SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1156

DUBOKI MODELI ZA SEMANTIČKU SEGMENTACIJU I DETEKCIJU ANOMALIJA

Marko Čengić

Zagreb, srpanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1156

DUBOKI MODELI ZA SEMANTIČKU SEGMENTACIJU I DETEKCIJU ANOMALIJA

Marko Čengić

Zagreb, srpanj 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 10. ožujka 2023.

ZAVRŠNI ZADATAK br. 1156

Pristupnik:	**Marko Čengić (0036532610)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	prof. dr. sc. Siniša Šegvić

Zadatak: **Duboki modeli za semantičku segmentaciju i detekciju anomalija**

Opis zadatka:

Semantička segmentacija važan je zadatak računalnog vida s mnogim zanimljivim primjenama. Međutim, standardno nadzirano učenje vrlo je osjetljivo na pojavu anomalija u ispitnih slikama. Zbog toga segmentacija anomalija predstavlja vrlo vrijedan dodatak standardnim postupcima za diskriminativnu gustu predikciju. U okviru rada, potrebno je odabrati okvir za automatsku diferencijaciju te upoznati biblioteke za rukovanje matricama i slikama. Proučiti i ukratko opisati postojeće segmentacijske arhitekture. Uhodati postupke učenja modela te validiranje hiperparametara. Isprobati različite pristupe za segmentaciju anomalija utemeljene na analizi predikcija modela. Vrednovati analizirane pristupe te prikazati i usporediti postignutu točnost. Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, kao i potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 9. lipnja 2023.

Zahvaljujem se prof. dr. sc. Siniši Šegviću, univ. bacc. ing. Jeleni Bratulić i mag. ing. Petri Bevandić na pomoći pri izradi rada.

Sadržaj

Uvod

Računalni vid je grana računalne znanosti koja se bavi analizom i interpretacijom vizualnih informacija. Velike inovacije u računalnom vidu omogućile su primjenu te tehnologije u raznim područjima, poput medicinske dijagnostike, autonomnih vozila, nadzora sigurnosti i drugih. Unatoč napretku, područje računalnog vida još uvijek ima mnogo otvorenih problema. Jedan od tih ključnih izazova je prepoznavanje i klasifikacija objekata na slici, odnosno semantička segmentacija.

Semantička segmentacija je proces klasifikacije piksela na slici ili u videu u određene semantičke kategorije, kao što su automobili ili ljudi. U ovom procesu najčešće se koriste duboki konvolucijski modeli.

Duboki konvolucijski modeli su neuronski mrežni modeli koji sadrže konvolucijske slojeve, tj. slojeve koji grupiraju informacije kako bi dobili generalizaciju. Takvi modeli su se pokazali izuzetno korisnima u računalnom vidu. Međutim, sami duboki konvolucijski modeli često nisu dovoljni za semantičku segmentaciju, jer imaju poteškoću u prepoznavanju nepoznatih kategorija, kao što su ovce na cesti.

Stoga se koriste metode detekcije anomalija u semantičkoj segmentaciji, što je upravo područje kojim se ovaj rad bavi.

1. Strojno učenje

Umjetna neuronska mreža je skup međusobno povezanih jednostavnih procesnih elemenata (neurona), najčešće poredanih u slojeve, čija se funkcionalnost temelji na biološkom neuronu i koji služe distribuiranoj paralelnoj obradi podataka[1]. Koristi različite matematičke i aktivacijske funkcije kako bi se odradio proces učenja.

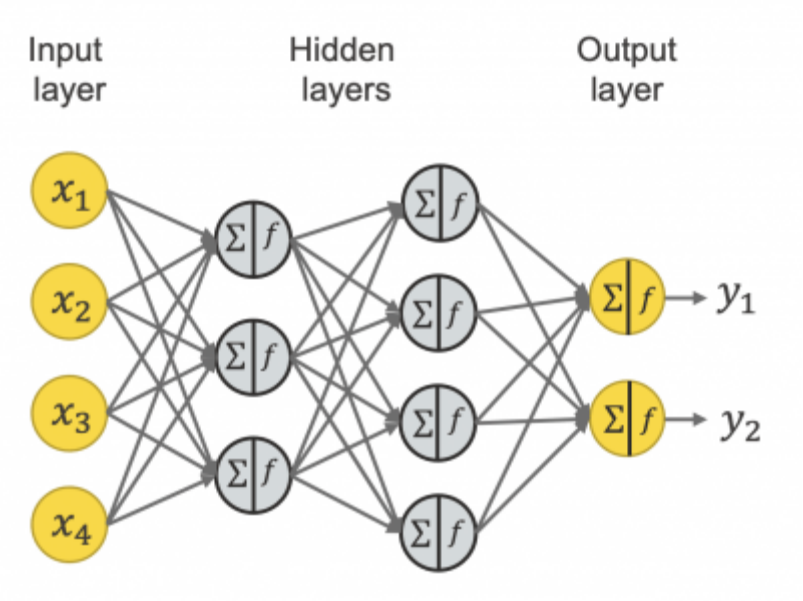
1.1. Učenje modela

Učenje umjetne neuronske mreže je proces prilagodbe težinskih vrijednosti neurona kako bi se postigao željeni izlaz na temelju ulaznih podataka. U početku su težine određene slučajno, ali se procesom treniranja prilagođavaju podatcima. U našem slučaju dobivamo neki ulaz x, s očekivanim izlazom y, na temelju kojih se prilagođava model. Takav tip učenja se naziva nadzirano učenje. Prvo se podatci propagiraju unaprijed, pri čemu se ulaz prosljeđuje kroz arhitekturu mreže kako je definirano pomoću funkcija i težina. Zatim zadnji sloj vraća neki rezultat koji se uspoređuje s dobivenom oznakom y. Nakon toga se izračunava gubitak na temelju neke funkcije gubitka i s tom funkcijom prolazi nazad po mreži tako da se prilagođavaju težinske vrijednosti neurona[2]. Poprilično je bitno da istrenirana mreža ima sposobnost rada s neviđenim podatcima, a ne samo onima koji su joj dani. To svojstvo mreže se zove generalizacija.

1.1.1. Modeli strojnog učenja

Računanje izlaza mreže

Umjetna neuronska mreža se sastoji od više slojeva. Prvi sloj se zove ulazni sloj, a u njega stavljamo podatke. Ostali slojevi su skriveni slojevi, koji aktiviraju određene funkcije nad svojim ulazima. Na kraju je izlazni sloj, koji vraća rezultate. Izlaz umjetne neuronske mreže je zapravo izlaz zadnjeg sloja neuronske mreže (Slika 1.1).



Slika 1.1: Slojevi potpuno povezane neuronske mreže, gdje je na ulaz prima 4 podatka koji se onda množe sa zadanim težinama i zbrajaju u sljedećem sloju. Zatim se zbroj provede kroz neku funkciju, i tako kroz oba skrivena sloja, sve dok ne dođe do 2 izlaza. Preuzeto sa:[17]

Izlaz neuronske mreže u tom slučaju je lista y izlaza (1.1):

$$y = \varphi\left(\sum_{i=1}^{n} x_i y_i + b\right) \tag{1.1}$$

Prijenosne funkcije

Prijenosne funkcije određuju izlaz neurona. Zbog prijenosnih funkcija omogućeno je modeliranje nelinearnih odnosa, što je nužno za ikakve kompleksnije probleme. To jest, model bez prijenosnih funkcija je čisto linearna kombinacija njegovih ulaza, te time ma koliko god slojeva nadodamo može se svesti na samo jedan. Najpoznatije prijenosne funkcije su[1]:

Funkcija skoka

Formula za funkciju skoka je (1.3):

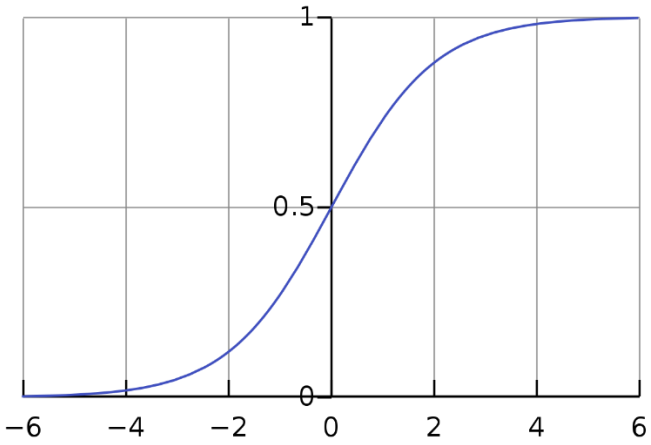
$$f(x) = \begin{cases} 0, x < 0 \\ 1, x > 0 \end{cases} \tag{1.3}$$

Jedna od osnovnih prijenosnih funkcija, koja se danas iznimno rijetko koristi zbog činjenice da nije derivabilna u točci x = 0, a u svim ostalim točkama ima gradijent 0. Time se ne može koristiti u postupcima učenja koji se temelje na gradijentima, poput propagacije unazad, jedne od najkorištenijih metoda današnjice.

Sigmoidalna funkcija

Formula za sigmoidalnu funkciju (Slika 1.2) je (1.4):

$$f(x) = \frac{1}{1+e^{-x}} \tag{1.4}$$



Slika 1.2: Sigmoidalna funkcija. Primijeti se da za svaki x manji od 0 daje vrijednost između 0 i 0.5, a za svaki x veći od nula, daje vrijednost između 0.5 i 1. Preuzeto sa: [18]

Jedna od danas popularnijih prijelaznih funkcija. Ograničuje vrijednost na interval od 0 do 1, ali za razliku od funkcije skoka, derivabilna je u svakoj točci, te time daleko preferirana pri korištenju propagacije unatrag. Jedna od mana joj je da kada su vrijednosti preblizu nule, poništava se djelovanje gradijenta, ali to nije toliki problem kao kod funkcije skoka. Specifično je pogodna za probleme binarne klasifikacije.

Softmax

Formula za softmax je (1.5):

$$f(s_j) = e^{s_j} / \sum_k^K e^{s_k} \tag{1.5}$$

Softmax je samo poopćenje sigmoidne funkcije na K klasa [3]. Tako, primjenom softmaxa dobivamo vjerojatnost pripadanja ulaza nekoj od K klasa. Često se koristi kao zadnja aktivacijska funkcija u modelu za klasifikaciju, tako da rezultat bude postotak.

Zglobnica

Formula za zglobnicu je (1.6):

$$f(x) = \max(0, x) \tag{1.6}$$

Danas najpopularnija prijenosna funkcija. Glavna prednost joj je kvalitetna i brza propagacije gradijenata, a glavna mana situacija u kojoj određeni neuroni postaju neaktivni u praktički svim situacijama.

1.1.2. Gubitak

Funkcije gubitka su jedan od osnovnih dijelova procesa učenja modela. One nam govore koliko nam je predikcija daleko od stvarne vrijednosti. Učenje se svodi na minimizaciju gubitka. Postoji mnogo takvih funkcija, pogodnih za specifične situacije.

