Segmentacija plovila iz satelitskih snimaka

Paladin, Luka

Master's thesis / Diplomski rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Rijeka, Faculty of Engineering / Sveučilište u Rijeci, Tehnički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:190:355397

Rights / Prava: Attribution 4.0 International/Imenovanje 4.0 međunarodna

Download date / Datum preuzimanja: **2024-07-24**

Repository / Repozitorij:

Repository of the University of Rijeka, Faculty of Engineering

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij elektrotehnike

Diplomski rad

SEGMENTACIJA PLOVILA IZ SATELITSKIH SNIMAKA

Rijeka, srpanj 2022.

Luka Paladin

0069070400

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Diplomski sveučilišni studij elektrotehnike

Diplomski rad

SEGMENTACIJA PLOVILA IZ SATELITSKIH SNIMAKA

Mentor: Prof. dr. sc. Zlatan Car

Rijeka, srpanj 2022.

Luka Paladin

0069070400

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
POVJERENSTVO ZA DIPLOMSKE ISPITE

Rijeka, 21. rujna 2021.

Zavod: **Zavod za automatiku i elektroniku**
Predmet: **Osnove robotike**
Grana: **2.03.06 automatizacija i robotika**

ZADATAK ZA DIPLOMSKI RAD

Pristupnik: **Luka Paladin (0069070400)**
Studij: Diplomski sveučilišni studij elektrotehnike
Modul: Automatika

Zadatak: **Segmentacija plovila iz satelitskih snimaka / Segmentation of vessels from satellite images**

Opis zadatka:

Predstaviti pregled dosadašnjih istraživanja iz područja segmentacije objekata sa satelitskih snimaka. Realizirati sustav za segmentaciju plovila sa satelitskih snimki primjenom algoritama temeljenih na konvolucijskim neuronskim mrežama. Komentirati dobivene rezultate i usporediti performanse različitih algoritama.

Rad mora biti napisan prema Uputama za pisanje diplomskih / završnih radova koje su objavljene na mrežnim stranicama studija.

Zadatak uručen pristupniku: 21. rujna 2021.

Mentor:

Prof. dr. sc. Zlatan Car

Predsjednik povjerenstva za diplomski ispit:

Prof. dr. sc. Viktor Sučić

IZJAVA

Izjavljujem da sam diplomski rad naslova „Segmentacija plovila iz satelitskih snimaka“ iz kolegija Osnove robotike izradio samostalno prema uputama za pisanje diplomskog rada koristeći navedenu literaturu i znanje stečeno tijekom studiranja na Tehničkom fakultetu u Rijeci uz konzultacije s mentorom: prof. dr. sc. Zlatanom Carom.

Rijeka, srpanj 2022.

Luka Paladin
0069070400

ZAHVALA

Ovom prilikom zahvaljujem se prof. dr. sc. Zlatanu Caru na mentorstvu i podršci pri izradi diplomskog rada.

Također zahvaljujem asistentu asist. dr. sc. Ivanu Lorencinu na ukazanom razumijevanju, strpljenju i neophodnoj pomoći tokom izrade ovog završnog rada.

Zahvaljujem se i svojim roditeljima što su mi bili podrška i oslonac tijekom godina studiranja te na financijskoj pomoći bez koje to ne bi bilo ostvarivo.

SADRŽAJ

1. UVOD 1

2. SEGMENTACIJA SLIKE 2

2.1. Mehanizam segmentiranja slike 3

2.2. Primjene segmentacije 4

3. PREGLED DOSADAŠNJIH ISTRAŽIVANJA 6

3.1. Matematičke metode 6

3.1.1. Segmentacija na temelju intenziteta 6

3.1.2. Segmentacija na temelju rubova 7

3.1.3. Segmentacija regija 8

3.1.4. Segmentacija grupiranjem 10

3.2. Metode temeljene na umjetnim neuronskim mrežama 11

3.2.1. Konvolucijske neuronske mreže (CNN) 12

3.2.2. RCNN 15

3.2.3. FCN 16

3.2.4. PSPNet 17

3.2.5. U-net 18

3.2.6. Segnet 19

3.2.7. E-net 20

3.2.8. DeepLabv3+ 20

3.2.9. Ostale metode 21

4. SEMANTIČKA SEGMENTACIJA OBJEKATA IZ SATELITSKIH SNIMAKA 23

4.1. Skup podataka 24

5. REALIZACIJA MODELA BAZIRANOG NA CNN-U 27

5.1. Priprema slikovnih podataka 27

5.2. Treniranje modela i preliminarni rezultati 30

5.3. Odabir funkcije gubitaka 35

5.4. Ostali parametri 36

6. REZULTATI 38
6.1. Usporedba modela 38
6.2. Usporedba težinskih funkcija 39
6.3. Usporedba seta ulaznih slika 40
6.4. Usporedba broja epoha treniranja 41
6.5. Konačni rezultati 42
7. ZAKLJUČAK 44
LITERATURA 45
SAŽETAK 48
ABSTRACT 48
DODATAK A - Programski kod za pripremu ulaznog seta podataka 49
DODATAK B - Programski kod za treniranje modela 52

1. UVOD

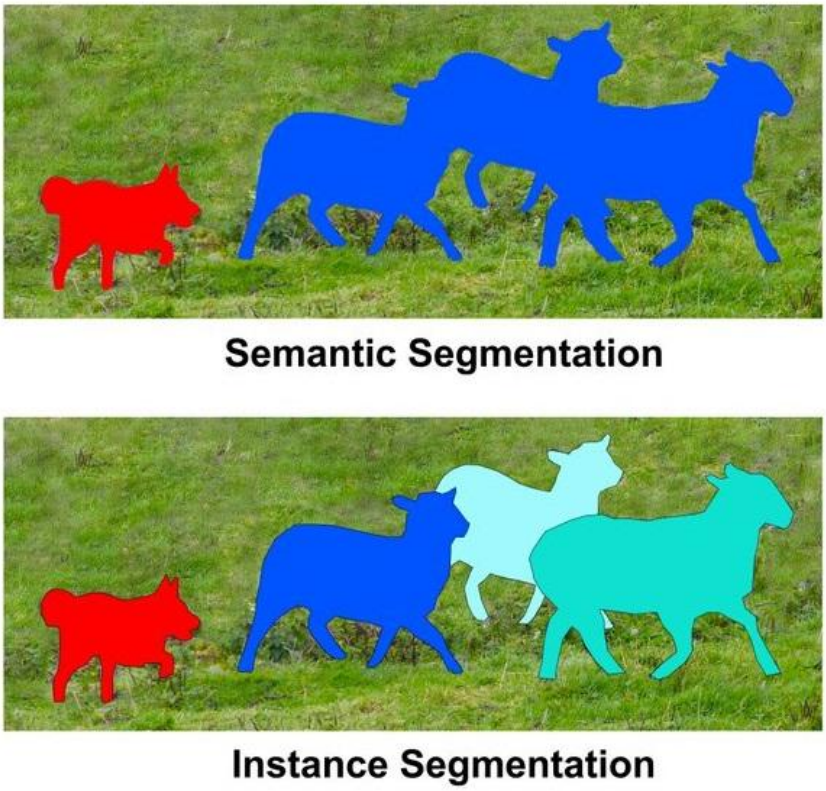
Napretkom tehnologija suvremenog doba sve se više zadataka obrade, analiziranja i razumijevanja digitalnih slika pokušava automatizirati korištenjem računala, takozvanim računalnim vidom i za određene zadatke primjenom algoritama umjetne inteligencije. Takvi algoritmi pri radu ne zahtijevaju ljudsku interakciju, a omogućavaju osjetno (za nekoliko redova veličine) brže izvršenje zadatka. Problemi računalnog vida koji se odnose na analizu slikovnih podataka kao što su klasifikacija slika, detekcija objekata te segmentacija objekata s digitalnih slika pokazali su se vrlo zahtjevnim. Takvi problemi zahtijevaju prilagodljive i brze algoritme visoke točnosti koji također zahtijevaju veliku količinu kvalitetnih slikovnih podataka za učenje. Postoje različiti algoritmi i postupci pripreme podataka, s prednostima i nedostatcima, koji se neprestano usavršavaju. Također se neprestano razvijaju nove metode. Razvoj tih metoda kroz vrijeme predstavit će se u ostatku rada.

Zadatak koji se u ovom radu opisuje i rješava je semantička segmentacija plovila iz satelitskih snimaka korištenjem dubokih neuronskih mreža. Segmentacija objekata odnosi se na proces pridjeljivanja semantičkih oznaka dijelovima slika, tj. pikselima slike. To podrazumijeva da se na slici precizno opisuje točno koji pikseli pripadaju traženom objektu. Kako se taj zadatak rješava korištenjem dubokih neuronskih mreža bit će pojašnjeno u ostatku rada. Sljedeće, izraz duboke neuronske mreže odnosi se na umjetne neuronske mreže s velikim brojem slojeva koje se u ovom slučaju primjenjuju za obradu slikovnih podataka.

2. SEGMENTACIJA SLIKE

Segmentacija slike je zadatak računalnog vida u kojem se područja slike kategoriziraju prema objektu koji se na njima nalazi. Točnije, cilj semantičke segmentacije slike je označiti svaki piksel slike odgovarajućom klasom onoga što sačinjava [1]. Budući da digitalna slika nije ništa drugo nego skup piksela, a segmentacija slike je proces razvrstavanja svakog piksela na slici određenoj klasi. Segmentacija se može smatrati problemom klasifikacije po pikselu [2]. Postoje dvije glavne tehnike segmentacije: semantička segmentacija i segmentacija instanci.

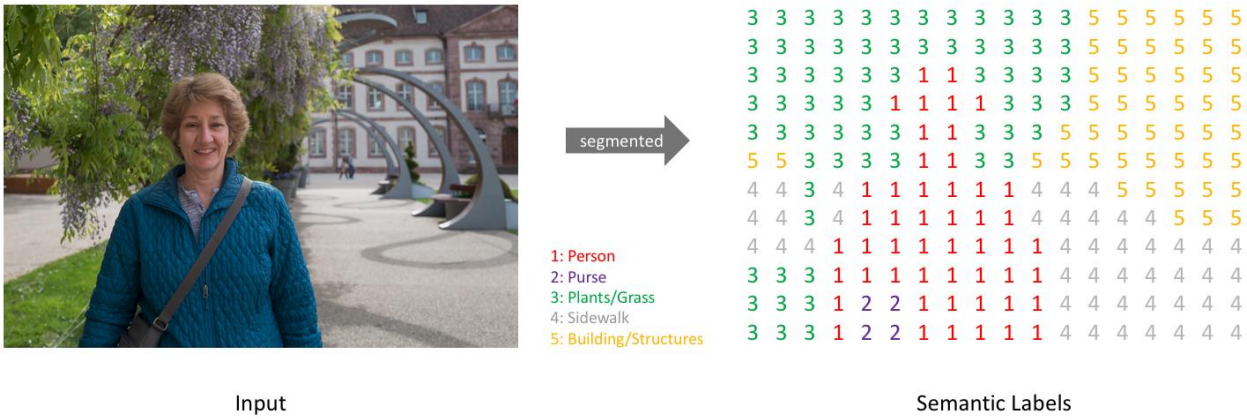
Korištenjem tehnike semantičke segmentacije odvojene se detekcije iste klase ne odvajaju, već se sve detekcije iste klase jednako kategoriziraju. Drugim riječima, ako na ulaznoj slici postoje dva objekta iste kategorije, metoda semantičke segmentacije ne razlikuje ih sama po sebi kao zasebne objekte, već im samo pripisuje kojoj kategoriji pripadaju. Ako se želi postići da se različiti objekti iste kategorije razlikuju, onda je potrebno koristiti se drugom metodom pod nazivom segmentacija instanci koja ima mogućnost razlikovati pojavljivanja odvojenih objekta iste klase na slici [1]. Razlika te dvije metode prikazana je na slici 2.1.



Slika 2.1. Razlika semantičke segmentacije i segmentacije istanci [2]

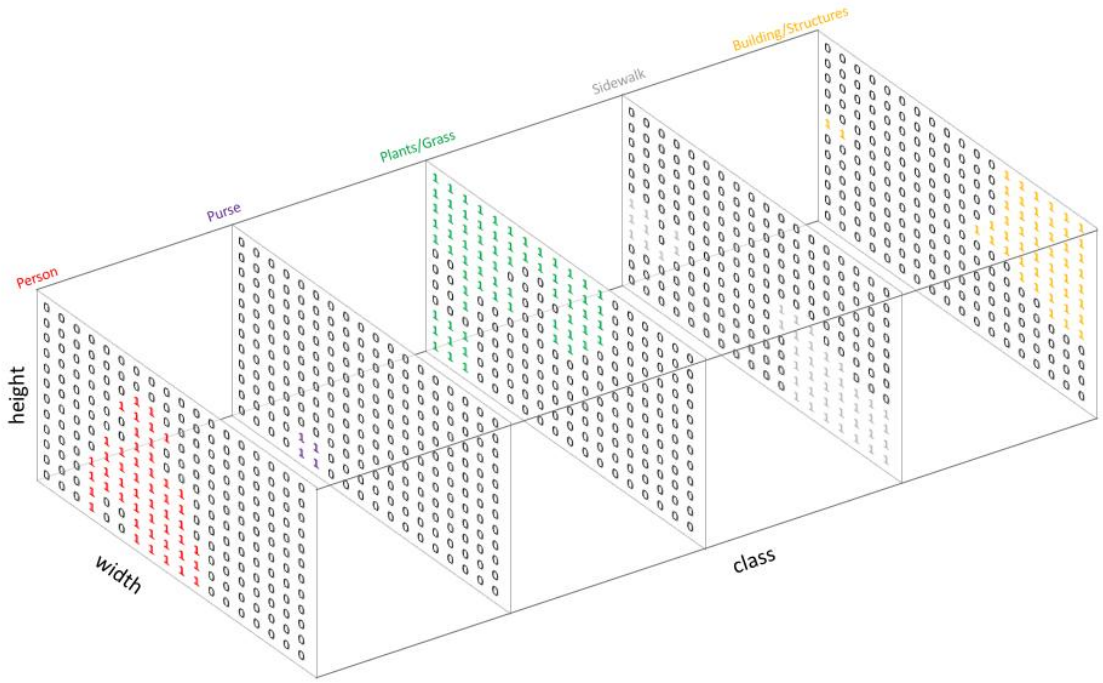
2.1. Mehanizam segmentiranja slike

Cilj segmentiranja je uzevši neku sliku kao ulaz, dobiti segmentacijsku mapu kao izlaz gdje svaki piksel sadrži oznaku klase kojoj pripada, gdje je svaka klasa predstavljena brojem kao što je pojednostavljeno prikazano na slici 2.2.



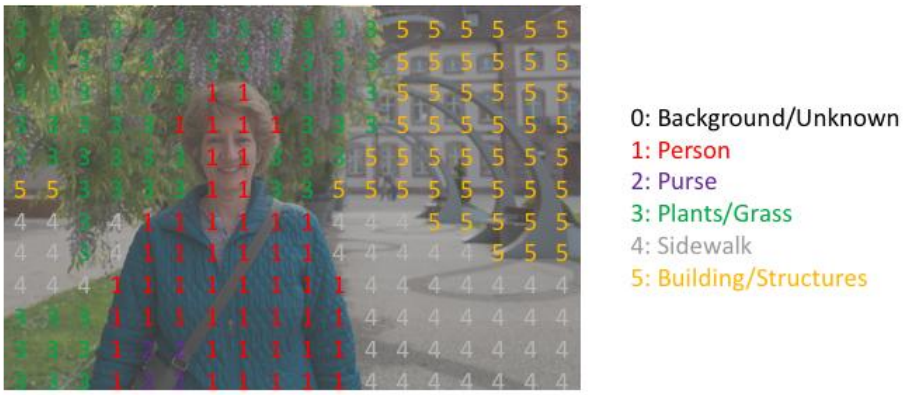
Slika 2.2. Ulazna slika i segmentacijska mapa kao dobiveni izlaz [1]

Slično kao kod standardnih kategorizacijskih metoda, stvara se izlazni kanal za svaku moguću klasu prikazano na slici 2.3. Predikcije se tada mogu sažeti u segmentacijsku mapu ako se uzima maksimalna vrijednost iz vektora vrijednosti za svaki piksel.



Slika 2.3. Izlazni kanali za svaku moguću klasu [1]

Tada se jednostavno može ispitati originalna sliku tako da ju se prekrije sa segmentacijskom mapom. To tada nazivamo maskom koja ističe regije slike gdje se određena klasa nalazi. To je prikazano na slici 2.4.



Slika 2.4. Maska koja ističe regije na slici [1]

2.2. Primjene segmentacije

Tehnika segmentiranja digitalnih slika ima široke primjene u modernom svijetu, neke od njih uključuju:

- autonomna vozila – potrebno je opremiti automobile potrebnom percepcijom kako bi bili u stanju razumjeti njihovo okruženje, identificirati trake, prometne znakove, pješake, ostale sudionike prometa i druge potrebne informacije. Kako bi se time samovozeći automobili mogli sigurno integrirati u naše postojeće ceste [3],

- medicinska dijagnostika - u medicinskom svijetu potreban je veliki broj stručnjaka koji mogu točno izvršiti analize snimaka, kao na primjer rendgenskih ili ultrazvučnih snimaka. Uvođenjem strojeva opremljenim s mogućnošću segmentiranja tih slika moguće je izvršiti većinski dio analize koju provode radiolozi, uvelike smanjujući vrijeme potrebno za pokretanje dijagnosticiranih testova [4],

- Prepoznavanje rukopisa - semantička segmentacija se koristi za izdvajanje riječi i redaka iz rukom pisanih dokumenata kako bi se prepoznali rukom pisani znakovi. To omogućuje brzu transkripciju rukom pisanih dokumenata u digitalni oblik [5],

- Odvajanje objekta od pozadine - Postoje mnogi slučajevi upotrebe u kojima je apsolutno neophodno odvojiti prednji plan od pozadine. Na primjer, u Googleovom portretnom modu stvara se zamagljena pozadina, dok prednji plan ostaje nepromijenjen [6].

- Detekcija oštećenja na proizvodima - Pri proizvodnji različitih proizvoda potrebno je locirati anomalije i oštećenja na proizvodima pri kontroli kvalitete [7].

- Virtualno isprobavanje šminke - Primjena različitih stilova šminke na lice osobe moguća je uz pomoć segmentacije slike [8].

- Virtualno isprobavanje odjeće - Virtualno isprobavanje odjeće zanimljiva je značajka koja je bila dostupna u trgovinama pomoću specijaliziranog hardvera koji stvara 3d model. S primjenom modela dubokog učenja i segmentacijom slike isto se može dobiti pomoću samo 2d slike [9].

