.
05. prosinca 2019.
06. prosinca 2019.

Zadatak zadao:

prof. dr. sc. Bojan Jerbić

Predsjednica Povjerenstva:

prof. dr. sc. Biserka Runje

SADRŽAJ

POPIS SLIKA

POPIS OZNAKA

Oznaka	Jedinica	Opis
x_i	-	element s nekom određenom bitnom karakteristikom
y_i	-	oznaka kojom se označava neki element
$\nabla f(w)$	-	gradijent funkcije
$\partial f(w)$	-	Opis oznake
η	-	stopa učenja
$x(t)$	-	položaj u nekom određenom trenutku
$s(t)$	-	konvolucijska funkcija
a	-	starost učenja za težinski faktor učenja
TP	-	točno određena pozitivna vrijednost segmentacije
TN	-	točno određena negativna vrijednost segmentacije
FN	-	netočno određena negativna vrijednost segmentacije

SAŽETAK

Ovaj rad posvećen je primjeni strojnog učenja u području biomedcinske segmentacije tkiva ljudskog mozga; prvenstveno se fokusira na segmentaciju ventrikula koje se mogu primijetiti na CT snimkama mozga. Opisan je način prikupljanja podataka, način na koji se ti podaci pripremaju za korištenje te sama umjetna neuronska mreža iskorištena za postupak segmentacije. Cilj rada je dokazati da se primjenom konvolucijske umjetne neuronske mreže može postići konkurentna točnost kod segmentacije ventrikula u mozgu, te postaviti temelje za eventualno izvođenje segmentacije drugih tkiva u mozgu.

Ključni pojmovi: strojno učenje, segmentacija ventrikula, biomedicinske snimke, konvolucijske umjetne neuronske mreže

SUMMARY

This diploma work is devoted to the application of machine learning in the field of biomedical tissue segmentation of the human brain with a primary focus on segmentation of the ventricles that can be seen on CT brain images. The paper contains a description of several thing: how the data is collected, how is this data prepared and what artificial neural network is used for the segmentation process. The aim of the paper is to prove that it is possible to achieve competitive accuracy in the segmentation of the ventricles in the brain using a convolutional artificial neural network, and to lay the groundwork for the eventual segmentation of other tissues in the brain.

Key ideas: machine learning, segmentation of ventricles, biomedical imaging, convolutional neural networks

1. UVOD

U ovome radu predložit će se primjena dubokog učenja kod segmentacije ventrikula koji se pojavljuju na CT snimkama ljudskog mozga. Na početku rada pojasnit će se što je strojno učenje. Nužno je razumijevanje osnovnih pojmova vezanih uz strojno učenje kako bi se ostatak rada mogao pratiti. Također će se objasniti što je duboko učenje, kako definiramo duboko učenje te koje su bitne značajke dubokog učenja. Nužno je pojasniti i razne arhitekture umjetnih neuronskih mreža koje se koriste u svrhe segmentacije slika. Prije nego što se može objasniti problem nužno je u potpunosti razumijeti teoriju. Poslije će se pojasniti problem unutar medicine koji se pokušava riješiti upotrebom strojnog učenja. Velik dio rješavanja tog problema pomoću umjetnih neuronskih mreža podrazumijeva i pravilnu pripremu podataka; bez adekvatno pripremljenih podataka umjetna neuronska mreža neće moći obaviti zadatak segmentacije. Zbog toga je nužno ući u problematiku korištenja medicinskih podataka, s obzirom to da se ne nalaze u obliku u kojem se standardno nalaze podaci koji se upotrebljavaju za treniranje umjetnih neuronskih mreža za segmentaciju slika. Za kraj će se pojasniti mreža upotrebljena u ovome radu; način učitavanja podataka te sami parametri upotrebljene mreže. Objasniti će se upotrebljene funkcije i raspored slojeva s kojima se postižu rezultati navedeni u radu.

2. STROJNO UČENJE

U ovom poglavlju definirat će se osnove strojnog učenja nužne za razumijevanje ostatka ovoga rada.

2.1. Definicija strojnog učenja

Strojno učenje može se definirati na dva načina [1]. Ono se definira kao:

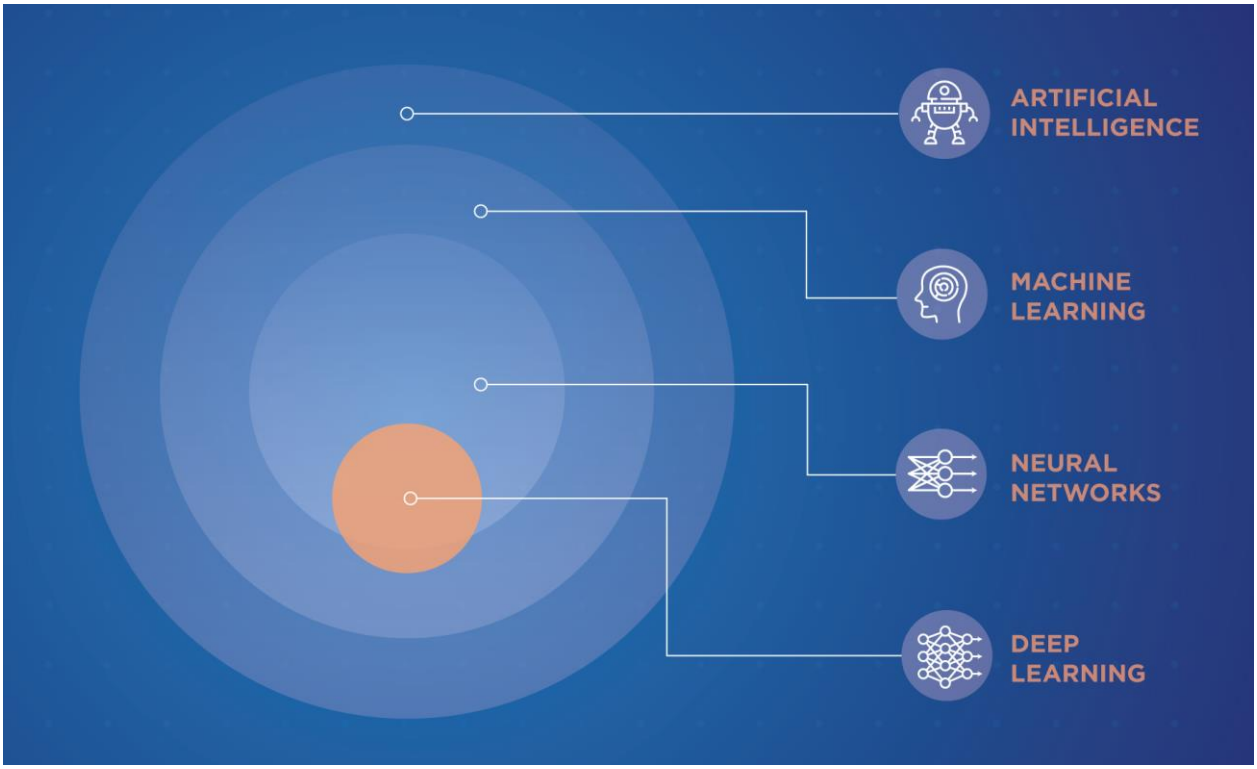
- potpodručje kompjuterskih znanosti vezanih uz izradu algoritama koji zahtijevaju niz primjera kako bi mogli naučiti rješavati neki problem

- proces rješavanja kompleksnog praktičnog problema izradom skupa ulaznih podataka na temelju kojih će se definirati matematički model koji je sposoban riješiti zadani problem

Iako se prethodne dvije formalno navode kao definicije strojnog učenja u odabranoj literaturi, gotovo svaki izvor ima neki određeni način na koji definira strojno učenje; neki izvori čak stavljaju veliki naglasak na definiranje onoga što strojno učenje nije. Pogotovo se naglašava kako strojno učenje ne predstavlja samo novu verziju statističkih modela [2]. Neovisno o tome na koji način ga definiramo strojno učenje će se svoditi na izlučivanje niza korisnih značajki nekog skupa podataka koje se mogu pružiti algoritmu kako bi se riješio neki određeni problem [3]. Na strojno učenje može se čak gledati i kao na neki oblik umjetne inteligencije (eng. artificial intelligence).

Dok se već duži period vremena relativno jednostavno rješavaju problemi koji su jasno definirani nekakvim matematičkim pravilima strojno učenje, pogotovo duboko učenje i umjetne neuronske mreže (eng. neural networks) koje će se spomenuti kasnije, pokušava riješiti niz problema koji su ljudima zapravo jednostavni za obaviti, ali teški za definirati [3].

To su zadaci koje ljudi obavljaju "automatski", bez razmišljanja te čije se rješavanje nama čini intuitivnim, kao npr. prepoznavanje lica ljudi na slikama [3].



Slika 1. Skupine unutar umjetne inteligencije [4]

2.1.2. Tipovi strojnog učenja

Strojno učenje se može podijeliti na [1]:

- učenje s nadzorom (eng. supervised learning)
- učenje s djelomičnim nadzorom (eng. semi-supervised learning)
- učenje bez nadzora (eng. unsupervised learning)
- učenje s pojačavanjem (eng. reinforcement learning)

Duboko učenje koje je nama bitno za ovaj rad ovisno o svojoj upotrebi može se iskoristiti kao učenje s nadzorom, učenje s djelomičnim nadzorom i kao učenje bez nadzora. Učenje s pojačavanjem bavi se rješavanjem drugih tipova problema (sekvencijalni problemi).

2.1.2.1. Učenje s nadzorom

U učenju s nadzorom set podataka koji se koristi sastoji se od niza označenih primjera (tzv. etiketirani primjeri) [1]. Ti etiketirani primjeri matematički se zapisuju kao :

$$\{(x_i, y_i)\}_{i=1}^{N} \quad \textbf{(1)}$$

x_i = element s nekom određenom relevantnom karakteristikom (eng. feature vector)

y_i = oznaka kojom se etiketira neki određeni element (eng.label)

Elementi x_i sadrže neke karakteristike koje su bitne za rješavanje određenog zadatka.Te karakteristike opisuju ono što mi promatramo [1]; ako npr. s x označimo neki stol kao predmet promatranja, onda će niz značajki x_i opisivati karakteristike toga stola kao što su recimo visina, duljina itd. Oznake za elemente y_i mogu se pojavljivati u više različitih oblika. One mogu biti članovi nekog skupa, realni brojevi pa čak i strukture poput vektora i matrica [1]. Ako nije drugačije navedeno za oznake se uzima da su element nekog skupa ili realni broj [1]. Cilj algoritma kod učenja s nadzorom je da se napravi model koji kao ulazne vrijednosti prima elemente s relevantnim karakteristikama x_i, a na svom izlazu izbacuje informacije nužne za dodjeljivanje oznake tom elementu [1].

2.1.2.2. Učenje s djelomičnim nadzorom

Kod ovakvog tipa učenja skup podataka koji se koristi za učenje sadrži i označene i neoznačene elemente; elemenata bez oznaka je redovito mnogo više od onih koji imaju svoju oznaku [1]. Činjenica da postoje elementi koji nisu označeni ne mijenja cilj kod izrade modela. Smatra se da će neoznačeni elementi pomoći pri učenju, a ne smetati algoritmu kod učenja.

To je zbog toga što algoritmu dajemo veći skup podataka iz kojeg može učiti: pretpostavka je da će algoritam to povećanje količine podataka iskoristiti kako bi bolje naučio rješavati problem [1].

2.1.2.3. Učenje bez nadzora

Učenje bez nadzora podrazumijeva učenje pomoću skupa neoznačenih elemenata [1]. Jednadžba koja se koristi je različita od jednadžbe 1.1. Razlog tome je što ne postoje oznake pa jednadžba poprima sljedeći oblik:

$$\{(x_i)\}_{i=1}^{N} \tag{2}$$

Kod ovakvog tipa učenja stvara se model koji uzima neoznačene elemente u obliku vektora kao ulazne vrijednosti, pa ih pretvara u druge vektore ili neku drugu vrijednost koja služi za rješavanje određenog zadatka [1]. Primjer ovoga je grupiranje (eng. clustering).

2.1.2.4. Učenje s pojačavanjem

Ovaj tip učenja se u potpunosti razlikuje od prije navedenih tipova učenja. Klasificira se u potpuno zasebno potpodručje kod strojnog učenja jer podrazumijeva skroz drukčiji pristup rješavanju problema. Stroj "živi" u nekoj okolini te definira stanje te okoline preko skupine vektora [1]. Obavljanjem zadataka on prelazi iz jednog stanja okoline u drugo stanje okoline. Zadatak ovakvog tipa učenja je da se stroj usmjeri prema rješavanju problema tako što ga se "nagrađuje" za dobra rješenja i prelazak u poželjno stanje okoline [1]. Pomoću ulaznih podataka bira se ono rješenje problema koje će dovesti do najbolje moguće "nagrade". Ono što razlikuje ovaj tip učenja je činjenica da donošenje odluke nije neovisno o odlukama donesenim u prošlosti; cilj je razviti mogućnost rješavanja sekvencijalnih problema [1]. Primjeri takvih zadataka su razni zadaci u logistici, umjetna inteligencija kod robota, umjetna inteligencija koja igra neku igru poput šaha itd.

2.2. Definicija algoritma učenja

Kod svakog algoritma učenja mogu se definirati tri stvari [3]:

- Zadatak algoritma
- Mjera uspjeha rada algoritma
- "Iskustvo" algoritma

2.2.1. Zadatak algoritma

Strojno učenje omogućava rješavanje zadataka i problema koji se ne mogu riješiti pomoću standardnih programa [3]. Osim toga, strojno učenje može se definirati kao uvod u inteligentno ponašanje te samim time kao i uvod u umjetnu inteligenciju [3]. Neki od tipičnih zadataka koji se rješavaju strojnim učenjem su [3]:

- Klasifikacija (eng. Classification)
- Klasifikacija s nedostajućim ulaznim podacima (eng. Classification with missing inputs)
- Regresija (eng. Regression)
- Transkripcija (eng. Transcription)
- Procjena gustoće (eng. Density estimation)
- Detekcija devijacija (eng. Anomaly detection)
- Sinteza i uzorkovanje (eng. Synthesis and sampling)

2.2.2. Mjera uspjeha rada algoritma

Nužno je moći kvantitativno definirati uspjeh rada nekog algoritma; u suprotnom nam je nepoznato da li taj algoritam obavlja svoj zadatak na odgovarajući način [3]. Uspjeh rada nekog algoritma se veže direktno na zadatak samog algoritma [3]. Tako se za zadatke kao što je klasifikacija kao mjera uspjeha definira točnost predviđanja korištenog algoritma, dok se za neke zadatke kao što je procjena gustoće uspjeh definira preko vjerojatnosti koju algoritam dodijeli nekom uzorku [3]. Najbitnije je paziti da se uspjeh rada algoritma procjenjuje preko testnog skupa podataka: podaci koji se koriste za procjenu uspjeha rada nekog algoritma ne smiju biti unutar skupa podataka pomoću kojih se algoritam trenirao [3].

Sa obzirom da se uspjeh rada algoritma definira drugačije za različite zadatke, on se općenito definira preko gubitka. Gubitak mijenja svoj oblik od zadatka do zadatka međutim uvijek je cilj nekog algoritma postići što manji gubitak [3]. Problem koji se javlja je određivanje gubitka:kako definirati gubitak za neki određeni zadatak. Iako na prvi pogled djeluje kao jednostavan zadatak, ako se gubitak za neki zadatak odredi nepravilno algoritam neće postići ono što se od njega traži [3]. Optimizacija algoritma definira se kao promjena parametara koja dovodi do manjeg gubitka, ali za to je potrebno imati gradijent ili nešto slično kako bismo znali u kojem smjeru treba pomaknuti parametre da bi se poboljšali rezultati rada algoritma. Bez toga nemoguće je riješiti zadatak na zadovoljavajućoj razini [3].

2.2.3. "Iskustvo" algoritma

Iako se u literaturi često definira kao "iskustvo" modela, ovaj dio zapravo podrazumijeva način na koji se algoritam veže sa skupom podataka preko kojeg bi ga se trebalo optimizirati [3]. U najjednostavnijem obliku to je obična veza između algoritma i nekog skupa podataka sa primjerima koji se koriste za optimizaciju algoritma, ali javljaju se i kompleksniji primjeri (traženje dodatnih primjera u slučaju aktivnog učenja i konstantna interakcija s okolinom kod učenja sa pojačavanjem). Skup podataka koji se koristi može se definirati na više načina. S obzirom na to da postoji više različitih zadataka koji se zadaju algoritmom najjednostavnije ga je definirati kao niz primjera: svaki primjer sadrži informacije koje algoritam može iskoristiti za učenje [3].

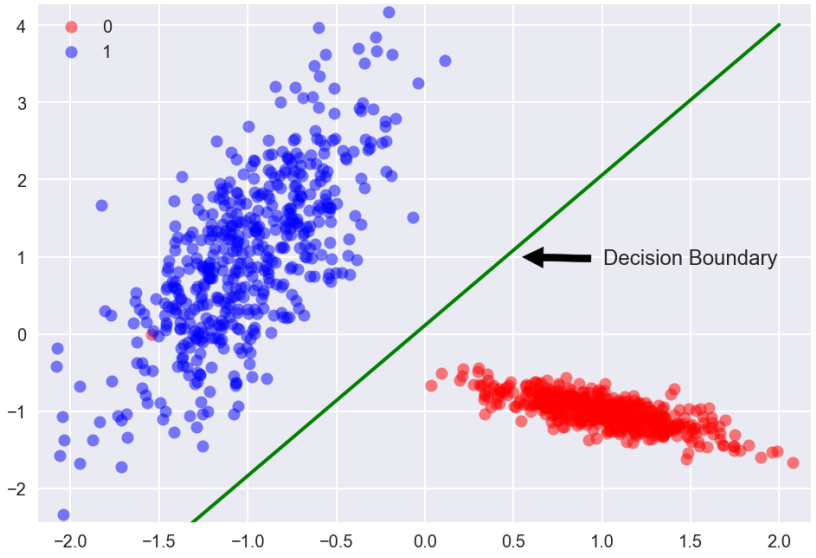
Te informacije javljaju se u obliku određenih značajki direktno vezanih s oznakama; iako oblik oznake varira od zadatka do zadatka ona uvijek doprinosi učenju algoritma tako što definira značajku na određeni za optimizaciju relevantan način[3].

2.4. Mogućnost generalizacije modela strojnog učenja

Nužno je da modeli funkcioniraju na novim i potpuno dotad neviđenim podacima. Kako bi objasnili zašto modeli funkcioniraju na potpuno novim i dotad neviđenim podacima koristit ćemo primjer kod učenja s nadzorom (zbog toga što je to tip učenja korišten u ovom radu). Prije izrade samog modela nužno je prikupiti sve potrebne podatke. Bez kvalitetnih skupova podataka nemoguće je izraditi model koji će obavljati željeni zadatak. U slučaju učenja s nadzorom to podrazumijeva skupljanje parova ulaznih i izlaznih podataka [1].

Osoba koja izrađuje model mora na temelju svog iskustva izabrati kako će elemente bitne za obavljanje zadatka iz stvarnog svijeta prikazati u obliku koji se može koristiti kod strojnog učenja [1]. U većini slučajeva, s obzirom na to da se radi s računalima, te karakteristike je obično nužno predstaviti u obliku brojeva.

Za primjer se može uzeti binarna klasifikacija. Ako članovima jednog skupa dodijelimo jednu klasu (recimo 0), a članovima drugog skupa dodijelimo drugu klasu (recimo 1) treniranjem modela stvorit će se tzv. granica odluke (eng. decision boundary) [1]. Ta granica odluke može poprimati više različitih oblika, ali radi jednostavnosti pretpostavimo u ovom trenutku da je oblika pravca. Takva granica će onda podijeliti članove dvaju skupova u dva potprostora. Uz pretpostavku da smo imali veliki broj primjera za učenje, velika je vjerojatnost da će se član koji spada u skup 0 nalaziti blizu nekog drugog člana iz skupa 0 te da će zbog toga biti smješten u odgovarajući podprostor. Zato je bitno imati što više podataka kod treniranja modela. Primjer gore navedenog može se vidjeti na slici 2.