Unakrsna entropija

Unakrsna entropija je najkorištenija funkcija gubitka za klasifikaciju. Pokazuje dobre rezultate i u binarnoj i u višeklasnoj klasifikaciji. Formula za jedan uzorak podataka izgleda ovako (1.7):

$$L_{CE} = -\sum_{n=1}^{K}(y_n \log \hat{y}_n) \tag{1.7}$$

y_n je očekivana vrijednost klase n dok je $\hat{y}_n$ predviđena vjerojatnost klase n. Za računanje te vjerojatnosti se koristi funkcija softmax iz prošlog poglavlja. Minus je prisutan zato da minimiziramo, a ne maksimiziramo gubitak. Minimiziranjem unakrsne entropije osiguravamo da se naše predikcije klasa približavaju raspodjeli gdje je jedna klasa 1, a sve ostale 0, što nam je i najkorisnija raspodjela vjerojatnosti.

1.1.3. Algoritmi optimizacije

Algoritmi optimizacije koriste se za pronalazak minimuma funkcije gubitka. Nakon svakog izračuna gubitka, taj gubitak se propagira unatrag.

Postupak propagacije pogreške unatrag je postupak učenja neuronskih mreža koji se temelji na učinkovitom izračunu svih parcijalnih derivacija (gradijenata) i njihovoj primjeni na određivanje iznosa kojim korigiramo svaku od težina[1].

Nakon propagacije unatrag slijedi pomicanje u smjeru suprotnom od gradijenta. Tada se koriste optimizacijske funkcije. Optimizacija se može provoditi na jedom podatku, manjoj grupi ili cijelom skupu podataka. Pokazalo se da je najbolji slučaj s manjom grupom podataka[4].

Gradijent spust

Najintuitivniji algoritam optimizacije u kojemu se pomiče za minus gradijent. Zato što funkcija gubitka ne mora biti konveksna, ne možemo biti sigurni da ćemo zapravo doći do globalnog minimuma.

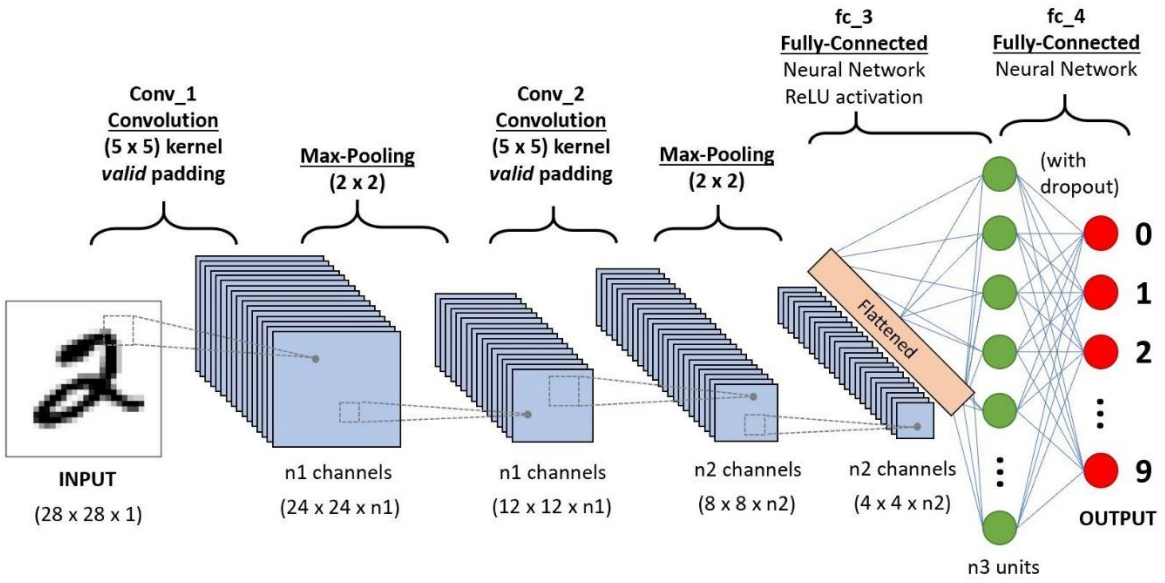
Adam

Adam je poprilično sličan gradijent spustu, samo ima zadržavanje zaleta. To znači da se pri odluci za trenutni pomak u obzir uzimaju i prijašnji gradijenti, a ne samo trenutni. Specifično što pamti je eksponencijalno padajući prosjek kvadrata prošlih gradijenata i eksponencijalno padajući prosjek prošlih gradijenata, koje onda koristi u svom izračunu. Adam je poprilično dobar algoritam optimizacije, ali nije prigodan za sve situacije. Pokazao se nešto gori u generalizaciji od drugih algoritama (SGD). Također se pokazalo da je dosta brz, što pridonosi njegovoj rasprostranjenosti.

1.2. Konvolucijske neuronske mreže

Konvolucijska neuronska mreža je vrsta dubokog modela koja obrađuje podatke oblika matrice. Digitalna slika je matrica u kojoj svako polje predstavlja svojstva piksela, te time pogodna za obradu konvolucijskom mrežom. Sama mreža je inspirirana vizualnim korteksom životinja[5], a zapravo je matematička konstrukcija koja se obično sastoji od tri vrste slojeva: konvolucijskih slojeva, slojeva sažimanja i potpuno povezanih slojeva. Konvolucijski sloj i sloj sažimanja, obavljaju izdvajanje značajki, dok treći, potpuno povezani sloj, mapira izdvojene značajke u konačni rezultat, poput klasifikacije (Slika 1.3). Kao što i samo ime kaže potpuno povezani sloj je sloj u kojemu je svaki neuron prošlog sloja povezan sa svim neuronima sljedećeg sloja. Taj potpuno povezani sloj je zadnji u konvolucijskoj mreži.

Konvolucijski modeli grupiraju piksele u značajke, te time brže uče i bolje generaliziraju naspram potpuno povezanih modela, koji zahtijevaju iznimno velik broj parametara za obradu slike. Konvolucijske mreže najčešće sadrže i neke nelinearne aktivacijske slojeve u kojima se primjenjuju neke od prije navedenih prijenosnih funkcija, npr. funkcija zglobnice. Još uvijek, glavno svojstvo i prednost konvolucijskih mreža su konvolucijski slojevi.



Slika 1.3: Osnovna arhitektura konvolucijske neuronske mreže koja se sastoji od 6 slojeva. Prvo dobiva na ulaz sliku broja od 28x28 piksela, a zatim provodi konvoluciju s jezgrom 5x5, pa sažimanje maksimalnom vrijednošću s filtrom veličine 2x2. Taj postupak se ponavlja dva puta, sve dok ne dođe do potpuno povezanih slojeva. Na kraju vraća šanse svakog broja da je ulaz u mrežu slika njega. Preuzeto sa: [19]

1.2.1. Konvolucijski sloj

Konvolucijski sloj (Slika 1.4) je specijalizirana vrsta linearnog sloja koji se koristi za ekstrakciju značajki. Tamo se mala matrica brojeva, koja se naziva jezgra K, primjenjuje na ulaz koji se naziva tenzor. Taj tenzor je veća matrica brojeva od same jezgre. U početku se redom prolazi kroz ulazni tenzor, tako da se kontinuirano uzimaju uzastopni isječci iz njega. Ti isječci moraju biti veličine jezgre. Zatim se svaki član tih isječka množi s članom na istoj poziciji jezgre, te se dobije nova matrica dimenzija jezgre. Sadržaj te matrice se samo zbroji i rezultat stavi u odgovarajuće polje nove matrice, koja se zove mapa značajki. Ovaj postupak se ponavlja primjenom višestrukih jezgara za formiranje proizvoljnog broja mapa značajki, koje predstavljaju različite karakteristike ulaznih tenzora. Različite jezgre se stoga mogu smatrati različitim ekstraktorima značajki.

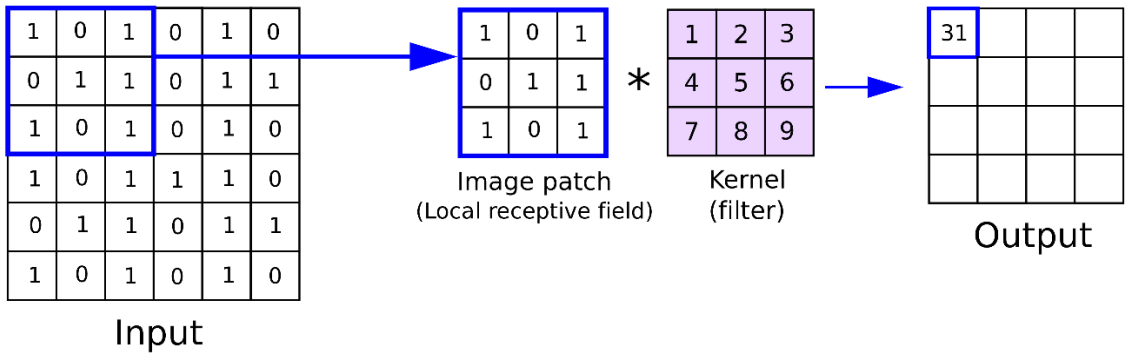
Dva ključna hiperparametra koja definiraju operaciju konvolucije su veličina i broj jezgara. Veličina jezgre je obično 3 × 3, ali ponekad 5 × 5 ili 7 × 7[5].

Udaljenost između dva uzastopna položaja jezgre naziva se korak. Uobičajeni izbor koraka je 1. Ponekad se koristi korak veći od 1 kako bi se postiglo smanjivanje uzorkovanja mapa

značajki. Daleko češće korištena alternativa za izvođenje smanjenja uzorkovanja je sloj sažimanja.

Zadnji bitan hiperparametar konvolucije je nadopunjavanje nulama. Tu se određuje koliko će se proširiti ulazni tenzor s nulama. To primarno služi za manipuliranje dimenzija mape značajki.

Kako je konvolucija linearna operacija nakon nje najčešće slijedi nelinearna aktivacijska funkcija.



Slika 1.4 Konvolucijski sloj s jezgrom 3x3. Prima ulaz veličine 6x6 i vraća mapu značajki veličine 4x4. Prvo uzima 3x3 dio ulazne matrice, zatim množi svako polje matrice s poljem jezgre, te ih sve zbroji i zapiše u mapu značajki. Preuzeto sa: [20]

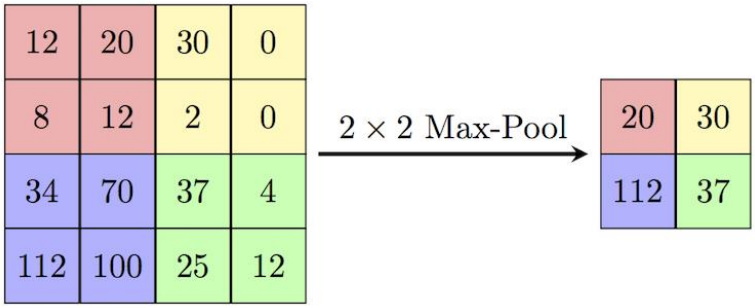
1.2.2. Sloj sažimanja

Sloj sažimanja smanjuje dimenzionalnost mapa značajki kako bi se uvela invarijantnost na male promjene i smanjio broj naknadnih parametara koji se moraju učiti. Postoje dvije popularne verzije sloja sažimanja: sažimanje maksimalnom vrijednošću i sažimanje srednjom vrijednošću.