- Vizualno pretraživanje slika - Ideja segmentiranja koristi se i u algoritmima za dohvaćanje slika u e-trgovini. Na primjer, prijenos bilo koje slike omogućuje stjecanje ponude srodnih proizvoda sličnog izgleda [10].

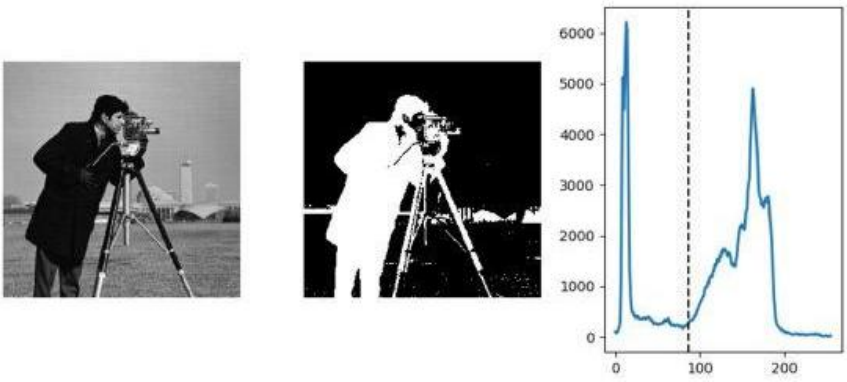
3. PREGLED DOSADAŠNJIH ISTRAŽIVANJA

U ovom poglavlju predstavljen je pregled razvoja metoda rješavanja problematike segmentacije slike od pojave digitalnih slika pa sve do danas. Započinje se kratkim pregledom jednostavnijih matematičkih metoda koje su dobro utvrđene, robusne i koriste se za rješavanje usko specijaliziranih problema. Tim metodama se mogu ostvariti različite vrste segmentiranja. Sljedeće, detaljnije će se prikazati do kakvih je napredaka u tom polju došlo primjenom umjetnih neuronskih mreža (eng. artificial neural networks) te kakve su metode razvijene za rješavanje tog problema.

3.1. Matematičke metode

3.1.1. Segmentacija na temelju intenziteta

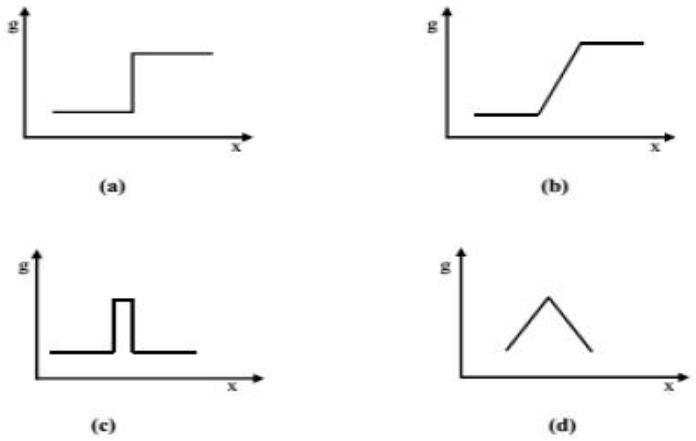
Jedan od najjednostavnijih pristupa segmentaciji je na temelju razine intenziteta i funkcionira kao segmentiranje na temelju praga. Tehnika temeljena na pragu klasificira sliku u dvije klase i radi na kriteriju da pikseli u određenom rasponu vrijednosti intenziteta predstavljaju jednu klasu, a ostatak piksela na slici predstavlja drugu klasu. Pragovi se mogu provesti ili globalno ili lokalno. Globalni prag razlikuje objekt i pozadinske piksele uspoređujući s odabranom graničnom vrijednošću i koristi binarnu particiju za segmentiranje slike. Pikseli čija je vrijednost iznad praga smatraju se objektom i dodjeljuje im se binarna vrijednost "1", dok se ostalim pikselima dodjeljuje binarna vrijednost "0" i tretiraju se kao pozadina. To je vidljivo na slici 3.1. Tehnike segmentacije temeljene na pragu su jednostavne, računalno brze i mogu se koristiti za primjene u stvarnom vremenu uz pomoć specijaliziranog hardvera [11]. Neki od nedostataka te metode su osjetljivost na šum, zanemarivanje prostorne informacije slike te uska primjena metode. Osim metode ručnog definiranja praga, postoje i računalne metode određivanja praga te adaptivne metode koje smanjuju neke od nedostataka.



Slika 3.1. Prikaz rezultata segmentacije na temelju intenziteta [4]

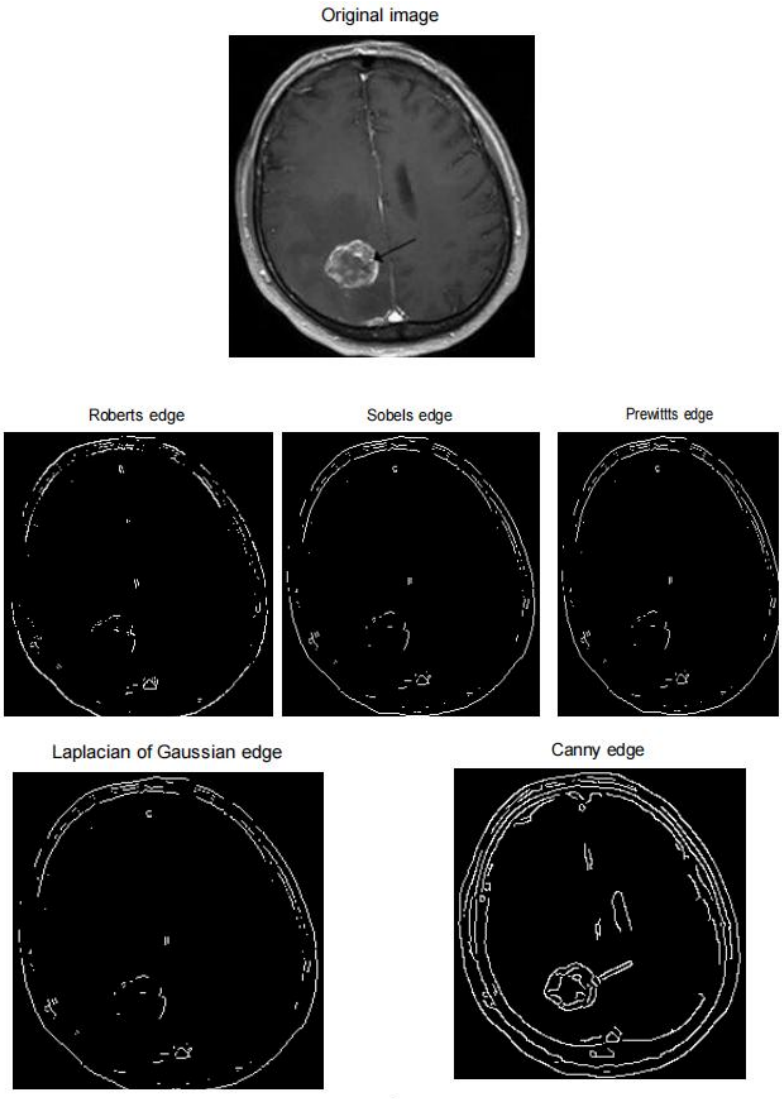
3.1.2. Segmentacija na temelju rubova

Metode segmentiranja slike na temelju rubova baziraju se na principu varijacije intenziteta među pikselima. Značajna promjena intenziteta među susjednim pikselima u određenom smjeru ukazuje na postojanje ruba nekakvog objekta na slici. Ti rubovi dovode do formiranja granica objekta na slici, te se tako objekt može segmentirati. Za pravilnu detekciju rubova potrebno je prvo izgladiti sliku filtrom kako bi se uklonio šum i pripremilo sliku za detekciju rubova pomoću filtra [12]. Postoje 4 osnovne vrste rubova koji se mogu nalaziti na slici. Oni su prikazani na slici 3.2 i to su: step, rampa, impuls i trokut.



Slika 3.2. Četiri osnovne vrste rubova prisutnih na slici [12]

Korištenjem određenih filtara jednostavno se može pronaći prisutnost tih rubova na slici. To se rješava operacijom konvolucije filtra sa slikom. Neki od jednostavnijih operatora su: Robertov, Prewittov i Sobelov operator. Oni koriste određene matrice (filtre) kako bi pronašli rubove objekta na slici u različitim smjerovima te njihovim spajanjem dobilo kompletne rubove objekta. Neki od naprednijih operatora su: Laplaceov, Gaussov i Canny operator. Laplaceov i Gaussov operator funkcioniraju na sličan način i traže istaknute regije s naglom promjenom intenziteta, što upućuje na postojanje rubova. Canny operator smatra se superiornijim operatorom za detekciju rubova zbog mogućnosti pronalaska slabih i jakih rubova na slici. To postiže zbog višekoračne operacije obrade slike. Na slici 3.3 prikazana je usporedba rezultata prethodno nabrojanih operatora.



Slika 3.3. Usporedba rezultata više operatora segmentiranja slike na temelju rubova [13]

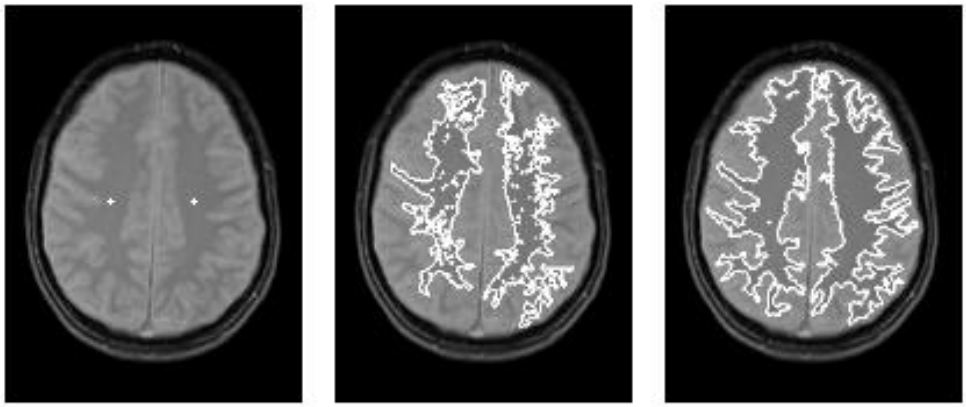
Prednost ove metode su pouzdani rezultati na slikama s dobrim kontrastom između objekata. Također, korisna je za računanje broja različitih objekata na danoj slici. Nedostatak je taj da rubovi objekta na slici moraju biti dovoljno jasni kao bi ih se u cjelini izdvojilo, pa se često događa da rubovi nisu konstantni, što ograničava primjenu te metode na slikama sa šumom. Nadalje, jedan operator ne može odgovarati svim slikama pa neće davati podjednako dobre rezultate bez ugađanja operatora [11].

3.1.3. Segmentacija regija

Ova metoda djeluje po principu homogenosti s obzirom na činjenicu da susjedni pikseli unutar nekih regija posjeduju slične karakteristike i razlikuju se od piksela u drugim regijama. Cilj segmentacije temeljene na regiji je proizvesti homogene regije koje su većih veličina i to rezultira s manjim brojem regija na slici. Najjednostavniji pristup segmentaciji regija bazira se na

pretpostavki sličnosti susjednih piksela, gdje se za svaki piksel provjerava sličnost sa susjednim. Ako je rezultat pozitivan onda se taj piksel dodaje regiji i ona raste [11].

U osnovi postoje dvije metode i to su: metoda uzgoja regija i metoda podjele i spajanja regija. Metoda uzgoja regija daje vrlo pouzdane rezultate. To je u osnovi vađenje regije iz slike koja koristi neke unaprijed definirane kriterije. Najjednostavnije, piksel se uspoređuje sa susjednim tako da se provjerava kriterij homogenosti dodijeljen klasi kojoj taj susjedni piksel pripada. Rezultati takve metode prikazani su na slici 3.4. Metoda podjele i spajanja regija također je bazirana na kriterijima homogenosti, ali radi na drugom principu. Ova metoda u početku cijelu sliku smatra jednom regijom, zatim se slika dijeli na četiri kvadranta na temelju unaprijed definiranih kriterija. Provjerava kvadrante prema definiranom kriteriju i ako je rezultat negativan dijeli ga dalje u četiri nova kvadranta i proces se nastavlja [11].



Slika 3.4. Prikaz rezultata segmentacije primjenom metode segmentacije regija [14]

Prednost metode segmentacije regija je da metoda većinom daje superiorne rezultate u usporedbi s ostalim metodama zato što segmentacija teče iz jedne točke, pa je velika vjerojatnost da će rezultat biti jasna granica objekata, pogotovo ako je dobro odabrano izvorno polazište. Također, metoda radi vrlo dobro i na slikama s umjerenom količinom šuma. S druge strane, dobar rezultat uvelike ovisi i dobrom odabiru izvornog polazišta ili sjemena regije (eng. region seed) i loš odabiri može dovesti do pogrešne segmentacije, pogotovo ako je na slici prisutan popriličan šum. Postupak odabira sjemena zahtijeva ručne intervencije i sklon je pogreškama. Ujedno, postupak je računski i vremenski zahtjevan.

3.1.4. Segmentacija grupiranjem

Grupiranje je proces organiziranja regija na temelju atributa. Cilj grupiranja je identificirati skup u podacima. Grupa obično sadrži skupinu sličnih piksela koji ili pripadaju jednoj regiji ili se dovoljno razlikuju od ostalih zbog karakteristika kao što su oblik, tekstura i slično. Postoje razne tehnike segmentacije grupiranjem, a najčešće su to algoritmi K-srednjih vrijednosti i Fuzzy C-srednjih vrijednosti.

Ti algoritmi funkcioniraju tako da se unaprijed definira broj grupa (K ili C) i iterativno ih se konvergira. Algoritam K-srednjih vrijednosti funkcionira tako da grupira točke najbliže težištu (središtu) koji je u osnovi srednja vrijednost svih točaka u toj grupi. Također, sadrži koordinate koje su aritmetička sredina od svih točaka u grupi, zasebno za svaku dimenziju sustava. Prvi korak je definiranje broja grupa i proizvoljno postavljanje njihovih središta na različita početna mjesta na slici. Kao drugi korak svaka se podatkovna točka dodjeljuje jednoj grupi čije je težište najbliže. Treći korak je novo računanje težišta grupa tako da budu na aritmetičkoj sredini svih podataka koji sačinjavaju tu grupu. Koraci 2 i 3 se iterativno izvršavaju sve dok se novim iteracijama ne dobivaju promjene ili se dostigne maksimalan broj iteracija. Na slici 3.5 je prikaz segmentacije slike grupiranjem na različit broj K-grupa. Algoritam fuzzy C-srednjih vrijednosti se razlikuje po tome da pikseli mogu istovremeno pripadati dvama ili više grupama s različitim stupnjem pripadnosti svakoj grupi [11].

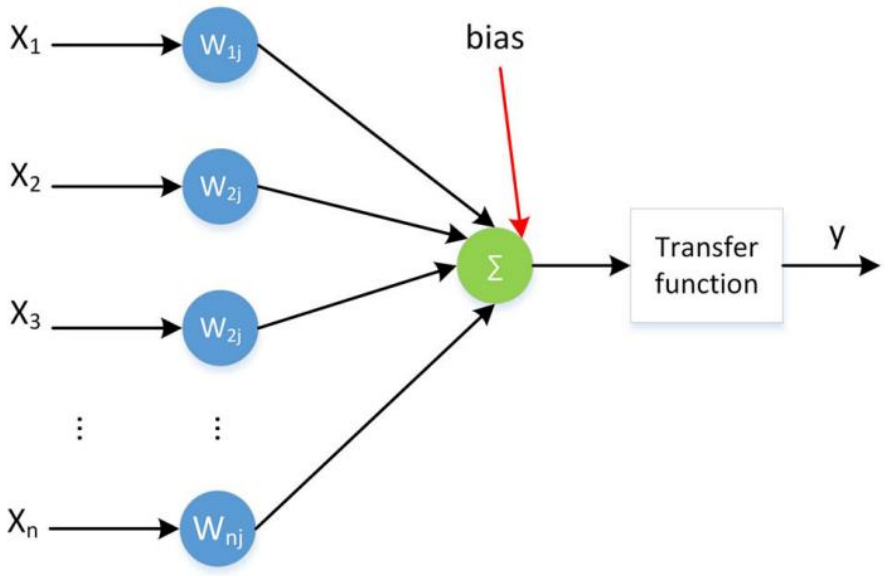
Slika 3.5.Prikaz rezultata segmentacije primjenom metode segmentacije slike grupiranjem na različit broj K-grupa [15]

Navedene metode su dobro dokazane i utemeljene te poprilično korisne za veliki broj primjena. Ali zahtijevaju dobro postavljanje računskih funkcija zbog iterativnog računanja kako bi se ostvarili željeni rezultati.

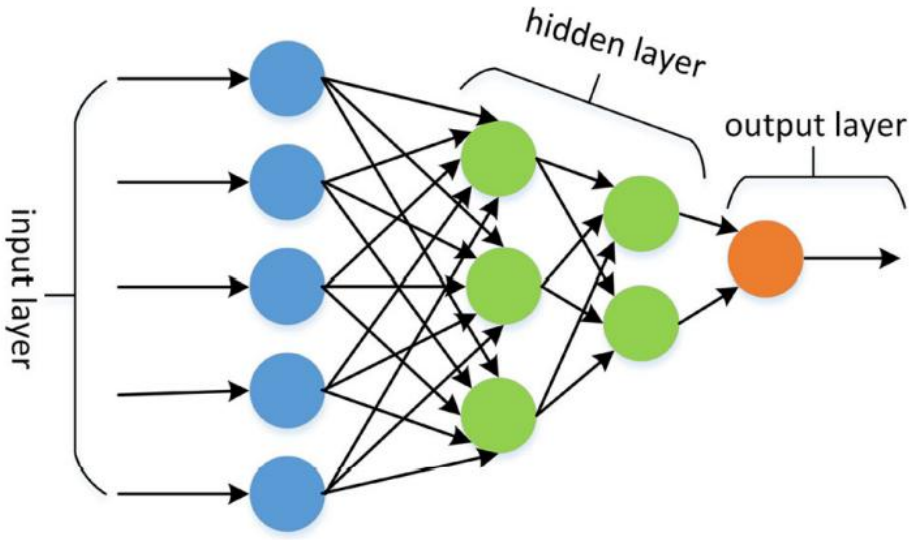
3.2. Metode temeljene na umjetnim neuronskim mrežama

Razvojem umjetnog neurona i time neuronskih mreža, otkrio se potencijal primjene neuronskih mreža sačinjenih od velikog broja neurona za analizu digitalnih slika, posebno konvolucijske neuronske mreže temeljene na algoritmima dubokog učenja. One su jednostavne za implementaciju zbog postojećih dobro utvrđenih knjižnica dostupnih u Pythonu, ali i drugim programskim jezicima te su prilagodljive za različite praktične aplikacije. Nedostatak korištenja neuronskih mreža za rješavanje problema segmentacije objekata na slikama je taj da je treniranje modela za pojedinačne primjene vrlo vremenski i računski zahtjevno [16].