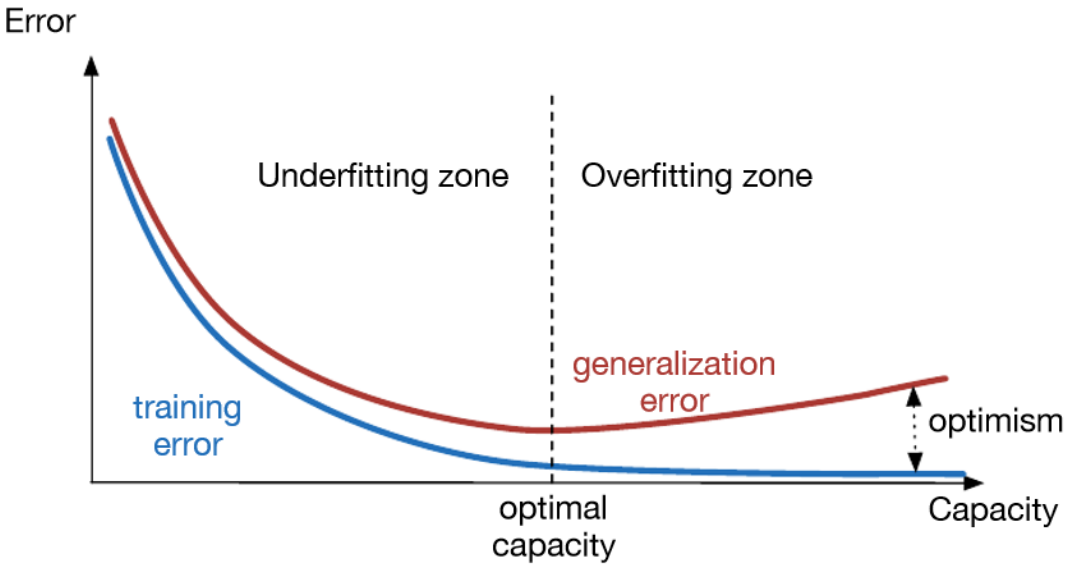


Slika 2. Klasifikacija i granica odluke [5]

Na slici 2. može se vidjeti koliko različito grupirani mogu biti uzorci odnosno elementi iz skupova. Usporedbom dva potprostora može se vidjeti da postoji mnogo veća varijacija kod uzoraka iz kategorije 1 nego iz kategorije 0; za generalizaciju je uvijek korisnije imati što veću količinu uzoraka te da ti uzorci unutar svoje kategorije variraju kolikog god mogu.

2.5. Kapacitet modela strojnog učenja

Kapacitet nekog modela podrazumijeva sposobnost modela da izdvoji iz skupa funkcija onu funkciju koja će biti optimalna za rješavanje određenog zadatka [3]. Ako se funkcija izdvaja iz skupa složenih funkcija bit će moguće iskoristiti znatno kompleksniju funkciju da definiranje nekog modela nego ako se koristi npr. linearna funkcija. To ne znači da je uvijek poželjnije koristiti što kompleksniju funkciju za definiranje modela [3]. Najbitnije je da funkcija dobro opisuje traženi problem odnosno da se ne javljaju problemi pretreniranja (eng. overfitting) i podtreniranja (eng. underfitting).



Slika 3. Pretreniranje i podtreniranje [3]

2.5.1. Pretreniranje

Pretreniranje je pojava koja se definira kao nesposobnost modela za generalizaciju [3]. Na prvi pogled čini se kao najjednostavnije rješenje uvijek uzeti za izbor skup funkcija u kojem se nalaze jako kompleksne funkcije, međutim kod takvog pristupa javlja se problem pretreniranja.

Ako se koristi skup funkcija, u kojem se nalaze jako kompleksne funkcije, za definiranje modela koji bi se mogao definirati i funkcijom koja se nalazi u skupu funkcija koji sadrži jednostavnije funkcije može se dogoditi da više funkcija iz kompleksnog skupa funkcija dobro odgovaraju modelu. U takvoj situaciji izbor prave funkcije svodi se na slučajnost: iako ih više odgovara kada je u pitanju skup podataka za treniranje samo je jedna optimalna za primjenu na nepoznatim podacima [3].

Hoće li se odabrati prava funkcija onda se svodi na čistu slučajnost. Zbog toga se uvijek kod izbora skupa funkcija okreće prema pravilu Occamove britve (eng. Occam's razor).

Treba izabrati skup funkcija koji je dovoljno velik da se u njemu nalazi samo jedna funkcija koja odgovara modelu pomoću kojega se treba riješiti zadatak [3]. Pojednostavljeno rečeno, ako je model prekompleksan za skup podataka na kojemu se pokušava trenirati on će zapamtiti skup podataka za treniranje, te zbog toga dati varljive rezultate: gubitak će biti veoma malen na skupu podataka za učenje, ali velik na skupu podataka za testiranje [6].

Problemu pretreniranja se može pristupiti, te ga pokušati riješiti na niz načina [6]:

- Pojednostavljenje modela
- Povećavanje skupa podataka za treniranje
- Metode regularizacije
- Rano zaustavljanje (kod iterativnog treniranja)

Pretreniranju je najlakše pristupiti u skladu s tim kako ga i prepoznajemo; budući da se kod pretreniranja javlja dobra točnost kod skupa podataka za treniranje, a velik gubitak kod skupa podataka za testiranje, kao potencijalno rješenje dodaje se skup podataka za validaciju kojim se optimalni kapacitet može promijeniti na empirijski način (gdje taj skup podataka predstavlja podatke koji se ne nalaze ni u skupu podataka za treniranje ni u skupu podataka za testiranje, te se koriste kao poseban skup podataka kojim se simulira testiranje modela za vrijeme treniranja) [3].

2.5.2. Podtreniranje

Podtreniranje je jednostavniji problem od pretreniranja: kod podtreniranja model nije dovoljno kompleksan da pronađe rješenje [3]. Jako ga je lako uočiti zbog toga što daje veliki gubitak i na skupu podataka za treniranje i na skupu podataka za testiranje [7]. Podtreniranje se manifestira na dva načina.

Ta dva načina su [3]:

- Skup funkcija iz kojeg se pokušava izdvojiti funkcija koja dobro odgovara modelu je previše ograničen odnosno ne dozvoljava potrebnu kompleksnost modela

- Iako je skup funkcija dovoljno velik i sadrži funkcije koje bi dobro odgovarale modelu, sam postupak optimizacije ne uspijeva izdvojiti traženu funkciju

Pojednostavljeno se problem može tretirati kao problem suprotan pretreniranju pa se tako pristupa i pokušavanju rješavanja tog problema (npr. može se pokušati povećati kompleksnost modela) [6].

3. DUBOKO UČENJE

Duboko učenje je podvrsta strojnog učenja, u kojemu se u svrhe rješavanja problema koriste umjetne neuronske mreže. Umjetne neuronske mreže spadaju u najranije i najuspješnije algoritme strojnog učenja [3]. Kada govorimo o dubokom učenju fokus će biti na učenju s nadzorom [3].

3.1. Osnove umjetnih neuronskih mreža

Umjetne neuronske mreže su se mogu definirati kao niz algoritama namijenjenih za prepoznavanje nekog uzorka [8]. Njihova građa zasniva se na građi ljudskog mozga. Koriste se umjetni neuroni koji su dizajnirani i pokušavaju imitirati rad bioloških neurona [9].

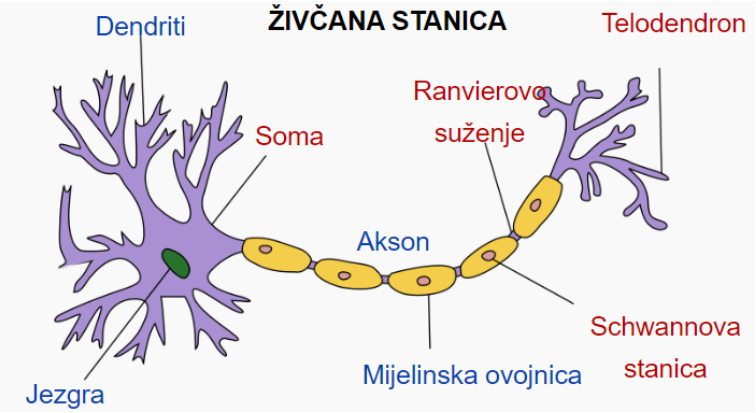
3.1.1. Biološki i umjetni neuroni

Elementi procesiranja umjetne neuronske mreže su umjetni neuroni koji svojim oblikom imitiraju izgled bioloških neurona. Kako bi pravilno razumjeli građu umjetnih neurona, treba poznavati i osnove građe bioloških neurona.

3.1.1.1. Biološki neuron

Poznavanje strukture bioloških neurona (živčanih stanica) nužno je kako bi se pravilno razumjelo umjetne neurone. Biološki neuroni sastoje se od nekoliko dijelova [9]:

- Dendriti
- Tijelo neurona
- Aksoni



Slika 4. Građa živčane stanice

Dendriti su kraći produžeci koji s osjetnih organa ili drugih živčanih stanica dovode živčano uzbuđenje na tijelo stanice [10]. Na završecima se nalazi dendritska spina ili trn; tako se postiže najbolji mogući kontakt s drugim neuronima [10]. Dendriti primaju ulazne podatke [10].

Tijelo neurona (poznato i kao soma) sastoji se jezgre stanice, od kromosoma i od jezgrice [10]. Soma je zadužena za obradu ulaznih podataka.

Aksoni su duži produžeci neurona koji prenose živčane impulse s tijela stanice do drugih živčanih stanica ili do izvršnih organa [10]. Preko njih do živčane stanice dolaze ulazni podaci. Obavijeni su mijelinskom ovojnicom[10]. Na mijelinskoj ovojnici postoje "rupe"; područja koja omogućavaju bržu kondukciju akcijskog potencijala. Ta područja nazivaju se Ranvierovim suženjima tj. čvorovima [10]. Između dva takva čvora nalaze se Schwannove stanice. Neuroprijenosnici se kod aksona nalaze unutar telodendrona koji se nalaze na samom kraju aksona (neuroprijenosnici se nalaze unutar malih vrećica na kvržicama teledendrona) [10].

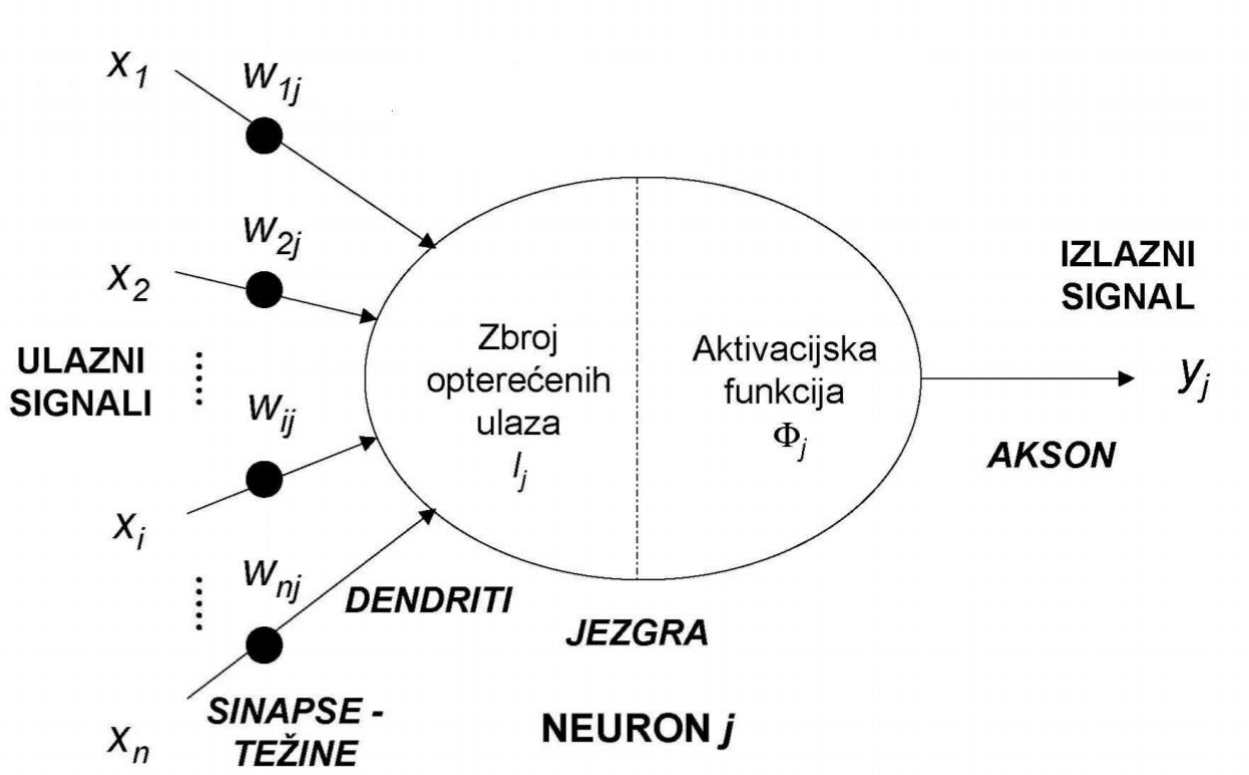
Sinapse su mjesta komunikacije između dva neurona koja se nalaze na krajevima aksona i sastoje se od primajućeg dijela (dendrit) i dijela koji odašilje signal (aksonski završetak). Sinapse su od ključne važnosti za rad neurona.

Rad bioloških neurona svodi se na polarizaciju i depolarizaciju. Taj kompletni proces naziva se akcijskim potencijalom i funkcionira prema zakonu "sve ili ništa" [10]. Zakon "sve ili ništa" definira ponašanje neurona pri interakciji stimulusa s neuronom. Svi neuroni imaju određeni prag podražljivosti. Kada se na neki neuron djeluje sa nekim određenim podražajem kod njega će se otvoriti pore za propust iona. Za koliko će se kanali otvoriti ovisi o odnosu između jačine podražaja i praga podražljivosti neurona [10]. Ako je podražaj dovoljno jak da se pređe prag podražljivosti, kanali se otvore u potpunosti i dolazi do depolarizacije neurona [10]. Nakon te depolarizacije ponovno će doći do polarizacije neurona i taj ciklus se ponavlja ovisno o dolasku podražaja na neki određeni neuron. Podražaj uvijek do neurona dolazi preko membranske ovojnice, koja se ovija oko cijelog neurona te je polupropusna [10]. Akcijski potencijal se ne događa po cijelom neuronu odjedanput, već samo na dijelu membrane. Nakon nastanka akcijski potencijal će putovati preko membrane; to putovanje akcijskog potencijala preko aksona i dendrita je osnova prijenosa informacija u mozgu [10].

3.1.1.2. Umjetni neuron

Budući da građa umjetnih neurona imitira građu bioloških, i način na koji se aktivira i radi sam neuron će također biti baziran na temelju rada bioloških neurona. Funkcionalnost biološkog neurona imitira McCulloch-Pitts model umjetnog neurona [11], koji se još naziva i Threshold Logic Unit (TLU). Analogija koja se koristi je [11]: signali se mogu opisati nekim numeričkim iznosom koji će se na ulazu u neuron množiti s nekakvim težinskim faktorom koji opisuje jakost sinapse. Ti signali pomnoženi težinskim faktorima se potom sumiraju sukladno tome kako se u tijelu bioloških neurona sumiraju potencijali, pa se konačni iznos uspoređuje s pragom osjetljivosti (podražljivosti). Ako je vrijednost veća od tog praga umjetni neuron će dati izlazni signal.

Osim standardne funkcije praga kod umjetnih neurona mogu se koristiti i određene prijenosne funkcije; uobičajeno je u današnje vrijeme koristiti neku vrstu aktivacijske funkcije.



Slika 5. Model umjetnog neurona

3.1.2. Građa umjetne neuronske mreže

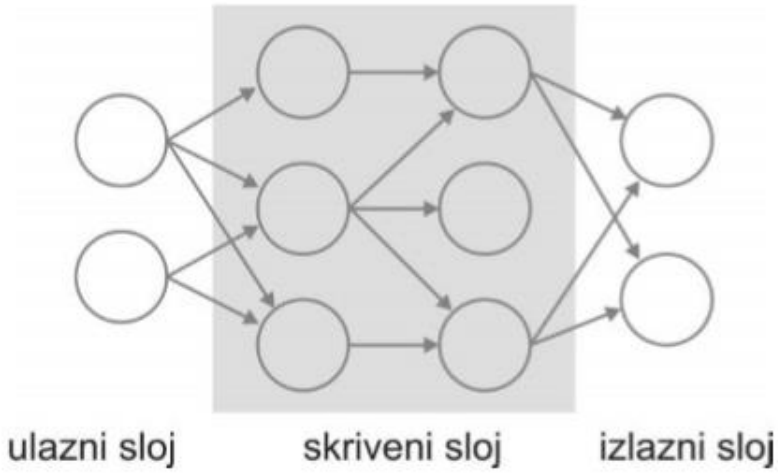
Za umjetne neuronske mreže se može reći da su paralelno distribuirane informacijske strukture koje se sastoje od elemenata procesiranja međusobno povezanih u slojeve [9]. Kod usporedbe ljudskog mozga s radom računala može se primijetiti jedna velika razlika: dok ljudski mozak procesiranje obavlja paralelno, računalo to radi sekvencijalno. Rad ljudskog živčanog sustava omogućuje čovjeku da daje bolje zaključke i aproksimacije i kod upotrebe djelomično nekompletnih i nekorektnih ulaznih podataka [9]. Slojevi koji sačinjavaju umjetnu neuronsku mrežu mogu se na osnovnoj razini podijeliti na ulazni sloj, skrivene slojeve i izlazni sloj [9]. Sama mreža ne mora nužno imati skrivene slojeve unutar svoje strukture; takve mreže sastoje se samo od ulaznog i izlaznog sloja. Kapacitet neke mreže bit će povezan s brojem slojeva te mreže i brojem neurona unutar pojedinih slojeva; mreža s manje slojeva može imati veći kapacitet od mreže s više slojeva ovisno o kombinaciji broja neurona i broja slojeva.

Osnovni tipovi arhitekture umjetnih neuronskih mreža su [11]:

- aciklička (eng. feedforward net) mreža
- mreža s povratnom vezom (eng. reccurent net)
- lateralno povezana mreža (rešetkasta)
- hibridne mreže

3.1.2.1. Aciklička mreža

Acikličke umjetne neuronske mreže su najjednostavniji tip neuronski mreža. Neuroni ovakve mreže ne sadrže povratne veze. Signali koji krenu od ulaznih neurona nakon određenog vremena stižu do izlaza mreže; događa se jednosmjerna propagacija signala [11]. Neuroni ulaznog sloja nemaju ulaznih signala tj. nemaju funkcionalnost neurona [11]. Pod ulazni sloj onda podrazumijevamo podatke organizirane u vektor konkretnih vrijednosti.



Slika 6. Aciklička arhitektura umjetne neuronske mreže [11]

Najjednostavniji oblik koji se može pojaviti je jednoslojni perceptron (eng. Single Layer Perceptron). Takav jednoslojni perceptron može poslužiti za klasifikaciju uzoraka koji su linearno separabilni [12]. Sastoji se od jednog neurona. Taj algoritam učenja, u slučaju linearne separabilnosti uzoraka, konvergira i postavlja ravninu koja će točno odijeliti dvije klase [12]. To znači da je linearna separabilnost nužan preduvjet za točnu klasifikaciju; naziva se još i teoremom linearne konvergencije [12].

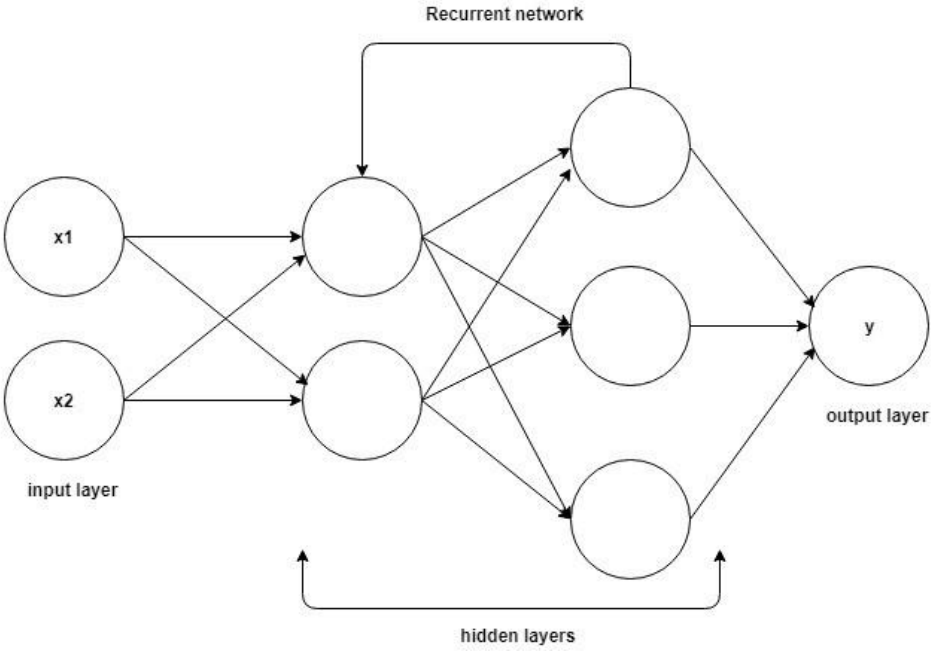
Kompliciraniji oblik je višeslojni perceptron (eng. Multilayer Perceptron) . Takve mreže imaju jedan ulazni sloj, jedan izlazni sloj i jedan ili više skrivenih slojeva [13]. Ovakve mreže se mogu koristiti za rješavanje raznih problema. Učenje se onda odvija pod nadzorom pomoću algoritma s povratnom propagacijom pogreške (eng. back-propagation algorithm). Višeslojni perceptron ima nelinearnu izlaznu karakteristiku koja je glatka (bitna razlika u usporedbi s Rosenblattovim perceptronom)[13]. Tako da , iako se na prvu čini da nema mana u usporedbi s jednoslojnim perceptronom, zapravo postoje određeni problemi koji se mogu primijetiti [13]:

- višestruke nelinearnosti i visoka povezanost otežava teoretsku analizu ovakve mreže
- korištenje skrivenih neurona otežava vizualizaciju procesa učenja
- proces učenja je teži zbog skrivenih neurona; mreža sama odlučuje što trebaju naučiti skriveni neuroni

3.1.2.2. Mreža s povratnom vezom

Ovakve mreže u svojoj strukturi sadrže minimalno jednu povratnu vezu. Mora postojati barem jedan tzv. čvor koji je takav da, ako pratimo njegov izlaz kroz sve moguće puteve, nakon konačnog broja koraka ponovno ćemo obići taj isti čvor [13]. Kod ovakvih mreža ne može se govoriti o ulaznom i izlaznom sloju, već se prije govori o vidljivim i skrivenim čvorovima [13]. Primjer ovakve mreže je Hopfieldova mreža. U osnovi mreže s povratnom vezom su sličnije funkcioniranju živih bića nego standardne acikličke neuralne mreže.