Sažimanje maksimalnom vrijednošću

Najpopularniji oblik operacije sažimanja je sažimanje maksimalnom vrijednošću. Ona prvo uzima manju matricu iz mape značajki, zatim iz te manje matrice uzima maksimalnu vrijednost, koju onda stavlja u novu izlaznu matricu, a sve ostale vrijednosti mape značajki u manjoj matrici ignorira. U praksi se obično koristi maksimalno udruživanje s filtrom veličine 2×2 i s korakom 2 (Slika 1.5). Time se smanjuje dimenzija mapa značajki u ravnini za faktor 2. Za razliku od visine i širine, dimenzija dubine mapa značajki ostaje nepromijenjena.

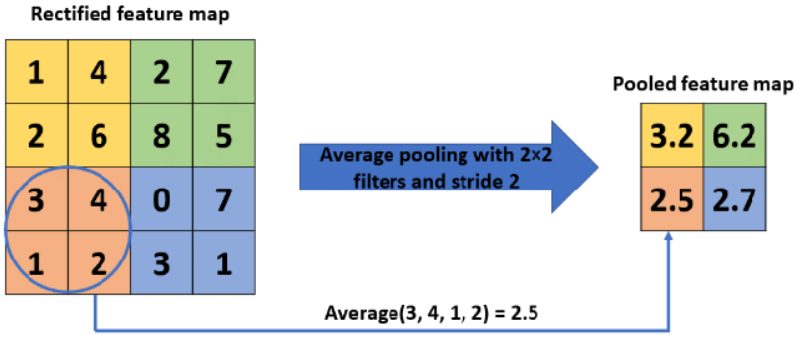
Sažimanje maksimalnom vrijednošću se specifično pokazalo bolje od sažimanje srednjom vrijednošću u generalizaciji. Činjenica da uzima samo maksimalnu vrijednost joj bolje omogućuje ignoriranje manjih promjena.



Slika 1.5: Sažimanje maksimalnom vrijednošću veličine filtra 2x2 i korakom 2. Prima matricu veličine 4x4, a vraća 2x2. Svako polje izlazne matrice je najveće polje jednako obojanog kvadrata ulazne matrice. Preuzeto sa: [21]

Sažimanje srednjom vrijednošću

Još jedna operacija sažimanja vrijedna pažnje je sažimanje srednjom vrijednošću. To je bila primarna metoda sažimanja prije sažimanje maksimalnom vrijednošću. Ona prvo uzima manju matricu iz mape značajki, zatim iz te manje matrice računa srednju vrijednost, te je onda stavlja u novu izlaznu matricu (Slika 1.6).

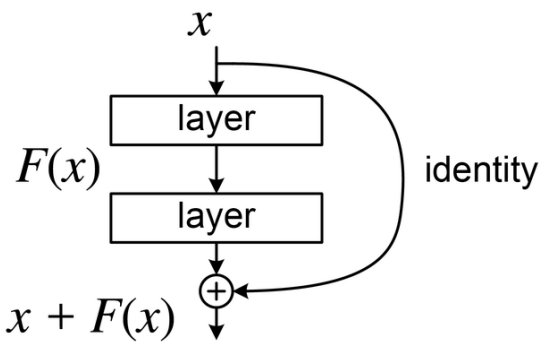


Slika 1.6: Sažimanje srednjom vrijednošću veličine filtra 2x2 i korakom 2. Prima matricu veličine 4x4, a vraća 2x2. Svako polje izlazne matrice je srednja vrijednost polja jednako obojanog kvadrata ulazne matrice. preuzeto sa: [22]

1.2.3. Resnet 18 i rezidualne mreže

Rezidualne mreže, poput Resneta18, su nastale kao rješenje problema učenja iznimno dubokih mreže. Problem je da duboke neuronske mreže, pa time i duboke konvolucijske mreže, povećanjem dubine, nakon određene granice, kreću davati lošije rezultate.

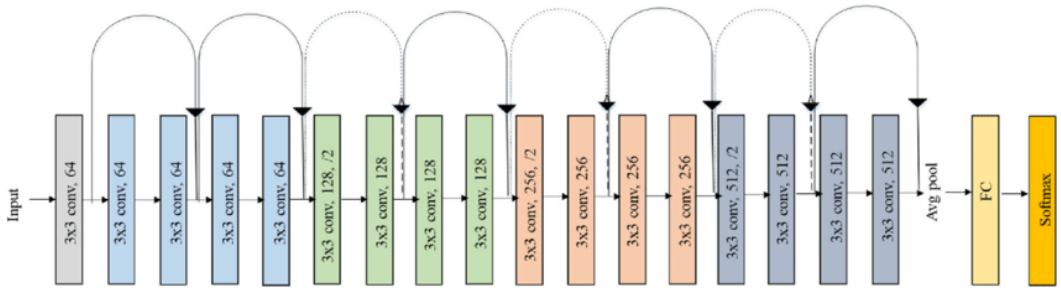
Rezidualne mreže taj problem rješavaju tako da uvode aditivne preskočne veze[6]. Arhitektura mreže je onda građena od rezidualnih blokova, manjih kombinacija slojeva u kojima se ulaz u te slojeve nadodaje na izlaz (Slika 1.7).



Slika 1.7: Rezidualni blok s ulazom x, koji prolaženjem kroz dva sloja vraća izlaz F(x). Na kraju se ulaz i izlaz zbrajaju da daju izlaz cijelog rezidualnog bloka. Preuzeto sa: [23]

Međutim, ponekad ulaz x i izlaz F(x) neće imati iste dimenzije. Podsjetimo se da operacija konvolucije obično smanjuje prostornu dimenzije slike. Tada se preslikavanje identiteta množi s linearnom projekcijom W, te time bi se proširili kanali prečaca kako bi odgovarali ostatku.

Resnet18 je samo rezidualna mreža s 18 slojeva. Postoji još mnogo popularnih Resnet modela, poput Resnet34, Resnet50 i Resnet101. Resnet18 je najbrži od njih, ali ovisno o situaciji ne i najpreciznji. Time je i resnet18 model dobra bazu modela za semantičku segmentaciju.



Slika 1.8: Arhitektura Resnet18, sa 18 slojeva. U početku se ulaz provuče kroz konvoluciju, a zatim kroz 8 rezidualnih blokova, nakon čega vraća rezultat. Preuzeto sa: [24]

2. Semantička segmentacija

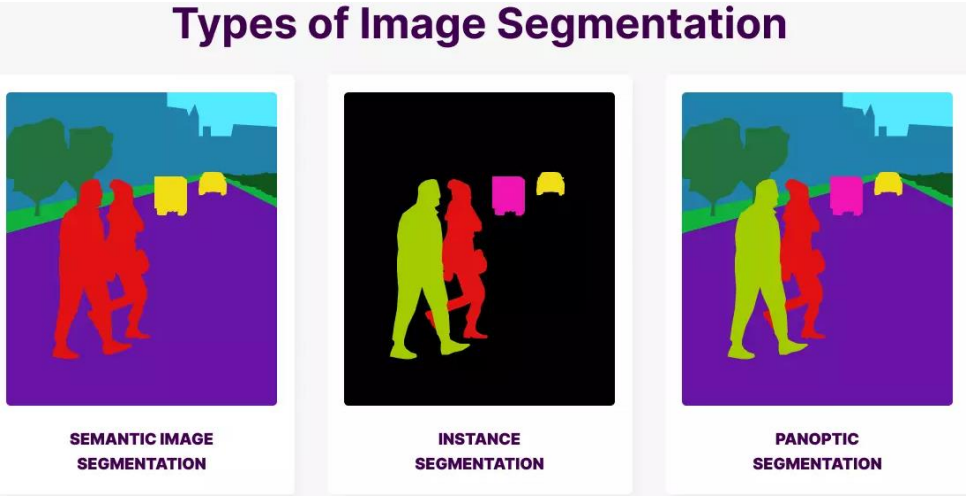
Često se u računalnom vidu susrećemo s problemom segmentacije. To jest, zanima nas klasifikacija dijelova slike. Postoje tri vrste segmentacija (Slika 2.1):

- Semantička segmentacija
- Segmentacija instanci
- Panoptička segmentacija

Semantička segmentacija je zadatak u računalnom vidu kojemu je cilj kategorizirati svaki piksel na slici u neku klasu. Rezultat semantičke segmentacije bi trebala biti gusta mapa segmentacije slike po pikselima, gdje je svaki piksel dodijeljen određenoj klasi. Npr. Pokazati koji sve pikseli na slici pripadaju ljudima.

Segmentacija instanci je pak zadatak u računalnom vidu kojemu je cilj kategorizirati svaki piksel na slici u neki objekt, neovisno o njegovoj klasi. Rezultat segmentacija instanci bi trebala biti gusta mapa segmentacije slike po pikselima, gdje je svaki piksel dodijeljen određenom objektu. Npr. Pokazati koji sve pikseli na slici pripadaju Anti, a koji Juri.

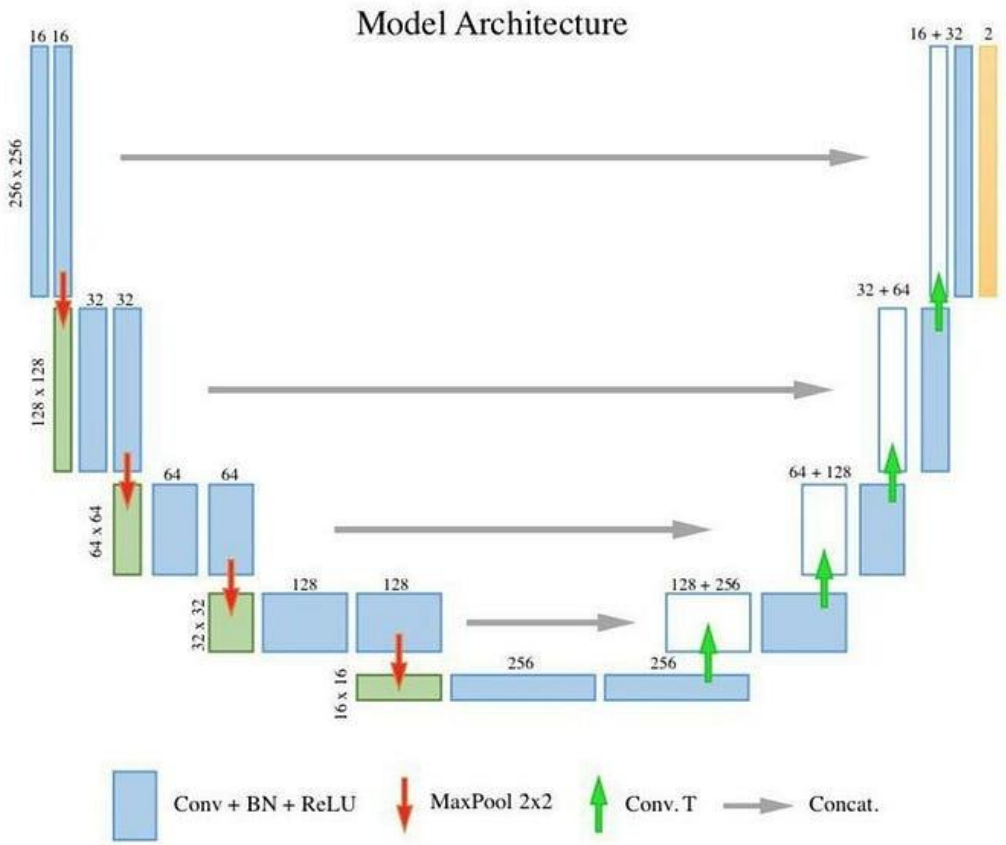
Panoptička segmentacija je samo kombinacija semantičke segmentacije i segmentacije instanci, gdje se kategoriziraju pikseli i po klasi i po objektu.