Umjetne neuronske mreže su nelinearne mreže koje pokušavaju imitirati funkcioniranje bioloških neuronskih mreža. Osnovni element je umjetni neuron. On se sastoji od tijela koji se ponaša kao sumator, ulaza u tijelo koji se sumiraju, te izlaza kao rezultata. Uz to svaki od ulaza ima pripisani težinski faktor, a tijelo aktivacijsku funkciju [17]. Prikaz jednog takvog neurona je na slici 3.6. Različitim slaganjem i međusobnim povezivanjem tih jednostavnih neurona dobiva se umjetna neuronska mreža čiji je primjer prikazan na slici 3.7.



Slika 3.6. Prikaz strukture jednog umjetnog neurona [17]



Slika 3.7. Prikaz strukture umjetne neuronske mreže sačinjene od više neurona [17]

Postoji mnogo vrsta neuronskih mreža raznih arhitektura, razina kompleksnosti i primjena. Neke od njih pojašnjene su u nastavku. Kod svih neuronskih mreža se koristi proces učenja za određivanje nepoznatih parametara te mreže kako bi ih se moglo upotrijebiti za postizanje željenih rezultata.

Pristup segmentaciji slika pomoću neuronskih mreža često se naziva prepoznavanjem slike. Metoda koristi umjetnu inteligenciju za automatsku obradu i identifikaciju komponenti slike poput objekata, lica, teksta i slično. Mogu se smatrati klasifikatorima koji izvlače hijerarhijske značajke iz neobrađenih podataka, u ovom slučaju vrijednosti piksela, i uče modele za različite zadatke povezane s vidom, u ovom slučaju segmentacija objekata [18]. Konvolucijske neuronske mreže posebno se koriste za ovaj proces zbog njihova dizajna za obradu slikovnih podataka visoke razlučivosti [16].

3.2.1. Konvolucijske neuronske mreže (CNN)

Konvolucijske neuronske mreže su unaprijedne neuronske mreže koje se sastoje od većeg broja slojeva sa svojstvima koja omogućuju efikasnije učenje i ostvarivanje boljih rezultata kod zadataka iz područja računalnog vida [19]. Neki od tih specijaliziranih slojeva su:

- ulazni sloj (eng. Input layer) predstavlja ulaznu sliku kao matricu određenih dimenzija na način da svaki piksel ima pripisanu numeričku vrijednost koja predstavlja njegov intenzitet,

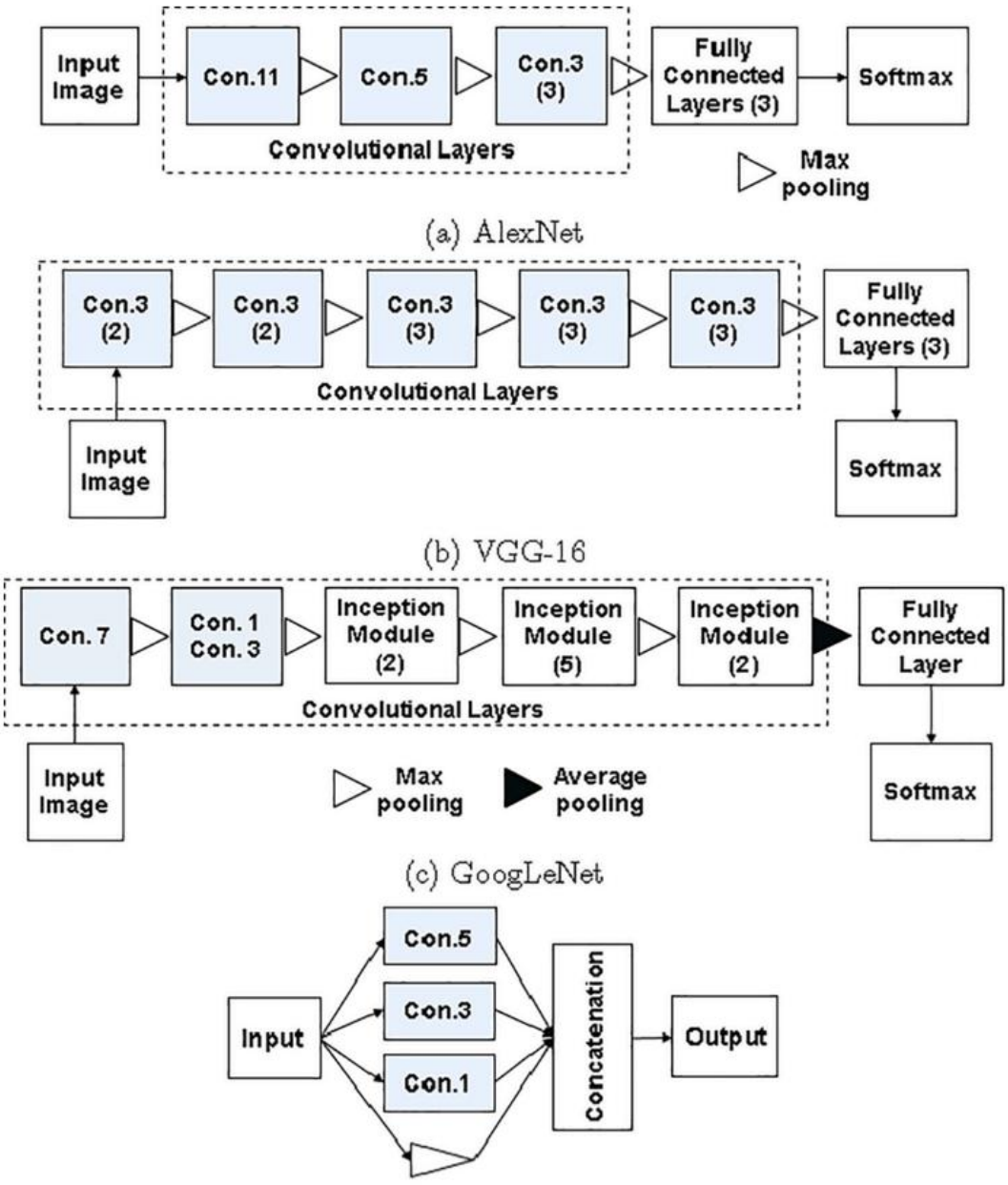
- konvolucijski sloj (eng. Convolution layer) uzima sliku iz prethodnog sloja te ju konvoluira s određenim filtrom, većinom mijenjajući joj dimenzije, i kreira izlaznu sliku tj. mapu značajki,
- aktivacijska funkcija (eng. Activation function), najčešće ReLU funkcija, ne mijenja dimenzije mape značajki i njezina uloga je dodavanje nelinearnosti u skrivene slojeve mreže i uklanjanje zasićenosti gradijenta,
- sloj udruživanja (eng. Pooling layer) uzima mape značajki iz prethodnog sloja i sažima ih na manju dimenziju korištenjem filtra i
- potpuno povezani sloj (eng. Fully connected layer) nalazi se na kraju mreže i sadrži neurone koji su povezani sa svim neuronima iz prethodnog sloja i izlazna jedinica mu je vektor rezultata [20].

Svi Konvolucijski slojevi u mreži zajedno sa aktivacijskom funkcijom i slojem udruživanja se najčešće kolektivno nazivaju jednom riječju kao skriveni slojevi. Dok postoji samo jedan ulazni i jedan izlazni sloj, skrivenih slojeva može biti i više. Konvolucijski slojevi služe kao izlučivači značajki (engl. Feature extractors). Slojevi bliže ulazu prepoznaju jednostavnije značajke kao što su rubovi i neki drugi jednostavniji uzorci. Izlazi slojeva se zato nazivaju se mape značajki (engl. feature maps). Svaki sljedeći sloj koristi mape značajki prethodnog sloja za prepoznavanje značajki više razine. Kod klasifikacije se na kraju mreže koristi jedan ili više potpuno povezanih slojeva koji služe za konačnu klasifikaciju na temelju značajki prepoznatih konvolucijskim dijelom mreže. Izlaz mreže koja se koristi za klasifikaciju je obično vektor dimenzije jednake broju razreda, pri čemu redni broj komponente s najvećom vrijednošću odgovara rednom broju razreda u koji je slika klasificirana [19].

Pošto je za predviđanja kod segmentcije potrebno zadržati dimenzije slike, da bi se segmentirani objekti prikazali u izvornom obliku, najjednostavniji način bio bi konstruirati neuronsku mrežu koja kroz slojeve ne smanjuje dimenzije slike i kao izlaz daje segmentacijsku mapu dimenzija ulazne slike. Međutim, taj pristup je računski zahtijevan i ne bi urodio dobrim rezultatima. Zato se kod CNN-a implementiraju prethodno navedeni slojevi koji rezultiraju dubokom neuronskom mrežom gdje raniji slojevi (veće dimenzije slike) uče značajke niske razine, dok kasniji slojevi (manje dimenzije slike) uče značajke visoke razine. To rezultira pouzdanijom i računski efektivnijom mrežom, ali kao rezultat u kasnijim slojevima ostaju podaci malih dimenzija, dok je za zadatak segmentacije potrebno kao rezultat dobiti mapu dimenzija ulazne slike.

Zbog svoje izvrsne strukture, CNN kao enkoderi ostvaruju izvrsne rezultate u klasifikaciji slika, dok nisu predviđene za primjene u segmentaciji kao takve. Kod semantičke segmentacije potrebno je da se svaki piksel posebno klasificira. Zato je potrebno proširiti i unaprijediti postojeće CNN modele sa dekoderom za tu primjenu. U nastavku su navedeni takvi modeli.

Neki od najpoznatijih CNN-a dizajniranih za klasifikaciju su AlexNet, VGGNet i GoogleNet čije su strukture prikazane na slici 3.8. Mnoga daljnja unaprjeđenja mreža za segmentaciju temeljena su na tim mrežama, jer temeljne ideje tih arhitektura podržavaju razvoj dubokih neuronskih mreža za segmentaciju slike [21].

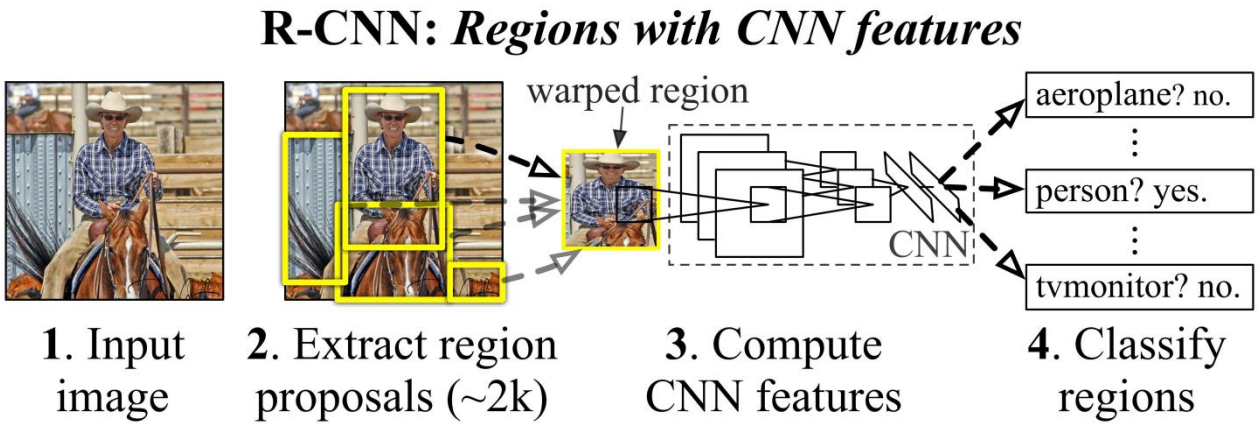


Slika 3.8. Struktura AlexNet, VGGNet i GoogleNet CNN-a [21]

3.2.2. RCNN

RCNN je neuronska mreža bazirana na izvlačenju regija i slijedi princip segmentacije pomoću prepoznavanja. Takva mreža prvo prostoručno izdvaja regije sa slike i opisuje ih, nakon čega slijedi klasifikacija na bazi regija. Pri testiranju, predikcije na temelju regija se preračunavaju u predikcije piksela, najčešće prema regiji s najvišim rezultatom koja ga sadrži [22].

Preciznije, RCNN mreže čija je struktura prikazana na slici 3.9 izvršavaju semantičku segmentaciju na temelju rezultata detekcije objekata. Prvo koristi selektivno pretraživanje kako bi se izdvojila velika količina prijedloga objekata i izračunale CNN značajke za svaki od njih. Konačno, klasificira svaku regiju koristeći linearne SVM (support-vector machines) specifične za klase.



Slika 3.9. Struktura RCNN mreže [23]

U usporedbi s tradicionalnim CNN strukturama koje su pretežito namijenjene za klasifikaciju slika, RCNN dodavanjem pretkoraka izvlačenja regija ima mogućnost rješavati složenije zadatke kao što su detekcija i segmentacija objekata. Na taj je način postala bazna arhitektura na koje se nadovezuju i složeniji modeli za detekciju i segmentaciju. Osim toga RCNN se može nadograditi na bilo koju postojeću CNN strukturu kao što su AlexNet, VGG, GoogleNet, ResNet i slično [22].

Koristeći CNN mrežu kao oslonac dovodi do značajno dobrih rezultata za predikcije, ali i do nekih nedostataka. Značajke dobivene iz CNN-a nisu potpuno kompatibilne za zadatke segmentacije. Značajke ne sadržavaju dovoljno prostornih informacija za definiranje preciznih

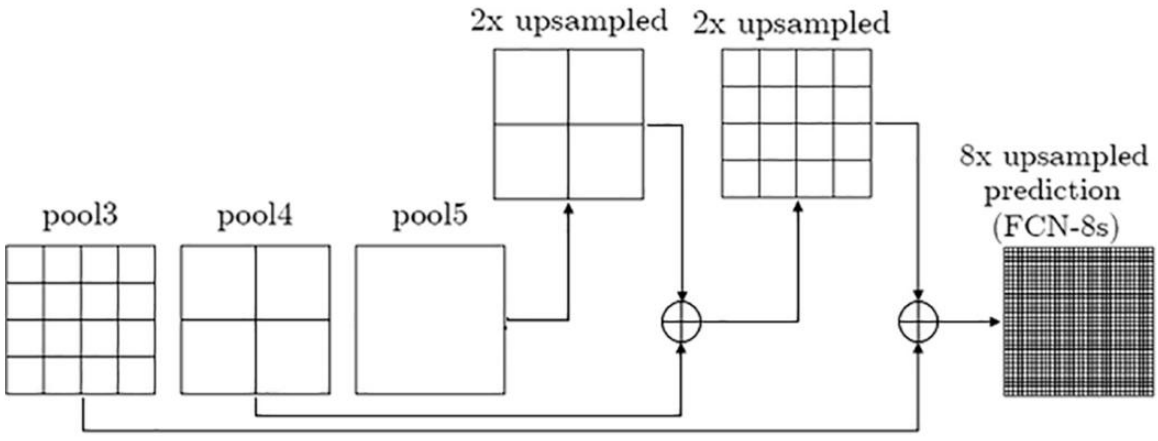
granica objekata te generiranje predikcija na temelju puno regija zahtijeva puno vremena i time smanjuje performanse pri korištenju te metode.

3.2.3. FCN

Prvo proširenje na AlexNet, VGGNet i GoogleNet bilo je razvoj potpuno konvolucijskih mreža (FCN) za semantičku segmentaciju slike. Opća ideja FCN-a sastoji se od 3 koraka: višeslojna konvolucija, dekonvolucija i spajanje (fuzija).

FCN zamjenjuje potpuno povezane slojeve s konvolucijskim slojevima. Konkretno konvolucijom dimenzija 1x1 zvana piksel konvolucija i koristi se za izračunavanje rezultata za svaku klasu na slici. Zbog slojeva udruživanja koji prate konvolucijske slojeve izlaz će biti manjih dimenzija od ulazne slike. Kako bi se rezultat vratilo u izvornu dimenziju ulazne slike koristi se proces dekonvolucije. Proces dekonvolucije posjeduje isti mehanizam kao i proces konvolucije ali djeluje na povećanje izlazne matrice dekonvolucijskim filtrima. Stoga, proces dekonvolucije rezultira matricom rezultata povećanih dimenzija, jednakih ulaznoj slici [21].

Kada bi čitav proces tu završio, izlazna matrica ne bi sadržavala jasne rezultate dobivene konvolucijskim procesom. Zato što se oni dekonvolucijom razrjeđuju i gube detalje. Kako bi se oporavili prostorni detalji tih rezultata koristi se takozvana fuzija semantičkih informacija iz trenutnog dubokog sloja s prostornim detaljima iz njegova prethodnog sloja. Iz te fuzije proizvodi se konačna segmentacija. Na slici 3.10 prikazano je kako se povećani trenutni sloj spaja s izlazom prethodnog sloja zbrajanjem njihovih individualnih elemenata [21].

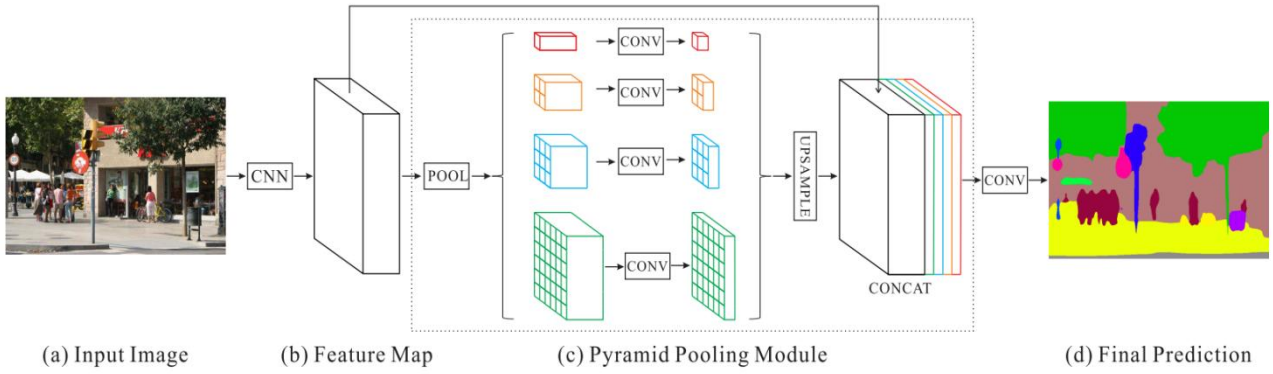


Slika 3.10. Shematska arhitektura FCN mreže [21]

3.2.4. PSPNet

PSPNet (eng. pyramid scene parsing network) ima specifičan pristup segmentaciji, umjesto arhitekture koja se bazira na enkoderu i dekoderu, ona upotrebljava segmentacijski model temeljen na modularno piramidalnom sloju udruživanja (eng. pyramid pooling). PSPNet je pokazao izvrsne performanse na popularnim skupovima slikovnih podataka. Izvlačenje globalnog konteksta ima izvanredne rezultate u segmentiranju pojedinih lokalnih komponenata, naročito iz satelitskih snimaka.