Za razliku od acikličkih mreža, koje ne posjeduju "pamćenje", mreže s povratnom vezom petljama simuliraju postojanje "pamćenja" i omogućavaju sadržavanje informacija unutar mreže [14].



Slika 7. Mreža s povratnom vezom [15]

Ovakve mreže se koriste u različite svrhe, od kojih su neke [16]:

- Prepoznavanje rukopisa
- Prevođenje jezika
- Komponiranje glazbe
- Prepoznavanje govora
- Upravljanje robotima

3.1.2.3. Lateralno povezane mreže i hibridne mreže

Ove tipove mreža treba spomenuti, međutim budući da se ne upotrebljavaju za probleme kojima se bavi ovaj rad neće biti objašnjavane u detalje. Za potrebe ovog rada nije bitno razumjeti kako one funkcioniraju.

3.2. Aktivacijske funkcije

Aktivacijske funkcije koriste se kod umjetnih neurona kako bi se omogućilo pravilno funkcioniranje višeslojne umjetne neuronske mreže. Kada se ne bi koristila aktivacijska funkcija sama umjetna neuronska mreža uvijek bi se mogla svesti na jedan perceptron; zbrajanjem više linearnih funkcija dobije se ponovno linearna funkcija. Praćenjem otežanih ulaza u neki neuron može se vidjeti da li oni prelaze neki prag osjetljivosti ili ne; ako prelaze preko nelinearne aktivacijske funkcije generirat će se izlazni signal umjetnog neurona. Koristi se niz različitih aktivacijskih funkcija koje se biraju ovisno o tome što je zadatak umjetne neuronske mreže [3]. Neke češće aktivacijske funkcije su [3]:

- Sigmoidalna funkcija
- Linearna rektifikacijska funkcija
- Softmax funkcija
- Hiperbolička tangentna funkcija

Neke manje česte aktivacijske funkcije su [3]:

- RBF funkcija
- Softplus funkcija
- Maxout funkcija

3.3. Funkcija gubitka

Preko funkcije gubitka se definira što želimo naučiti mrežu [3]. Za različite potrebe se upotrebljavaju različite funkcije gubitka. One su u direktnoj svezi s aktivacijskom funkcijom u izlaznom sloju. Jako je bitno kvalitetno izabrati funkciju gubitka kako bismo mogli pratiti da li se rad modela poboljšava [3]. Dva najčešća tipa su srednja kvadratna pogreška i unakrsna entropija [17]. Korištenje pojedinih funkcija gubitka može se podijeliti na sljedeći način [17]:

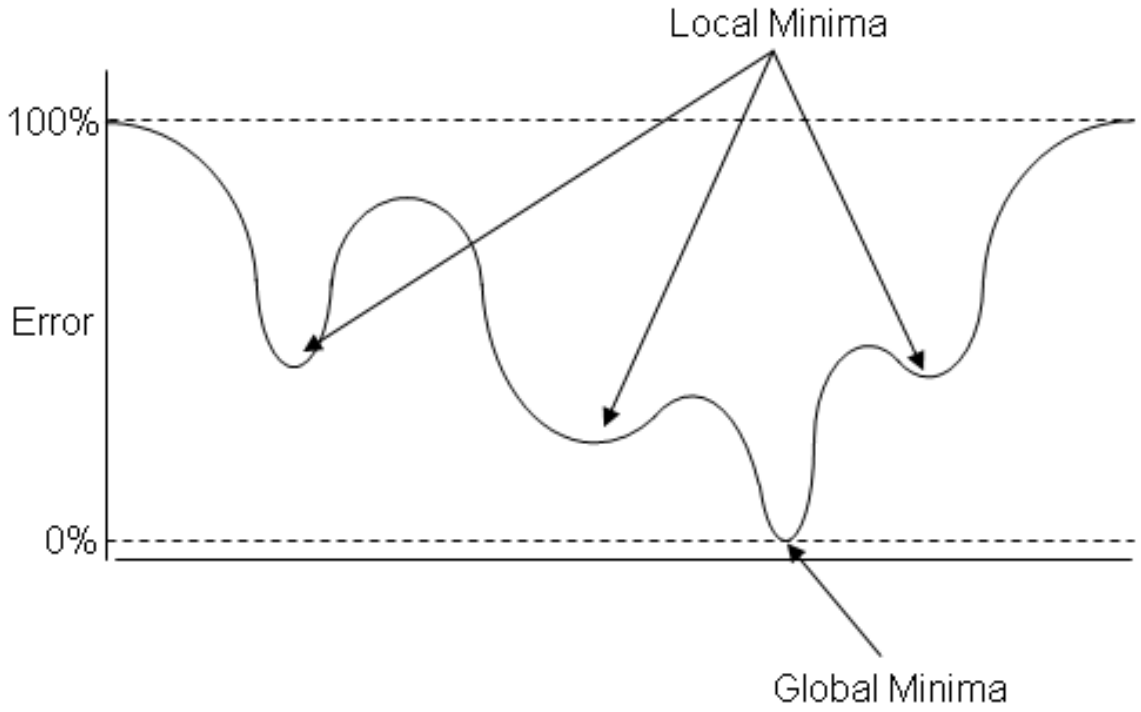
- Regresija - linearna aktivacijska jedinica u izlaznom sloju, koristi se srednja kvadratna pogreška (eng. Mean Squared Error)
- Binarna klasifikacija - sigmoidalna aktivacijska jedinica u izlaznom sloju, koristi se unakrsna entropija (eng. Cross-Entropy) poznata i kao logaritamski gubitak (eng. Logarithmic Loss)
- Klasifikacija u više kategorija - Softmax aktivacijska jedinica u izlaznom sloju, koristi se unakrsna entropija poznata i kao logaritamski gubitak

Srednja kvadratna pogreška prati srednju kvadratnu razliku između pretpostavki koje donosi umjetna neuronska mreža i stvarnog rezultata [17]. Unakrsna entropija funkcionira tako da se svaka donesena pretpostavka modela usporedi sa stvarnom vrijednosti koju mreža pokušava pretpostaviti: na temelju toga se računa određena "kazna" koja je logaritamskog tipa [17]. Kazna će biti manja za manja odstupanja, a za velika odstupanja znatno veća. Idealna unakrsna entropija bi bila ona za koju se postiže vrijednost unakrsne entropije od 0.0 [17].

3.4. Optimizacija

Kod umjetnih neuronskih mreža uvijek se sprovodi neki tip optimizacije; optimizacija umjetnih neuronskih mreža podrazumijeva promjenu vrijednosti x kod funkcije f(x) tako da se postigne minimalna vrijednost te funkcije [3]. Funkcija koju minimiziramo je u slučaju neuronskih mreža funkcija gubitka [3].

To u praksi znači da pokušavamo naći globalni minimum funkcije. Kod dubokog učenja problemi se stvaraju zbog kompleksnosti funkcija: često se trebaju optimizirati funkcije koje imaju više od jednog globalnog minimuma, te funkcije koje unutar sebe sadrže lokalne minimume unutar kojih se algoritam može "zaglaviti" [3]. Zato se u praksi zadovoljavamo s pronalaženjem vrijednosti koja je što bliže moguće globalnom minimumu [3].



Slika 8. Problem lokalnog minimuma [18]

3.4.1. Gradijentni spust

Baza optimizacije u dubokom učenju je gradijentni spust. To je iterativni algoritam koji se koristi za minimiziranje vrijednosti neke određene funkcije [19]. Ako uzmemo neku funkciju $f(w)$ tada možemo reći da za njen gradijent vrijedi sljedeća jednadžba:

$$\nabla f(w) = \left(\frac{\partial f(w)}{\partial w[1]}, \ldots\ldots, \frac{\partial f(w)}{\partial w[d]}\right) \quad \textbf{(3)}$$

Gradijent se onda može definirati kao vektor parcijalnih derivacija od f. Budući da je ovaj algoritam iterativan on će krenuti od neke početne vrijednosti parametra w (recimo 0) i u koracima minimizirati vrijednost funkcije:

$$w^{(t+1)} = w^{(t)} - \eta \nabla \mathrm{f}\left(w^{(t)}\right) \tag{4}$$

Iz gornje jednadžbe se vidi da će nova vrijednost parametra učenja biti trenutna vrijednost parametra učenja na koju se nadodaje gradijent [19]. Bitno je pratiti i stopu učenja η. Gradijentni spust ima određene mane: relativno je spor i nema sigurnu konvergenciju prema globalnom minimumu u slučaju postojanja više lokalnih minimuma. Zbog toga ga je naslijedio stohastički gradijentni spust.

3.4.2. Stohastički gradijentni spust

Stohastički gradijentni spust koristi više manjih gradijenata pogreške umjesto globalnog. U svrhu korištenja to znači da, dok standardni gradijentni spust korigira težine nakon izračuna na svim primjerima, stohastički gradijentni spust aproksimira gradijentni spust inkrementalnom korekcijom težina [19]. To znači da će u praksi SGD biti brži (jer standardni gradijentni spust računa sve težine odjednom). Matematički gledano razlika između stohastičkog i standardnog gradijentog spusta je u tome što se kod stohastičkog gradijentnog spusta ne prilagođavaju parametri tako da se prati gradijent cijele funkcije već se koristi nasumični vektor za kojeg jedino mora vrijediti da će njegova očekivana vrijednost biti subgradijent funkcije na trenutačnom vektoru [19]. Pojednostavljeno to znači da nećemo računati gubitak cijele skupine podataka, već ćemo računati gubitak samo nasumično malog dijela skupine podataka; pretpostavka je da ćemo se praćenjem gradijenta tog malog dijela seta podataka kretati u smjeru minimuma. To često nije točno, čak naprotiv nekad nas takve inkrementalne promjene čak znaju udaljiti od minimuma.

Međutim ako se koriste dovoljno mali pomaci te se ne prolazi kroz cijeli set podataka već kroz mali, nasumični dio seta podataka konvergencija prema minimumu će biti značajno brža [19].

3.4.3. Moderni algoritmi za optimizaciju

U današnje vrijeme postoji niz modernih algoritama koji se koriste u optimizacijske svrhe kod umjetnih neuronskih mreža (tako da se stohastički gradijentni spust i ne koristi pretjerano često) [3]. Neki od tih algoritama su [3]:

- Adam
- AdaGrad
- RMSProp

Od gore nabrojanih najčešće se koristi Adam: za njega se smatra da kombinira dobre strane i AdaGrad i RMSProp algoritma: na njega se može gledati kao na njihovu kombinaciju [3]. Ime je dobio po tome što koristi adaptivnu stopu učenja: ona se proračunava zasebno za svaki parametar. Adam koristi pretpostavljene vrijednosti prvog i drugog momenta gradijenta kako bi prilagodio svaki zasebni težinski faktor unutar umjetne neuronske mreže [3].

3.4.4 Povratna propagacija

Povratna propagacija (eng. Backpropagation) koristi se za podešavanje težinskih faktora unutar umjetne neuronske mreže, pomoću nekakvog algoritma koji se temelji na gradijentu, te na temelju tzv. pravila lanca (eng. chain rule) [3]. Pravilo lanca je formula za izračunavanje derivacije od kompozicije dvaju ili više funkcija [3]. Kod povratne propagacije obavlja se parcijalno deriviranje u smjeru suprotnom od onoga u kojemu mreža na temelju neurona pokušava dati točnu pretpostavku [3].

Povratnom propagacijom se zapravo proračunava smjer najstrmijeg spusta gradijenta koji koriste algoritmi poput stohastičkog gradijentnog spusta; na taj način se obavlja treniranje mreže [3].

3.5. Regularizacija

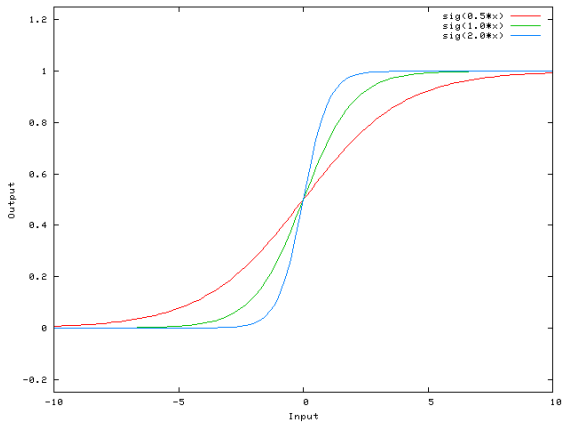
Regularizacija se smatra najboljom strategijom za postizanje dobre sposobnosti generalizacije kod umjetnih neuronskih mreža [3]. Pod regularizaciju smatramo postavljanje određenih "prepreka" učenju umjetne neuronske mreže (postavljanjem posebnih zahtjeva na određene parametre i slično) [3]. Ako se ti dodatni zahtjevi postave kako spada mogu jako pomoći učenju umjetne neuronske mreže te postizanju boljih rezultata na skupu podataka za testiranje [3]. To u praksi znači da, pogotovo u slučaju dubokog učenja, način za postizanje optimalnih rezultata često se svodi na više od samo određivanja optimalnog broja parametara. Zapravo je često bolje imati veći broj parametara koji su na neki način ograničeni [3]. Na osnovnoj razini regularizacija se može opisati kao unošenje dodatnog težinskog faktora koji se zove pomak (eng. bias) kako bi se ograničila varijanca.

3.5.1. Pomak

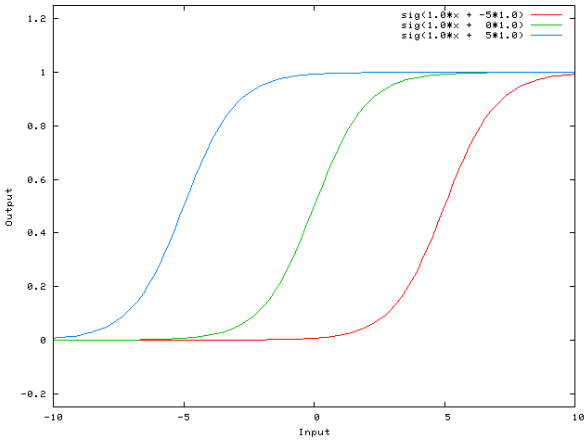
Pomak se koristi kao dodatan način za utjecanje na izlaz iz nekog neurona s obzirom na to da se dodaje na sumu ulaza pomnoženih s težinskim faktorima [3]. Vizualno se utjecaj pomaka može vidjeti kao pomak aktivacijske funkcije unutar koordinatnog sustava. Takvim pomakom može se funkcija podesiti kako bi bolje odgovarala zadatku odnosno bolje odgovarala ulaznim podacima [3]. Pomoću njega se da podesiti granična vrijednost pri kojoj se aktivira neuron [3].

Kao primjer može se uzeti situacija sa sigmoidalnom funkcijom. U takvom slučaju nagib sigmoidalne funkcije može se podesiti podešavanjem težinskih faktora [3]. Ponekad se zadatak ne može riješiti tako jednostavno. U takvim slučajevima potrebno je moći pomaknuti cijelu krivulju u nekom određenom smjeru [3]. Mreža se onda podesiti da za vrijednost $x = 2$ izbacuje kao izlaz nulu.

Takvim podešavanjem se indirektno podešava mreža; omogućava se postizanje raznih kombinacija koje bez pomaka ne bi bile moguće [3].



Slika 9. Podešavanje funkcije bez korištenja pomaka



Slika 10. Podešavanje funkcije uz korištenje pomaka

3.5.2. Varijanca

Varijanca se može definirati kao razlika između greške kod skupa podataka za treniranje i greške kod skupa podataka za validaciju. Kada se razlika između tih grešaka povećava kažemo da se varijanca povećava (potencijalno dolazi do pretreniranja) [3].

3.5.3. Normalizacijske metode kod umjetnih neuronskih mreža

Jako je bitno normalizirati podatke prije nego se iskorištavaju u umjetnim neuronskim mrežama. Ako podaci nisu normalizirani mogu se pojaviti razni problemi. Zbog toga je bitno podatke prebaciti u oblik koji odgovara domeni u kojoj želimo te podatke predstaviti; svi podaci koji se učitavaju u mrežu moraju se nalaziti unutar usporedivog raspona. Normalizacijske metode koje se koriste kod umjetnih neuronskih mreža su [21]:

- Normalizacija hrpa (eng. Batch normalization)
- Normalizacija težinskih faktora (eng. Weight normalization)
- Normalizacija slojeva (eng. Layer normalization)
- Normalizacija primjerka (eng. Instance normalization)
- Normalizacija grupa (eng. Group normalization)
- Renormalizacija hrpa (eng. Batch renormalization)
- Normalizacija hrpa i uzoraka (eng. Batch-Instance normalization)
- Promjenjiva normalizacija (eng. Switchable normalization)
- Spektralna normalizacija (eng. Spectral normalization)

Razne metode koriste se u raznim situacijama; najčešće korištena metoda je normalizacija hrpa iako ona sama po sebi nije optimalna u svim uvjetima [21].

4. PRIMJENA KONVOLUCIJE U DUBOKOM UČENJU

Dosad smo raspravljali o općim oblicima umjetnih neuronskih mreža, međutim sada se treba posvetiti tipu umjetne neuronske mreže koji će biti korišten u ovome radu. Riječ je o konvolucijskim umjetnim neuronskim mrežama. Konvolucijske umjetne neuronske mreže predstavljaju algoritme dubokog učenja koje se koriste u korist raspoznavanja različitih objekata na slikama [21]. One su s vremenom postale najpopularniji tip mreža za upotrebu kod bilo kakvih zadataka koji zahtijevaju klasifikaciju i/ili segmentaciju objekata na slici. Ime su dobile prema matematičkoj operaciji koja se upotrebljava u ovim mrežama: konvoluciji [3]. Ove mreže je najjednostavnije definirati kao umjetne neuronske mreže koje umjesto standardnog načina množenja matrica (koji se obavlja u feed-forward neuronski mrežama) koriste konvoluciju [3]. One će svejedno na svom kraju prije samih krajnjih izlaza sadržavati gusto povezan sloj neurona, poput onih u standardnim feed-forward mrežama [3].

4.1. Konvolucija

Svaka slika može se definirati kao matrica vrijednosti piksela [21]. Zadatak konvolucije kod konvolucijskih umjetnih neuronskih mreža je da svedu količinu podataka koje mreža treba obraditi na što je manje moguću, a da se pritom ne izgube podaci koji su mreži nužni kako bi vršila svoju funkciju (klasifikaciju, segmentaciju itd.) [21]. Konvolucija se matematički gledano može definirati kao operacija dvije funkcije na nekom argumentu [3]. Preko dvije funkcije dobije se treća funkcija koja pokazuje kako oblike prve funkcije ovisi o obliku druge funkcije. Zapis jedne konvolucije izgleda na sljedeći način:

$$s(t) = (x * w)(t) \tag{5}$$

Kao matematička oznaka za konvoluciju koristi se $*$. Konvoluciju se najlakše da objasniti na primjeru. Pretpostavimo da postoji senzor koji definira udaljenost nekog vozila od mjesta mjerenja. S obzirom na to da se položaj vozila može promijeniti u bilo kojem trenutku mi možemo taj položaj u nekom određenom trenutku definirati kao $x(t)$. U stvarnoj situaciji uvijek će postojati nekakav šum kod mjerenja.

Zbog toga ćemo, umjesto da koristimo jednu vrijednost, izmjeriti više puta kako bi dobili srednju vrijednost, te kako bi nam pretpostavka bila što točnija. U obzir se trebaju uzeti trenutci mjerenja. Svako sljedeće mjerenje relevantnije je od onoga prije, pa se zato koriste težinski faktori kako bi se tim kasnijim mjerenjima kod obavljanja proračuna pridodala veća važnost kod konačne procjene pozicije. Težinska funkcija se može zapisati kao $w(a)$, gdje taj a predstavlja starost nekog mjerenja.