Slika 2.1: Usporedba semantičke segmentacije, segmentacije instanci i panoptičke segmentacije. U prvoj su sve instance iste klase označene istom bojom, u drugoj različitom bojom, ali bez klasa koje nemaju instance, a u trećoj je i svaka instanca i svaka klasa različite boje. Preuzeto sa: [25]

2.1. U-net model

U-net je najpopularniji model s ljestvičastim naduzorkovanjem koji se koristi za provođenje semantičke segmentacije, te je time dobar primjer modela s ljestvičastim naduzorkovanjem. On se gradi na temelju konvolucijske mreže. Glavno svojstvo, po kojemu je i dobio ime je njegova U arhitektura (Slika 2.2), koja se sastoji od enkoderskog puta i dekoderskog puta. Enkoderski put slijedi tipičnu arhitekturu konvolucijske mreže. Dok u svakom koraku enkoderskog puta udvostručujemo broj značajnih kanala, u svakom koraku dekoderskog puta pak povećavamo mape značajki i radimo 2x2 konvoluciju koja prepolovljuje broj kanala značajki[7].



Slika 2.2: Arhitektura U-net modela. Prima ulaz dimenzija 256x256 i provodi ga dva puta kroz Conv + BN + Relu pa kroz jedan sloj sažimanja maksimalnom vrijednošću. To radi sve dok ne dođe do 256 značajnih kanala kada prestaje raditi sažimanje i kreće sa konvolucijom u putu naduzorkovanja. Izlazi iz enkoderskog puta se onda pribrajaju izlazima iz dekodirskog puta, sve dok ne dođemo do kraja. Preuzeto sa: [26]

Konvolucija u putu naduzorkovanja je samo proces suprotan dosadašnjoj konvoluciji. Glavna ideja iza dekoderskog puta je da slojevi tog puta povećavaju razlučivosti izlaza.

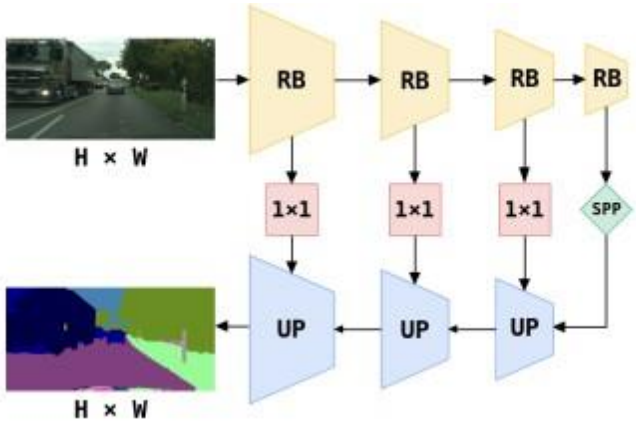
Sada, da bi se taj izlaz s povećanom razlučivosti lokalizirao, kombinira se sa značajkama visoke razlučivosti iz enkoderskog puta[7]. To kombiniranje se naziva preskočna veza. Zbog tih svojstva U-net se pokazao precizniji od konvolucijskih mreža u području semantičke segmentacije.

2.2. Swiftnet

Swiftnet[8] je efikasan model za semantičku segmentaciju s ljestvičastim naduzorkovanjem, te dizajniran za predikcije u stvarnom vremenu. Sastoji se od segmentacijskog enkodera, dekodera za naduzorkovanje i modula za povećanje receptivnog polja. Enkoder je MobileNet V2 ili Resnet18 model. Oba su predtrenirana na ImageNetu. Dekoder se sastoji od tri grupe za naduzorkovanje. Te grupe imaju dva ulaza: značajke niske razlučivosti, nad kojima će se provesti naduzorkovanje i značajke iz slojeva enkodera. Značajke niske razlučivosti prvo se naduzorkuju s bilinearnom interpolacijom na istu razlučivost kao i značajke enkodera. Zatim se ti elementi naduzorkovanih značajki zbrajaju s elementima značajki enkodera. Na kraju se samo rezultat tog zbrajanja stapa pomoću 3x3 konvolucije.

Swiftnet se može vrtjeti u dvije konfiguracije: model s piramidalnom fuzijom i model s prostornim piramidalnim sažimanjem. Obe metode se koriste za povećanje receptivnog polja. U svojim eksperimentima sam koristio Swiftnet baziran na ResNet18 s prostornim piramidalnim sažimanjem.

Način na koji taj model radi je da se prvo slika provuče kroz enkoder, baziran na resnet18, gdje u svakoj konvolucijskoj grupi, osim zadnje, postoji preskočna veza sa slojem naduzorkovanja. Zadnja konvolucijska grupa, pak svoj rezultat šalje u funkciju prostornog piramidalnog sažimanja. Na kraju se rezultat te funkcije provuče kroz tri grupe za naduzorkovanje, čime ovaj proces završava (Slika 2.3).



Slika 2.3: Arhitektura swiftneta sa prostornim piramidalnim sažimanjem. Preuzeto sa: [7]

2.3. Primjene semantičke segmentacije

Semantička segmentacija se često koristi u kompleksnim zadatcima računalnog vida. Tako detaljno poznavanje sadržaja slike je potrebno u mnogim situacijama, od kojih ću neke navesti ovdje.

Autonomna vožnja je vrlo kompleksni zadatak koji zahtjeva reagiranje u konstantno promjenjivim uvjetima. Zadatak prepoznavanja treba se obaviti s najvećom preciznošću, jer je sigurnost u pitanju. Semantička segmentacija pruža informacije o slobodnom prostoru na cestama, kao i za otkrivanje oznaka traka i prometnih znakova, te time omogućuje autonomnu vožnju (Slika 2.4).

Slika 2.4: Semantička segmentacija u autonomnoj vožnji. Primijeti se da su sve instance iste klase iste boje. To je st svi auti su plave, cesta ljubičaste itd. Time bi rezultat ove slike mogao očito koristiti pri donošenju odluka auto u vožnji, npr. da ne skreće sada lijevo, jer je auto tu. Preuzeto sa: [27]

Semantička segmentacija omogućuje brzu obradu radioloških slika, što za uzvrat daje bržu dijagnostiku.

Ponekad je potrebno na temelju satelitskih slika odrediti koji dio te slike je šuma, koji cesta, koji ravnica itd. Te slike bi se onda mogle koristiti za pomoć pri upravljanju prometa.

Semantička segmentacija bi mogla prepoznati određene probleme u rendgenskim slikama proizvoda.

3. Prepoznavanje anomalija

Nedavni napredci u dubokom učenju, računalnom vidu i semantičkoj segmentaciji omogućili su primjene tih tehnologija u sve više raznolikim i dinamičnim područjima. Taj napredak pak dolazi s novim problemima. Jedan od glavnih je pojava potpuno nepoznatih objekata, koji su izvan mogućnosti naših mreža da ih procesiraju smisleno. Zato je poprilično bitno imati metode koje u najmanju ruku prepoznaju da se pojavio jedan od takvih objekata. To područje se naziva detekcija anomalija.

Cilj mu je prepoznati primjere koji se nalaze izvan distribucije. Anomalije ne moraju nužno sadržavati ekstremne vrijednosti, nego samo neuobičajeno poredane podatke. Ovisno o tome ima li nam skup podataka za treniranje samo podatke unutar distribucije, imamo dvije vrste detekcije anomalija: detekcija izvanrednih vrijednosti (eng. outlier detection) i detekcija novina (eng. novelty detection) to jest detekcija izvan distribucije.

Detekcija izvanrednih vrijednosti se temelji na detekciji anomalija iz klasične statistike. Provodi se nad skupom podataka za treniranje, tako da prepozna vrijednosti u tom skupu koje su izvan distribucije. Također se zove i nenadzirana detekcija anomalija.

Ali detekcija anomalija koja nas više zanima je detekcija novina, to jest polu-nadzirana detekcija anomalija. U detekciji novina anomalije se detektiraju nakon što je model već istreniran, tako da se prepozna da novi podatci ne spadaju u istu distribuciju kao i skup na kojem smo trenirali. Ovaj rad se specifično bavi dijelom detekcije novina kojemu je cilj prepoznati semantički nepoznate objekte na slici, što se naziva segmentacija anomalija[10].

Za ovaj zadatak predloženo je nekoliko pristupa koji se temelje na kvantifikaciji nesigurnosti, generativnim modelima ili korištenju već istreniranog modela za otkrivanje anomalija.

Većina metoda segmentacije anomalija se temelji na kvantifikaciji nesigurnosti, što ima određenog intuitivnog smisla. Razumno je očekivati da ako se model susretne s nečime što nikada do sada nije vidio, da će biti nesiguran u svoju predikciju. U početku se za ovaj proces koristilo Bayesovo duboko učenje, ali zbog računalne kompleksnosti te metode, ona se napustila.

Već istrenirani model se može koristiti za detekciju anomalija tako da mu se promjeni arhitektura, ili korištenjem novih podataka nad kojima uči detektirati anomalije. U prvom

slučaju modeli postaju dosta sporiji, zbog dodatne arhitekture, a u drugom je vrlo teško napraviti dobre podatke za generalizaciju svih anomalija.

Zadnja metoda segmentacije anomalija koristi generativne modele da prepozna anomalije. Pretpostavka je da će generativni modeli koji rekonstruiraju originalne slike bolje očuvati poznate dijelove slike, od nepoznatih. Anomalije se onda prepoznaju razlikama između piksela nove i stare slike. Glavno ograničenje ovog načina je proces prepoznavanja relevantnih razlika između nove i stare slike, te se treba još vidjeti hoće li raditi u kompleksnijim situacijama.

Ovdje ćemo proučiti tri metrike za provođenje segmentacije anomalija koje se sve temelje na kvantificirati nesigurnost: max-softmax, max-logit i entropija.

3.1. Max-softmax

Maksimalna softmax vjerojatnost je jedna od najkorištenija metoda segmentacije anomalija, najčešće korištena kao osnovna crta za druge metode. Intuicija iza maksimalne softmax vjerojatnosti, kaže da je vjerojatnost da je piksel anomalija suprotno proporcionalna vjerojatnosti najvjerojatnije klase. Vrijednost anomalije piksela se onda ovom metodom računa ovako (3.1):

$$a_i = 1 - \max_{k \in K} softmax(y_i) \tag{3.1}$$

y_i su sve vrijednosti klasifikacije tog piksela, a K sve klase.

Glavni problem te metode su slučajevi kada je model uvjeren da neki piksel pripada nekom od dva objekta, samo nije uvjeren kojem[11]. U tom slučaju bi ova metoda detektirala anomaliju, kada je zapravo piksel jedan od dva prije ponuđena objekta. Taj problem se često pronalazi na semantičkim granicama.

3.2. Max-logit

Maksimalna logit vrijednost koristi sličnu funkciju kao i maksimalna softmax vjerojatnost, samo sa čistim logitom, te glasi ovako (3.2):

$$a_i = -\max_{k \in K} y_i \tag{3.2}$$

Maksimalna logit vrijednost se pokazala boljom od maksimalna softmax vjerojatnosti na velikim skupovima podataka[12]. Još uvijek sadrži problem da lažno detektira anomalije na semantičkim granicama, ali je taj problem nešto blaži.