Za razliku od tipične arhitekture, PSPNet se sastoji od enkodera s CNN osloncem s konvolucijama za udruživanje, piramidalnog modula za udruživanje te dekodera. Takva arhitektura je prikazana na slici 3.15. [24].



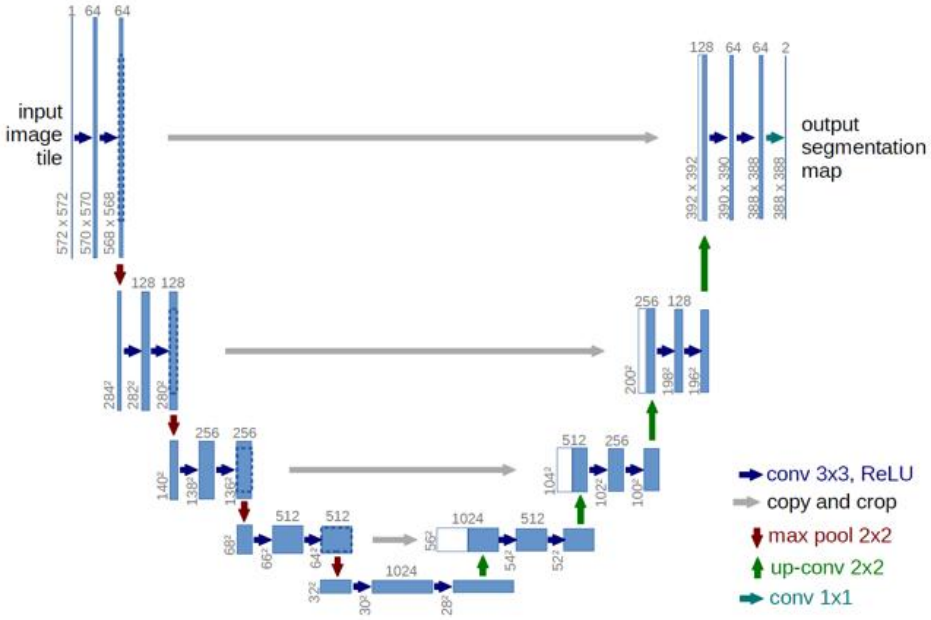
Slika 3.15. Tipična arhitektura PSPNet mreže [24]

U zadnjem sloju CNN mreže, mijenja se tipični konvolucijski sloj s konvolucijom za udruživanje, što pomaže u povećanju receptivnog polja. Piramidalni modul za udruživanje je glavni dio ovog modela i on pomaže modelu da uhvati globalni kontekst tako da klasificira piksele na temelju globalnih informacija prisutnih na slici [24].

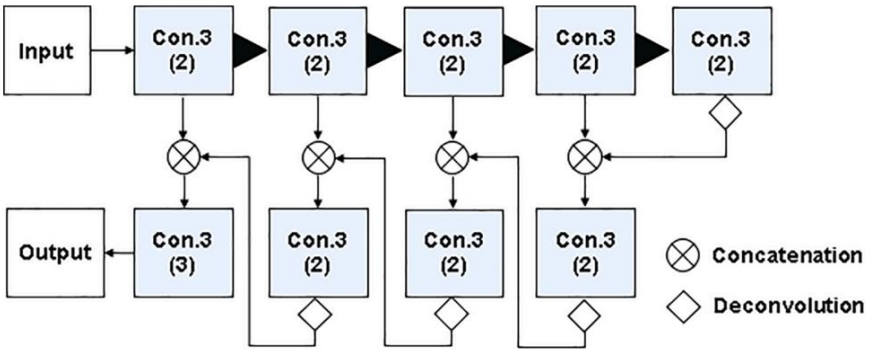
Kao što je vidljivo na slici 3.15 mape značajki iz CNN oslonca su sažete na više različitih dimenzija i kao takve zasebno prolaze kroz konvolucijski sloj. Nakon toga, izvodi se uzorkovanje prema gore kako bi ih se vratilo na dimenzije originalnih mapa značajki. Konačno, te mape značajki su ulančane s originalnim i prosljeđuju se dekoderu. Ova tehnika spaja značajke prisutne na više razina i stoga skuplja cjelokupni kontekst slike [24]. Završno, dekoder uzima te mape značajki i pretvara ih u predikcije prolazeći ih kroz mrežu dekodera.

3.2.5. U-net

U-net arhitektura usmjerena je na treniranje s malim setom slikovnih podataka. Sastoji se od strukture koja spaja enkoder i dekoder, tj. konvolucijski i dekonvolucijski dio, slično kao i FCN. Enkoder se kao i standardna arhitektura za klasifikaciju sastoji od slojeva koji postepeno smanjuju dimenzije slike, dok je zadaća dekodera iz tih rezultata postepeno povećavati dimenzije slike do originalnih dimenzija [25]. Glavna razlika je ta da je U-net simetričan i veze između uzorkovanje prema dolje i prema dolje koriste ulančavanje umjesto sume. Te veze pružaju informacije preuzete iz lokalnog u globalni tijekom faze uzorkovanja prema gore i zbog svojih simetričnih karakteristika ima značajan broj mapa značajki u fazi uzorkovanja prema gore [26]. Takva arhitektura je vizualno prikazana na slici 3.11 Dok je shematski prikaz na slici 3.12.



Slika 3.11. Tipična arhitektura U-net mreže [25]



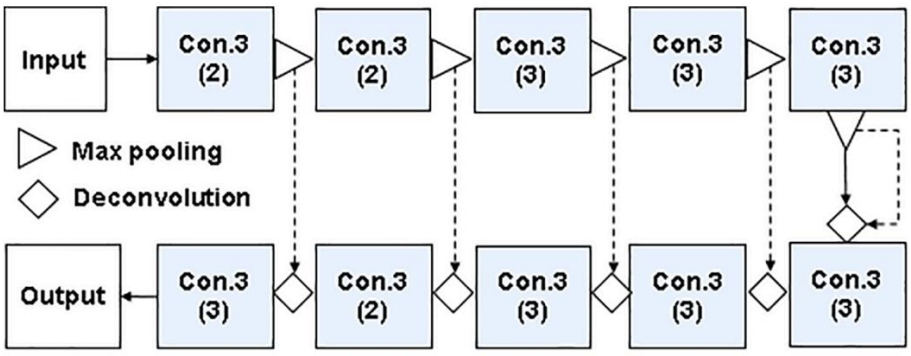
Slika 3.12. Shematska arhitektura U-net mreže [21]

Te veze između slojeva na različitim razinama mreže omogućuju da izlaz uzima u obzir značajke iz različitih slojeva (značajke visoke razine i niske razine). Time je omogućeno da se na izlazu ostvaruje i dobra lokalizacija i dobra predikcija u isto vrijeme. To je ključno za dobivanje dobrih rezultata i zato je ta metoda vrlo poželjna i često primjenjivana za rješavanje problema semantičke segmentacije [25].

Sa strane dekodera (desne strane mreže) slojevi udruživanja zamijenjeni su slojevima proširenja. Ti slojevi povećavaju dimenzije izlaza. Uz informacije dobivene preko veza sa strane enkodera uzastopni sloj konvolucije tada može naučiti sastaviti precizniji izlaz. Kao posljedica toga strana enkodera je približno simetrična strani dekodera i to daje arhitekturi oblik slova U. Od tuda potječe naziv U-net. Na samom kraju mreže dobivena segmentacijska mapa sadrži samo piksele objekata, za koje je kontekst dobiven preklapanjem s ulaznom slikom i tako se dobiva konačna segmentacija. Ta strategija omogućuje besprijekornu segmentaciju proizvoljno velikih slika [25].

3.2.6. Segnet

Segnet je model temeljen na strukturi enkodera i dekodera koji ima gotovo identičnu strukturu U-netu. Model je osmišljen tako smanjuje potrebe za velikom memorijom pri treniranju [26]. Mreža enkodera u Segnet modelu sadrži brojne konvolucijske slojeve i slojeve sažimanja za izvlačenje značajki. Po tome prati arhitekturu FCN-a. Ali kako dublji slojevi mreže izvlače značajke većeg semantičkog značaja, tako istodobno gube na prostornim informacijama. Segnet metoda uvodi ideju pohranjivanja indeksa elemenata, tj. pohranjuje lokaciju elementa unutar okvira filtra. I tu informaciju koristi pri postupku uzorkovanja prema gore u dekoder mreži. Struktura dekodera simetrično prati strukturu enkodera, slično kao i U-net mreža [21]. Takva struktura vidljiva je na slici 3.13.

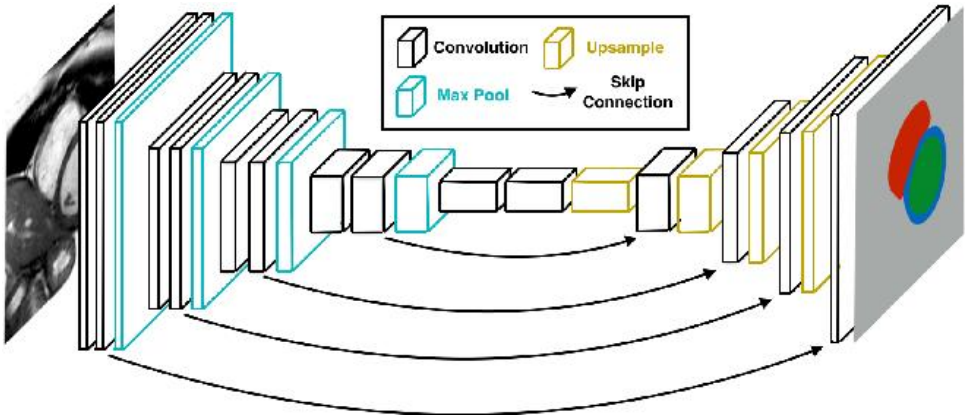


Slika 3.13. Shematska arhitektura Segnet mreže [21]

3.2.7. E-net

E-net arhitektura je također bazirana na strukturi enkodera i dekodera, kao i prethodno navedene. To je suvremena arhitektura koja je pokazala izvrsne rezultate u usporedbi s istodobnim modelima temeljenim na enkoderu i dekoderu, pretežito na popularnim skupovima podataka satelitskih snimaka kao što su Cityscapes i CamVid. Ideja iza E-neta je kompaktna arhitektura prikazana na slici 3.14 koja primjenjuje različite optimizacijske elemente za rješavanje nekih razvojnih problema [27].

Najveća razlika u odnosu na prethodno navedene metode je ta da mreža nije simetrična, već se sastoji od velikog enkodera i relativno manjeg dekodera. Takva je mreža privlačna za primjenu jer je strana enkodera ta koja radi predikcije, dok je strana dekodera potrebna samo za uzorkovanje izlaza prema gore kako bi se očuvali detalji predikcije na originalnoj rezoluciji. Stoga nije potrebno da je strana dekodera velika kako enkoder. Proširena konvolucija je korištena kako bi se proširio kontekst izlaza enkodera tako da ima šire receptivno polje pošto sama mreža već šteti točnosti smanjenjem uzorkovanja mapa značajki. Koristi i prostorni dropout sloj kao regulator na kraju konvolucijskih grana [26].

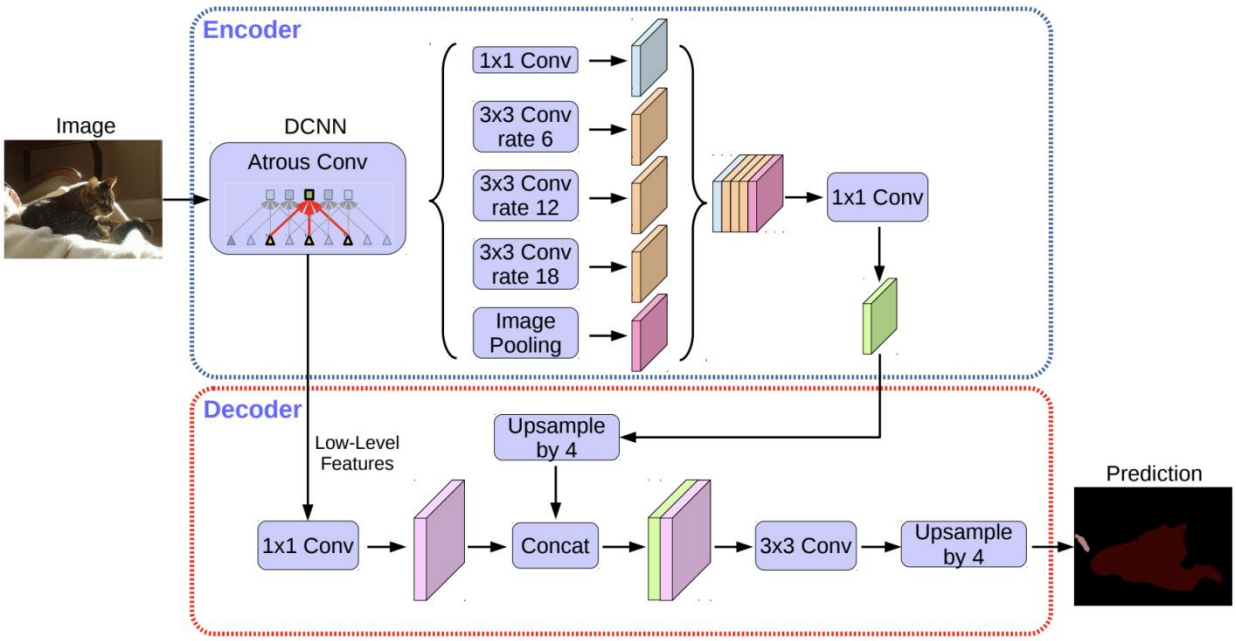


Slika 3.14. Tipična arhitektura E-net mreže [28]

3.2.8. DeepLabv3+

DeepLabv3+ je u ovom trenutku najsuvremenija generacija DeepLab modela. U grubom opisu ovaj model stapa koncept enkoder-dekoder modela i modela s piramidalnim slojem udruživanja. Najnovija generacija nadodaje jednostavan, ali učinkovit modul dekodera za usavršavanje rezultata segmentacije, posebno uzduž granica objekta [26].

Model implementira nekoliko parelelnih konvolucija sa različitim stopama učenja (eng. Atrous spatial pyramid pooling). Za razliku od PSPNet-a koji izvršava operacije udruživanja na različitim dimenzija, DeepLab koristi konvolucije (eng. Atrous convolutions) koje izvršavaju konvolucije na mapama značajki sa različitim stopama učenja. Iako se važne semantičke informacije nalaze u zadnjoj mapi značajki PSPNet-a, detaljne informacije vezane za granice objekata se gube zbog konvolucija i udruživanja unutar oslonca mreže. DeepNet model to ublažuje dodavanjem prije navedenih atroznih konvolucija za izvlačenje gušćih mapa značajki. Model je vidljiv na slici 3.16.



Slika 3.16. Tipična arhitektura DeepLab mreže [29]

Modele na bazi enkoder-dekodera definira brži izračun u enkoderu, pošto se nikakve mape značajki ne proširuju. Takvi modeli onda postepeno obnavljaju granice objekata u dekoderu. U ovoj strukturi, moguće je samovoljno kontrolirati rezoluciju izvučenih značajki iz enkodera pomoću atroznih konvolucija i tako regulirati kompromis između preciznosti i brzine računanja, dok to nije moguće kod postojećih enkoder-dekoder modela.

3.2.9. Ostale metode

Postoji još puno metoda segmentacije objekata temeljenih na umjetnim neuronskim mrežama koje u ovom radu nisu navedene. Većinom se radi o manjim razlikama između metoda ili samo nadogradnji postojećih, stoga nije potrebno razmatrati bas svaku metodu zasebno. Neovisno o tome, nekoliko metoda koje zaslužuju napomenu su; DialatedNet, DeconvNet, DenseNet,

ParseNet, GCN, MCNN, FPN, LinkNet, Mask R-CNN, Fast-RCNN, Up-sample metoda, Piramidalne metode i slabo nadzirana segmentacija.

Najveći dio metoda za segmentaciju objekata temelji se na CNN mrežama, te primjenjuje različite alate, module ili dodatne korake za savladavanje prepreka pri uporabi takvih mreža za potrebe segmentacije. Iz tog razloga, razvio se jako veliki broj metoda za taj zadatak, koje se i dalje usavršavaju i nadograđuju. Ne može se reći da je neka od njih apsolutno bolja od ostalih. Samim time što i unutar polja znanosti kao što je segmentacija, postoji popriličan broj specijaliziranih zadataka ovisno o skupu podataka s kojima se radi te kakve se rezultate očekuje.

U nastavku je predstavljena primjena semantičke segmentacije na užem području segmentacije plovila iz satelitskih snimaka.

4. SEMANTIČKA SEGMENTACIJA OBJEKATA IZ SATELITSKIH SNIMAKA

Satelitske snimke se obrađuju kako bi se identificirali različiti obrasci, objekti, geografske konture, informacije o tlu, posljedice prirodnih katastrofa i slično. Te se informacije kasnije koriste za poljoprivredu, rudarstvo, geo istraživanja, procijene štete, uz ostalo [16].

Proces provedbe semantičke segmentacije objekata iz satelitskih snimaka korištenjem metoda baziranih na CNN mrežama najjednostavnije je prijenosom učenja (eng. Transfer learning) s naprednih modela dobivenih evaluacijom na velikim već postojećim skupovima podataka korištenih za slične zadatke kao što je segmentacija satelitskih snimaka. Cjelovito se prenose arhitekture postojećih neuronskih mreža za taj zadatak, s pred istreniranim težinskim faktorima. Tada se te mreže dodatno treniraju na pripremljenoj bazi podataka za željeni zadatak.

Za dodatno treniranje takvih mreža za segmentaciju potrebno je pripremiti bazu podataka koja sadrži ulazne slike i pripadajuće segmentacijske mape (eng. ground truth), čiji je primjer prikazan na slici 4.1. Poželjno je imati što veći broj slika. Također slike moraju biti kvalitetne i jasne, s točnim segmentacijskim mapama, čija je kreacija sama po sebi izazovna kada se slične baze podataka sastoje od više desetaka tisuća slika, pa i milijuna slika. Stoga, umjesto samostalnog pripremanja takve slikovne baze podataka, odlučeno je koristiti postojeću bazu koja je predstavljena u nastavku.