Pomoću ove dvije funkcije dobivamo tzv. otežanu funkciju procjene srednje vrijednosti; preko dvije funkcije dobiti će se treća funkcija koja predstavlja pretpostavku o mjestu nalaženja vozila u nekom trenutku sljedećega oblika:

$$s(t) = \int x(a)w(t-a)da \tag{6}$$

Konvolucija se može definirati samo kada se gornji integral može definirati (ne moraju se nužno koristiti vrijeme t i starost procjene a, mogu se uzeti i neki drugi faktori). Konvolucijske umjetne neuronske mreže koriste i određenu terminologiju vezanu uz same konvolucije. Tako se prvi argument naziva ulaznim argumentom (eng. input), a drugi argument naziva se konvolucijskom jezgrom (eng. kernel). Gornja jednadžba, kao i cijeli primjer, predstavljaju idealizaciju. U umjetne neuronske mreže podaci će se uvijek učitavati u nekim malim diskretnim trenutcima u vremenu, pa se onda opći oblik jednadžbe mijenja u tzv. diskretnu konvoluciju:

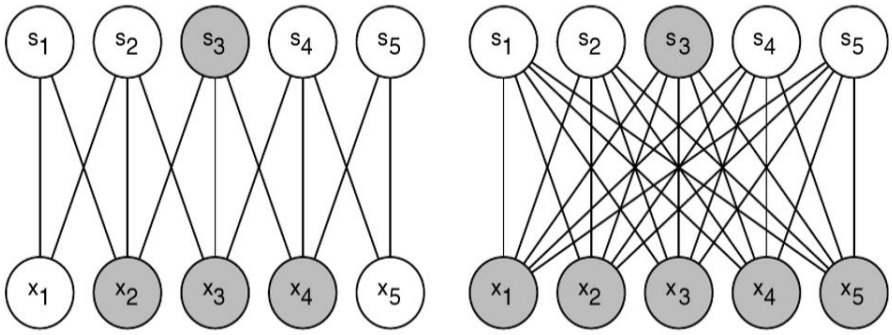
$$s[t] = (x * \mathrm{w})(t) = \sum_{a=-\infty}^{\infty} x[a]w[t-a] \tag{7}$$

U slučaju umjetnih neuronski mreža input će biti podaci koji se učitavaju u mrežu , a kernel će predstavljati skup parametara koje neka mreža može naučiti. Dimenzionalnost kernela ovisit će o dimenzionalnosti podataka: za 2D podatke koristit će se 2D kernel, za 3D podatke 3D kernel itd.

Konvolucija se očituje u upotrebi umjetnih neuronskih mreža na tri načina [3]:

- Rijetke interakcije (eng. sparse interactions)
- Dijeljenje parametara (eng. parameter sharing)
- Ekvivarijantno predstavljanje (eng. equivariant representations)

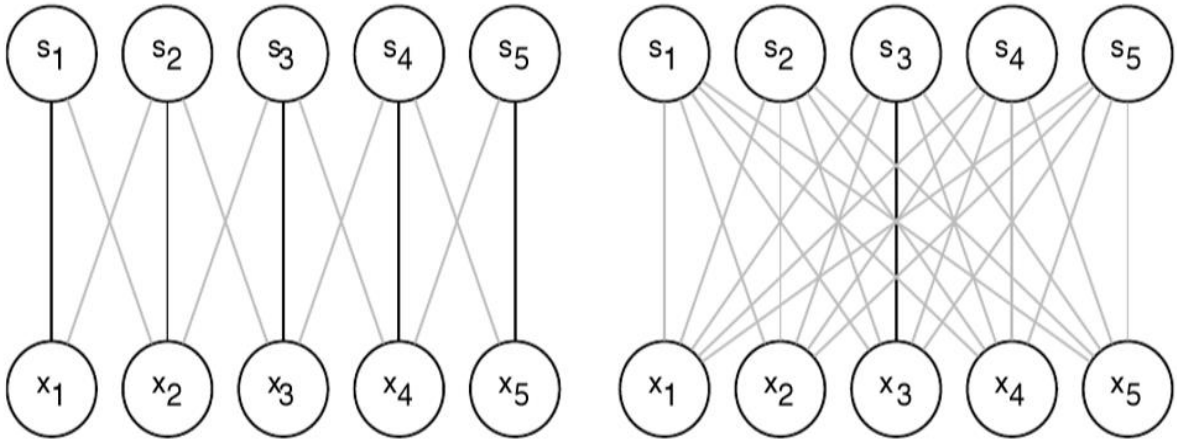
Rijetke interakcije su posljedica korištenja kernela koji je manje veličine od ulaznih podataka. Dok se kod standardnih umjetnih neuronskih mreža povezuju, te postoji interakcija između svake ulazne i svake izlazne vrijednosti, kod konvolucijskih mreža se javlja drugačija situacija. Često su slike puno većih dimenzija nego što je potrebno kako bi se pronašle karakteristike bitne za obavljanje zadatka, pa se slike umanjuju pomoću kernela čime se smanjuje broj podataka koji mreža treba obraditi. Na taj način se može ubrzati rad umjetne neuronske mreže [3].



Slika 11. Rijetke interakcije [3]

Dijeljenje parametara je jednostavan koncept: kod konvolucijskih mreža, za razliku od standardnih mreža, pojavljuje se veza između parametara tako da se vrijednost težinskog faktora primijenjena na neku ulaznu vrijednost nalazi u direktnoj svezi s vrijednosti težinskog faktora primjenjenoj drugdje [3]. Budući da se svaki član kernela upotrebljava na svakoj poziciji ulaznih podataka, ne mora se učiti zaseban set parametara za svaku zasebnu lokaciju već se uči jedan set parametara [3].

Iako dijeljenje parametara ne čini rad bržim, čini mrežu memorijski efikasnijom [3].



Slika 12. Dijeljenje parametara [3]

Ekvivarijantno predstavljanje znači da, zbog toga što postoji dijeljenje parametara, kod konvolucijskih umjetnih neuronskih mreža dolazi do pojave kod slojeva poznate kao ekvivarijantnost [3]. Ekvivarijantnost znači da će promjena ulaznog parametara uzrokovati promjenu izlaznog parametra [3]. Matematički se ekvivarijantnost prikazuje kao:

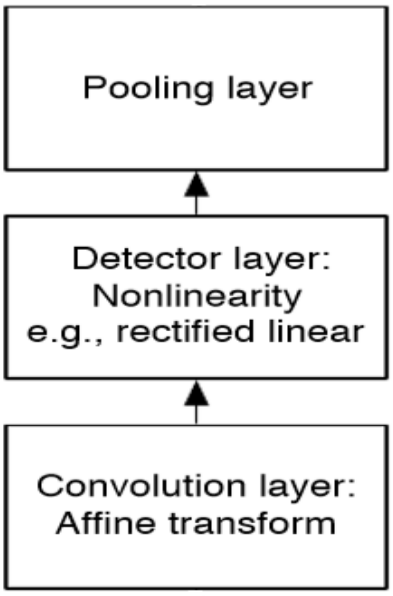
$$f\big(g(x)\big) = g(f(x)) \tag{8}$$

Treba uzeti u obzir da ekvivarijantnost kod konvolucije vrijedi samo u slučaju translacije, ali ne i drugih transformacija (kao recimo rotacije ili umanjivanja elemenata) [3]. Gornju jednadžbu najlakše je objasniti na primjeru. Ako $g(x)$ definiramo sa jednadžbom $g(x)[i] = [i-1]$, te kažemo da je $g(x)$ ekvivarijantan s konvolucijom, dobiti ćemo jednaki rezultat neovisno o redoslijedu obavljanja operacija. Ako prvo obavimo transformaciju za *x* preko jednadžbe pa potom obavimo konvoluciju dobit ćemo isti rezultat kao da smo prvo obavili konvoluciju, a onda obavili transformaciju za *x* [3]. U praksi to znači da, kada promatramo elemente na nekoj slici te uočimo neki element kao rub na slici u prvom sloju, taj rub biti će prepoznat gdje god da je translatiran na slici [3]. To nije uvijek korisno jer se moraju sačuvati i informacije o prostornom rasporedu stvari na slici (npr. donji rub lica neke životinje mreža mora razlikovati od gornjeg ruba lica neke životinje) [3].

Konvolucijski slojevi nisu univerzalno upotrebljivi. Za određene upotrebe umjetnih neuronskih mreža mogu čak biti i štetni kod učenja. Međutim, činjenica da nam omogućavaju da učimo mrežu u slučajevima u kojima standardne mreže pokazuju jako slabe rezultate čini ih gotovo nezamjenjivima u nekim određenim situacijama [3].

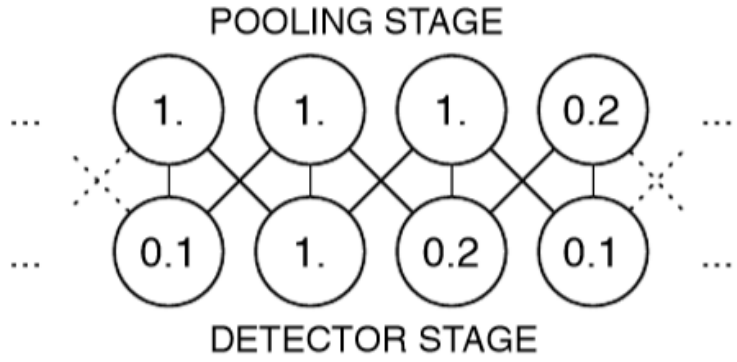
4.2. Sažimanje

Sažimanje (eng. pooling) predstavlja zadnji stadij jednog sloja konvolucijske umjetne neuronske mreže [3]. Prvi stadij je definiran kroz paralelno obavljanje procesa konvolucije kako bi se dobile neke vrijednosti, koje se u drugom stadiju propuštaju kroz određenu aktivacijsku funkciju [3]. U trećem stadiju odvija se sažimanje.



Slika 13. Stadiji konvolucijskog sloja [3]

Sažimanje dodatno modificira izlaznu vrijednost iz nekog sloja [3]. Umjesto praćenja pojedinih vrijednosti, može se prostor promatranja podijeliti na regije, pa se iz svake regije uzima samo najveća vrijednost [3]. Ta najveća vrijednost uzima se kao izlazna vrijednost za sve pozicije unutar te regije [3]. Najpoznatiji tip sažimanja je tzv. max pooling. Max pooling uzima kvadratne regije promatranja, te prati najveću izlaznu vrijednost koja se kod njih pojavljuje. Postoje i drugi tipovi, međutim max pooling se najčešće upotrebljava [3].



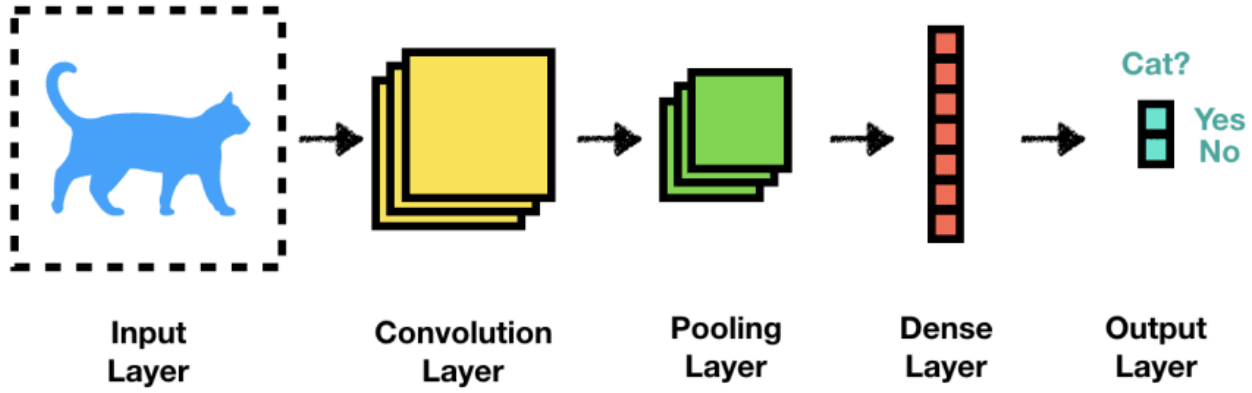
Slika 14. Max Pooling [3]

Na gornjoj slici vrijednosti u donjem redu predstavljaju one vrijednosti koje izlaze nakon prva dva stadija konvolucijskog sloja, a gornji red predstavlja vrijednosti nakon operacije sažimanja. Kao što se može vidjeti, od svih vrijednosti koje ulaze u neki neuron kao krajnja vrijednost tog neurona uzima se najveća ulazna vrijednost. Sažimanjem se postiže invarijantnost za male translacije. Budući da se uzima najveća vrijednost neke regije, mali pomaci neće izazvati promjenu vrijednosti same regije, zbog toga što će se na kraju opet uzeti ista maksimalna vrijednost unutar te regije. Ovo je jako korisno kada nas više zanima prisutnost nekog elementa unutar neke regije, nego gdje se taj element unutar te regije nalazi [3]. Posebnu korist pronalazi kada se obavlja postupak poput klasifikacije [3]. Kada se obavlja postupak poput segmentacije, gdje je jako bitna lokacija nekog elementa, moraju se dodati određeni elementi u mrežu kako se lokacija elementa ne bi izgubila (više o tome će se objasniti u dijelu vezanom za mreže koje se upotrebljavaju za segmentaciju). Sažimanje je nužno kako bi se omogućilo korištenje ulaznih varijabli različitih veličina [3]. Raznim načinima na koji se sažimanje može podesiti osigurava se da će zadnji sloj uvijek na izlazu imati jednak broj varijabli, neovisno o tome kolika je veličina ulaza (npr. uvijek će imati četiri izlaza neovisno o tome koja su dimenzije slike preko kojih se mreža trenira).

4.3. Upotreba konvolucijskih slojeva kod slika

Postoji jako veliki broj varijanti konvolucijskih slojeva koji se daju modificirati [3]. Za potrebe ovog rada bitno je razumjeti kako konvolucijski slojevi funkcioniraju kada se upotrebljavaju za klasifikaciju i segmentaciju slika.

Primjer se nalazi na slici 15.

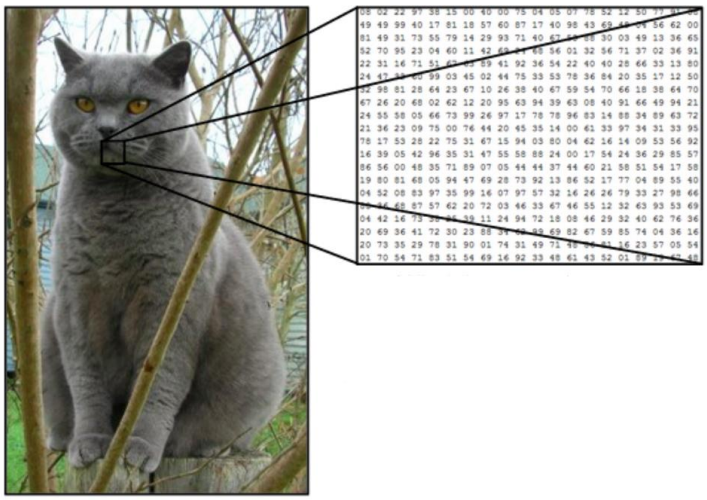


Slika 15. Prikaz prolaska jedne slike kroz konvolucijsku mrežu [22]

4.3.1 Konvolucija elemenata slika

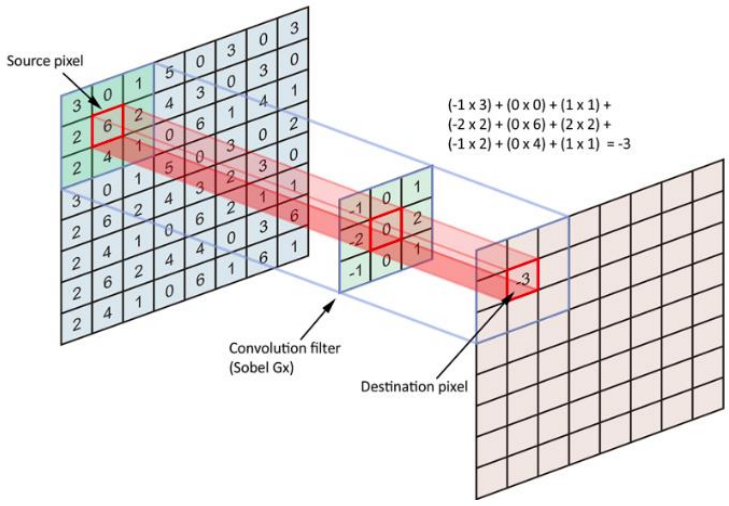
Svaka slika u konvolucijskoj umjetnoj neuronskoj mreži bit će predstavljena s brojkama. Mreže ne percipiraju slike kao slike, već kao niz matrica. U ranijim poglavljima opisali smo što je konvolucija te ju detaljnije pojasnili. Kod slika će nam ulazne vrijednosti biti pikseli. Konvolucija kod slika je karakteristična po tome što se mora uzeti u obzir da slike posjeduju podatke kroz tri područja: to je zbog toga što se sve slike definiraju kao kombinacija određene količine crvene, plave i zelene boje (eng. RGB slike) [22]. U sva tri područja određeni piksel poprima tri vrijednosti koje variraju od 0-255 (po jednu za svako područje). Vrijednost 255 predstavlja bijelu boju, a vrijednost 0 crnu boju.

Ako slike nisu RGB, one će imati samo jedno područje unutar kojeg će se svi pikseli moći zamijeniti brojkom između 0 i 255.



Slika 16. Način na koji računalo vidi slike [20]

Ako se za svaki piksel uzme da predstavlja jednu ulaznu vrijednost, onda možemo reći da ćemo RGB sliku dimenzija 512 x 512 zapravo predstaviti kao matricu dimenzija 512 x 512 x 3. Budući da za svaki piksel uzimamo po jedan neuron smanjenjem piksela u upotrebi kroz konvoluciju smanjit će se potreban broj neurona za rad mreže; točnost neće biti kompromitirana, a mreža će puno brže učiti. Jedan primjer može se vidjeti na slici 17.



Slika 17. Konvolucija kod slika [23]

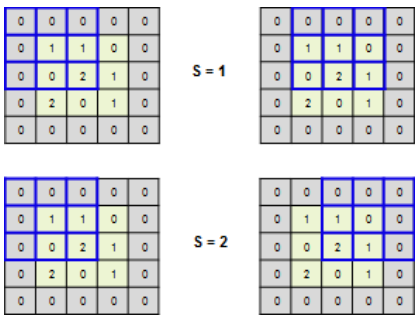
Lijevo se nalazi izvorna slika kod koje je svaki piksel predstavljen s nekom brojkom [23]. U sredini se vidi konvolucijski filter, odnosno matrica s kojom se množe vrijednosti slike koju analiziramo [23]. Desno se nalazi rezultat.

4.3.2. Rezultat konvolucije slika

Na tri načina (pomoću tri hiperparametra) se regulira izlazna vrijednost [24]:

- Dubina (eng. depth)
- Korak (eng. stride)
- Popunjavanje (eng. padding)

Dubina podrazumijeva količinu filtera iskorištenih na nekoj slici [24]. Budući da nas zanimaju razni podaci može se izvesti konvolucija na istoj slici sa različitim filterima, što će utjecati na izlazni rezultat [24]. Filter konvolucije pomiče se po slici određenim korakom. Korak vrijednosti jedan značiti će da se događaju pomaci za po jedan piksel, korak veličine dva da se događaju pomaci po dva piksela itd. [24]. Pomak se bira kako bi najbolje odgovarao pojedinoj situaciji, ali postoje preferirane vrijednosti pomaka. Što je pomak veći, to će se konvolucijom više smanjiti dimenzije neke slike [24].



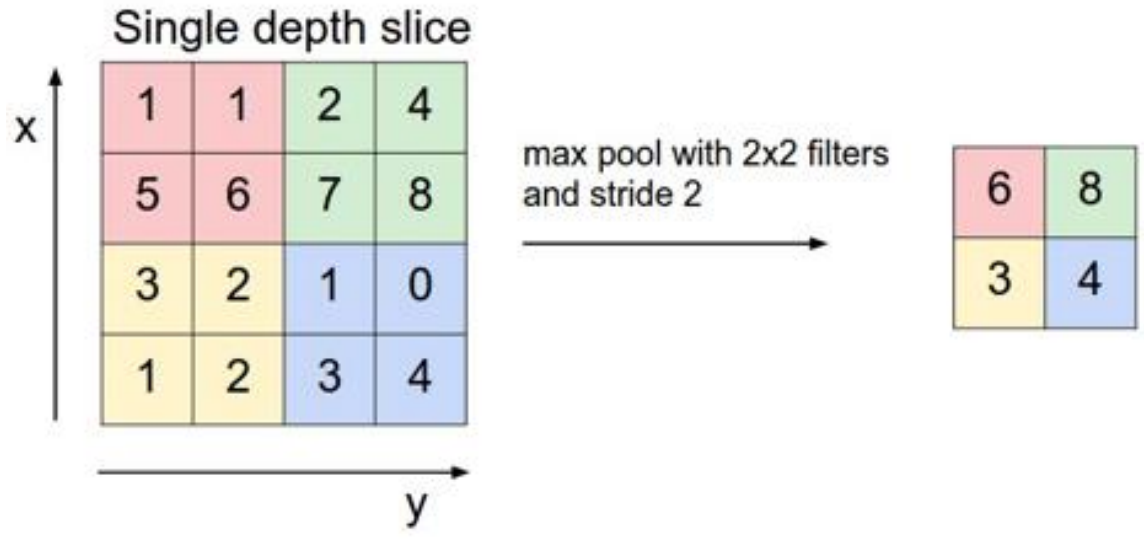
Slika 18. Pomak kod konvolucije [24]

Popunjavanje često se koristi kako bi postojala kontrola nad brojem izlaznih neurona. Najčešće se upotrebljava popunjavanje nulama (eng. zero padding). Kao što se može vidjeti na gornjoj slici matrice se popune s nulama kako bi oblikom odgovarale filterima. Popunjavanjem se postižu tri efekta [25]:

- Sprječava se prebrzo smanjivanje veličine ulaznih podataka
- Omogućuje se korištenje mreža s više slojeva
- Bolje se mogu očuvati podaci koji se nalaze uz rubove

4.3.3. Sažimanje kod analize slika u konvolucijskim mrežama

Nakon što se obavi konvolucija i na rezultat primjeni neka funkcija, kao treći stadij dolazi sažimanje. Sažimanje će smanjiti dimenzije slika, te će služiti kao dodatna zaštita od pretreniranja [23] . Upotrebom maksimalnog sažimanja uzima se najveća vrijednost unutar nekog područja.