3.3. Entropija

Entropija se koristi za segmentaciju anomalija tako da se za svaki piksel izračuna entropija vjerojatnosti njegovih klasa računicom (3.3):

$$H(X) = -\sum_{k \in K} p(k) \log p(k) \tag{3.3}$$

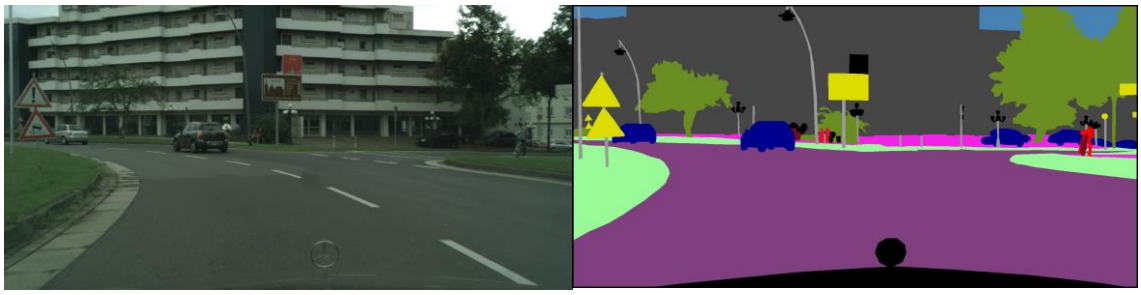
Ta entropija je onda vrijednost anomalije piksela. Ova metoda bi također trebala imati problema pri granicama, ali manje nego ostale u slučaju većeg broja klasa.

4. Skupovi podataka

U ovom radu sam koristio dva skupa podataka, cityscapes podatke i fishyscapes podatke.

4.1. Cityscapes

Za semantičku segmentaciju sam koristio Cityscapes[9] skup podataka. Cityscapes je skup slika ulica uzetih iz 50 različitih gradova u Njemačkoj, u različitim godišnjim dobima, uvijek po danu i u dobrim vremenskim uvjetima. Specifično u Cityscapesu sam koristio gtFine_trainvaltest i leftImg8bit_trainvaltest skupove. GtFine_trainvaltest se sastoji od 5000 maska slika, 3475 iz skupa train i val i 1525 iz skupa test. LeftImg8bit_trainvaltest je isti kao i gtFine_trainvaltest, samo umjesto maska, u njemu se nalaze prave slike u RGB formatu. Sve slike su uzete iz perspektive lijevog dijela auta. Sam cityscapes klasificira slike u 30 klasa, ali sam prilikom njegova korištenja, trebao taj broj klasa smanjiti na 19 + 1 klasu za ignoriranje. Slike su same po sebi raspoređene u skupove za validaciju, treniranje i testiranje, pa sam ih samo trebao koristiti (Slika 4.1).



Slika 4.1: Cityscapes slika i maska. Primijeti se da je auto na kojem je kamera zapravo crne boje, a ne plave kao ostatak auta. To je zato što on spada u klasu za ignoriranje. Preuzeto sa [9]

4.2. Fishyscapes

Fishyscapes lost and found validacijski skup podataka[11] sam koristio za prepoznavanje anomalija. To je skup od 100 maska slika baziranih na lost and found leftImg8bit skupu podatka, gdje su samo 3 klase:

- Anomalija, označena s indeksom 1
- Cityscapes klase, označene s indeksom 0
- Klasa za ignoriranje, označena s indeksom 255

Lost and found je skup podataka od 2104 slika ceste s nekim anomalijama na cesti, te će zato model istreniran na Cityscapesu biti primjenjiv i na lost and found. Ovaj skup podataka je specifično napravljen za detekciju anomalija.

5. Programska izvedba

Model koji sam koristio za semantičku segmentaciju i detekciju anomalija je Swiftnet baziran na ResNet18 s prostornim piramidalnim sažimanjem. Implementacija programa je bila u pythonu na github linku: https://github.com/thrg/swiftnet-mc53261. Također sam koristio nekoliko biblioteka s alatima koji su mi bili potrebni. Bitne za napomenuti biblioteke su Numpy, Pytorch, Sklearn, OpenCv, Pillow i Matplotlib.

Python

Python je programski jezik visoke razine. Naglašava čitljivost koda svojom sintaksom. Poznat je po svojoj jednostavnosti pisanja. Dinamički je tipkan jezik isto kao i većina modernih jezika, te ima skupljač smeća. Podržava više paradigmi od kojih su najpopularnije objektno orijentirana paradigma i funkcijska paradigma. Sadrži iznimno velik broj lako dostupnih biblioteka, koje se gotovo sve mogu instalirati pomoću pip-a. Zbog tih svojstava najrasprostranjeniji je jezik u strojnom učenju.

Numpy

NumPy, kao što i samo ime naznačuje, je biblioteka za programski jezik Python. Ona nudi podršku za velike, višedimenzionalne nizove i matrice, zajedno s velikom zbirkom matematičkih funkcija za rad s tim nizovima. NumPy je softver otvorenog koda i ima mnogo suradnika. Temeljna funkcionalnost NumPya je njegov "ndarray" struktura podataka za n-dimenzionalni niz. Takvi nizovi su brzi pogledi na memoriju koji su za razliku od Pythonove ugrađene liste homogeno tipizirani. Vrlo je popularna biblioteka u strojnom učenju, jer može raditi brze operacije nad velikim nizom podataka.

Pytorch

Pytorch je okvir za strojno učenje, temeljen na torch biblioteci. Najbitnija je biblioteka za strojno učenje koju koristim. Ima nekoliko bitnih modula koji zajedno omogućuju gotovo sve potrebno za strojno učenje. Glavni moduli su optim i nn. Optim je modul koji sadrži funkcije optimizacije, poput Adam-a koji se onda samo stave u kod. Nn modul služi za jednostavno stvaranje neuronskih mreža definiranjem slojeva mreže. Pytorch sadrži sve od predtreniranih modela i funkcija gubitka do modela za podatke i dataloadera. Čak i omogućuje specificiranje je li bi htio koristiti gpu ili cpu za obradu podataka.

Podatci u pytorchu se spremaju u tenzore, koji omogućavaju poprilično kompleksne operacije u kratkom vremenu. Čak sadrži i torchvision, koji je specifična biblioteka pytorcha za računalni vid. Torchvision sadrži razne modele, skupove podataka i transformacije nad podatcima potrebne za računalni vid.

Sklearn

Sklearn, točnije scikit-learn, je biblioteka u pythonu koja omogućuje razne stvari vezane uz strojno učenje. Koristi NumPy za rad s podatcima i poprilično je optimizirana, čak s nekim dijelovima u Cythonu. Najbitnija stvar vezana za sklearn u ovom projektu je da ima mogućnost računanja AP i AUROC metrika, koje ću objasniti kasnije.

OpenCy

OpenCv je biblioteka pythona otvorenog koda, koja se koristi za računalni vid u stvarnom vremenu. Napisana je u C++, ali se može koristiti i u Pythonu. U Swiftnetu se ta biblioteka koristi za brzu normalizaciju, pretvorbe boja i takve stvari.

Pillow

Pillow je biblioteka za Python koja se koristi u obradi slika. Ima sposobnost brzo očitati podatke iz piksela.

Matplotlib

Matplotlib je biblioteka za crtanje grafova u Pythonu. Koristi klase Numpya za prikaz podataka u grafu. Dopušta različite matematičke prikaze podataka, poput histograma, linijskih grafova, 3D grafova, točkastih grafova i još mnogih drugih.

Općenito o implementaciji

Moj kod se temeljio na prije spomenutom Swiftnetu, te glavni nadodatci na njega su kod za obradu fishyscapes lost and found skupa podataka i evaluate_anomly funkcija, zajedno s njenim pomoćnim funkcijama max_softmax, max_logit i entropy. Evaluate_anomly funkcija se nalazi u evaluate.py te računa AP i AUROC, a pomoćne funkcije računaju vrijednosti anomalije piksela po svojoj metodi.

6. Eksperimenti

U nastavku donosimo pregled eksperimenata koje sam provodio za ovaj završni. Prvo ćemo proći kroz određene metrike nužne za određivanje kvalitete kasnijih metoda, zatim ćemo proći kroz rezultate istreniranog modela za semantičku segmentaciju i na kraju usporediti različite metode detekcije anomalija. Ali, najprije je potrebno znati određene detalje o treniranju samog modela.

Model sam trenirao na Cityscapes gtFine i left8bit train podskupu. Treniranje je bilo na 30 epoha i trajalo tri sata. Model koji sam uzeo je bio najbolji po mIoU na val skupu unutar tih 30 epoha. Kod se nalazi na githubu, a vrtio sam ga na kagglu s GPU T4 x2 grafičkom. Model je treniran na Cityscapes slikama s punom rezolucijom (1024 x 2048). Slike su bile grupirane u grupe od 14 slika, stopa učenja je na početku bila 4e-4 s propadanjem težina od 10^{-4}. Minimalna stopa učenja je od 10^{-6}.

6.1. Metrike

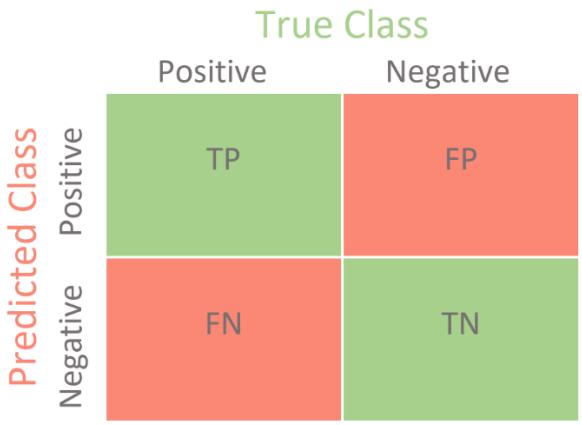
Za početak ćemo definirati korištene mjere kvalitete semantičke segmentacije i detekcije anomalija. Ali, prije objašnjenja korištenih metrika potrebno je objasniti matricu zabune, jer se sve naše mjere temelje na njoj.

Matrica zabune

Matrica zabune je matrica koja prikazuje rezultat izvedbe modela strojnog učenja, tako da je svako polje u matrici određeno predviđenom klasom ulaza i očekivanom klasom ulaza. Svako polje matrice sadrži broj ulaza koji spadaju u tu kategoriju.

Tako imamo četiri kategorije u matrici (Slika 6.1):

- TP – true positive, gdje je model točno klasificirao ulaz u klasu koju trenutno gledamo
- FP – false positive, gdje je model krivo klasificirao ulaz kao trenutnu klasu
- FN – false negative, gdje je model krivo klasificirao ulaz, koji je zapravo trenutne klase
- TN – true negative, gdje je model točno klasificirao klasu koju trenutno ne gledamo



Slika 6.1: Matrica nesigurnosti. X dimenzija ove matrice je očekivana klasa, a y predviđena klasa. Preuzeto sa: [13]

Točnost

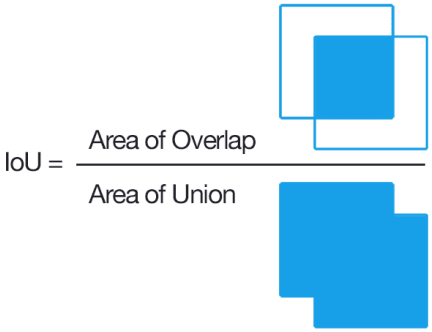
Najpoznatija mjera iz matrice zabune je točnost. Ona je zapravo broj točno klasificiranih ulaza kroz sveukupni broj ulaza (6.1).

$$Acc = \frac{TP+FP}{TP+FP+FN+TN} \tag{6.1}$$

U semantičkoj segmentaciji se ova metrika zove točnost piksela. Ima veliki problem da nije toliko korisna kada postoje klase koje dominiraju u učestalosti nad drugima, što je česti slučaj u semantičkoj segmentaciji.