Slika 4.1. Usporedba ulazne slike i pripadajuće segmentacijske maske [26]

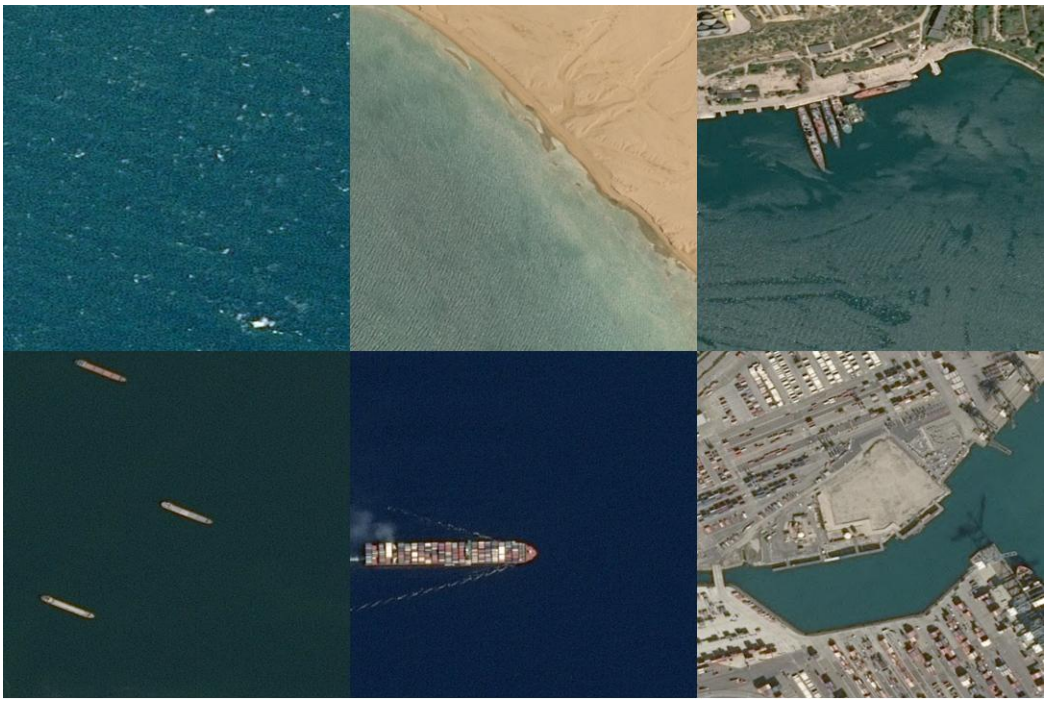
4.1. Skup podataka

Jedna od ključnih stvari za postizanje dobrih rezultata je dobar skup podataka. Može se reći da je ostvareni model dobar samo koliko je dobar i skup podataka korišten za treniranje tog modela. Zato je za ovaj projekt odabran skup podataka korišten za „Airbus ship detection challenge“ dostupna na stranici Kaggle [30].

Taj je skup slikovnih podataka pripremljen od strane tvrtke Airbus za potrebe natjecanja namijenjenog da potakne programere iz cijelog svijeta da se natječu u izradi modela koji će detektirati sve brodove na satelitskim snimkama, u što kraćem vremenu. Motivacija za izazov dolazi iz razloga da se brodski promet sve više povećava, brodske luke zahtijevaju metodu praćenja prometa u njima, kao i na otvorenom moru. Postojanje većeg broja plovila povećava šanse za prekršaje na moru kao što su ekološki razorne brodske nesreće, piratstvo, ilegalni ribolov, trgovina drogom i ilegalno kretanje tereta. To je prisililo mnoge organizacije, od agencija za zaštitu okoliša do osiguravajućih društava i nacionalnih državnih tijela, da pobliže paze na otvoreno more.

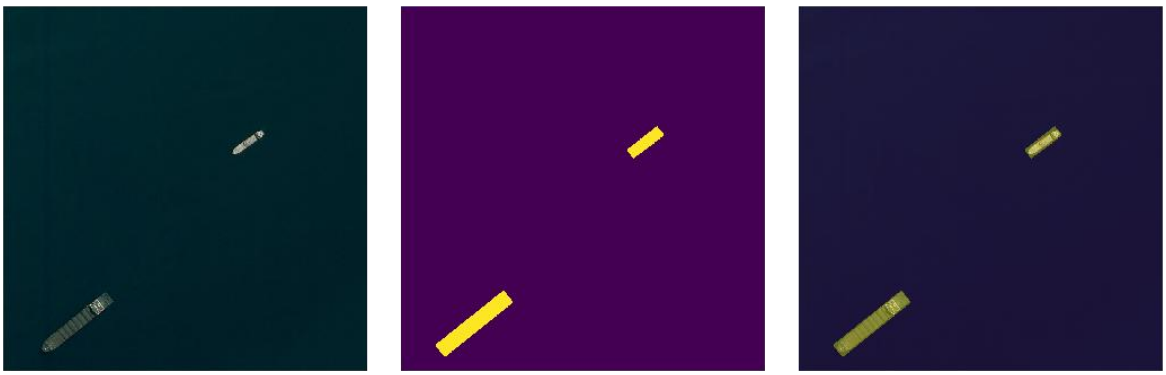
Airbus nudi sveobuhvatne usluge pomorskog praćenja izgradnjom smislenog rješenja za široku pokrivenost, sitne detalje, intenzivno praćenje, vrhunsku reaktivnost i interpretaciju. Kombiniranjem svojih podataka s visokoobrazovanim analitičarima, oni pomažu u pružanju potpore pomorskoj industriji u povećanju znanja, predviđanju prijetnji, aktiviranju upozorenja i poboljšanju učinkovitosti na moru [30].

Navedeni skup podataka sastoji se od 208.162 slika podijeljenih u dvije skupine. Skupina za treniranje koja sadrži 192.556 slika i skupina za testiranje koja sadrži 15.606. Sve su slike jednake rezolucije 768x768 piksela. Većina slika ne sadrži plovila dok neke sadrže jedno ili više plovila. Plovila na slici se međusobno mogu drastično razlikovati po veličini (koliki postotak slike zauzimaju) i mogu se nalaziti na otvorenom moru, na dokovima, marinama, itd. Nekoliko primjera slika iz baze prikazano je na slici 4.2.



Slika 4.2. Nekoliko primjera slika iz korištene baze podataka [30]

Uz sve slike u ovoj bazi podataka pripremljena je *.csv* datoteka koja sadrži informacije o segmentacijskoj mapi (maski) ili temeljnoj istini (eng. ground truth) svih slika i opisuje koji pikseli na slici predstavljaju plovilo, a koji pozadinu. Točnije, te informacije su zapisane u RLE formatu (eng. Run-length encoding). To je oblik kompresije podataka bez gubitaka, koristi se za sekvencije u kojima se ista vrijednost podataka javlja u mnogim uzastopnim podatkovnim elementima. Taj način kompresije je učinkovit na podacima kao što su elementi na digitalnim slikama, ili u ovom slučaju, maske objekata za segmentaciju. Jedna slika iz odabrane baze i njena maska prikazane su na slici 4.3.



Slika 4.3. Primjer slike s maskom i njihovo preklapanje [30]

Jedan od izazova kod korištenja ove baze podataka bit će dekodiranje tih segmentacijskih maski iz RLE formata. Drugi i veći izazov je snažna neuravnoteženost podataka. Većina slika ne sadrži plovila. Dok na slikama koje i sadrže neko plovilo, omjer piksela plovila i pozadine je ~ 1:1000. S tim i drugim problemima suočit će se pri realizaciji algoritma za segmentaciju plovila u 5. poglavlju.

5. REALIZACIJA MODELA BAZIRANOG NA CNN-U

Smisao ovog rada je istražiti, ispitati i usporediti različite metode segmentacije objekata iz satelitskih snimaka. U 3. poglavlju predstavljeno je nekoliko takvih metoda. Prvo će se predstaviti potrebna priprema i vizualizacija slikovnih podataka za nastavak. Sljedeće, uvoženje nekoliko metoda za segmentaciju, implementacija na korišteni skup podataka i preliminarni rezultati. Uz to, predstavit će se problematika tog procesa i problematika korištenja nekih metoda za ovaj zadatak.

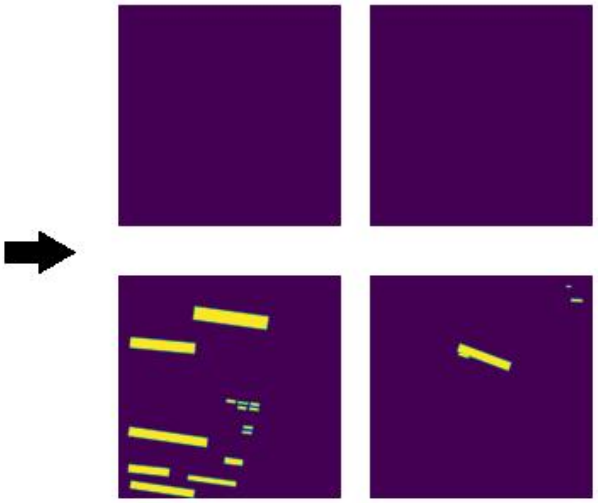
5.1. Priprema slikovnih podataka

Kako bi se brzo i bez poteškoća ispitalo više modela za segmentaciju, potrebno je najprije pripremiti odabrani skup podataka tako da je lako uporabljiv za treniranje modela. Za pojednostavljenje procesa treniranja i evaluacije različitih modela i parametara odlučeno je koristiti postojeću knjižnicu pod nazivom keras segmentation [31]. Ta knjižnica omogućuje implementaciju nekoliko segmentacijskih modela baziranih na dubokom učenju, ali zahtijeva specifičnu pripremu slikovnog skupa podataka.

Potrebno je slike i maske iz odabranog skupa podataka [30] spremiti u .png formatu u dvije mape. Jedna za originalne slike i druga za pripadajuće maske pod istim imenom. Dok su slike u odabranom skupu podataka, kao što je pojašnjeno u poglavlju 4, zapisane u .jpg formatu. A sve maske u jednoj .cvs datoteci u RLE (eng. Run-length encoding) formatu. Postupak pripreme slikovnog skupa podataka nalazi se u online bilježnici na stranici Kaggle [32] i u dodatku A. Postupak je izrađen po uzoru na postojeće Kaggle online bilježnice [33, 34].

Priprema slika započinje uvoženjem potrebnih python knjižnica i definiranjem funkcija za dekodiranje RLE formata i kreiranje slika maski iz toga. Na slici 5.1 prikazano je kako izgleda zapis maski unutar .cvs datoteke te kako izgledaju dekodirane maske kao slike. Vidljivo je da .cvs datoteka sadrži 2 stupca. U lijevom stupcu zapisana je oznaka (ime) slike kojoj ta maska odgovara kako bi se programski lako odredilo pripadne slike i maske. U desnom stupcu zapisani su podaci u RLE formatu koji označavaju koji pikseli slike pripadaju objektu, u ovom slučaju plovilu.

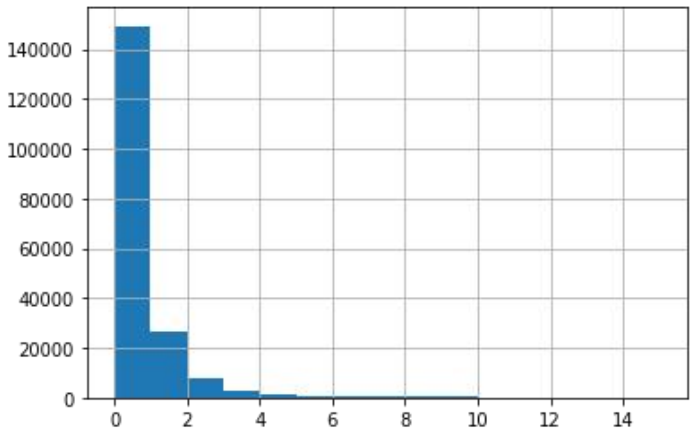
192556 unique values	[null] 65% 43801 1 44567 4 45... 0% Other (81721) 35%
00003e153.jpg	
0001124c7.jpg	
000155de5.jpg	264661 17 265429 33 266197 33 266965 33 267733 33 268501 33 269269 33 270037 33 270805 33 271573 33 ...
000194a2d.jpg	360486 1 361252 4 362019 5 362785 8 363552 10 364321 10 365090 9 365858 10 366627 10 367396 9 368165...



Slika 5.1. Zapis maski u RLE formatu (lijevo) i dekodirane maske (desno)

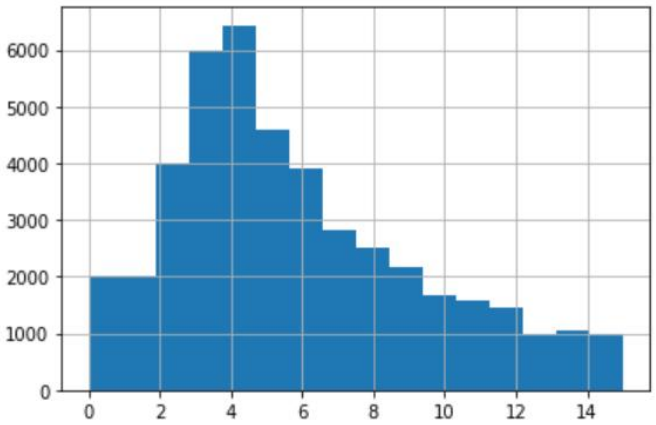
Na lijevoj strani slike 5.1 u tablici vidljivo je da prve dvije slike nemaju nikakav zapis u desnom stupcu, to znači da na tim slikama nema prisutnih plovila, što potvrđuje desna strana slike. Zapravo, na vrhu desnog stupca na slici zapisano je da 65% slika nema nikakvog zapisa o prisutnim plovilima što znači da samo 35% slika sadrži jedno ili više plovila. Uz to, na slikama koje sadrže plovila, velika je razlika u broju piksela koji pripadaju kategoriji pozadina ili kategoriji plovilo. Odlučeno je da se ta velika uneravnoteženost na neki način smanji.

Za početak, ispitane su sve maske iz skupa podataka za treniranje po tome koliko zasebnih instanci plovila sadržavaju. Graf koji prikazuje koliko slika sadrži određeni broj plovila prikazan je na sljedećoj slici 5.2.



Slika 5.2. Broj slika sortiranih prema broju istanci plovila prije selekcije

Na gornjoj slici vidljivo je da oko 150.000 slika ne sadrži niti jedno plovilo, oko 25.000 slika sadrži jedno, oko 8000 slika sadrži 2, i dalje se to za svaku kategoriju smanjuje. Kako bi se za učenje bolje iskoristile one slike koje sadrže više brodova, odlučeno je programski povećati udio slika s većim brojem brodova naprema slikama s manje ili bez brodova. Time se povećava kvaliteta skupa slikovnih podataka, a ujedno se smanjuje ukupan broj slika čime će se ubrzati buduće treniranje modela. Na slici 5.3 prikazan je graf koji predstavlja koliko slika sadrži određeni broj plovila nakon odrađene selekcije. Vidi se da je dobiveni skup slikovnih podataka puno više izbalansiran. Skup sadrži samo 2000 slika bez brodova, 2000 s jednim plovilom, 4000 slika s 2 plovila, 6000 slika s 3 plovila i tako dalje. Ukupno se tim postupkom skup podataka smanjio s 192.556 na 44.202 slike.



Slika 5.3. Broj slika sortiranih prema broju istanci plovila nakon selekcije

Sljedeći je korak definiranje funkcije koja uzima ulazne slike i ulazne maske te korištenjem prije definiranih funkcija za dekodiranje RLE formata, sprema parove slika i maski u generator funkciju. Iz te je generator funkcije tada jednostavno iteracijski izvući parove slika i maski i spremiti ih u .png formatu u dvije mape pod istim imenom kao što je zahtijevano od prije spomenute keras segmentation knjižnice.

Na taj se je način za potrebe daljnjeg rada izradio novi skup podataka koji je kasnije jednostavno uvesti u novoj online kaggle bilježnici za treniranje modela. Odlučeno je da se slike odmah pri pohranjivanju u .png formatu raspoređuju u 3 para mapi; za treniranje, za validaciju i za testiranje, svaka s određenim brojem slika. Taj raspored slika prikazan je u tablici 5.1.

Tablica 5.1. Raspored slika u kreiranom skupu podataka

Ime datoteke	Broj slika	Sadržaj mape
train_images	20.000	Satelitske snimke za treniranje modela
train_masks	20.000	Maske za treniranje modela
valid_images	5.000	Satelitske snimke za validaciju modela
valid_masks	5.000	Maske za validaciju modela
test_images	5.000	Satelitske snimke za testiranje modela
test_masks	5.000	Maske za testiranje modela

Iz priložene tablice se može uočiti da kreirani skup slika sadrži ukupno 30.000 slika i 30.000 pripadnih maski, dok je prethodno selekcijom bilo kreirano 44.202 maski. To toga je došlo iz jednostavnog razloga da kaggle bilježnice ograničavaju veličinu izlaznih podataka na 20 GB, i time kreiranje skupova podataka do maksimalno te veličine. Izračunato je da 30.000 parova slika i maski rezultira skupom veličine nešto manje od 20GB. Procijenjeno je da je ta količina slika dovoljna za treniranje kvalitetnih modela pa je taj skup slika kreiran. Skup je dostupan za pregled na kaggle stranici [35]. 20.000 slika za treniranje i 5.000 slika za validaciju koristit će se pri svakom epohu treniranja modela. A 5000 slika za testiranje koristit će se za evaluaciju točnosti i kvalitete modela nakon odrađenog treniranja. Podaci dobiveni evaluacijom koristit će se za usporedbu modela temeljenih na različitim metodama segmentacije.

5.2. Treniranje modela i preliminarni rezultati

Za potrebe treniranja izrađena je nova kaggle bilježnica [36] dostupna u dodatku B, koja dozvoljava uvoženje prethodno kreiranog skupa, i keras segmentation knjižnice [31]. Ta knjižnica omogućuje implementaciju više metoda segmentacije temeljenih na dubokim neuronski mrežama. Uz to omogućuje jednostavno definiranje različitih parametara poput temeljne CNN mreže, dimenzija ulaznih slika, funkcije gubitaka, broj klasa i drugo. Također, omogućuje jednostavnu augmentaciju ulaznih slika, uvođenje težinskih faktora iz unaprijed istreniranih mreža, jednostavnu evaluaciju modela te još nekoliko dodatnih stvari.

Navedena online kaggle bilježnica započinje uvoženjem nekoliko potrebnih python knjižnica, keras segmentation knjižnice te kreiranog slikovnog skupa. Definiraju se nazivi datoteka prema tablici 5.1 i uvoze se pripadne datoteke iz skupa. Sljedeće, jednostavno se definira i uvozi željeni model koji odgovara metodi i parametrima koji se trenutno ispituju. U tablici 5.2 prikazane su

sve dostupne kombinacije segmentacijskog modela i temeljne CNN mreže. Dostupni segmentacijski modeli (metode) su FCN8, FCN32, PSPNet, U-Net i Segnet i te će se metode primjenjivati i uspoređivati u daljnjem radu. Dostupne temeljne CNN arhitekture su osnovna CNN, VGG 16, Resnet-50 i MobileNet.