Slika 19. Primjer sažimanja kod slika [23]

4.4. Upotreba čvrsto povezanih slojeva kod analize slika u konvolucijskim mrežama

Standardno se kao zadnji sloj kod konvolucijskih neuronskih mreža uzima čvrsto vezani sloj neurona [3]. Zadnji sloj obavlja klasifikaciju i potpuno je iste građe kao čvrsto vezani sloj kod standardnih feed-forward umjetnih neuronskih mreža.

Konvolucijski slojevi zaduženi su za učenje karakteristika. Oni ne obavljaju klasifikaciju, već služe kako bi mreža naučila koje elemente pokušava naći na nekoj određenoj slici.

Ovo je najlakše objasniti na primjeru umjetne neuronske mreže koju pokušavamo istrenirati da klasificira slike u dvije kategorije. Neka prva kategorija bude snimak mozga, a druga kategorija neka bude snimak pluća. Kada ubacujemo podatke u konvolucijsku umjetnu neuronsku mrežu, konvolucijski slojevi biti će zaduženi za to da nauče koji elementi su na slikama bitni, odnosno koji elementi razlikuju pluća od mozga. Na temelju toga kakvi se elementi nalaze na slici, zadatak zadnjeg čvrsto povezanog sloja je da odluči da li je riječ o snimci pluća ili mozga.

5. KONVOLUCIJSKE UMJETNE NEURONSKE MREŽE

Ovaj rad se bavi temom uporabe umjetnih neuronskih mreža za potrebe obavljanja zadatka segmentacije medicinskih snimaka. Kako bi se pravilno pristupilo tome problemu treba prvo objasniti što podrazumijeva segmentacija slika, prije nego što se uđe u detalje oko toga kako se segmentacija može obaviti pomoću dubokog učenja. Segmentacija slika je jedan od problema kojima se bavi područje kompjuterske vizije. Prije nego što se objasni sama segmentacija, moraju se nužno objasniti i zadaci klasifikacije i detekcije objekata zbog boljeg razumijevanja razlike između njih i segmentacije i zbog načina na koji umjetne neuronska mreža pristupa problemu segmentacije.

5.1. Klasifikacija

Zadatak klasifikacija slika može se definirati kao zadatak sparivanja slika s kategorijama unutar kojih te slike spadaju. U kompjuterskoj viziji se pomoću umjetnih neuronskih mreža taj problem može riješiti na sljedeći način [24]. Mreži se da jako puno primjera kod kojih je već označeno kojoj kategoriji pripadaju. Za vrijeme treniranja ona uči prepoznavati karakteristike na temelju kojih se neka slika svrstava u pojedinu kategoriju. Kvalitetno istrenirana mreža poslije može točno svrstati dotad neviđene primjere u jednu od kategorija, često s veoma visokom točnosti. Najpopularniji tip umjetne neuronske mreže koji se koristi u svrhu klasifikacije slika su konvolucijske umjetne neuronske mreže (eng. Convolutional Neural Networks).

5.2. Detekcija objekata

Detekcija objekata omogućava određivanje prisutnosti traženog objekta na nekoj slici. Složeniji je zadatak od klasifikacije i veoma često se na jednoj slici treba identificirati više objekata koji spadaju u različite kategorije. Kako bi se mreža istrenirala treba joj se dati mnogo primjera slika koje na sebi imaju označene objekte. Bitna razlika između detekcije i klasifikacije je da se kod klasifikacije cijela slika uvijek svrstava u neku kategoriju, dok se kod detekcije pojedini dijelovi slike svrstavaju unutar određene kategorije. Prema tome je sličnija segmentaciji nego klasifikaciji.

Područje na kojem se nalazi traženi objekt kod segmentacije je puno preciznije određeno. Kod detekcije objekata je označena samo općenita lokacija nekog objekta.

5.3. Segmentacija objekata

Segmentacija slika podrazumijeva dekompoziciju slike na određene regije (segmente) [25]. Segmenti na slici moraju zadovoljiti određene kriterije.

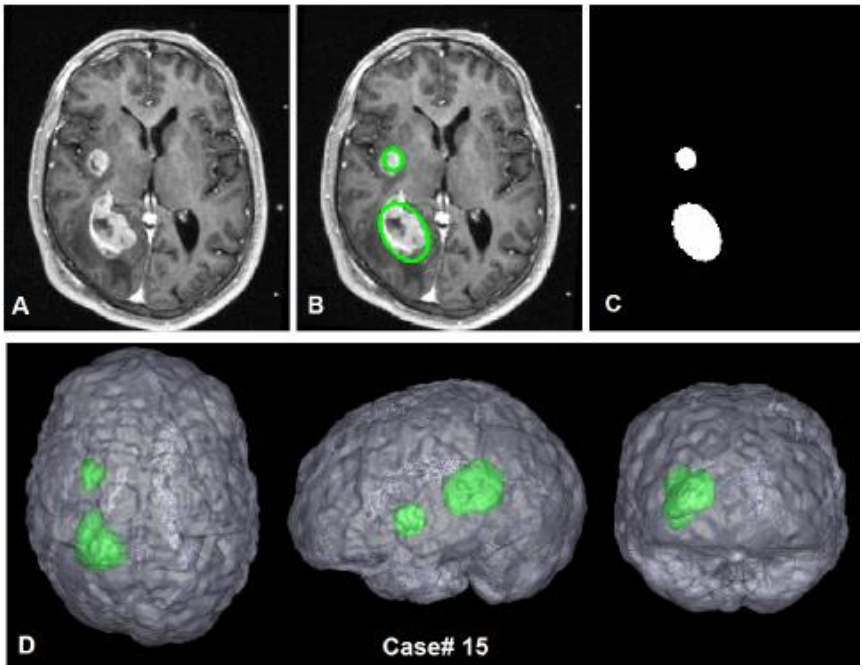
Kriteriji koji se moraju zadovoljiti su [25]:

- Regije moraju biti uniformne s obzirom na neko svojstvo
- Granice regija moraju biti relativno jednostavne
- Regije ne smiju imati male otvore
- Susjedne regije se moraju značajno razlikovati jedna od druge

Postoje dva tipa segmentacije [24]:

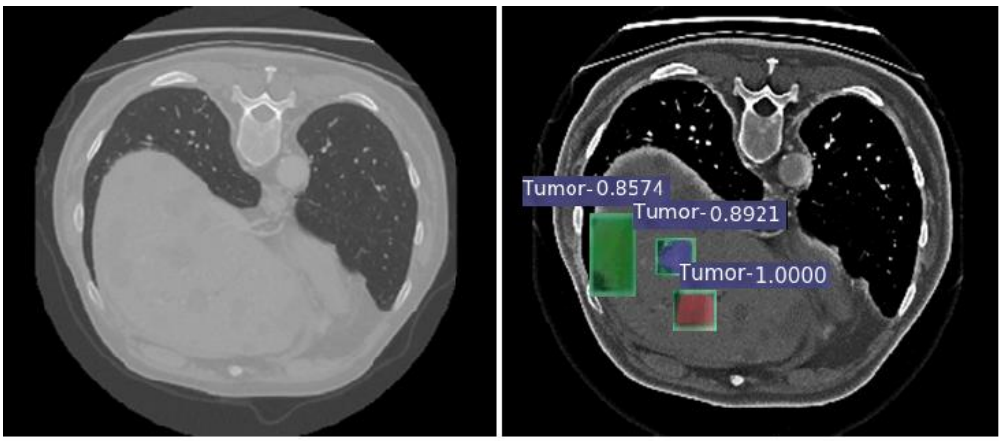
- Semantička segmentacija (eng. Semantic segmentation)
- Pojedinačna segmentacija (eng. Instance segmentation)

Semantička segmentacija se može tretirati kao klasifikacija na razini piksela. To znači da se svakom pikselu na slici dodjeljuje nekakva oznaka. Mana je što se ne može razdijeliti više članova koji spadaju u jednu kategoriju. Pojedinačna segmentacija s druge strane može prepoznati i razliku između članova koji spadaju unutar jedne kategorije. Najlakše je razliku primijetiti usporedbom slika koje predstavljaju semantičku segmentaciju i pojedinačnu segmentaciju (koja je zapravo kombinacija detekcije objekata i segmentacije slika).



Slika 20. Semantička segmentacija tumora na mozgu [26]

Kao što se može vidjeti na slici 26, iako postoje dva odvojena tumora na mozgu, kada se obavi semantička segmentacija ona će oba područja označiti istom kategorijom. S druge strane ako se na sličnom primjeru (u ovom slučaju tumori na jetri) obavi pojedinačna segmentacija onda rezultat izgleda nešto drugačije, kao što se može vidjeti na slici 27.



Slika 21. Instance Segmentation tumora na jetri [27]

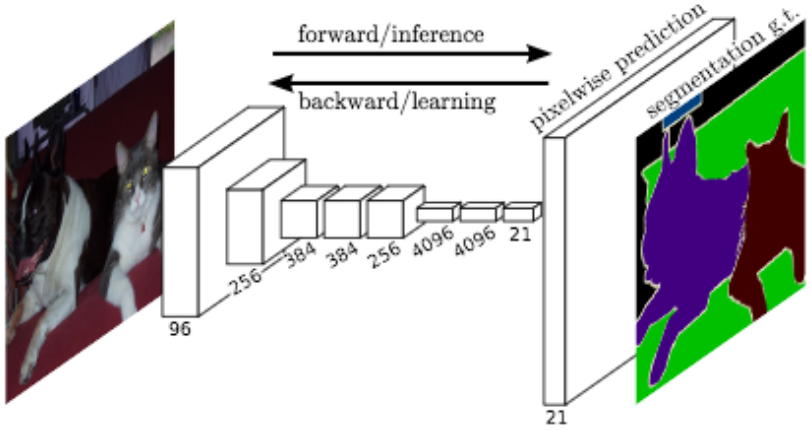
5.4. Arhitekture umjetnih neuronskih mreža koje se upotrebljavaju u svrhu segmentacije

Postoje razni algoritmi koji se upotrebljavaju u svrhu rješavanja problema segmentacije slika. Dolazi do nastanka raznih arhitektura neuronskih mreža kojima je cilj riješiti taj problem na što efektivniji način. Važno je napomenuti da, iako stalno nastaju nove arhitekture, zbog same prirode umjetnih neuronskih mreža i neke starije arhitekture su uz određene preinake konkurentne najnovijim arhitekturama. U ovom poglavlju spomenuti ćemo i ukratko objasniti arhitekture specijalizirane za segmentaciju kronološkim redoslijedom [28]:

- Potpuno konvolucijske mreže (FCN)
- ParseNet
- U-Net
- Feature Pyramid Network
- Pyramid Scene Parsing Network (PSPNet)
- Mask R-CNN
- DeepLab, DeepLabv3 i DeepLabv3+
- Path Aggregation Network (PANet)
- Context Encoding Network (EncNet)

5.4.1. FCN

Poseban tip mreža koje karakterizira činjenica da se sastoje isključivo od konvolucijskih slojeva, predložena u radu " Fully Convolutional Networks for Semantic Segmentation" 2015. godine [29]. Ovakve mreže imaju mogućnost uzimanja slike bilo kojih dimenzija i prezentiranje segmentirane slike istih tih dimenzija kao rezultata rada mreže. Riječ je o mrežama koje osim konvolucijskih slojeva koriste i tzv. dekonvolucijske slojeve te preskočne veze (eng. skip connections).

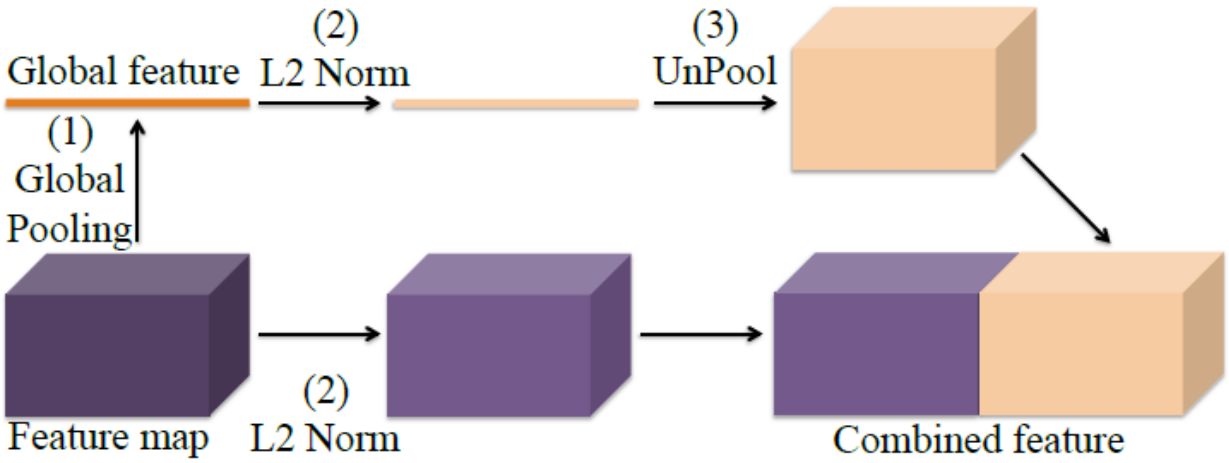


Slika 22. FCN [28]

5.4.2. ParseNet

Ova mreža zapravo predstavlja FCN s određenim poboljšanjima koja su predložili autori znanstvenog članka po imenu "ParseNet: Looking Wider to See Better" [30]. Mreža je drukčije arhitekture što joj omogućava da uzme u obzir kontekst onoga što se nalazi na slici, dok FCN u potpunosti ignoriraju globalne informacije o slikama [30]. To ne stvara probleme u slučaju da je ono što želimo segmentirati na slici jako različito od ostatka slike. Međutim što je veća sličnost među regijama koje želimo segmentirati, korištenje FCN-a će biti manje efektivno, upravo zbog toga što taj tip mreže ignorira kontekst unutar kojega se neki dio slike nalazi [31]. Prijedlog je koristiti umjesto konvolucijskih slojeva tzv. module.

Ti moduli omogućavaju zadržavanje globalnog kontektsta, a građa im se može vidjeti na slici 23.



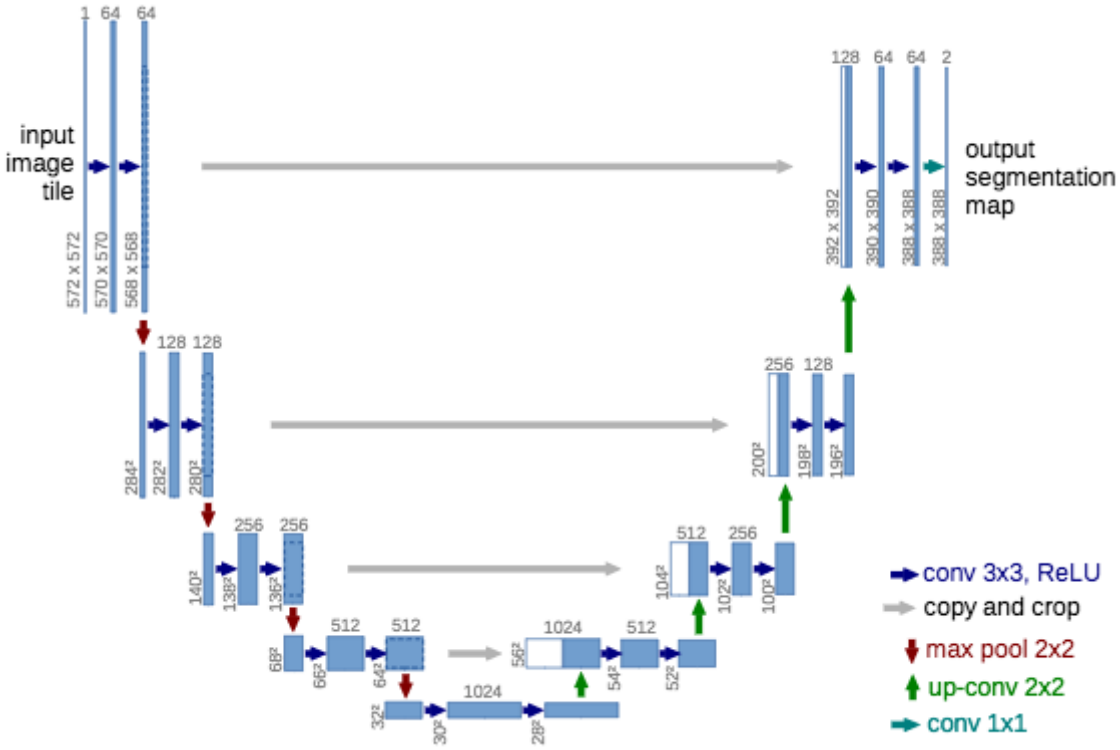
Slika 23. Modul koji koristi ParseNet [31]

Kao što se može vidjeti na slici 8. umjesto standardne konvolucije ovdje se sloj koji bi inače bio konvolucijski dijeli na dva "puta": gornji i donji put. U principu se primjenjuju dva različita postupka na isti ulazni element, čiji se rezultati lančano vežu (eng. concatenate). Rezultat korištenja ovakve strukture mreže je veća točnost mreže kod segmentacije.

5.4.3 U-Net

Ova mreža tip je FCN-a prilagođen upotrebi u medicinske svrhe. Predložena je u radu po imenu "U-Net: Convolutional Networks for Biomedical Image Segmentation" [32]. Ime je dobila zbog toga što svojom strukturom podsjeća na slovo U. Ova mreža posjeduje značajno veću moć lokalizacije od prije spomenutih mreža. Svaki piksel na slici se tretira kao središte minijaturne regije, kojoj se onda dodjeljuje određena oznaka [32]. Zbog toga što se kao ulazni podaci koriste te male regije (eng. patches), dobiva se nešto veći broj ulaznih podataka nego kad bi se cijele slike koristile kao ulazni podaci (ovisno o razlici između veličine samih regija i cijele slike) [32].

U-Net mreža se sastoji od dva područja: područje skupljanja (eng.contracting path ili downsampling) i područje širenja (eng. expanding path ili upsampling) [32]. Područje skupljanja smanjuje dimenziju slike (odnosno u ovom slučaju regije) [32], a usput zadržava bitne karakteristike ulaznih podataka. Posljedica toga je da se dolazi do spoznaje da li se tražena karakteristika nalazi na slici, ali se gubi podatak gdje se ta karakteristika nalazi (što je problem jer je cilj segmentacije označiti na orignalnoj slici određenu regiju s određenom oznakom) [32].



Slika 24. U-Net struktura [32]

Zbog toga se nakon skupljanja provodi postupak širenja. Obavlja se dekonvolucija slike. Iz umanjenih slika ponovno se dobivaju uvećane slike na kojima se uspješno može odrediti područje koje neka regija zauzima na slici [32]. Bitno je uključiti između blokova koji se nalaze u downsamplingu i upsamplingu tzv. preskočne veze (eng. skip-connections) kako bi se izbjegao gubitak podataka o strukturi slike na svakoj pojedinoj razini downsamplinga [31].

Često se na podacima sprovodi i tzv. augmentacija podataka [32]. Kako bi se postigla što veća robusnost mreže i omogućilo joj se postizanje što veće sposobnosti generalizacije. U svrhu augmentacije se koriste razne metode: linearne metode poput uvećavanja pojedinih područja i rotiranja slika, prostorne deformacije kao što je elastična deformacija (koja se pokazala jako korisnom u svrhe rada ove mreže) [32].

U-Net briljira kod segmentacije s jako malom količinom podataka za treniranje. Zbog toga što se u području medicine često barata s malim brojem podataka U-Net je jedna od mreža koje se uobičajeno koriste u svrhu segmentacije u medicini (gdje je U-Net isproban u svrhu rješavanja više različitih problema te je pokazao jako visoke stope uspješnosti) [32].

5.4.4. Feature Pyramid Network

Ova mreža, prezentirana u radu "Feature Pyramid Networks for Object Detection" sastoji se od dva dijela: bottom-up pathway i top-down pathway (pritom je proces neovisan o tome koja konvolucijska mreža služi kao osnova) [33]. Namijenjena je za upotrebu u područjima detekcije objekata na slikama, ali i segmentacije slika.

Bottom-up pathway i top-down pathway međusobno su povezani s lateralnim vezama koje omogućuju povezivanje karakteristika slika manjih dimenzija sa slikama većih dimenzija [33].