IoU

IoU (intersection over union) je najbitnija metrika za semantičku segmentaciju. Kao što i samo ime kaže ona je presjek predviđene i očekivane klase kroz njihova unija (Slika 6.2).



Slika 6.2: IoU je slikovito prikazan kao presjek kroz unija. Preuzeto sa: [14]

Računica je onda (6.2):

$$IoU = \frac{TP}{TP+FP+FN} \tag{6.2}$$

Ova metrika je bolja od točnosti, jer za svaku klasu zasebno daje koliko je predikcija uspješna. Trebali bi još da dobijemo metriku za cijelu našu predikciju, a ne samo za predikciju određene klase, izračunati srednju vrijednost svih IoU, što se naziva mIoU (mean IoU).

Preciznost

Uz IoU dobro je imati i neke pomoćne metrike poput preciznosti. Preciznost je zapravo svi točno predviđeni pikseli neke klase kroz svi pikseli koje je model predvidio da su te klase. To jest (6.3):

$$preciznost = \frac{TP}{TP+FP} \tag{6.3}$$

Ako nas zanima preciznost cijelog modela samo izračunamo srednju vrijednost preciznosti svake klase. Preciznost se često uparuje s odzivom.

Odziv

Odziv je svi točno predviđeni pikseli neke klase kroz svi pikseli te klase (6.4)

$$odziv = \frac{TP}{TP+FN} \tag{6.3}$$

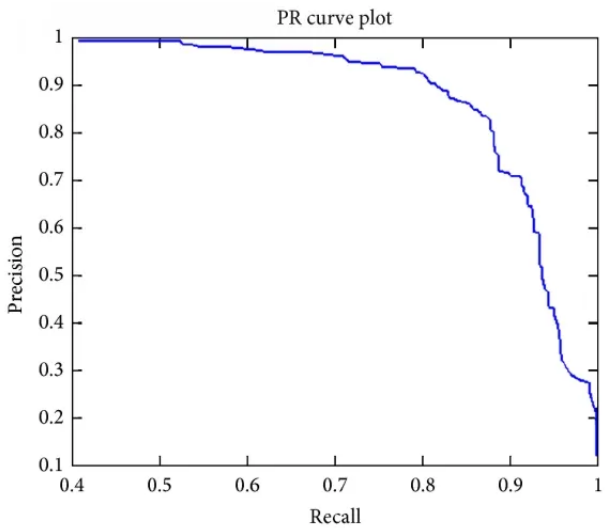
Isto kao i obije metrike prije poželjno je izračunati srednju vrijednost odziva svih klasa.

Metrike za segmentaciju anomalija

Prepoznavanje anomalija zahtjeva druge metrike od semantičke segmentacije, jer želimo izračunati koliko nam je općenito dobra segmentacija anomalija, a ne samo za jedan graničnu vrijednost. U tu svrhu koristimo nove metrike AP i AUROC.

AP

AP (average precision) ili prosječna preciznost predstavlja koliko dobro detekcija anomalija detektira anomalije, bez da ih lažno detektira. Računa se kao površina ispod preciznost-odziv krivulje[15]. Preciznost-odziv krivulja je krivulja koja opisuje razmjenu postotaka između preciznosti i odziva na nekom modelu (Slika 6.3).



Slika 6.3: Preciznost-odziv krivulja. Primijetite da je u početku preciznost 1 sve dok ne kreće povećanje odziva smanjivati preciznost, a na kraju je odziv 1 i smanjenje preciznosti prestaje povećavati odziv. Preuzeto sa: [15]

Kada je površina ispod krivulje 1, onda to znači da nam model potpuno savršeno predviđa anomalije. To jest da je vrijednost anomalije piksela na svakom pikselu koji je anomalija inf, a na svakom koji nije -inf. Vrijednost 0 bi pak značila da uopće ne razlikuje između anomalija i ne anomalija. Najčešće se računa promjenom granice kojom se nešto smatra anomalija, te time dobivanjem puno parova odziva i preciznosti, ili procesiranjem puno primjera u kojima su prisutne razni odzivi i preciznosti. U praksi se računa kao srednja vrijednost preciznosti na različitim odzivima.

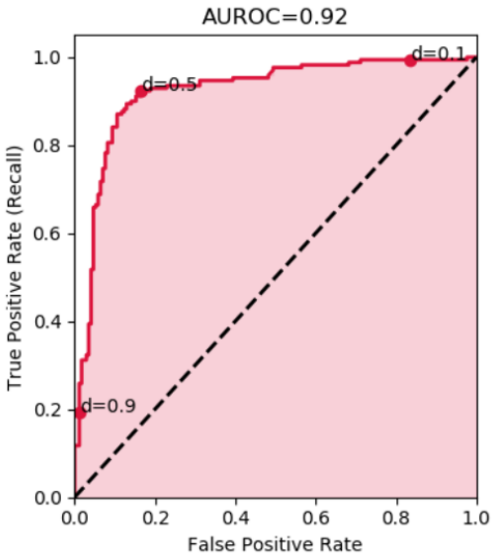
FPR

FPR (false positive rate) se ne koristi kao metrika u segmentaciji anomalija, samo ga je potrebno razumjeti prije AUROC-a. Računa se kao (6.4):

$$FPR = \frac{FP}{FP+TN} \tag{6.4}$$

AUROC

AUROC (area under the curve of receiver operator characteristics) predstavlja koliko točno model rangira anomalije. Računa se kao površina ispod odziv-FPR krivulje[16]. Odziv-FPR krivulja je krivulja koja opisuje razmjenu između odziva i FPR-a na nekom modelu.



Slika 6.4: Odziv-FPR krivulja. Crticama je označena krivulja ispod koje je površina 0.5. Preuzeto sa: [16]

Kada je površina ispod krivulje 0.5, to znači da model uopće nema sposobnost razlikovanja anomalija od ne anomalija. Kada je 0 to pak znači da uvijek svaku anomaliju klasificira kao ne anomaliju i obrnuto. Ako je pak površina 1, to je uvijek savršena klasifikacija. Računa se isto kao i AP samo na odziv-FPR krivulji.

6.2. Rezultati semantičke segmentacije

Nakon što je bio istreniran model, trebalo ga je nekako evaluirati. Za to sam koristio val podskup Cityscapesa, te će svi bitni rezultati za semantičku segmentaciju biti na tom skupu.

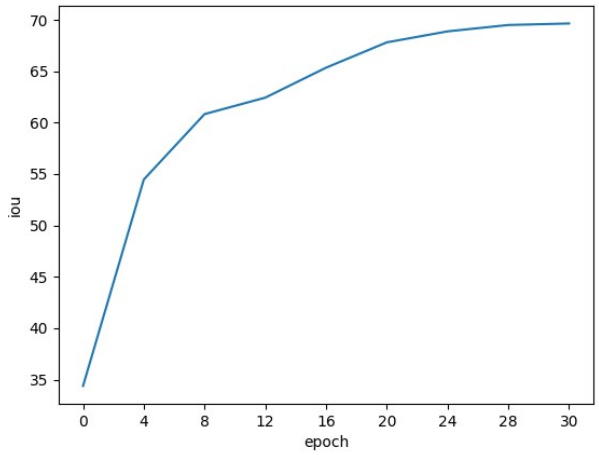
Računanjem točnosti piksela, mIoU(Slika 6.5), preciznosti i odziva na train i val skupu sam dobio ove rezultate(Slika 6.6):

Tablica 6.1: Metrike rezultata semantičke segmentacije na train i val skupu

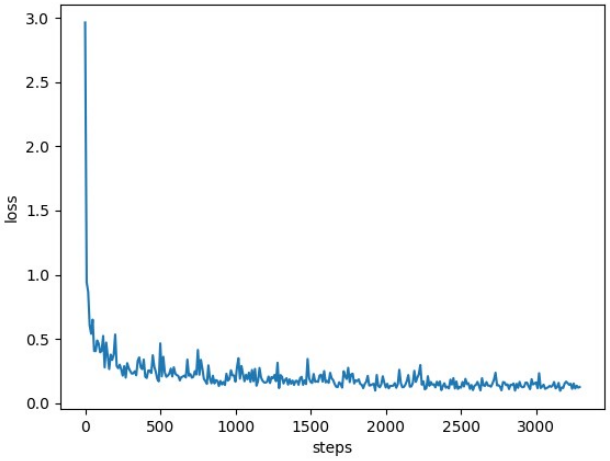
	train	val
Točnost piksela	95.84%	94.83%
mIoU	75.83%	69.65%
preciznost	87.41%	78.29%
odziv	83.89%	84.36%

Kao što je bilo i očekivano model radi bolje na train nego na val podskupu i točnost piksela ima najveći postotak, jer model ima preferenciju klasificirati piksel u klase koje se češće pojavljuju.

Brzina obrade jedne slike je bila 0.77 sekundi.



Slika 6.5: miou validacije kroz epohe, mjeren svake 4 epohe. Kao što je očekivano kontinuirano i sve sporije raste.



Slika 6.6: Gubitak tijekom treniranje, mjeren svakih 10 koraka. U početku naglo pada, a s vremenom sve manje se naglo mijenja. Također nakon početnog naglog pada kreće sporije i manje konzistentno padati.

6.3. Usporedba pristupa za detekciju anomalija

Model sam evaluirao na Fishyscapes lost and found podatcima, to jest na 100 slika u njemu. Model koji je bio korišten je bio onaj iz semantičke segmentacije. Prvo je model obradio

danu sliku, a onda su se koristile metode za detekciju anomalija da se dobiju vrijednosti anomalije svakog piksela. Metode koje sam koristio su max-softmax, max-logit i entropiju.

Bitno bi bilo za napomenuti da pri računanju AP i AUROC, nisam računao te dvije vrijednosti izravno na svim slikama, zbog opterećenja koje je nastalo radom s toliko piksela, koje je onda rušilo pokrenuti cpu na kagglu. Umjesto toga sam izračunao AP i AUROC za svaku sliku i uprosječio ih po skupu podataka.