Tablica 5.2. Kombinacije segmentacijskog modela i temeljne CNN mreže

Ime modela	Temeljna CNN mreža	Segmentacijski model
fcn_8	Vanilla CNN	FCN8
Fcn_8_vgg	VGG 16	FCN8
Fcn_8_resnet50	Resnet-50	FCN8
Fcn_8_mobilenet	MobileNet	FCN8
fcn_32	Vanilla CNN	FCN32
Fcn_32_vgg	VGG 16	FCN32
Fcn_32_resnet50	Resnet-50	FCN32
Fcn_32_mobilenet	MobileNet	FCN32
pspnet	Vanilla CNN	PSPNet
vgg_pspnet	VGG 16	PSPNet
resnet50_pspnet	Resnet-50	PSPNet
unet_mini	Vanilla Mini CNN	U-Net
unet	Vanilla CNN	U-Net
vgg_unet	VGG 16	U-Net
resnet50_unet	Resnet-50	U-Net
mobilenet_unet	MobileNet	U-Net
segnet	Vanilla CNN	Segnet
vgg_segnet	VGG 16	Segnet
resnet50_segnet	Resnet-50	Segnet
mobilenet_segnet	MobileNet	Segnet

Nakon odabira i uvoženja modela iz keras segmentation knjižnice te definiranja dimenzija ulazne slike, model se kompilira s definiranom funkcijom gubitaka, optimizatorom i metrikom koja se ispisuje pri treniranju. Tada se sljedećom funkcijom započinje treniranje modela. Unutar te funkcije potrebno je definirati više parametara:

- lokacije datoteka koje sadrže ulazne slike i maske za treniranje i validaciju,
- veličina uzorka slika koje algoritam istodobno prosljeđuje kroz mrežu,
- broj koraka (uzimanja uzorka slika) u jednoj epohi treniranja,
- broj epoha i
- dodatni proizvoljni parametri.

Nakon treniranja modela, njegova arhitektura i težinski faktori se sahranjuju kako bi se lako u nekom drugom koraku pristupilo tom istreniranom modelu. Sljedeće, model se procjenjuje na 5000 slika iz datoteke za testiranje, funkcijom koja kao ulazne parametre uzima lokacije datoteka koje sadrže ulazne slike i maske za testiranje. Ta evaluacija na slikama koje model nije koristio za treniranje daje realne podatke o kvaliteti odabrane metode, odabranih parametara i samog procesa treniranja. Ti su podatci predočeni u obliku IoU metrike i taj će se rezultat, uz ostalo, koristiti za međusobnu usporedbu metoda.

IoU (eng. Intersection-Over-Union) metrika mjerna je vrijednost koja se uobičajeno koristi za evaluaciju semantičke segmentacije slike [37]. Predstavlja omjer presjeka i unije predikcije i originalne maske slike. Originalna jednadžba opisana je izrazom (5.1):

$$IoU = \frac{|T \cap P|}{|T \cup P|}, \tag{5.1}$$

gdje je:

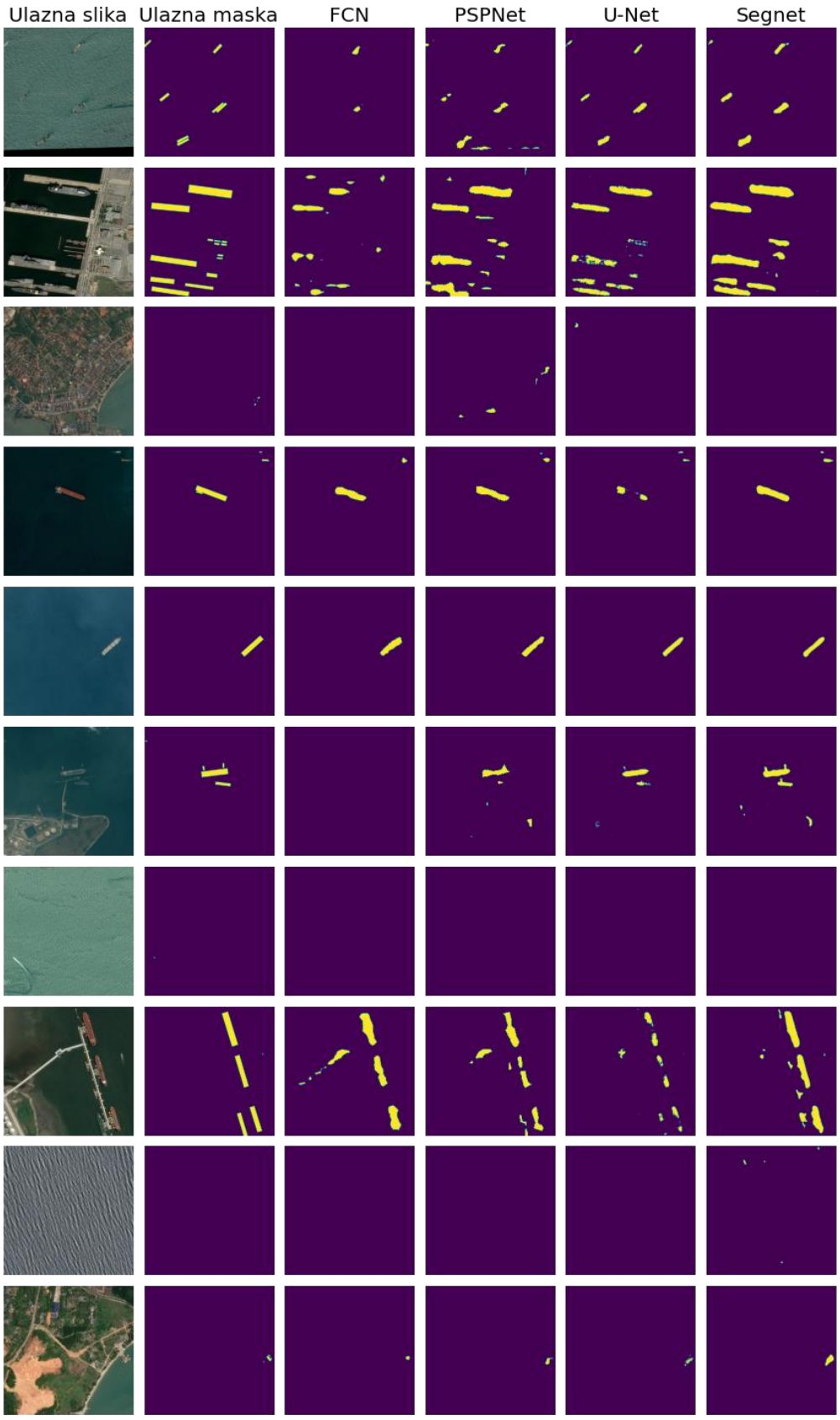
- IoU dobivena vrijednost između 0 i 1,
- T je ulazna maska (temeljna istina) i
- P je dobivena predikcija [2].

Iz tih elemenata izračunate su 3 vrste IoU metrike:

- IoU vrijednost za svaku klasu (eng. Class wise IoU),
- Srednja IoU vrijednost (eng. Mean IoU) i
- Srednja IoU vrijednost uravnotežena prema omjeru pojavljivanja svake klase (eng. Frequency weighted IoU). Koristi se u primjeru neuravnoteženog pojavljivanja klasa, kao što je u slučaj u ovom radu [2].

Odlučeno je za početak istrenirati 4 modela s osnovnim parametrima kako bi se ukratko pregledalo njihov razlike u izlaznim predikcijama, IoU vrijednostima i vremenu treniranja.

Izlazne predikcije na nekoliko različitih satelitskih slika za FCN32, PSPNet, U-net i Segnet metode sa VGG 16 temeljnom mrežom prikazane su na slici 5.4.



Slika 5.4. Ulazne slike i maske te izlazne predikcije različitih metoda

IoU vrijednosti i vrijeme treniranja za navedene metode s osnovnim parametrima prikazani su u sljedećoj tablici 5.3.

Tablica 5.3. Rezultati treniranja s osnovnim parametrima

Ime modela	fcn_32	vgg_pspnet	vgg_unet	vgg_segnet
Broj parametara	134 milijuna	17 milijuna	12.3 milijuna	11.6 milijuna
Class wise IoU	{0.996, 0.409}	{0.995, 0.444}	{0.997, 0.563}	{0.997, 0.595}
Mean IoU	{0.703}	{0.719}	{0.780}	{0.796}
Frequency weighted IoU	{0.992}	{0.992}	{0.995}	{0.994}
Vrijeme treniranja 1 epoha	3.566 sec (59min 26s)	1.803 sec (30min 03s)	2.320 sec (38min 40s)	2.117 sec (35min 17s)

Sva 4 modela trenirana su 5 epoha s originalnim dimenzijama ulazne slike (768x768) te na VGG 16 temeljnoj CNN mreži. Sa 20.000 slika za treniranje, 5.000 za validaciju i 5.000 za evaluaciju. Iz slike 5.4 i tablice 5.3 uočavamo neke glavne nedostatke svake metode.

FCN metoda u slučaju prve, treće, četvrte i osme slike nije segmentirala sve instance plovila. U slučaju šeste slike nije segmentirala niti jedno plovilo dok su ostale metode to uspješno odradile. Iz tablice se iščitava da se FCN metoda sastoji od mreže puno većeg broja parametara od ostalih i potrebno joj je ~54% više vremena za treniranje od sljedeće najsporije metode (U-net). Također, prema IoU vrijednosti za klasu plovila postiže najnižu vrijednost.

PSPNet metoda u slučaju prve, druge, treće, šeste i osme slike napravila je više predikcija polovila nego što se na slici nalazi. Segmentacije dobivene metodom generalno ne predstavljaju dobar oblik plovila, ali segmentirane su gotovo sve instance plovila. PSPNet metoda postiže nešto bolju srednju IoU vrijednosti i IoU vrijednost za klasu plovila of FCN metode te je najbrža pri treniranju od 4 ispitane metode.

U-Net metoda pokazala se bolja u izdvajanju zasebnih instanci susjednih plovila i segmentiranju obrisa plovila. Jedino ta plovila nisu segmentirana kao 1 cjelina, već razdvojena na 2 ili više odvojenih dijelova, kao što se vidi na drugoj, četvrtoj i osmoj slici. Također, generalno metoda nije segmentirala dijelove slike koji originalno ne sadrže plovila. Prema tablici 5.3 metoda je

nešto sporija od PSPNeta, ali postiže poprilično bolje rezultate na temelju IoU vrijednosti za klasu plovila te srednje IoU vrijednosti.

Segnet metoda prema slici 5.4 dobro je segmentirala gotovo sve instance plovila, kao jednu cjelinu i zadržavajući im oblik. To je vidljivo na prvoj, drugoj, četvrtoj, šestoj i osmoj slici, dok je, kao što se vidi na šestoj, osmoj i devetoj slici, segmentirala i manje dijelove slike koji originalno ne sadrže plovila. Prema tablici 5.3 postiže najbolje rezultate na temelju IoU vrijednosti za klasu plovila te srednje IoU vrijednosti. Sadrži arhitekturu s najmanjim brojem parametara i vrijeme treniranja je drugo najbrže, nakon PSPNet-a.

Iako su dobiveni rezultati adekvatni za usporedbu različitih metoda, uočava se jedan veliki nedostatak kod svih metoda pri treniranju s osnovnim parametrima. Taj je nedostatak predstavljen niskom IoU vrijednosti za klasu plovila, u usporedbi s klasom pozadine. Najviši rezultat postiže Segnet s vrijednošću 0.595, naprema 0.997 za klasu pozadine. Takav se rezultat moglo očekivati iz prije navedenog razloga da klase nisu uravnotežene prema ukupnom postotku piksela slike koje predstavljaju. Ta će se nejednakost pokušati kompenzirati pažljivim odabirom funkcije gubitaka i izmjenom nekih drugih parametara.

5.3. Odabir funkcije gubitaka

Za rezultate dobivene na slici 5.4 i tablici 5.3 pri treniranju tih modela korištena je zadana funkcija gubitaka pod nazivom categorical crossentropy. Funkcija gubitaka je u osnovi jednadžba koju algoritam koristi pri procesu treniranja kao bi model izračunao koliko dobro radi predikcije. Funkcija gubitaka daje povratne informacije modelu tijekom procesa nadziranog učenja (učenja s pred označenim podacima) koliko dobro konvergira na optimalne parametre modela. Koristi se za vođenje modela u potrazi za što boljom aproksimacijom koja mapira ulazne podatke u izlazne podatke. U slučaju segmentacije, ulazne slike u maske [38].

Funkcija gubitaka categorical crossentropy, kao i binary crossentropy, jednostavna je funkcija koja računa prosjek klasifikacije za svaki piksel na slici. Ta se funkcija često koristi ali najbolje funkcionira na balansiranim podacima, tj. zahtijeva dobro balansirane klase prema ukupnom postotku piksela slike koje predstavljaju [2]. Zbog tog je nedostatka za zadatak segmentacije plovila sa satelitskih snimaka potrebno probati pronaći učinkovitiju funkciju gubitaka. Moguće je funkciju prilagoditi tako da se definiraju težišni faktori svake klase, ali u ovom slučaju je razlika u pojavljivanju svake klase na razini piksela uistinu prevelika.

IoU koeficijent predstavljen prethodno u potpoglavlju 4.2 može se implementirati u funkciju gubitaka. Koeficijent predstavlja omjer presjeka i unije predikcije i originalne maske slike te takva funkcija u nekim slučajevima predstavlja prednosti nad nešto slabije balansiranim setovima podataka.

Dice koeficijent, sličan je IoU koeficijentu i koristi se za računanje sličnosti dviju slika. Koeficijent je implementiran u funkciju gubitaka i računa sličnosti između predikcije i ulazne maske [39]. U suštini, koeficijent mjeri preklapanje dvaju slika. U slučaju segmentacije, koeficijent vrijednosti 1 predstavlja savršeno preklapanje, tj. savršenu predikciju. Postoji i varijanta dice funkcije pod nazivom soft dice funkcija gubitaka. Ta funkcija računa dice koeficijent za svaku klasu zasebno i iz njih srednju vrijednost kao rezultat.

Focal loss funkcija gubitaka osmišljena je tako da natjera mrežu da se više fokusira na teže primjere, tako da im pridodaje veće težinske faktore. Također funkcija je osmišljena za slučajeve s jako nebalansiranim setovima podataka gdje klase nisu uravnotežene prema ukupnom postotku piksela slike koje predstavljaju [2]. Takva se funkcija gubitaka u teoriji pokazala obećavajućom za zadani zadatak.

Uz navedene funkcije postoji mnogo drugih funkcija gubitaka koje se ovdje neće navesti. Usto, za svaku je funkciju moguće definirati određeni broj parametara što ima utjecaj i na rezultate modela. Također, moguće je i kreirati takvu funkciju koja kombinira dvije ili više funkcija gubitaka. Na te je načine moguće kreirati takvu funkciju gubitaka koja je specifična za zadani problem. Odlučeno je isprobati utjecaj navedenih funkcija gubitaka, te nekih kombinacija istih, na rezultate treniranja modela. Pošto je ideja usporediti različite funkcije, a ne ostvariti najbolje moguće rezultate, koristit će se standardni parametri, tj. funkcije u svojem najosnovnijem obliku. Pošto je u prvoj rundi treniranja Segnet postigla najbolje rezultate te će se funkcije ispitati samo na toj metodi i rezultati tih treniranja biti će prikazani u poglavlju 6.

5.4. Ostali parametri

Ostali parametri čiji će se utjecaj ispitati na modelu koji je od opisanih metoda i funkcija gubitaka postigao najbolje rezultate su:

- omjer seta za treniranje i validaciju,
- veličina ulaznog skupa podataka,
- dimenzije ulazne slike i
- vrijeme (broj epoha) treniranja.

Promjenom omjera seta za treniranje i validaciju praktički se može povećati set slika za treniranje i time modelu pružiti bolju šansu za učenjem. Istodobno se set slika za validaciju nesmije previše smanjiti kako nebi na taj način nehotice samnjili kvalitetu modela.

Smanjenjem dimenzija ulaznih slika može se značajno ubrzati proces treniranja, potrebno je ispitati isplativost tog postupka. Postupak je isplativ ako ne utječe značajno na kvalitetu rezultata. Ako se smanjuju dimenzije ulaznih slika, tada se to vrijeme može iskoristiti za treniranje s većim brojem epoha. Potrebno je ispitati te parametre i naći optimalni kompromis među njima. Rezultati tih ispitivanja bit će prikazani u poglavlju 6.

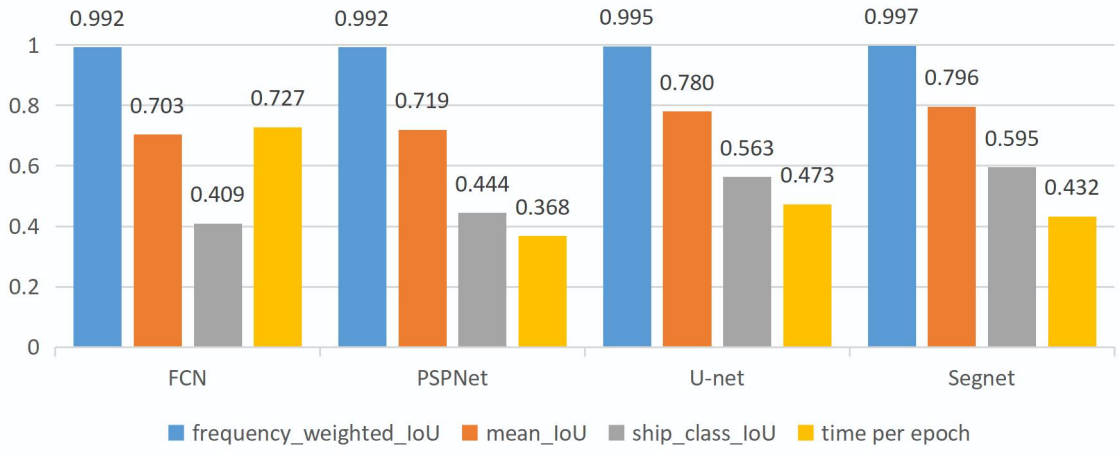
6. REZULTATI

Odlučeno je da se evaluacija rezultata provodi na temelju sljedećih faktora :

- srednje IoU vrijednosti uravnotežene prema omjeru pojavljivanja svake klase,
- sveukupne srednje IoU vrijednosti,
- IoU vrijednosti za klasu plovila,
- vremena potrebnog za treniranje modela i
- subjektivnom ocjenom segmentacije na odabranim slikama iz test skupa podataka.