Slika 25. Feature Pyramid Network [33]

5.4.5. PSPNet

Ova mreža predložena je u radu " Pyramid Scene Parsing Network" [34]. Glavni cilj ovakve strukture je da se što bolje očuva globalni kontekst elemenata koji se nalaze na nekoj slici [31]. Ova mreža ne radi neke pogreške koje se mogu dogoditi mrežama koje su dosad navedene. Dok bi dosad spomenute mreže mogle na slici neki brod mogle slučajno svrstati u kategoriju auta, ako je oblikom jako sličan autima, ova mreža omogućava uzimanje pravilnog konteksta u obzir (činjenica da se vozilo nalazi na vodi) pa se tako smanjuje broj grešaka [34]. Uzorci se izvlače iz slike pomoću metode navedene u radu "Deep Residual Learning for Image Recognition" [35] koja se zove "dilated network strategy" [35]. Zahtjeva se uporaba pooling metode piramidalnog oblika. Sažimanje se odvija na četiri razine koje predstavljaju četiri različite veličine istog ulaznog podatka [31].

Može se primijetiti kako je ovaj tip mreže jako koristan kada se obavlja segmentacija većeg broja regija na jednoj slici [34]. Segmentacijska mapa koja se dobije na taj način jako je precizna; postiže vrhunske rezultate na različitim tipovima skupova podataka.

5.4.6. Mask R-CNN

Ova mreža, predložena u radu "Mask R-CNN" [36], u trenutku objave je srušila sve rekorde po pitanju točnosti na velikom broju skupova podataka koji se koriste kao mjerilo za testiranje mreža koje se upotrebljavaju za segmentaciju slika [31].

Dok su dosadašnje mreže jako dobro rješavale probleme identifikacije objekata i semantičke segmentacije, Mask R-CNN je prva koja je rješavala određene zadatke pojedinačne segmentacije na visokoj razini [36]. To znači da ova mreža uspješno rješava oba zadatka u isto vrijeme. Pravilno identificira objekte koji se nalaze na slici i određuje njihov položaj tj. obavlja segmentaciju. Temelj ove mreže je Faster R-CNN, predložen u radu po imenu " Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" u svrhu identifikacije objekata na slici [36].

Proces rada Mask R-CNN može se svesti na dva koraka:

- Identifikacija objekata: odrede se svi objekti na slici (mora se odrediti i zasebno označiti i kada se pojavljuje više objekata koji spadaju u istu klasu)
- Semantička segmentacija unutar svakog područja koje je označeno kao područje nekog objekta

Ovakvom podjelom rada postiže se velika točnost: budući da se može postići velika točnost identifikacije objekata na slici, mreži je puno lakši zadatak analizirati svako područje nekog objekta kao zasebno područje na kojem ima neko glavno područje koje treba segmentirati (objekt lociran unutar tog područja). Budući da se ovaj rad bavi problemom semantičke segmentacije neće se dalje ulaziti u detalje oko Mask R-CNN mreže. Slojevi i određene metode koji se koriste u ovoj mreži uporabljeni su isključivo zato što se rješava zadatak pojedinačne segmentacije, te kao takvi nisu bitni za ono što ovaj rad obrađuje.

5.4.7. DeepLab, DeepLabv3 and DeepLabv3+

Ova mreža temelji se na radu "Feature Pyramid Networks for Object Detection" [37]. Kada se govori o DeepLab mreži zapravo će se pričati o DeepLabv2 zbog toga što je pod tim imenom navedena u radu " DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs" [38]. DeepLab bavi se rješavanjem tri problema koristeći pritom tri modifikacije koje bi trebale omogućiti veću preciznost umjetnih neuronskih mreža kod rješavanja problema semantičke segmentacije [38].

Tri problema semantičke segmentacije su [38]:

- Smanjenje rezolucije značajki
- Postojanje objekata pri različitim razinama uvećanja
- Prostorna invarijantnost (eng. Spatial invariance)

6. OBRADA MEDICINSKIH PODATAKA

U ovom poglavlju detaljnije će biti objašnjeni podaci koji se koriste u radu, te kako se oni pripremaju za umjetne neuronske mreže. Pojasniti će se razlika između standardnih tipova podataka koji se koriste kod konvolucijskih umjetnih neuronskih mreža, te podataka koji se dobivaju snimanjem pomoću raznih medicinskih uređaja. Bitno je razumijeti format unutar kojega podaci dolaze da bi se njima moglo pravilno baratati.

6.1. Medicinske baze podataka

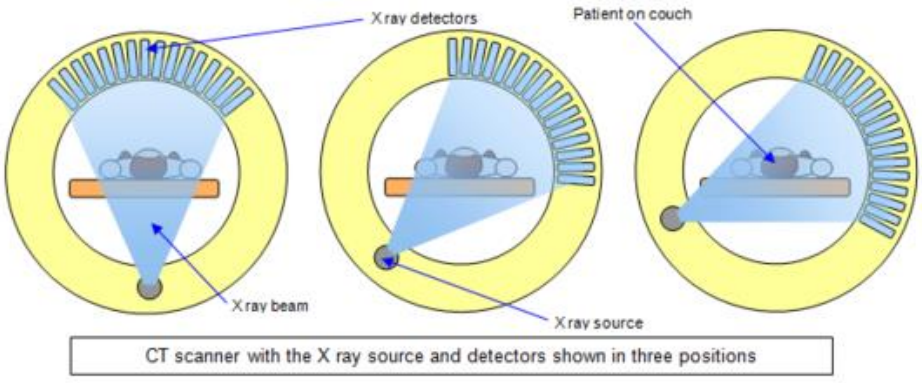
Baze podataka koje se koriste u medicini imaju niz karakteristika koji ih razlikuje od baza podataka koje se koriste u druge svrhe. Bez poznavanja tih razlika ne može se razviti efektivan način pretprocesiranja podataka.

6.1.1. Medicinski uređaji za snimanje

Kako bi razumjeli podatke koji se koriste u medicinske svrhe (u slučaju ovog rada bit će riječ o snimkama mozga) treba posjedovati osnovno znanje o načinu funkcioniranja uređaja koji snimaju pacijente, te izbacuju podatke u formatu karakterističnom za upotrebu u medicini. Budući da su u ovom radu korišteni podaci tj. snimke dobivene od strane uređaja kompjutorizirane tomografije treba objasniti osnovni princip funkcioniranja jednog takvog uređaja. Uređaj za računalnu tomografiju (eng. computed tomography scan odnosno CT scan) koristi se u radiologiji za dobivanje snimaka raznih organa unutar ljudskoga tijela [39]. Računalna tomografija je radiološka metoda oslikavanja koja daje slojevni prikaz tijela koristeći ionizirajuće zračenje za nastanak slike [39]. CT uređaj sastoji se od pomičnog stola, na kojemu leži pacijent, i od tzv. tunela uređaja [39]. Pomični stol služi kako bi pacijent na njemu ležao za vrijeme snimanja, dok se CT uređajem zapravo smatra tunel uređaja. Taj tunel unutar sebe sadrži rendgensku cijev i detektore zračenja [39].

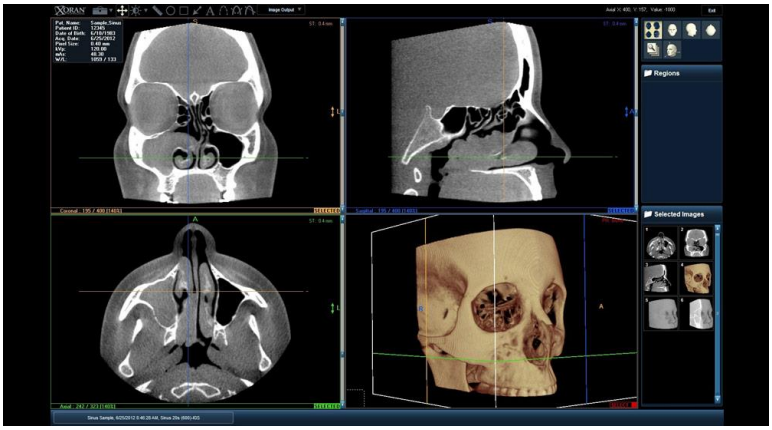
Slika 26. Uređaj za računalnu tomografiju [40]

Pacijent treba nepomično ležati na pomičnom stolu i onda se taj pomični stol polako giba kroz tunel CT uređaja. U kućištu koje čini tunel nalazi se rendgenska cijev koja za vrijeme cijelog pregleda kruži oko pacijenta [39]. Ta cijev proizvodi rendgenske zrake koje prolaze kroz tijelo pacijenta [39]. Nasuprot toj cijevi nalaze se detektori koji bilježe rendgensko zračenje. Što je dio tijela veće gustoće to rendgenske zrake više slabe prolaskom kroz njega, što će se primijetiti na detektorima. U detektorima se zračenje pretvara u električni signal. Rezultat ovog postupka je nastanak dvodimenzionalne slike snimljenog područja u nijansama sive boje. Svaka nijansa sive boje odgovarat će jedinici apsorpcije rendgenskih zraka (Housefieldove jedinice) [39].



Slika 27. Snimanje pacijenta CT uređajem [41]

Iako je prikaz dvodimenzionalan većina uređaja danas može na temelju CT snimki pacijenta obaviti tzv. multiplanarne rekonstrukcije; može se dobiti trodimenzionalni prikaz snimljenog organa [39]. Trodimenzionalni prikaz može biti jako koristan u pojedinim slučajevima. Poželjno je prikazati rezultat segmentacije obavljen pomoću umjetne neuronske mreže na trodimenzionalnom modelu organa koji je sniman. Takvi prikazi jako su korisni kod predoperativne pripreme; kirurzi pomoću modela mogu značajno jednostavnije isplanirati postupak te smanjiti opasnost po pacijenta.



Slika 28. Trodimenzionalni prikaz ljudske glave [42]

6.1.2 Format medicinskih snimaka

Biomedicinske snimke karakteristične su po standardu koji se koristi. One se najčešće spremaju prema DICOM standardu (eng. Digital Imaging and Communications in Medicine) [43]. Ovaj standard karakterizira [43]:

- mogućnost dijeljenja informacija između strojeva različitih proizvođača
- konstantan napredak i poboljšavanje formata kako bi bio usklađen s najnovijim medicinskim standardima
- format je besplatan za korištenje

Korištenjem DICOM formata omogućava se brza i efikasna komunikacija liječnika što dovodi do veće uspješnosti kod liječenja pacijenata [43]. DICOM je vrlo opsežni standard koji se koristi za većinu procedura u radu s medicinskim podacima [44]. Vlasnik prava na DICOM format je organizacija NEMA (National Electrical Manufacturers Association) [44]. Informacije u DICOM formatu grupiraju se u skupove podataka (eng. datasetove). Na taj način se povezuju slikovni podaci s metapodacima [44]. Metapodaci uključuju ime pacijenta, ID, datum snimanja itd [44]; podaci se povezuju s informacijama koje ih opisuju (slično kao oznake kod JPEG formata) [44]. Većina atributa u DICOM formatu su metapodaci, uz iznimku podataka vezanih za slikovne vrijednosti piksela (eng. pixel data). DICOM standard podržava sve bitne metode koje se koriste u medicinskoj vizualizaciji. Te metode se nazivaju modalitetima [44]. Modaliteti podržani od strane DICOM standarda su [45]:

- BI - Biomagnetsko oslikavanje (engl. Biomagnetic Imaging)
- CR - Računalna radiografija (engl. Computed Radiography)
- CT - Računalna tomografija
- DG - Diafanografija (eng. Diaphanography)
- DX - Digitalna radiografija
- EM - Elektronski mikroskop (eng. Electron Microscope)
- ES - Endoskopija
- GM - Općeniti mikroskop
- LS - Površinski laserski sken
- MG - Mamografija
- MR - Magnetska rezonanca
- NM - Nuklearna medicina

- OT - Ostalo
- PT - PET (eng. Positron Emission Tomography)
- RF - Radio fluoroskopija
- RG - Radiografsko oslikavanje
- RT - Terapija ozračivanjem
- SC - Sekundarno oslikavanje (eng. Secondary Capturing)
- SM - Slide Microscopy
- TG - Termografija
- US - Ultrazvuk
- VL - Vidljiva svjetlost
- XA - Rendgenska angiografija
- XC - Vanjska kamera (fotografija)

Postoje posebni programi koji se koriste za čitanje ovog formata (postoje i besplatni programi i programi koji se plaćaju). Za potrebe ovog rada korišten je program OsiriX. Programi koji se koriste su [44] :

- 3D Slicer
- Ambra Health
- CinePaint
- GIMP
- Ginkgo CADx
- Irfan View

- MicroDicom
- Noesis
- OsiriX
- Orthanc
- InVesalius
- IDL

6.2. Pretvorba CT snimaka u format prilagođen radu umjetne neuronske mreže

Već prije objasnili smo kako funkcionira snimanje pacijenata CT uređajem. Ono što treba dodatno pojasniti je u kojem se obliku dobiva CT snimka. CT snimanje stroj obavlja s nekim određenim pomakom. Sama CT snimka dobiti će se u obliku niza režnjeva (eng. sliceova). Svaki režanj je dvodimenzionalni prikaz ljudskog mozga u nijansama sive boje u nekom određenom trenutku tijekom snimanja, odnosno na nekoj određenoj poziciji udaljenoj od početne pozicije snimanja za pomak snimanja. Podaci se, kako bi se koristili u umjetnim neuronskim mrežama, trebaju transformirati u oblik koji je pogodan za rad s umjetnom neuronskom mrežom. U slučaju ovoga rada podaci se trebaju pretvoriti u oblik pogodan za rad s konvolucijskim umjetnim neuronskim mrežama. Kako bi se to napravilo korišten je program OsiriX [46]. OsiriX omogućava učitavanje podataka u DICOM formatu te pretvaranje tih podataka u neki drugi format. Podaci se mogu pretvoriti u JPG, u TIFF format te u druge formate kako bi se mogli iskoristiti za razne potrebe [46]. Pretvorba DICOM režnjeva u slike obavlja se tako da svaki režanj postaje jedna zasebna slika. Izabrani format za pretvorbu slika u ovom radu je TIFF (Tagged Image File) format [47]. Umjesto korištenja JPEG formata kod kojeg će nam u pretvorbi ostati 8-bitna slika, korištenjem TIFF formata može se sačuvati slika kvalitete 16-bita [47]; većina današnjih uređaja koji koriste DICOM format podatke o pikselima spremaju u 16-bitnom formatu. Problem ovog formata je veličina. Veličina 16-bitne slike značajno je veća od 8-bitne slike. Međutim budući da je skup podataka korišten u ovom radu jako limitiran ta razlika u veličini ne stvara primjetne poteškoće.

6.3. Augmentacija kod pretprocesiranja podataka

Kod izrade same umjetne neuronske mreže jako je bitno uzeti u obzir podatke koji su na raspolaganju za treniranje i testiranje te umjetne neuronske mreže. To je pogotovo bitno kod biomedicinskih snimaka. Zbog toga što za svaku snimku treba tražiti dozvolu od pacijenta kako bi se ona koristila u znanstvene svrhe, pa tako i u svrhu treniranja i testiranja neke umjetne neuronske mreže, gotovo nikad nema dovoljno podataka za treniranje umjetnih neuronskih mreža. U današnje vrijeme pojavljuju se neke veće javne baze podataka, međutim te baze podataka su prilično specijalizirane.

One pokrivaju područje najčešćih primjena umjetnih neuronskih mreža za svrhe medicinske segmentacije. Zbog toga je jako bitno optimizirati pretprocesiranje podataka kako bi se podaci koje imamo mogli što bolje iskoristiti za treniranje određenih umjetnih neuronskih mreža. Kod učitavanja podataka u mrežu kao dio pretprocesiranja obaviti će se i augmentacija podataka, kako bi se na par različitih načina povećao efektivan set podataka za treniranje umjetne neuronske mreže. Najbolji način za augmentaciju podataka, koji još uvijek osigurava zadržavanje svih bitnih karakteristika medicinske slike, odavno je tema raznih istraživanja.

Kao neke od najboljih metoda za augmentaciju podataka spominju se [48]:

- Smicanje (eng. Shear)
- Gaussov filter (eng. Gaussian filter)
- Rotacijski postupci (eng. Rotation)
- Rastezanje i skupljanje (eng. Scaling)

Postoje i druge metode koje se mogu koristiti (kao npr. dodavanje šuma) koje su manje korisne [48]. Priča za sebe su elastična deformacija i augmentacija kontrasta [49], koje

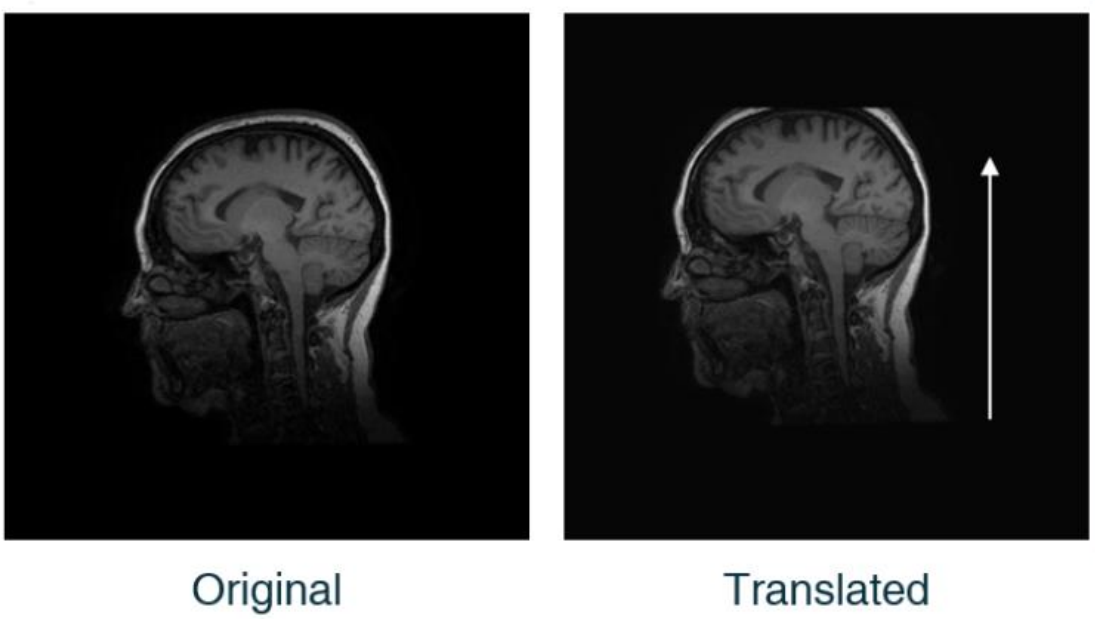
se znatno razlikuju u uporabi od prije navedenih. U ovome radu bit će upotrebljene određene metode augmentacije podataka kako bi se maksimalno optimizirao rad mreže i postiglo da se mreža može kvalitetno istrenirati sa što manjom količinom podataka. Sljedeće metode augmentacije podataka su uzete u obzir za potrebe ovog rada:

- Translacija
- Rotacija
- Zrcaljenje
- Rastezanje
- Smicanje

Gore navedene metode se mogu iskoristiti u svrhe augmentacije podataka bez ikakve brige zbog toga što predstavljaju linearne transformacije, što znači da ne mogu stvoriti problema segmentaciji. S druge strane postupak kao elastična deformacija spada u prostornu augmentaciju podataka i još uvijek se raspravlja da li ih je sigurno koristiti u svrhe augmentacije medicinskih podataka ili ne.

6.3.1. Translacija

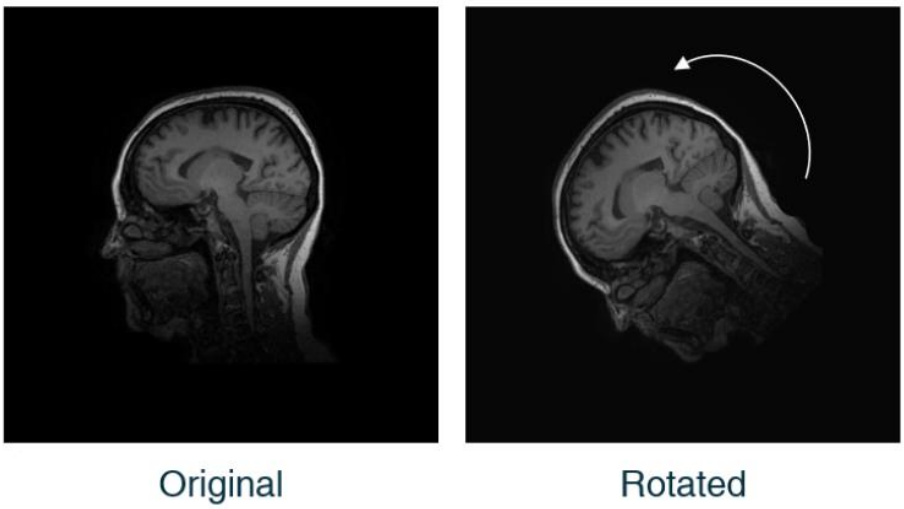
Translatirani podaci (eng. translated) se dobivaju na jednostavan način; područje interesa na nekoj slici se pomiče (translatira) u odnosu na središte slike [49]. Ovo je jedna od najjednostavnijih transformacija koje se mogu iskoristiti u svrhu pomaganja mreži da postigne što veću sposobnost generalizacije [49].