Dobio sam ove rezultate:

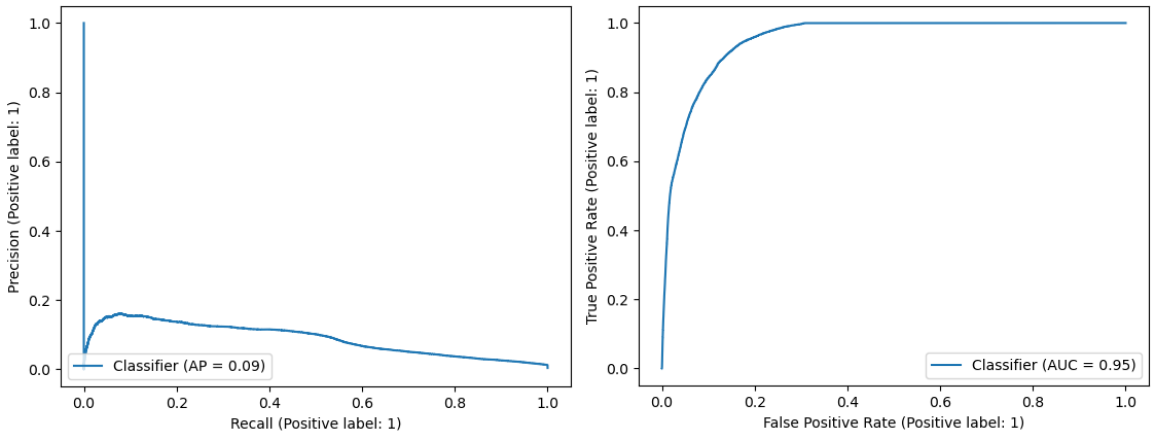
Tablica 6.2: Metrike rezultata segmentacije anomalija pomoću metoda detekcije anomalija

	AP	AUROC
Max-softmax	4.82%	82.76%
Max-logit	14.64%	88.16%
Entropija	9.02%	84.19%

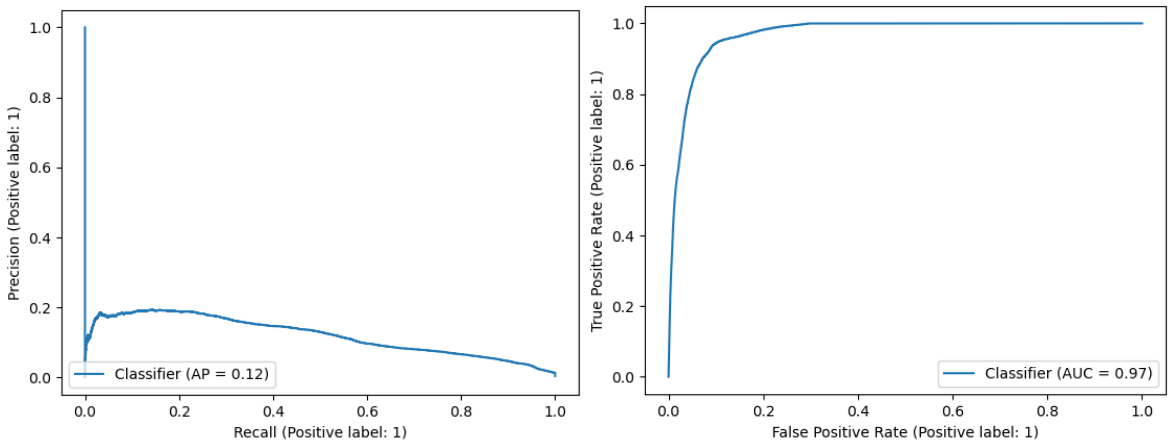
Kao što je bilo i očekivano, max-softmax je ispao najgora metoda (Slika 6.7)(Slika 6.10)(Slika 6.13). Entropija me iznenadila (Slika 6.8)(Slika 6.11)(Slika 6.14), očekivao sam da bi mogla i biti bolja od max-logita, ali je ispala primjetljivo gora. Vjerojatno zbog svojstva da ako je model nesiguran između samo dvije od devetnaest klasa max.logit bi to lagano smatrao anomalijom, a entropija ne.

Time se max-logit pokazao najboljom funkcijom za detekciju anomalija u ovom slučaju (Slika 6.9)(Slika 6.12)(Slika 6.15).

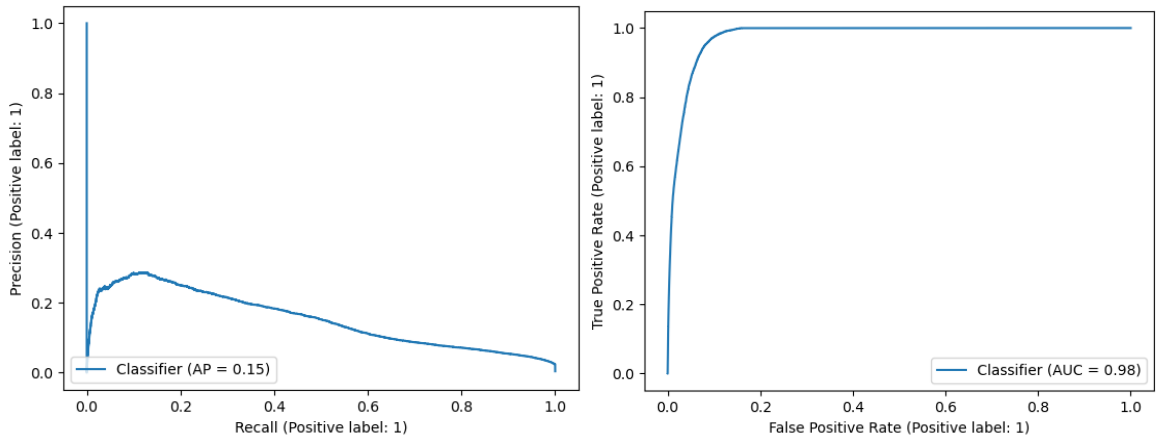
Histogrami vrijednosti anomalija piksela su poprilično slični u skupu sa anomalijama i skupu bez anomalija, zato što na svakoj slici sa anomalijama, same anomalije čine jako mali dio.



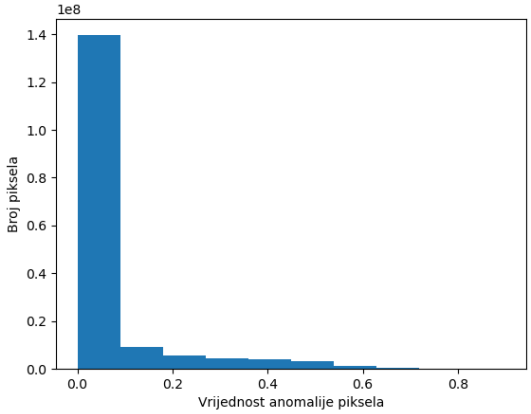
Slika 6.7: AP i AUROC dobiveni metodom maksimalnog softmaxa na trećoj slici lost and found skupa podataka..



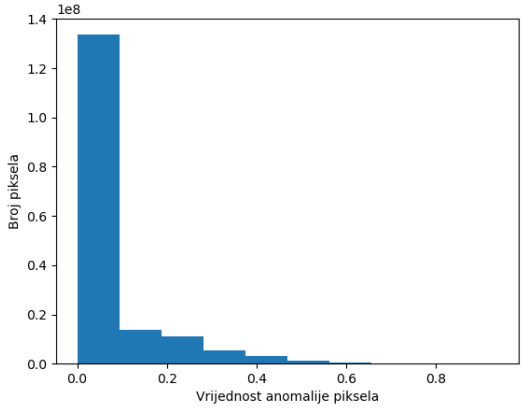
Slika 6.8: AP i AUROC dobiveni metodom entropije na trećoj slici lost and found skupa podataka.



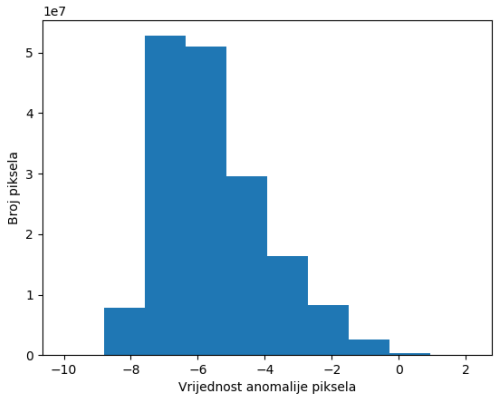
Slika 6.9: AP i AUROC dobiveni metodom maksimalnog logita na trećoj slici lost and found skupa podataka.



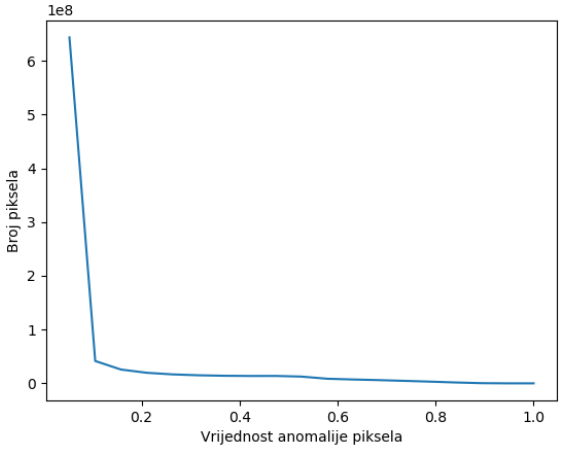
Slika 6.10: Histogram vrijednosti anomalija piksela koje su dobivene softmaxom na skupu sa anomalijama. Primijeti se da ih je većina oko 0 što znači da nema puno piksela sa anomalijama. Ovi između 0.1 i 0.3 su vjerojatno samo pikseli rjeđih klasa ili na semantičkim granicama.



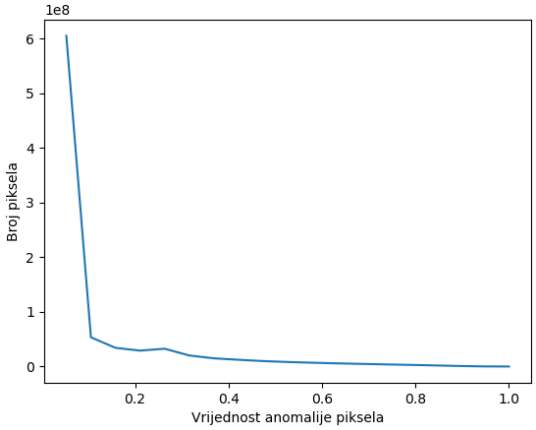
Slika 6.11: Histogram vrijednosti anomalija piksela koje su dobivene entropijom na skupu sa anomalijama. Za razliku od softmax-a pikseli su koncentriraniji u prostoru od 0 do 0.3, ali zato ih oko 0.2 ima dosta više nego u softmaxu.



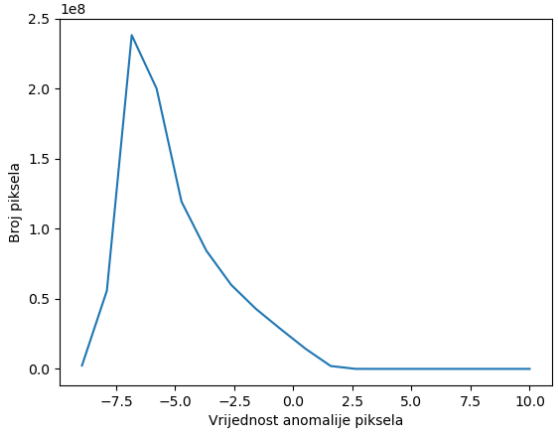
Slika 6.12: Histogram vrijednosti anomalija piksela koje su dobivene maksimalnim logitom na skupu sa anomalijama. Za razliku od softmax-a i entropije dosta preciznije prikazuje podatke koji gotovo sigurno nisu anomalije (od -2 na dalje je ekvivalent softmaxova od 0 do 0.1). Također ima najmanje vrijednosti koje su vjerojatno anomalija (od 0 na dalje).