6.1. Usporedba modela

Za početak, na slici 6.1 ponovno su prikazani na jasniji način preliminarni rezultati prethodno prikazani u 5. poglavlju. Rezultati su dobiveni treniranjem FCN, PSPNet, U-Net i Segnet modela s osnovnim parametrima. Za svaki model prikazane su četiri brojčane vrijednosti. IoU vrijednosti su prethodno pojašnjene u 5. poglavlju. Veća IoU vrijednost predstavlja bolji rezultat i kreće se u rasponu od 0 do 1. Posljednja vrijednost ("time per epoch") predstavlja vrijeme utrošeno za jedan epoh treniranja modela. Veća vrijednost predstavlja lošiji rezultat (više utrošenog vremena). Vrijednost je izračunata je tako da uzeta srednja vrijednost za sva četiri modela i ta je vrijednost skalirana na broj 0.5. Tada, prikazane vrijednosti za sve modele skalirane su na jednak način i dobivena brojčana vrijednost prikazana na grafu.



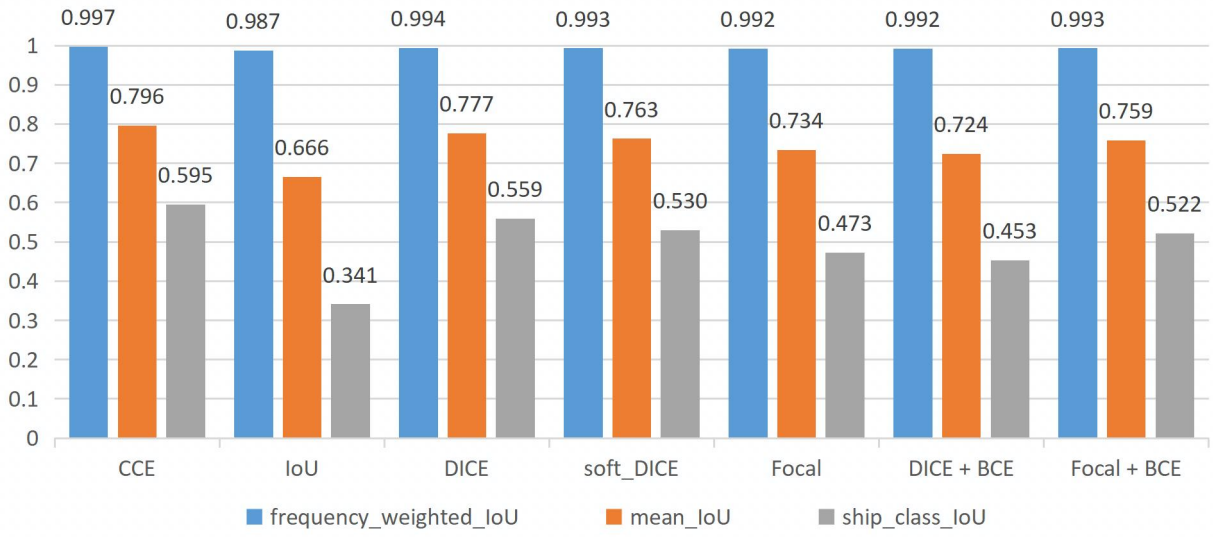
Slika 6.1. Rezultantne IoU vrijednosti i vrijeme treniranja za ispitane modele

Ukratko, kod FCN metode utrošeno je najviše vremena za treniranje i metoda postiže najlošije rezultate. Kod PSPNet metode utrošeno je najmanje vremena za treniranje i metoda postiže bolje

rezultate od FCN metode, ali lošije od U-neta i Segneta. U-net i Segnet metode postižu najbolje rezultate, dok je kod Segnet metode ipak utrošeno nešto manje vremena za treniranje i metoda postiže nešto bolje rezultate.

6.2. Usporedba težinskih funkcija

Sljedeće, odlučeno je ispitati utjecaj težinskih funkcija na treniranje modela. Pošto Segnet metoda postiže najbolje rezultate u prvom ispitivanju, odlučeno je daljnja ispitivanja odraditi samo na toj metodi. Na slici 6.2 prikazani su rezultati dobiveni treniranjem Segnet metode sa šest različitih funkcija gubitaka. CCE označava osnovnu funkciju pod nazivom categorical crossentropy koja je prethodno upotrebljena za treniranje. Sljedeće su funkcije pod nazivom IoU loss, Dice loss, soft Dice loss i Focal loss. Uz to, generirane su dvije funkcije gubitaka kombiniranjem dvoje standardnih funkcija gubitaka. Prva je kombinacija funkcija gubitaka Dice loss i binary crossentropy. Druga je kombinacija funkcija gubitaka Focal loss i binary crossentropy.



Slika 6.2. Rezultati dobiveni treniranjem s različitim funkcijama gubitaka

Na gornjoj slici 6.2 očito je da se najbolji rezultati ostvare primjenom categorical crossentropy funkcijom gubitaka. Ta je funkcija korištena pri početnom treniranju odabranih modela. Ispitivanje drugih funkcija gubitaka nije dovelo do poboljšanja rezultata treniranja Segnet modela. Neke su funkcije poput Dice, soft Dice i kombinacije Focal i BCE funkcije dosegle približno dobre rezultate kao i CCE. Dok je IoU funkcija gubitaka dosegla znatno manje rezultate. Postoji mogućnost da korištenje određenih funkcija gubitaka zahtijeva duže vrijeme

treniranja modela kako bi se ostvarilo bolje rezultate. Unatoč tome, iskušano je pronaći drugi način za poboljšanje rezultata treniranja modela.

6.3. Usporedba seta ulaznih slika

Sljedeće, ispitan je utjecaj izmjene omjera seta slika za treniranje i validaciju. Prethodno se set slika za treniranje sastojao od 20.000 slika a set za validaciju od 5.000 slika. Ukupno to čini 25.000 slika podijeljeno u omjeru 80:20%. Novi je set slika podijeljen u omjeru 90:10%. To rezultira setom za treniranje od 22.500 slika i setom za validaciju od 2.500 slika. Takva raspodjela slika rezultira većim setom slika dostupnih modelu za treniranje, što bi u teoriji trebalo rezultirati boljim modelom, ali i nešto dužim treniranjem. Istodobno, premali set slika za validaciju može pogoršati kvalitetu modela. Na slici 6.3 prikazani su dobiveni rezultati. Uz to, odlučeno je ispitati utjecaj rezolucije ulaznih slika na rezultate treniranja. Smanjenje rezolucije s 768x768 piksela na 384x384 piksela, rezultira četiri puta manjim brojem piksela na svakoj ulaznoj slici. Time je omogućeno puno brze treniranja s istim brojem ulaznih slika. Rezultati tog ispitivanja također su prikazani na slici 6.3.

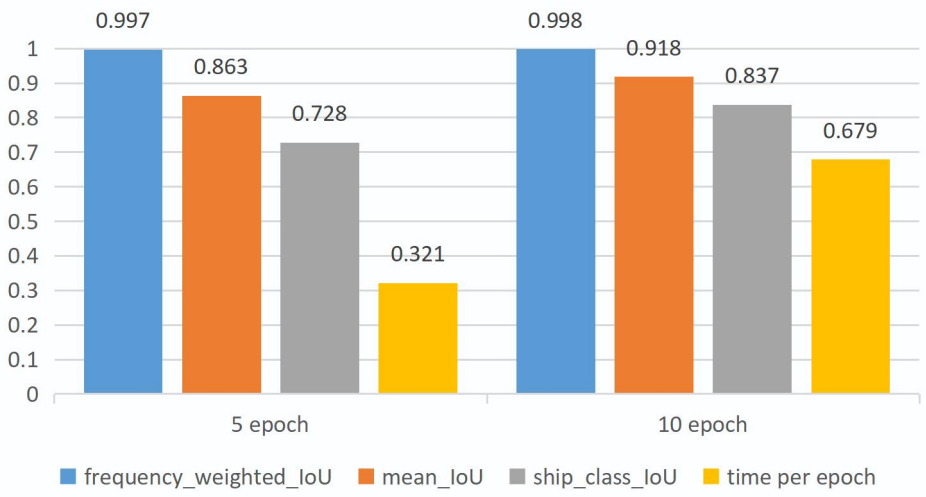
Slika 6.3. Rezultati dobiveni treniranjem s različitim brojem i rezolucijom ulaznih slika

Iz gornje slike očito je kako se povećanjem seta slika za treniranje za 12.5% i smanjenjem seta slika za validaciju za 50% postižu značajno bolji rezultati. Točnije, mean IoU povećao se za ~5,8%, a ship class IoU za ~15,3%. Uz povećanje IoU vrijednost, vrijeme jednog epoha treniranja povećalo se za 12,1%.

Ispitivanjem utjecaja smanjenja rezolucije ulazne slike ustanovilo se kako nema drastičnog pada rezultata u usporedbi s istovjetnim modelom treniranim na slikama veće rezolucije. Dok nema pada kvalitete modela, vrijeme treniranja jednog epoha drastično se smanjilo za ~58%. Time se potvrdila isplativost smanjenja rezolucije ulaznih slika. Uštađeno vrijeme može se upotrijebiti za povećanje broja epoha treniranja modela. Iz tih je rezultata odlučeno daljnje ispitivanje odraditi na modelu u posljednjom stupcu slike 6.3 sa 22.500 slika u setu za treniranje i 2.500 slika u setu za validaciju rezolucije 384x384.

6.4. Usporedba broja epoha treniranja

Odlučeno je ispitati isplativost produljenja vremena treniranja modela povećanjem broja epoha treniranja. Na svim se modelima do sada treniralo 5 epoha. Za potrebe ovog ispitivanja broj epoha treniranja povećan je na 10, čime se približno udvostručuje vrijeme treniranja modela. Na slici 6.4 prikazni su dobiveni rezultati.



Slika 6.4. Rezultati dobiveni s različitim brojem epoha treniranja modela

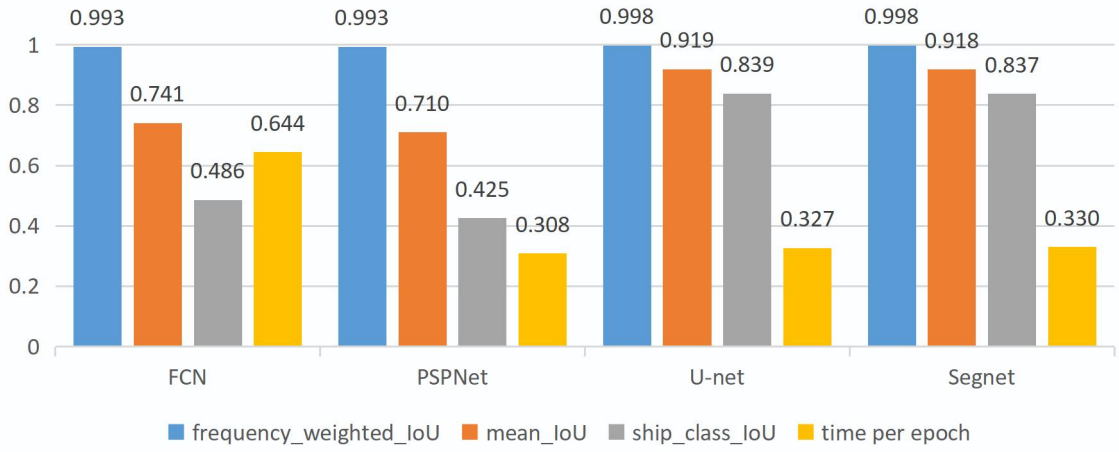
Na gornjoj slici uočljivo je da se povećanjem broja epoha treniranja s 5 na 10 približno dvostruko povećalo vrijeme treniranja. Unatoč tom nedostatku rezultantne IoU vrijednosti značajno su porasle. Srednja IoU vrijednost porasla je za ~6,4% na 0,918 a IoU vrijednost za klasu plovila porasla je za ~15% na 0,837. Iz tih je odličnih rezultata zaključeno da je veći broj epoha treniranja isplativ u pogledu kvalitete dobivenog modela ako je dostupno dovoljno vremena za dugotrajniji proces treniranja.

6.5. Konačni rezultati

Posljednje, pomoću informacija prikupljenih uslijed ispitivanja metoda povećanja kvalitete modela odlučeno je ponoviti treniranje FCN, PSPNet, U-Net i Segnet modela sa sljedećim parametrima:

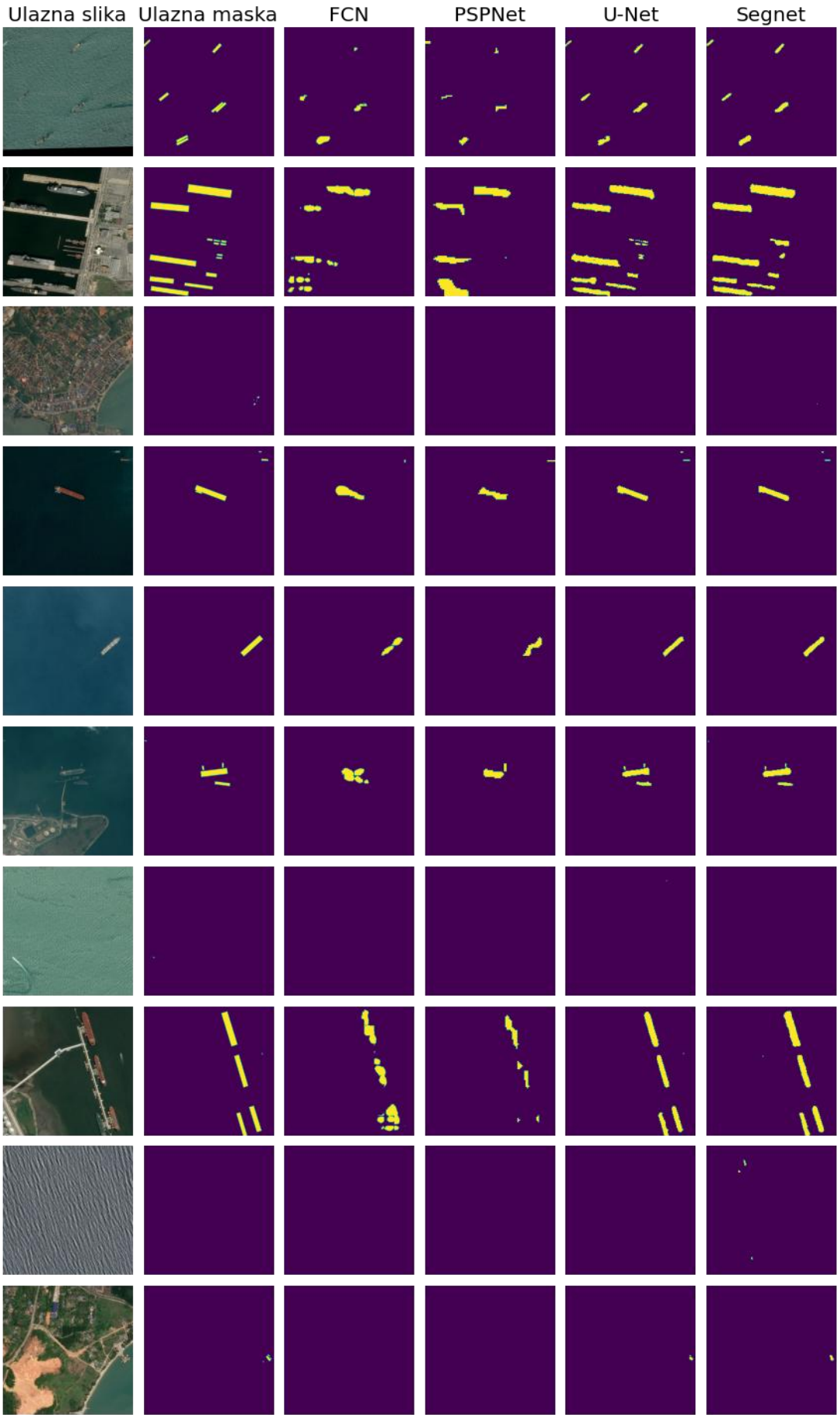
- omjer seta za treniranje i validaciju od 90:10% (22.500 : 2.500 slika),
- categorical crossentropy funkcija gubitaka,
- ulazne slike rezolucije 384x384 i
- 10 epoha treniranja.

Dobiveni rezultati tog treniranja prikazani su na slici 6.5.



Slika 6.5. Konačni rezultati ostvareni primjenom FCN, PSPNet, U-Net i Segnet metoda

Kada se dobiveni rezultati usporede s preliminarnim rezultatima sa slike 6.1 može se uočiti kako je s novo izabranim parametrima FCN metoda ostvarila donekle bolje ali ne značajno dobre rezultate. Vremenski se i dalje pokazala kao najzahtjevnija metoda. PSPNet metoda začudno postiže mrvicu gore rezultate, moguće da metoda jednostavno bolje djeluje na ulaznim slikama veće rezolucije. U-Net i Segnet ponovo postižu najbolje rezultate i to značajno bolje od preliminarnih rezultata. Rezultati dobiveni objema metodama gotovo su jednaki, međutim U-Net metoda postiže najbolje konačne rezultate u pogledu IoU vrijednosti i vremena treniranja. Dobiveni rezultati su uz graf na slici 6.5 dodatno vizualno prikazani na slici 6.6 putem izlaznih predikcija na nekoliko različitih satelitskih slika iz test seta za treniranje. Očito je da U-Net i Segnet metode odabranim parametrima postižu vrlo kvalitetne i zadovoljavajuće rezultate.



Slika 6.6. Ulazne slike i maske te izlazne predikcije različitih metoda

7. ZAKLJUČAK

U ovom radu predstavljeno je pojašnjenje segmentacije slike, pregled razvoja metoda segmentacije slika, uključujući matematičke metode i metode temeljene na umjetnim neuronskim mrežama. Predstavljena je primjena takvih metoda za segmentiranje plovnih objekata sa satelitskih snimaka u morskim, lučkim i priobalnim okruženjima. Konačno, realizirano je nekoliko suvremenih modela temeljenih na umjetnim neuronskim mrežama, ispitane njihove performanse na danom zadatku i prikazani ostvareni rezultati.

Tijekom izrade rada došlo se do zaključka da ne postoji jedna najbolja metoda koja se može koristiti za rješavanja svih problema vezanih za obradu i analizu slike. Različiti problemi zahtijevaju različit pristup zadatku. Nekada je smislenije upotrijebiti neku od matematičkih metoda a nekada neku od naprednijih metoda temeljenih na umjetnim neuronskim mrežama. Neophodno je svakom rješenju pristupiti individualno, s razumijevanjem. Uz to, potrebno je razviti znanje o mogućim rješenjima i pristupima, da se intuitivno može pristupiti rješavanju problema. Također je potrebno biti u toku s razvojem novih tehnologija u ovom brzo razvijajućem polju znanosti.