Slika 29. Translacija slike [49]

6.3.2. Rotacija slike

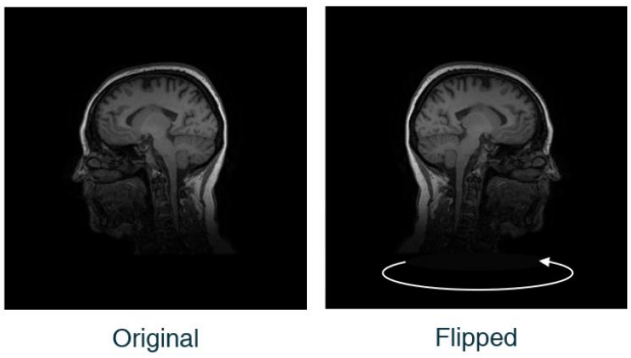
Ova metoda augmentacije slika sastoji se od rotiranja slike za nasumičan broj stupnjeva [49]. Tako se poboljšava sposobnost generalizacije jer se često događa da pacijenti nisu snimljeni apsolutno ravno već pod nekakvim malim kutom [49].



Slika 30. Rotacija slike [49]

6.3.3. Zrcaljenje

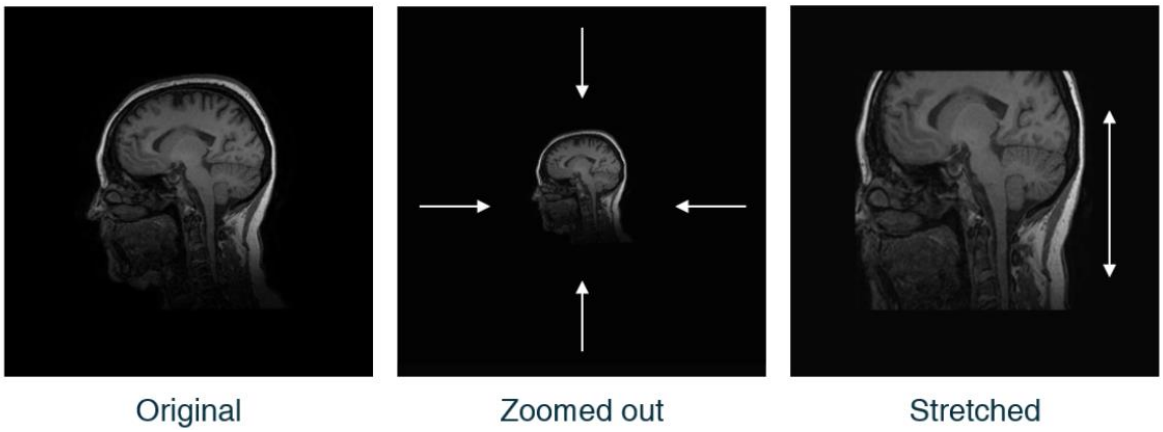
Metoda augmentacije slika koja je posebno pogodna za upotrebu kod relativno simetričnih slika (kao što su npr. medicinske snimke jer na sebi sadrže anatomske strukture koje su većinom simetrične) [49]. Posebna korist zrcaljenja kod medicinskih snimki je što onemogućavaju mreži da greškom nauči prioritizirati jednu polovicu simetrične strukture u odnosu na drugu kod postupka pronalaženja regije interesa [49].



Slika 31. Zrcaljenje slika [49]

6.3.4. Rastezanje

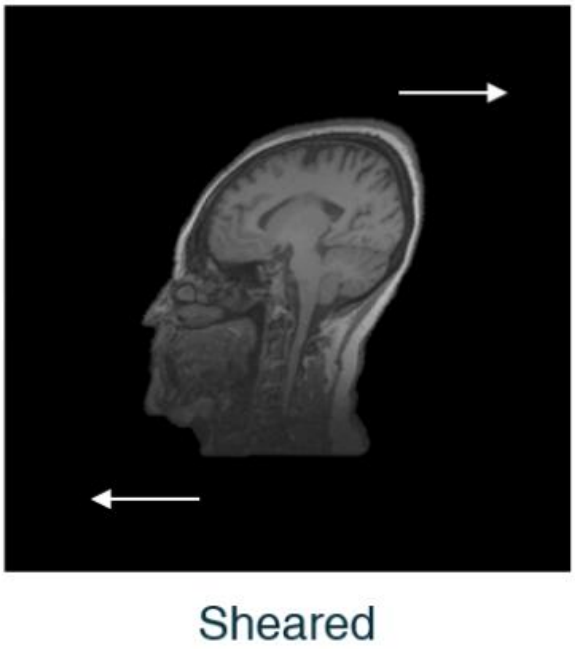
Često veličina pacijenta varira, pa je zbog toga korisno sprovesti postupke uvećavanja ili umanjivanja. Ako se ne trudimo sačuvati originalne omjere između visine i širine pojavljuje nam se rastezanje slike u nekom smjeru. Kod ove metode jako je bitno pravilno odrediti varijabilnost [49]; neki dio snimke ne smije se toliko uvećati ili umanjiti da se izgubi poveznica sa stvarnim objektom (npr. mozak se ne može augmentirati tako da završi s volumenom od dvije litre jer će to zbuniti mrežu kada dobije snimke pravih mozgova) [49].



Slika 32. Rastezanje slika [49]

6.3.5. Smicanje

Dok se kod rastezanja slika deformira u jednom smjeru, kod smicanja će se slika deformirati u dva različita smjera [49].



Slika 33. Smicanje slike [49]

6.3.6. Posebne metode augmentacije podataka

Dosad spomenute metode augmentacije nazivaju se linearnim metodama augmentacije podataka [49]. Smatraju se konvencionalnim metodama augmentacije podataka. Postoje ekstremnije metode augmentacije podataka kao što su [49]:

- Elastična deformacija
- Augmentacija kontrasta

Ove ekstremnije metode jako je opasno koristiti. Ako se krivo postave parametri onda se podaci mogu u potpunosti kompromitirati, odnosno rezultati treniranja umjetne neuronske mreže postaju neprimjenjivi u praksi (podaci na kojima se mreža trenira bit će previše različiti od izvornih podataka) [49].

7. DIZAJNIRANJE MREŽE ZA SEGMETNACIJU TKIVA

Na temelju dosad navedene teorije može se dizajnirati umjetna neuronska mreža namijenjena segmentaciji ventrikula na CT snimkama pacijenata.

7.1. Priprema podataka

Prije nego što se mogu iskoristiti u svrhe treniranja umjetne neuronske mreže podaci kojima raspolažemo moraju se kvalitetno pripremiti. Snimke s CT uređaja dolaze u DICOM formatu pa je prvi korak prebaciti ih u TIFF format. Nakon toga treba podijeliti podatke u dvije kategorije: moramo ih razdijeliti na slike i na tzv. segmentacijske maske. Svaka CT snimka sastoji se od niza režnjeva (eng. slice). Broj tih režnjeva primarno ovisi o tome kako je namješten CT uređaj (koliki je korak snimanja određenog CT uređaja). Za potrebe ovog rada svaki režanj bit će pretvoren u jednu sliku. Ovisno o različitim Housefieldovim jedinicama svaka slika sadržavat će prikaz u nijansama sive boje (npr. crna boja predstavljati će zrak).

Slika 34. Prikaz režnja s ventrikulom

S druge strane u programu OsiriX obavljena je ručna segmentacija ventrikula kako bi se dobili podaci za treniranje mreže. Budući da je segmentacija medicinskih snimaka značajno kompliciraniji zadatak od klasifikacije običnih slika nikada se ne dobiva točnost koja je veoma blizu sto postotnoj točnosti.

Kompetitivni rezultati variraju od 92-97 %. Zbog toga je dogovoreno da će se kod izrade ručne segmentacije odvojiti malo dodatnog prostora oko ventrikule kao određena vrsta faktora sigurnosti. Na taj način će se osigurati da se kod medicinskih postupaka sigurno ne zahvati područje ventrikula. Područje interesa na tim slikama su područja ventrikula. Rezultat ručne segmentacije bit će po jedna slika za svaki režanj. Međutim u slučaju segmentiranih slika na tim slikama će sve osim područja interesa biti definirano kao pozadina.

Slika 35. Prikaz segmentacijske maske

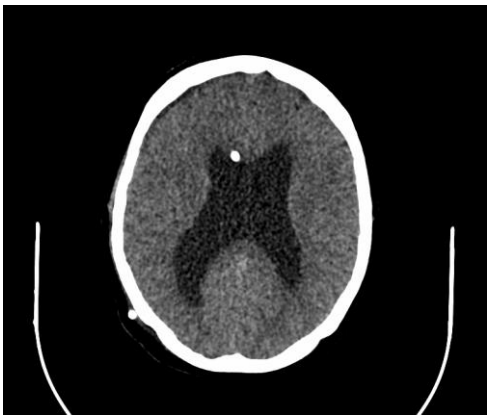
Treniranje mreže obavlja se s nadzorom. To znači da će se mreža trenirati pomoću primjera povezanih s oznakama. U slučaju segmentacije ventrikula primjeri će biti slike koje predstavljaju režnjeve, a oznake će biti segmentacijske maske. Sve slike se u krajnjem obliku mogu vidjeti kao slike nijansa sive (eng. grayscale image), a ne kao slike u boji (eng. RGB image).

7.2. Mreža za segmentaciju ventrikula

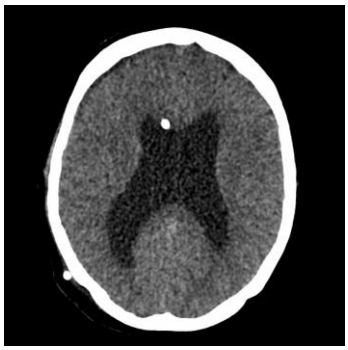
Za segmentaciju ventrikula koristit će se modificirana verzija mreže U-Net [32]. Modifikacije su sprovedene kako bi se arhitektura mreže prilagodila zadanom zadatku. Zadatak segmentacije ventrikula značajno je jednostavniji od zadataka kao što je npr. segmentacija tumora.Ventrikul se značajno razlikuje od tkiva koja se vide na CT snimkama mozga, pa je zbog toga moguće uspješno istrenirati umjetnu neuronsku mrežu sa manjom količinom podataka. Zbog relativne jednostavnosti zadatka U-Net se može modificirati; uz pravilno postavljanje hiperparametara može se za potrebe segmentacije ventrikula iskoristiti umjetna neuronska mreža s manjim brojem slojeva od originalnog U-Neta.

7.2.1. Učitavanje podataka u mrežu

Prije nego što se pređe na samu arhitekturu mreže treba se pojasniti način učitavanja podataka u mrežu. Podaci se moraju moći učitati na efikasan način te ih se treba podvrgnuti transformacijama navedenim u poglavlju 7.3. Prije svega treba napomenuti format slika. Transformacijom režnjeva dobivene su slike raznih dimenzija (667 x 662 ,708 x 662 itd.) Dimenzije variraju od pacijenta do pacijenta. Odmah se može uočiti da se slike mogu obrezati (eng. crop) kako bi se izbacio višak koji ne igra nikakvu ulogu kod treniranja mreže (izbacuje se što je više moguće sa slike a da se ipak obuhvati cijeli mozak i da se na njoj nalazi minimalno artefakata i zraka), te kako bi se slike svele na jednake dimenzije [50]. Također je bitno da te dimenzije budu prikladne za korištenje kod svih pacijenata. Kroz testiranje došlo se do zaključka kako su optimalne dimenzije na koje se slike mogu obrezati 576 x 576. Jako je bitno sve postupke sprovesti i na segmentacijskim maskama koje označavaju slike kako bi one i dalje kvalitetno predstavljale segmentirani ventrikul.



Slika 36. Prikaz režnja prije obrezivanja



Slika 37. Prikaz režnja poslije obrezivanja

Poslije obrezivanja slikama se mogu promijeniti dimenzije (isti postupci sprovode se i na segmentacijskim maskama koje odgovaraju slikama). Što su slike veće to će zauzimati više memorije kada se bude trenirala umjetna neuronska mreža. Slike dimenzija 576 x 576 nepotrebno su velike za potrebe segmentacije ventrikula. Smanjivanjem slika štedi se na memoriji te se povećava efikasnost rada mreže [50]. Testiranjem se zaključilo da je optimalna veličina slika 128 x 128; sve potrebne informacije i dalje su sačuvane, a rad mreže će biti puno efikasniji.

Slika 38. Slika poslije promjene dimenzija

Nakon umanjivanja slika i segmentacijskih maski one su spremne za učitavanje u mrežu. Za učitavanje slika u mrežu iskorištena je klasa ImageDataGenerator unutar programa Python [51]. Ta klasa nalazi se unutar sučelja Keras i omogućava kreiranje hrpa (eng. batch) podataka koje se učitavaju u umjetnu neuronsku mrežu, s dodatnom mogućnosti provođenja niza postupaka augmentacije na tim hrpama podataka [51].

ImageDataGenerator ne predstavlja aditivnu operaciju. Učitana hrpa podataka se transformira, te se samo transformirani podaci učitavaju u mrežu. To znači da se ne učitavaju u isto vrijeme i originalni i transformirani podaci [52].

7.2.2. Arhitektura mreže

Za potrebe segmentacije ventrikula koristiti će se U-Net s određenim modifikacijama. Te modifikacije su:

- Smanjeni broj slojeva
- Dodana normalizacija hrpa u slojevima kod puta kontrakcije
- Upotrebljeni drugi tipovi augmentacija
- Druga funkcija gubitka

U-Net je prvotno zamišljen kao mreža za segmentaciju stanica [32], te se koristila arhitektura navedena u poglavlju 6.4.3. Međutim korištenjem takve arhitekture pojavio se problem pretreniranja u slučaju segmentacije ventrikula.

Mreže se onda treba modificirati prema pravilima navedenim u poglavlju 3.5.1. Nakon testiranja više različitih kombinacija hiperparametara kod treniranja mreže donesen je izbor mreže koja se sastoji od dva sloja na putu kontrakcije, jednog centralnog sloja koji povezuje put kontrakcije i put ekspanzije, te dva sloja na putu ekspanzije (uz preskočne veze koje povezuju slojeve kontrakcije i ekspanzije). Za odabir početnih vrijednosti težinskih faktora izabrana je He inicijalizacija [53]. Ovaj tip inicijalizacije se preferira za upotrebu kod umjetnih neuronskih mreža koje koriste ReLu kao aktivacijsku funkciju [53]. U ovome radu za potrebe normalizacije podataka upotrebljena je normalizacija hrpa. To znači da će se normalizacija odvijati na razini hrpa koje se učitavaju u mrežu na način naveden u poglavlju 7.2.1. Kao veličina hrpe odabrana je vrijednost od šesnaest slika. U slučaju učitavanja veće količine podataka (ili slika većih dimenzija) trebala bi se smanjiti veličina hrpa koje se učitavaju u mrežu zbog ograničenja očvrsja (eng. hardware).

Mreža je testirana i s većim slikama u manjim hrpama te njima odgovarajućim tipovima normalizacije, međutim pokazalo se da manja veličina slika kombinirana s normalizacijom hrpa daje najbolje rezultate jer normalizacija hrpa dodatno pomaže protiv pretreniranja [54]. Za razliku od originalnog rada kod zadatka segmentacije ventrikula nije upotrebljena elastična deformacija kao metoda augmentacije [32]. Za to postoje dva razloga. Prvi razlog je što su se dobri rezultati dobili korištenjem manje ekstremnih metoda augmentacije. Drugi razlog je što ImageDataGenerator od Kerasa ne sadrži elastičnu deformaciju kao jednu od ponuđenih opcija, pa bi bilo nužno kodirati vlastiti generator hrpa podataka. Kreiranje vlastitog generatora hrpa podataka zakompliciralo bi kod vezan za umjetnu neuronsku mrežu, u slučaju da se ova mreža kasnije želi prilagoditi za rješavanje nekog drugog sličnog problema. U originalnom radu kao funkcija gubitka korištena je unakrsna entropija [32]. U slučaju ovog rada primijenjen je tzv. Sørensen–Dice koeficijent (eng. Sørensen–Dice coefficient) [55]. Ovaj gubitak jako je praktičan za primjenu u svrhu određivanja gubitka koji se pojavljuje kod segmentacije nekih slika.

On predstavlja u kolikoj mjeri se preklapaju prava segmentacijska maska (eng. ground truth) i segmentacijska maska predviđena od strane mreže [56]. Budući da se kod segmentacije obavlja klasifikacija svakog zasebnog piksela, te da postoji disbalans između broja traženih piksela i ostalih piksela (segmentirani dio je relativno mali u odnosu na veličinu cijele slike) koristi se otežana verzija funkcije gubitka temeljena na Dice koeficijentu (eng. Dice coefficient) [57]. Na kraju će se gledati srednja vrijednost preklapanja preko unije (eng. mean IOU) kao metoda koja se najčešće koristi kod zadataka segmentacije kao etalon za provjeru toga koliko dobro neka mreža obavlja zadatak semantičke segmentacije [58]. Kao algoritam za optimizaciju izabran je algoritam Adam naveden u poglavlju 3.4.3. u ovome radu. Faktor učenja je izabran nakon provođenja nekoliko testiranja kao 0.000005.

7.2.3. Dice koeficijent i mIOU

Dice koeficijent i mIOU mogu se predstaviti preko vrijednosti istinski pozitivnih rezultata (eng. True Positive), istinski negativnih rezultata (eng. True Negative) i lažno negativnih rezultata (eng. False Negative). Preko ove tri vrijednosti može se definirati gubitak upotrebljen za treniranje umjetne neuronske mreže.

$$Dice = \frac{2TP}{2TP + TN + FN} \quad \textbf{(9)}$$

$$IOU = \frac{TP}{TP + TN + FN} \quad \textbf{(10)}$$

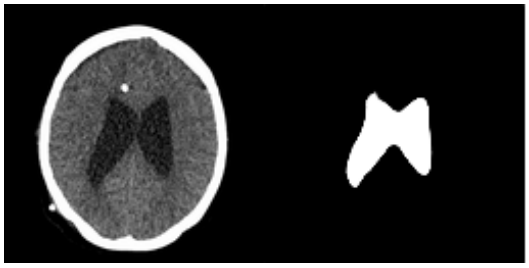
TP znači da je za piksel određeno da spada u područje koje se želi segmentirati, te da je to uistinu tako. TN znači da je za piksel određeno da ne spada u područje koje se želi segmentirati, te da je to uistinu tako. FN znači da je za piksel određeno da ne spada u područje koje se želi segmentirati, iako on zapravo spada u to područje.

7.2.4. Rezultati

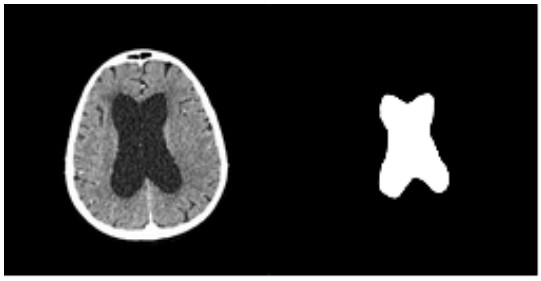
Mreža je trenirana u dvadeset epoha (eng. epoch), gdje svaka epoha predstavlja jednu iteraciju kroz sve podatke koji se koriste za treniranje. Treniranje je zaustavljeno nakon dvadeset epoha zbog toga što su rezultati počeli fluktuirati, što znači da bilo kakvo daljnje treniranje dovelo do gubitka u točnosti rada mreže.

Podaci iz epoha uključeni su u prilozima ovom radu. Mreža je uspješno istrenirana na visoku razinu točnosti. Gubitak je manji od jedan posto (0.05763), a vrijednost mIOU je devedeset i tri posto (0.9332). Mreža je trenirana s dvadeset i jednom slikom u skupu podataka za treniranje (eng. training set) i šest slika u skupu podataka za potvrđivanje (eng. validation set).

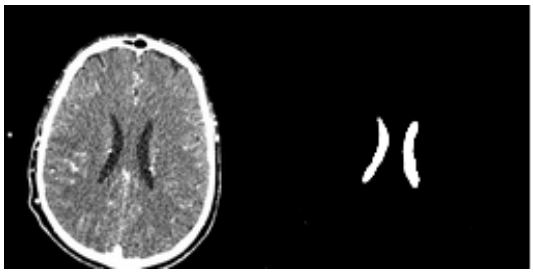
Za usporedbu se može reći da je originalni U-Net je treniran na 30 slika [32]. Slike za treniranje i validaciju obuhvaćaju tri pacijenta. Mreža je testirana na slikama uzetim od pacijenata koji nisu bili uključeni niti u skup podataka za treniranje niti u skup podataka za potvrđivanje.