Slika 6.13: Histogram vrijednosti anomalija piksela koje su dobivene softmaxom na skupu bez anomalija. Kao što je i očekivano, gotovo svi pikseli su na 0.



Slika 6.14: Histogram vrijednosti anomalija piksela koje su dobivene entropijom na skupu bez anomalija. Kao što je i očekivano, gotovo svi pikseli su na 0. Sa entropijom postoji ih nešto više oko 0.2.



Slika 6.15: Histogram vrijednosti anomalija piksela koje su dobivene maksimalnim logitom na skupu bez anomalijama. Velika većina vrijednosti je između -10 i 0, sa jako malo između 0 i 1, te dalje 0.

Zaključak

Ovaj rad istražuje nekoliko metoda detekcije anomalija u semantičkoj segmentaciji. Semantička segmentacija sama po sebi je kompliciran problem u domeni računalnog vida s mnogo primjena, a ne bi bila potpuna bez detekcije anomalija. Cilj ovoga rada je tako bio isprobati i usporediti različite metode detekcije anomalija.

Za ovaj problem sam koristio arhitekturu Swiftnet baziranu na ResNet18 s prostornim piramidalnim sažimanjem. Iako je primarna svrha Swiftneta omogućavanje semantičke segmentacije u stvarnom vremenu, ima i iznimno veliku točnost, te brzo uči, zbog čega je bio dobar izbor za ovaj rad. Slike i njihove maske sam preuzeo iz Cityscapesa, te na njima istrenirao model. Nakon 30 epoha najveći prosječni IoU validacijskog skupa je bio 69%, a točnost piksela 94.83%.

Cityscapes skup za treniranje se sastojao od 2975 slika s ulice, iz perspektive auta, rezolucije 1024 x 2048. Fishyscapes lost and found, na kojemu sam evaluirao svoje metode detekcije anomalija je sadržavao 100 maska Cityscapes podataka.

Platforma korištena za učenje modela je bila kaggle s grafičkom. Programska implementacija je bila napisana u Pythonu uz pomoć biblioteka Sklearn i Pytorch. Sklearn sam koristio za izračun metrika za detekciju anomalija, a Pytorch tenzore u funkcijama za detekciju anomalija.

Nakon što sam dobio taj prihvatljivi mIoU, prešao sam na detekciju anomalija. Tamo sam koristio tri metode za dobivanje vrijednosti anomalije piksela: maksimalni softmax, maksimalni logit i entropiju.

Od te tri metode maksimalni logit najbolje klasificira anomalije, s AP 14.64% i AUROC 88.16%, a zatim ide entropija, pa maksimalni softmax. Danji napredak s ovog rada bi se mogao postići isprobavanjem ovih metoda na većem skupu podataka, korištenjem preciznijeg modela za semantičku segmentaciju te isprobavanjem drugih metoda detekcije anomalija.

Literatura

[1] Čupić M., Šnajder J., Bašić B.D. *Umjetne neuronske mreže*, Uvod u umjetnu inteligenciju, 2019. Poveznica: https://www.fer.unizg.hr/_download/repository/UI_12_UmjetneNeuronskeMreze.pdf ; pristupljeno 2. lipnja 2023.

[2] Hinton G. E. *How neural networks learn from experience. Scientific American*, 267(3), 144-151. (1992). Poveznica: https://www.jstor.org/stable/24939221; pristupljeno 2. Lipnja 2023.

[3] Šegvić S., Šarić J., Grubišić I., Oršić M., Bevandić P., *Nulta labaratorijska vježbna iz dubokog učenja*, Duboko učenje, (2023, veljaća) Poveznica: http://www.zemris.fer.hr/~ssegvic/du/lab0.shtml; pristupljeno 2. Lipnja 2023

[4] Šegvić S., *Optimizacija parametara modela*, Duboko učenje, (2019, travanj) Poveznica: http://www.zemris.fer.hr/~ssegvic/du/du3optimization.pdf; pristupljeno 2. Lipnja 2023.

[5] Yamashita, R., Nishio, M., Do, R.K.G. et al. *Convolutional neural networks: an overview and application in radiology*. Insights Imaging 9, 611–629 (2018). Poveznica: https://doi.org/10.1007/s13244-018-0639-9; pristupljeno: 3. Lipnja 2023.

[6] He K., Zhang X., Ren S., i Sun J. *Identity mappings in deep residual networks*, Arxiv, (2016.) Poveznica: https://arxiv.org/abs/1603. 05027; pristupljeno: 3. Lipnja 2023.

[7] Ronneberger O., Fischer P., Brox T., U-Net: *Convolutional Networks for Biomedical Image Segmentation*, Arxiv, (May 2015). Poveznica: https://arxiv.org/abs/1505.04597v1; pristupljeno 3. Listopada 2023.

[8] Oršić M., Krešo I., Bevandić P., Šegvić S., *In Defense of Pre-trained ImageNet Architectures for Real-time Semantic Segmentation of Road-driving Images*, Arxiv, (ožujak 2019) Poveznica: https://arxiv.org/abs/1903.08469; pristupljeno 3. Lipnja2023.

[9] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "*The Cityscapes Dataset for Semantic Urban Scene*

Understanding," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. [Bibtex]

[10] Chan, R., Uhlemeyer, S., Rottmann, M., Gottschalk, H. (lipanj 2022). *Detecting and Learning the Unknown in Semantic Segmentation*. In: Fingscheidt, T., Gottschalk, H., Houben, S. (eds) Deep Neural Networks and Data for Automated Driving. Springer, Cham. Poveznica: https://doi.org/10.1007/978-3-031-01233-4_10; pristupljeno 4. Lipnja Lipnja 2023.

[11] Fontanel D., Cermelli F., Mancini M., Caputo B. *Detecting Anomalies in Semantic Segmentation with Prototypes*, Arxiv, (lipanj 2021.), Poveznica: https://arxiv.org/abs/2106.00472; pristupljeno: 4. Lipnja 2023.

[12] Hendrycks, D., Basart, S., Mazeika, M., Zou, A., Kwon, J., Mostajabi, M., Steinhardt, J; Song, D.. (2022). *Scaling Out-of-Distribution Detection for Real-World Settings*. Proceedings of the 39th International Conference on Machine Learning, in Proceedings of Machine Learning Research 162:8759-8773 Poveznica: https://proceedings.mlr.press/v162/hendrycks22a.html; pristupljeno 4. Lipnja 2023.

[13] Narkhede S. *Understanding Confusion Matrix*, Towards Data Science, (svibanj 2019), Poveznica: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62; pristupljeno 7. Lipnja 2023.

[14] Tiu, E. *Metrics to Evaluate your Semantic Segmenatation Model*, Towards Data Science, (kolovoz 2019) Poveznica: https://towardsdatascience.com/metrics-to-evaluate-your-semanticsegmentation-model-6bcb99639aa2; pristupljeno: 7. Lipnja 2023.

[15] Liu Y., *The Confusing Metrics of AP and mAP for Object Detection / Instance Segmentation*, Medium, Poveznica: https://yanfengliux.medium.com/the-confusing-metrics-of-ap-and-map-for-object-detection-3113ba0386ef; pristupljeno: 8. Lipnja 2023.

[16] Draelos R., *The Complete Guide to AUC and Average Precision: Simulations and Visualizations*, Glass box, (srpanj 2020.), Poveznica: https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/; pristupljeno: 8. Lipnja 2023.

[17] @matelabs_ai, *Everything you need to know about Neural Networks*, hackernoon, (studeni 2017), Poveznica: https://hackernoon.com/everything-you-need-to-know-about-neural-networks-8988c3ee4491; pristupljeno: 2. Lipnja 2023.

[18] *Logistic function*, wikipedia, (lipanj 2023), Poveznica: https://en.wikipedia.org/wiki/Logistic_function#; pristupljeno: 2. Lipnja 2023.

[19] Ferdinand N., *A Simple Guide to Convolutional Neural Networks*, Towards Data Science, (siječanj 2020.), Poveznica: https://towardsdatascience.com/a-simple-guide-to-convolutional-neural-networks-751789e7bd88; pristupljeno: 2. Lipnja 2023.

[20] Reynolds A. H., *Convolutional Neural Networks*, Anh H. Reynolds blog, (2019), Poveznica: https://anhreynolds.com/blogs/cnn.html; pristupljeno: 3. Lipnja 2023.

[21] *File:MaxpoolSample2.png*, Computer science wiki, (veljaća 2018.), Poveznica: https://computersciencewiki.org/index.php/File:MaxpoolSample2.png; pristupljeno: 3. Lipnja 2023.

[22] Gholamalinejad H., Khosravi H., *Pooling Methods in Deep Neural Networks, a Review*, Researchgate, (rujan 2020), Poveznica: https://www.researchgate.net/figure/Example-of-Average-Pooling-operation_fig1_344277235; pristupljeno: 3. Lipnja 2023.

[23] *File:ResBlock.png*, Wikipedia, (studeni 2015), Poveznica: https://en.wikipedia.org/wiki/File:ResBlock.png; pristupljeno: 3. Lipnja 2023.

[24] Ramzan, Farheen & Khan, Muhammad Usman & Rehmat, Asim & Iqbal, Sajid & Saba, Tanzila & Rehman, Amjad & Mehmood, Zahid. (2019). *A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer's Disease Stages Using Resting-State fMRI and Residual Neural Networks*. Journal of Medical Systems. 44. 10.1007/s10916-019-1475-2

[25] *What is Image Segmentation*: The Basics and Key Techniques, Mindy News Blog, (listopad 2022), Poveznica: https://mindy-support.com/news-post/what-is-image-segmentation-the-basics-and-key-techniques/; pristupljeno: 3. Lipnja 2023.

[26] Dwivedi P., *Semantic Segmentation — Popular Architectures*, Towards Data Science, (ožujak 2019), Poveznica: https://towardsdatascience.com/semantic-segmentation-popular-architectures-dff0a75f39d0; pristupljeno: 3. Lipnja 2023.

[27] Musale A., *Semantic Segmentation On Indian Driving Dataset!*, Medium, (ožujak 2021), Poveznica: https://atharvamusale.medium.com/semantic-segmentation-on-indian-driving-dataset-3054cb2e70a7; pristupljeno: 3. Lipnja 2023.

Sažetak

U ovom radu primijenjene su metode dubokog učenja za segmentaciju anomalija. Specifično je korišten model arhitekture Swiftnet sa s ljestvičastim naduzorkovanjem. Nadodane su na njega tri metode detekcije anomalija: maksimalni softmax, maksimalni logit i entropija Takav je model onda korišten za detekciju anomalija na cesti.

Ključen riječi: detekcija anomalija, semantička segmentacija, segmentacija anomalija, maksimalni softmax, maksimalni logit, entropija, ljestvičasto naduzorkovanje, U-Net, Swiftnet, konvolucijske neuronske mreže, duboko učenje

Summary

In this work, deep learning methods were applied for anomaly segmentation. The Swiftnet architecture model with ladder oversampling was specifically used. Three anomaly detection methods were added to it: max-softmax, max-logit and entropy. Such a model was then used to detect anomalies on the road.

Keywords: anomaly detection, semantic segmentation, anomaly segmentation, max-softmax, max-logit, entropy, scalar causality, U-Net, Swiftnet, convolutional neural networks, deep learning