Po završetku rada zaključujem da su metode segmentacije temeljene na umjetnim neuronskim mrežama izvrstan alat za potrebe zadatka segmentacije slike. Pokazalo se da primjena tih metoda zahtijeva pripremu velikog i kvalitetnog skupa slikovnih podataka. Uz to, važno je pažljivo definirati parametre te osigurati dovoljno vremena za treniranje kvalitetnog modela. Po pitanju segmentiranja plovnih objekata iz satelitskih snimaka primjenom U-Net i Segnet metoda postižu se izvrsni rezultati.

LITERATURA

[1] Jordan, J. (2018). An overview of semantic image segmentation. Data Science, 1-21.

[2] Matcha, A. C. N. (2021). A 2021 guide to Semantic Segmentation. AI & Machine Learning Blog.

[3] Ha, Q., Watanabe, K., Karasawa, T., Ushiku, Y., & Harada, T. (2017, September). MFNet: Towards real-time semantic segmentation for autonomous vehicles with multi-spectral scenes. In 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 5108-5115). IEEE.

[4] Aggarwal, P., Vig, R., Bhadoria, S., & Dethe, C. G. (2011). Role of segmentation in medical imaging: A comparative study. International Journal of Computer Applications, 29(1), 54-61.

[5] Lu, Y., & Shridhar, M. (1996). Character segmentation in handwritten words—an overview. Pattern recognition, 29(1), 77-96.

[6] Datar, S. V., & Bernal, J. G. (2022). Portrait Segmentation Using Deep Learning. arXiv preprint arXiv:2202.02705.

[7] Nakazawa, T., & Kulkarni, D. V. (2019). Anomaly detection and segmentation for wafer defect patterns using deep convolutional encoder–decoder neural network architectures in semiconductor manufacturing. IEEE Transactions on Semiconductor Manufacturing, 32(2), 250-256.

[8] Zhao, Y., Tang, F., Dong, W., Huang, F., & Zhang, X. (2019). Joint face alignment and segmentation via deep multi-task learning. Multimedia Tools and Applications, 78(10), 13131-13148.

[9] Meng, Y., Mok, P. Y., & Jin, X. (2010). Interactive virtual try-on clothing design systems. Computer-Aided Design, 42(4), 310-321.

[10] Li, Y., Xu, S., Luo, X., & Lin, S. (2014). A new algorithm for product image search based on salient edge characterization. Journal of the Association for Information Science and Technology, 65(12), 2534-2551.

[11] Khan, A. M., & Ravi, S. (2013). Image segmentation methods: A comparative study.

[12] Kelian, V. H., Keraf, N. D., Darus, H., Noor, N. M., Nie, E. L. Y., & Ehkan, P. (2020). Proposal for Automatic Vehicle Number Plate Recognition System in POLIMAS. Journal of Engineering and Science Research, 4(1).

[13] Kumar, M. J., Kumar, D. G. R., & Reddy, R. V. K. (2014). Review on image segmentation techniques. International Journal of Scientific Research Engineering & Technology, 3(6), 993-997.

[14] Epicier, T., Faraz, K., Grenier, T., & Ducottet, C. (2021, July). Multiple Object Tracking of Supported Nanoparticles during in situ Environmental TEM Studies of Nanocatalysts. In Microscience Microscopy Congress (mmc) 2021.

[15] Srivastava, A. (2021). Image Segmentation Using K-Means. Blog.

[16] Prasad, S. (2021). What is Image Segmentation. URL: https://www. analytixlabs. co. in/blog/what-is-image-segmentation.

[17] Liu, X., Deng, Z., & Yang, Y. (2019). Recent progress in semantic image segmentation. Artificial Intelligence Review, 52(2), 1089-1106.

[18] Muruganandham, S. (2016). Semantic segmentation of satellite images using deep learning.

[19] Grubišić, I. (2016). Semantička segmentacija slika dubokim konvolucijskim mrežama (Doctoral dissertation, University of Zagreb. Faculty of Electrical Engineering and Computing).

[20] Džomba, K. (2018). Konvolucijske neuronske mreže (Doctoral dissertation, University of Zagreb. Faculty of Science. Department of Mathematics).

[21] Yuan, X., Shi, J., & Gu, L. (2021). A review of deep learning methods for semantic segmentation of remote sensing imagery. Expert Systems with Applications, 169, 114417.

[22] Guo, Y., Liu, Y., Georgiou, T., & Lew, M. S. (2018). A review of semantic segmentation using deep neural networks. International journal of multimedia information retrieval, 7(2), 87-93.

[23] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

[24] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2881-2890).

[25] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[26] Saifi, M. Y., & Singla, J. (2020, March). Deep learning based framework for semantic segmentation of satellite images. In 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC) (pp. 369-374). IEEE.

[27] Rahnemoonfar, M., Chowdhury, T., Murphy, R., & Fernandes, O. (2020). Comprehensive semantic segmentation on high resolution UAV imagery for natural disaster damage assessment. arXiv preprint arXiv:2009.01193.

[28] Lieman-Sifry, J., Lê, M., Lau, F., Sall, S., & Golden, D. (2017). FastVentricle: Cardiac Segmentation with ENet. ArXiv, abs/1704.04296.

[29] Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., & Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In Proceedings of the European conference on computer vision (ECCV) (pp. 801-818).

[30] S interneta: https://www.kaggle.com/c/airbus-ship-detection

[31] Gupta, D. (2017). Image segmentation keras. GitHub repository with implementation of PSPNet.

[32] S interneta: https://www.kaggle.com/code/lukapaladin/diplomski-priprema-slika/notebook

[33] S interneta: https://www.kaggle.com/code/inversion/run-length-decoding-quick-start

[34] S interneta: https://www.kaggle.com/code/hmendonca/u-net-model-with-submission /notebook

[35] S interneta: www.kaggle.com/dataset/ae39860b693075da1e5537cca6ab641a6f61d53f487f758c45e4e5d810177e84

[36] S interneta: https://www.kaggle.com/code/lukapaladin/diplomski-gupta-segmentation-models?scriptVersionId=100028305

[37] S interneta: https://keras.io/api/metrics/segmentation_metrics/#meaniou-class

[38] Espressius, D. (2022). 3 Common Loss Functions for Image Segmentation. Blog.

[39] Jadon, S. (2020, October). A survey of loss functions for semantic segmentation. In 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB) (pp. 1-7). IEEE.

SAŽETAK

U ovom se diplomskom radu predstavio zadatak segmentacije plovnih objekata iz satelitskih snimaka. Segmentacija slike je proces razvrstavanja svakog piksela na slici određenoj klasi. U primjeru ovog rada potrebno je bilo predstaviti i realizirati metode segmentiranja plovnih objekata od ostatka slike. Za početak je u širem smislu pojašnjena segmentacija i njezine primjene u stvarnom svijetu. Nakon toga, predstavljen je pregled dosadašnjeg razvoja od matematičkih metoda raznih kompleksnosti do metoda temeljenih na umjetnim neuronskim mrežama. Sljedeće je predstavljena segmentacija u užem smislu segmentacije objekata iz satelitskih snimaka. Završno je kreiran opsežan i kvalitetan skup slikovnih podataka koji je omogućio treniranje i usporedbu FCN, PSPNet, U-Net i Segnet metoda. Danjim ispitivanjem i prilagodbom raznih parametara primjenom U-Net i Segnet metoda postižu se odlični rezultati.

Ključne riječi: segmentacija slike, umjetne neuronske mreže, satelitske snimke, U-Net, Segnet.

ABSTRACT

In this graduate thesis the task of segmentation of sea vessels from satellite imagery is presented. Image segmentation is the process of sorting each pixel in an image to a specific class. In the example of this work, it was necessary to present and achieve the methods of segmenting sea vessels from the rest of the image. To begin with, segmentation and its applications in the real world are explained in a broader sense. After that, an overview of the development so far from mathematical methods of various complexities to methods based on artificial neural networks was presented. Next, segmentation is presented in the narrow sense of segmentation of objects from satellite imagery. Finally, an extensive and high-quality set of imaging data was created that enabled the training and comparison of FCN, PSPNet, U-Net and Segnet methods. By examining and adjusting various parameters, excellent results are achieved using U-Net and Segnet methods.

Keywords: image segmentation, artificial neural networks, satellite imagery, U-Net, Segnet.

DODATAK A - Programski kod za pripremu ulaznog seta podataka

```
import os
import numpy as np
import pandas as pd
from skimage.io import imread
import matplotlib.pyplot as plt
from skimage.segmentation import mark_boundaries
from skimage.util import montage as montage
import gc; gc.enable()
from skimage.morphology import label

## dekodiranje slika i maski

montage_rgb = lambda x: np.stack([montage(x[:, :, :, i]) for i in range(x.shape
[3])], -1)ship_dir = '../input'

train_image_dir = os.path.join(ship_dir, '../input/airbus-ship-detection/train_v2
')
test_image_dir = os.path.join(ship_dir, '../input/airbus-ship-detection/test_v2')

def multi_rle_encode(img):
    labels = label(img[:, :, 0])
    return [rle_encode(labels==k) for k in np.unique(labels[labels>0])]

def rle_encode(img):
    pixels = img.T.flatten()
    pixels = np.concatenate([[0], pixels, [0]])
    runs = np.where(pixels[1:] != pixels[:-1])[0] + 1
    runs[1::2] -= runs[::2]
    return ' '.join(str(x) for x in runs)

def rle_decode(mask_rle, shape=(768, 768)):
    s = mask_rle.split()
    starts, lengths = [np.asarray(x, dtype=int) for x in (s[0:][::2], s[1:][::2])]
    starts -= 1
    ends = starts + lengths
    img = np.zeros(shape[0]*shape[1], dtype=np.uint8)
    for lo, hi in zip(starts, ends):
        img[lo:hi] = 1
    return img.reshape(shape).T

def masks_as_image(in_mask_list):
    # Take the individual ship masks and create a single mask array for all ships
    all_masks = np.zeros((768, 768), dtype = np.int16)
    #if isinstance(in_mask_list, list):
    for mask in in_mask_list:
        if isinstance(mask, str):
            all_masks += rle_decode(mask)
    return np.expand_dims(all_masks, -1)

## ispis broja slika u uveženom setu za treniranje

print('training data:')
train = os.listdir('../input/airbus-ship-detection/train_v2')
print(len(train), 'training images')
print('-' * 80)
```

```
train_masks = pd.read_csv('../input/airbus-ship-detection/train_ship_segmentation
s_v2.csv')
print(train_masks.shape[0], 'training masks')
print('-' * 80)

## sažimanje i balansiranje seta slika

train_masks['ships'] = train_masks['EncodedPixels'].map(lambda c_row: 1 if isinst
ance(c_row, str) else 0)
unique_img_ids = train_masks.groupby('ImageId').agg({'ships': 'sum'}).reset_index
()
unique_img_ids['has_ship'] = unique_img_ids['ships'].map(lambda x: 1.0 if x>0 els
e 0.0)
unique_img_ids['has_ship_vec'] = unique_img_ids['has_ship'].map(lambda x: [x])#
unique_img_ids['file_size_kb'] = unique_img_ids['ImageId'].map(lambda c_img_id:
             os.stat(os.path.join(train_image_dir,c_img_id)).st_size/1024)

unique_img_ids = unique_img_ids[unique_img_ids['file_size_kb']>50]
unique_img_ids['file_size_kb'].hist()
train_masks.drop(['ships'], axis=1, inplace=True)
unique_img_ids.sample(5)

unique_img_ids['ships'].hist(bins=unique_img_ids['ships'].max())

SAMPLES_PER_GROUP = 2000
balanced_train_df = unique_img_ids.groupby('ships').apply(lambda x: x.sample(SAMP
LES_PER_GROUP) if len(x) > SAMPLES_PER_GROUP else x)
balanced_train_df['ships'].hist(bins=balanced_train_df['ships'].max()+1)

train_df = pd.merge(train_masks, balanced_train_df)
train_df['ships'].hist(bins=train_df['ships'].max()+1)
print(train_df.shape[0], 'training masks')

## dekoddiranje i izrada slika i maski iz ulaznih podataka

BATCH_SIZE = 1
IMG_SCALING = (1,1)

def make_image_gen(in_df, batch_size = BATCH_SIZE):
    all_batches = list(in_df.groupby('ImageId'))
    out_rgb = []
    out_mask = []
    while True:
        np.random.shuffle(all_batches)
        for c_img_id, c_masks in all_batches:
            rgb_path = os.path.join(train_image_dir, c_img_id)
            c_img = imread(rgb_path)
            c_mask = masks_as_image(c_masks['EncodedPixels'].values)
            if IMG_SCALING is not None:
                c_img = c_img[::IMG_SCALING[0], ::IMG_SCALING[1]]
                c_mask = c_mask[::IMG_SCALING[0], ::IMG_SCALING[1]]
            out_rgb += [c_img]
            out_mask += [c_mask]
            if len(out_rgb)>=batch_size:
                yield np.stack(out_rgb, 0)/255.0, np.stack(out_mask, 0)
                out_rgb, out_mask=[], []
                train_gen = make_image_gen(train_df)

gc.collect()
```

```
## spremanje slika u zasebne datoteke za treniranje, validaciju i testiranje

os.makedirs('./train_images'), os.makedirs('./train_masks')
os.makedirs('./valid_images'), os.makedirs('./valid_masks')
os.makedirs('./test_images'), os.makedirs('./test_masks')

TRAIN_IMAGES_DIR = './train_images', TRAIN_MASKS_DIR = './train_masks'
VALID_IMAGES_DIR = './valid_images', VALID_MASKS_DIR = './valid_masks'
TEST_IMAGES_DIR = './test_images', TEST_MASKS_DIR = './test_masks'

from keras.preprocessing.image import save_img
from time import sleep
from progressbar import progressbar

i=0
for img in progressbar(train_gen):

    if i < 22500:
        image=img[0]
        image=image[0,:,:,:]
        train_image_path=os.path.join(TRAIN_IMAGES_DIR, 'image{}.png'.format(i))
        save_img(train_image_path, image)

        mask=img[1]
        mask=mask[0,:,:,:]
        mask=np.clip(mask, 0, 1)
        train_mask_path=os.path.join(TRAIN_MASKS_DIR, 'image{}.png'.format(i))
        save_img(train_mask_path, mask, scale=False)

    elif 22500 <= i < 25000:
        image=img[0]
        image=image[0,:,:,:]
        valid_image_path=os.path.join(VALID_IMAGES_DIR, 'image{}.png'.format(i))
        save_img(valid_image_path, image)

        mask=img[1]
        mask=mask[0,:,:,:]
        mask=np.clip(mask, 0, 1)
        valid_mask_path=os.path.join(VALID_MASKS_DIR, 'image{}.png'.format(i))
        save_img(valid_mask_path, mask, scale=False)

     elif 25000 <= i < 30000:
        image=img[0]
        image=image[0,:,:,:]
        test_image_path=os.path.join(TEST_IMAGES_DIR, 'image{}.png'.format(i))
        save_img(test_image_path, image)

        mask=img[1]
        mask=mask[0,:,:,:]
        mask=np.clip(mask, 0, 1)
        test_mask_path=os.path.join(TEST_MASKS_DIR, 'image{}.png'.format(i))
        save_img(test_mask_path, mask, scale=False)

    elif i == 30000:
        break

    i+=1
```

DODATAK B - Programski kod za treniranje modela

```
import numpy as np
import pandas as pd
import os
import cv2

!cp -r ../input/keras-segmentation/* ./
import keras_segmentation

train_images = '../input/ship-images-with-masks-225k25k-768x768-png/train_images'
train_masks = '../input/ship-images-with-masks-225k25k-768x768-png/train_masks'
valid_images = '../input/ship-images-with-masks-225k25k-768x768-png/valid_images'
valid_masks = '../input/ship-images-with-masks-225k25k-768x768-png/valid_masks'
test_images = '../input/ship-images-with-masks-225k25k-768x768-png/test_images'
test_masks  = '../input/ship-images-with-masks-225k25k-768x768-png/test_masks'

## odabir modela i dimenzija ulazne slike

MODEL_NAME = 'vgg_segnet'
input_dimensions= 384

print('Odabrani model za ovu verziju je: ', MODEL_NAME)

# FCN

if MODEL_NAME == 'fcn_32':
    from keras_segmentation.models.fcn import fcn_32
    model = fcn_32 (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'fcn_32_vgg':
    from keras_segmentation.models.fcn import fcn_32_vgg
    model = fcn_32_vgg (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'fcn_32_resnet50':
    from keras_segmentation.models.fcn import fcn_32_resnet50
    model = fcn_32_resnet50 (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)
# PSPNet

elif MODEL_NAME == 'pspnet':
    from keras_segmentation.models.pspnet import pspnet
    model = pspnet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'vgg_pspnet':
    from keras_segmentation.models.pspnet import vgg_pspnet
    model = vgg_pspnet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)
```

```
elif MODEL_NAME == 'resnet50_pspnet':
    from keras_segmentation.models.pspnet import resnet50_pspnet
    model = resnet50_pspnet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

# U-Net

elif MODEL_NAME == 'unet':
    from keras_segmentation.models.unet import unet
    model = unet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'vgg_unet':
    from keras_segmentation.models.unet import vgg_unet
    model = vgg_unet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'resnet50_unet':
    from keras_segmentation.models.unet import resnet50_unet
    model = resnet50_unet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

# Segnet

elif MODEL_NAME == 'segnet':
    from keras_segmentation.models.segnet import segnet
    model = segnet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'vgg_segnet':
    from keras_segmentation.models.segnet import vgg_segnet
    model = vgg_segnet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

elif MODEL_NAME == 'resnet50_segnet':
    from keras_segmentation.models.segnet import resnet50_segnet
    model = resnet50_segnet (n_classes=2,
                    input_height=input_dimensions,
                    input_width=input_dimensions)

else:
    print(' *MODEL_NAME* nije iz liste dopuštenih modela')

# kompiliranje i treniranje modela

model.compile()
model.summary()

model.train(
    train_images = train_images  ,
    train_annotations = train_masks ,
    batch_size=4,
```

```
    steps_per_epoch=5625,
    val_images=valid_images,
    val_annotations=valid_masks,
    val_batch_size=4,
    val_steps_per_epoch=625,
    checkpoints_path = "/tmp/temp_model",
    verify_dataset=False,
    epochs=10,
    )

model.save('{}.h5'.format(MODEL_NAME))

## evaluacija modela na test skupu podataka

print(model.evaluate_segmentation( inp_images_dir=test_images  , annotations_dir=
test_masks ) )

## evaluacija modela na odabranim slikama

from keras.preprocessing.image import save_img
import matplotlib.pyplot as plt
from PIL import Image

images_list=[25001,25011,25017,25019,25027,25031,25040,25043,25050,25056]

for i in images_list:

    inp='../input/ship-images-with-masks-train-20000-768-png/test_images/image{}.
        png'.format(i)
    out= model.predict_segmentation(inp=inp)
    out = (out*255).astype(np.uint8) # scale to 0-255  range and convert to int
    cv2.imwrite('prediction{}.png'.format(i), out) # better use png
```