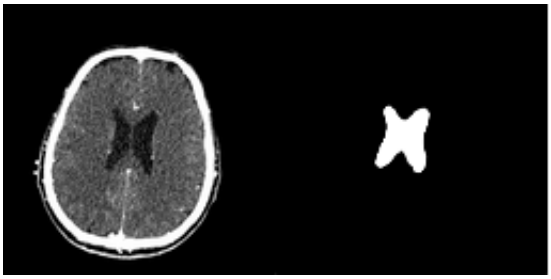
Slika 39. Prvi primjer segmentacije



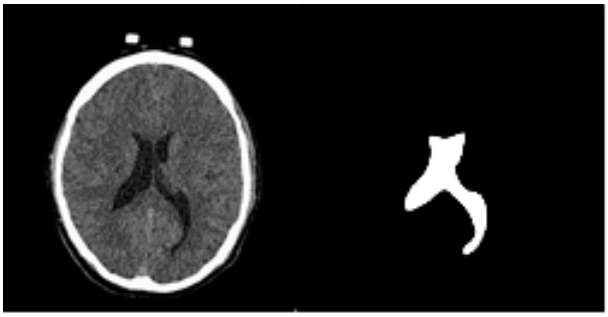
Slika 40. Drugi primjer segmentacije



Slika 41. Treći primjer segmentacije

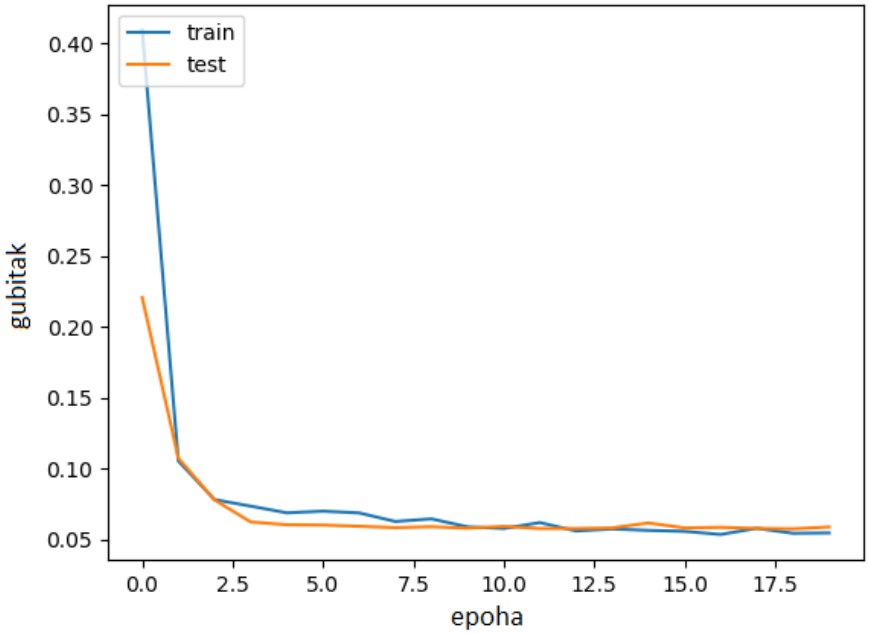


Slika 42. Četvrti primjer segmentacije

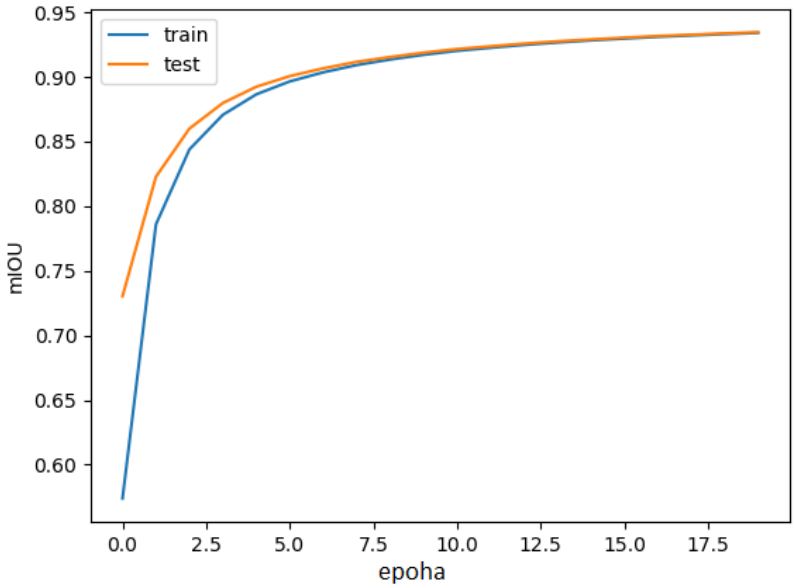


Slika 43. Peti primjer segmentacije

Rezultati se mogu i vizualno prikazati kroz dva grafa. Jedan graf prikazuje promjenu vrijednosti gubitka, a drugi graf prikazuje promjenu vrijednosti mIOU. U oba grafa promjena tih vrijednosti prati se u usporedbi s epohama treniranja.



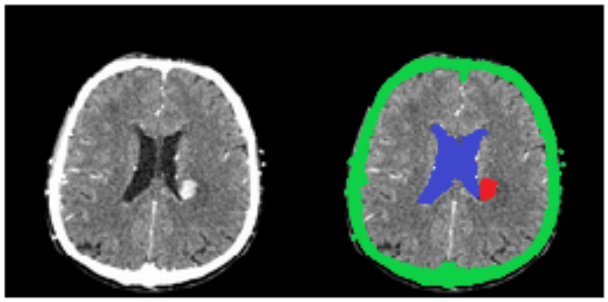
Slika 44. Promjena gubitka kroz epohe



Slika 45. Promjena mIOU kroz epohe

7.2.5. Potpuna segmentacija CT snimaka

Osim segmentacije ventrikula, na koju se fokusira ovaj rad, sprovedena je i segmentacija cijele CT snimke; korištenjem iste neuronske mreže obavilo se skidanje lubanje (eng. skull-stripping) i segmentacija tumora. Iako su dobiveni jako dobri rezultati (oko 90% točnosti) na tim segmentacijama, uzorak pacijenata korišten za treniranje i testiranje nije dovoljno velik kako bi se moglo sa sigurnošću reći da on predstavlja sve moguće primjere tumora i lubanja koje se mogu pojaviti na CT snimkama. Primjer takve segmentacije može se vidjeti na slici 46.



Slika 46. Potpuna segmentacija CT snimke

8. ZAKLJUČAK

Umjetne neuronske mreže nalaze široku primjenu u različitim poljima ljudskih djelatnosti. Jedno od tih polja je i medicina. Napretkom medicinske tehnologije poboljšava se kvaliteta života ljudi, te se pomaže mnogima kojima se nekada ne bi moglo pomoći. S vremenom se pojavio neočekivani problem; različiti operativni postupci koje liječnici obavljaju u današnje vrijeme zahtijevaju sve veću i veću preciznost te predoperativnu pripremu koju je jako teško obaviti ljudima bez neke dodatne pomoći. Ovaj rad posvećen je korištenju umjetnih neuronskih mreža za pronalazak ventrikula na CT snimkama, te sukladno tome obavljanje segmentacije istoga. Budući da se ventrikul ne pojavljuje na svim režnjevima dobivenim snimanjem pacijenta bitno je točno identificirati režnjeve na kojima se nalazi ventrikul, te onda obaviti segmentaciju ventrikula na istima. Ovim radom dokazano je kako se taj zadatak može obaviti određenim modifikacijama mreže po imenu U-Net koja je originalno dizajnirana za značajno drugačiji zadatak segmentacije od onoga u ovome radu. Treniranjem mreže te kasnijim testiranjem postigla se konkurentna razina točnosti. Rezultatima ovoga rada potvrđuje se mogućnost korištenja umjetnih neuronskih mreže za analizu CT snimaka, što znači da bi se u budućnosti moglo pristupiti i rješavanju sličnih problema kojima dosad nije pridavano previše pažnje (npr. pokušaj segmentacije tumora na CT snimkama, a ne na MR snimkama).

LITERATURA

[1] Burkov, Andriy, The hundred page machine learning book

[2] Davison, Joe, No, Machine Learning is not just glorified Statistics, Towards Data Learning
https://towardsdatascience.com/no-machine-learning-is-not-just-glorified-statistics-26d3952234e3

[3] J. Bengio, I. J.Goodfellow, A.Courville, Deep Learning, MIT Press, 10.11. 2016.

[4] The Scientist, A Primer: Artificial Intelligence Versus Neural Networks
https://www.the-scientist.com/magazine-issue/artificial-intelligence-versus-neural-networks-65802

[5] A. Dertat, Applied Deep Learning - Part 1: Artificial Neural Networks,Towards Data Learning
https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6

[6] Overfitting in Machine Learning: What It Is and How to Prevent It, Elite Data Science,2019.
https://elitedatascience.com/overfitting-in-machine-learning

[7] Underfitting, Data Robot,2019.
https://www.datarobot.com/wiki/underfitting/

[8] A Begginer 's Guide to Neural Networks nad Deep Learning, Skymind, 2019.
https://skymind.ai/wiki/neural-network

[9] D.Brezak,T.Staroveški,ROS predavanja, Fakultet Strojarstva i Brodogradnje, 2018.

[10] Neuron, Wikipedija, 2019.
https://hr.wikipedia.org/wiki/Neuron

[11] B.D. Bašić, M. Čupić, J. Šnajder, Umjetne neuronske mreže, predavanja, Fakultet Elektrotehnike i Računarstva, Zagreb svibanj 2008.

[12] prof.dr.sc. B. Dalbelo Bašić, mr.sc. M. Čupić, mr.sc. J. Šnajder, Umjetna Inteligencija, Fakultet Elektrotehnike i Računarstva, Zagreb, svibanj 2008.
https://www.aes.hr/_download/repository/04-Perceptron-1s.pdf

[13] Prof. dr.sc. S. Lončarić, Neuronske mreže: Višeslojni perceptron, Fakultet Elektrotehnike i Računarstva, 2019.
https://www.aes.hr/_download/repository/06-ViseslojniPerceptron-1s.pdf

[14] S. Banerjee, An Introduction to Reccurent Neural Networks, Machine Learning, 23.05.2018.
https://medium.com/explore-artificial-intelligence/an-introduction-to-recurrent-neural-networks-72c97bf0912

[15] C. Maklin, LSTM Reccurent Neural Network Keras Example, Towards Data Science, 14.07.2019.
https://towardsdatascience.com/machine-learning-recurrent-neural-networks-and-long-short-term-memory-lstm-python-keras-example-86001ceaaebc

[16] Wikipedia, Reccurent Neural Network, 2019.
https://en.wikipedia.org/wiki/Recurrent_neural_network#Applications

[17] J. Brownlee, Loss and Loss Functions for Training Deep Learning Neural Networks, Machine Learning Mastery, 23.10.2019.
https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/

[18] Lazy Programmer, Deep Learning Tutorial part 3: Deep Belief Networks, 15.06.2015. https://lazyprogrammer.me/tag/restricted-boltzmann-machines/

[19] S.Shalew-Schwartz, S. Ben-David, Understanding machine learning, From Theory to Algorithms, Cambridge University Press, 2014.

[20] K. Kurita, An Overview of Normalization Methods in Deep Learning, Machine Learning Explained, 30.11.2018. https://mlexplained.com/2018/11/30/an-overview-of-normalization-methods-in-deep-learning/

[21] S. Saha, A Comprehensive Guide to Convolutional Neural Networks - the ELI5 way, Towards Data Science, 15.12.2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[22] Shashikant, Convolutional Neural Network: A Step By Step Guide, Towards Data Science, 17.03.2019. https://towardsdatascience.com/convolutional-neural-network-a-step-by-step-guide-a8b4c88d6943

[23] D. Cornelisse, An intuitive guide to Convolutional Neural Networks, freeCode Camp, 24.04.2018. https://www.freecodecamp.org/news/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050/

[24] Missinglink.ai, Image Segmentation in Deep Learning: Methods and Applications, Computer Vision, 2019. https://missinglink.ai/guides/computer-vision/image-segmentation-deep-learning-methods-applications/

[25] Prof.dr.sc. S. Lončarić, Segmentacija slike, Fakultet Računarstva i Elektrotehnike, 2019. https://www.fer.unizg.hr/_download/repository/09-OI-SegmentacijaSlike%5B2%5D.pdf

[26] C.Yu, G. Ruppert, C. Robert, N. Dan, X. Alexandre, L. Yanxi, 3D Blob based Brain Tumor Detection and Segmentation in MR Images, Research Gate, 29.06.2014. https://www.researchgate.net/figure/A-case-15-original-image-showing-2-tumors-in-a-representative-slice-B-the_fig5_259482926

[27] R. Mina, Y. Haojin, M. Christoph, Instance Tumor Segmentation using Multitask Convolutional Neural Network, Research Gate, 08.06.2018. https://www.researchgate.net/figure/Instance-tumor-segmentation-result-by-proposed-method-on-the-2D-slice-of-abdomen-CT-of_fig4_326931235

[28] O. Arthur, Review of Deep Learning Algorithms for Image Segmentation, Medium, 11.12.2018. https://medium.com/@arthur_ouaknine/review-of-deep-learning-algorithms-for-image-semantic-segmentation-509a600f7b57

[29] L.Jonathan, S. Shelhamer, D. Trevor, Fully Convolutional Networks for Semantic Segmentation, UC Berkeley, 07.06.2015. https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

[30] L.Wei, R. Andrew, C.B. Alexander, ParseNet: Looking Wider to See Better, Cornell University, 15.06.2015. https://arxiv.org/pdf/1506.04579.pdf

[31] S. Tsang, Review: ParseNet — Looking Wider to See Better (Semantic Segmentation),Medium, 17.11.2018. https://medium.com/datadriveninvestor/review-parsenet-looking-wider-to-see-better-semantic-segmentation-aa6b6a380990

[32] O.Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional Networks for Biomedical Image Segmentation, University of Freiburg, Germany, 18.05.2015.
https://arxiv.org/pdf/1505.04597.pdf

[33] T.Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature Pyramid Networks for Object Detection, Cornell University, 19.04.2017.
https://arxiv.org/pdf/1612.03144.pdf

[34] Z. Hensuang, S. Jianping, Q. Xiaojuan, W. Xiaogang, J. Jiaya, Pyramid Scene Parsing Network, The Chinese University of Hong Kong, 27.04.2017.
https://arxiv.org/pdf/1612.01105.pdf

[35] H. Kaiming, Z. Xiangyu, R.Shaoqing, S. Jian, Deep Residual Learning for Image Recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 27.06.2016.
https://arxiv.org/pdf/1512.03385.pdf

[36] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, Facebook AI Research (FAIR), 24.01.2018.
https://arxiv.org/pdf/1703.06870.pdf

[37] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature Pyramid Networks for Object Detection, Facebook AI Research (FAIR), 19.04.2017.
https://arxiv.org/pdf/1612.03144.pdf

[38] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs, IEEE Transactions on Pattern Analysis and Machine Learning Intelligence,Volume 40, Issue 4, 01.04.2018.
https://arxiv.org/pdf/1606.00915.pdf

[39] Sv. Katarina, Specijalna Bolnica
https://www.svkatarina.hr/hr/centar-za-radiologiju-i-dijagnostiku/kompjutorizirana-tomografija

[40] CT Scan in Dallas, TX, What is Computed Tomography, Southwest Diagnostic Center for Molecular Imaging
https://swdcmi.com/diagnostic-services/computed-tomography/

[41] E.Swanson, CT Imaging, Openwetware, 2019.
https://openwetware.org/wiki/CT_Imaging,_by_Elizabeth_Swanson

[42] Xoran products, MiniCAT^{IQ}, Xoran Technologies LLC, 2019.
https://xorantech.com/products/minicat/

[43] DICOM: Digital Imaging and Communications in Medicine, DICOM, DICOM current edition, 2019.
https://www.dicomstandard.org/

[44] Wikipedia, DICOM, 2019.
https://en.wikipedia.org/wiki/DICOM#Image_display

[45] NEMA: „Digital Imaging and Communications in Medicine (DICOM) - Part 1: Introduction and Overview“, NEMA, Rosslyn, USA, 2009.

[46] OsiriX DICOM Viewer, 2019.
https://www.osirix-viewer.com/

[47] Wikipedia, TIFF, 2019.
https://en.wikipedia.org/wiki/TIFF

[48] Z. Hussain, F.Gimenez, D. Yi, D. Rubin, Differential Data Augmentation Techniques for Medical Imaging Classification Tasks, AMIA Annual Symposium Proceedings Archive, 16.04.2018.
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5977656/

[49] J. Tol, Image Augmentation: How to overcome small radiology datasets?, Quantib, 28.06.2019.
https://www.quantib.com/blog/image-augmentation-how-to-overcome-small-radiology-datasets

[50] B. Nikhil, Image Data Pre-Processing for Neural Networks, Becominghuman.ai, 11.09.2017.
https://becominghuman.ai/image-data-pre-processing-for-neural-networks-498289068258

[51] Keras Documentation, Image Preprocessing, Keras.io, 2019.
https://keras.io/preprocessing/image/

[52] A. Rosebrock, Keras ImageDataGenerator, Pyimagesearch, 08.07.2019.
https://www.pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/

[53] S. Yadav, Weight Initialization Techniques in Neural Networks, Towards Data Science, 09.11.2018.
https://towardsdatascience.com/weight-initialization-techniques-in-neural-networks-26c649eb3b78

[54] FD, Batch Normalization in Neural Networks, Towards Data Science, 20.10.2017.
https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c

[55] Wikipedia, Sørensen–Dice coefficient, 2019.
https://en.wikipedia.org/wiki/S%C3%B8rensen%E2%80%93Dice_coefficient

[56] J. Jordan, An overview of semantic image segmentation, Data Science, 21.05.2018. https://www.jeremyjordan.me/semantic-segmentation/#loss

[57] H. Phan, M. Krawczyk-Becker, T. Gerkmann, and A. Mertins, DNN And CNN With Weighted and Multi-task Loss Functions for Audio Event Detection, University of Lubeck, Institute for Signal Processing, Lubeck, Germany and University of Hamburg, Department of Informatics, Hamburg, Germany, 18.10.2017. https://arxiv.org/pdf/1708.03211.pdf

[58] D. Higham, How we use image semantic segmentation, Digital Bridge, 21.02.2018. https://medium.com/digitalbridge/how-we-use-image-semantic-segmentation-e85fac734caf

PRILOZI

Prilog 1: Arhitektura mreže

```
def Hook_net_ventrikul_2 (input_size=(128, 128, 1)):
  nb_filter = [128, 256, 512]

  inputs = Input(input_size)

  down0 = Conv2D(nb_filter[0], 3, padding='same', kernel_initializer='he_normal')(inputs)
  down0 = BatchNormalization()(down0)
  down0 = Activation('relu')(down0)
  down0 = Conv2D(nb_filter[0], 3, padding='same', kernel_initializer='he_normal')(down0)
  down0 = BatchNormalization()(down0)
  down0 = Activation('relu')(down0)
  down0_pool = MaxPooling2D((2, 2), strides=(2, 2))(down0)

  down1=Conv2D(nb_filter[1],3,padding='same',kernel_initializer='he_normal')(down0_pool)
  down1 = BatchNormalization()(down1)
  down1 = Activation('relu')(down1)
  down1 = Conv2D(nb_filter[1], 3, padding='same', kernel_initializer='he_normal')(down1)
  down1 = BatchNormalization()(down1)
  down1 = Activation('relu')(down1)
  down1_pool = MaxPooling2D((2, 2), strides=(2, 2))(down1)

  center=Conv2D(nb_filter[2],3,padding='same',kernel_initializer='he_normal')(down1_pool)
  center = Activation('relu')(center)
  center = Conv2D(nb_filter[2], 3, padding='same', kernel_initializer='he_normal')(center)
  center = Activation('relu')(center)
```

```
up1 = UpSampling2D((2, 2))(center)
up1 = concatenate([down1, up1], axis=3)
up1 = Conv2D(nb_filter[1], 3, padding='same', kernel_initializer='he_normal')(up1)
up1 = Activation('relu')(up1)
 up1 = Conv2D(nb_filter[1], 3, padding='same', kernel_initializer='he_normal')(up1)
 up1 = Activation('relu')(up1)
 up1 = Conv2D(nb_filter[1], 3, padding='same', kernel_initializer='he_normal')(up1)
 up1 = Activation('relu')(up1)

 up0 = UpSampling2D((2, 2))(up1)
 up0 = concatenate([down0, up0], axis=3)
 up0 = Conv2D(nb_filter[0], 3, padding='same', kernel_initializer='he_normal')(up0)
 up0 = Activation('relu')(up0)
 up0 = Conv2D(nb_filter[0], 3, padding='same', kernel_initializer='he_normal')(up0)
 up0 = Activation('relu')(up0)
 up0 = Conv2D(nb_filter[0], 3, padding='same', kernel_initializer='he_normal')(up0)
 up0 = Activation('relu')(up0)

 up_last = Conv2D(1, 1, activation='sigmoid')(up0)

 model = Model(inputs=inputs, outputs=up_last)

 model.compile(optimizer=Adam(lr=0.000005),loss=weighted_dice_loss,
 metrics=[mean_iou])

 model.summary()
 return model
```

Prilog 2: Generatori hrpa

```
image_generator = image_datagen.flow_from_directory(
train_path,
classes=[image_folder],
class_mode=None,
color_mode=image_color_mode,
target_size=target_size,
batch_size=batch_size,
save_to_dir=save_to_dir,
save_prefix=image_save_prefix,
seed=seed)

mask_generator = mask_datagen.flow_from_directory(
train_path,
classes=[mask_folder],
class_mode=None,
color_mode=mask_color_mode,
target_size=target_size,
batch_size=batch_size,
save_to_dir=save_to_dir,
save_prefix=mask_save_prefix,
seed=seed)

train_generator = zip(image_generator, mask_generator)
for (img, mask) in train_generator:
img, mask = adjustData(img, mask, flag_multi_class, num_class)
yield (img, mask)
```

Prilog 3: Augmentacija podataka

```
augmentations = dict(rotation_range=0.1,
                     width_shift_range=0.05,
                     height_shift_range=0.05,
                     shear_range=0.05,
                     zoom_range=0.05,
                     horizontal_flip=True,
                     fill_mode='nearest')
```