Zahvaljujem mentoru prof. dr. sc. Siniši Šegviću i mag. ing. Mateju Grciću na pomoći i strpljenju pri pisanju ovog rada.

Sadržaj

Uvod

Računalni vid je grana umjetne inteligencije koja proučava kako izlučiti i analizirati korisnu informaciju iz slike uporabom računala. Cilj računalnog vida je oblikovati model stvarnog svijeta na temelju slika. Prije značajnog razvoja područja dubokog učenja, zadaci računalnog vida rješavali su se rekonstrukcijskim pristupima te metodama klasičnog strojnog učenja. Međutim, u zadnje vrijeme iznimne rezultate pokazuju duboki konvolucijski modeli [20].

Jedan od temeljnih zadataka računalnog vida je semantička segmentacija slike [26]. Cilj semantičke segmentacije je odrediti značenje svakog dijela neke zadane slike. Po načinu na koji radi, ona spada u modele guste predikcije. Primjena semantičke segmentacije je široka i uključuje područja od autonomne vožnje do medicinske dijagnostike.

Duboki konvolucijski modeli koji se koriste u semantičkoj segmentaciji znatan dio zaključivanja provode na poduzorkovanoj reprezentaciji [19]. Model provodi zaključivanje nad slikom smanjene rezolucije koja sadrži izvučene značajke. Unatoč dobrim rezultatima, glavna prepreka prema boljoj generalizaciji modela je njegovo nepouzdano ponašanje u slučaju izvandistribucijskog ulaza.

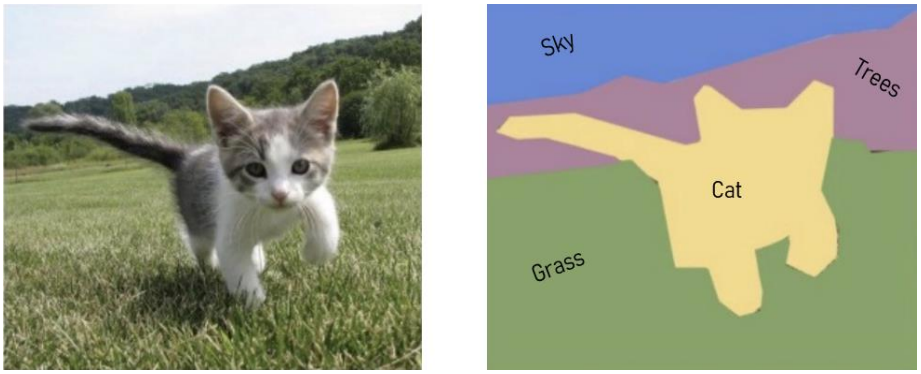
U ovom radu ću prikazati moguće rješenje problema nepouzdanog ponašanja modela u slučaju izvandistribucijskog ulaza [4]. Guste predikcije Softmaksa izražene kosinusnom sličnošću u odnosu na naučena središta semantičkih razreda možda mogu ublažiti taj problem [10]. Usporedit ću generalizaciju arhitekture SwiftNet [24] s njenom modificiranom inačicom nad skupom podataka CamVid [2]. Objasnit ću i opisati semantičku segmentaciju, svojstva, građu i moderne primjere dubokih konvolucijskih modela.

1. Semantička segmentacija

U ovom poglavlju opisujem i objašnjavam temeljni zadatak rada, semantičku segmentaciju. Opisujem povezanost semantičke segmentacije i modernih dubokih konvolucijskih modela.

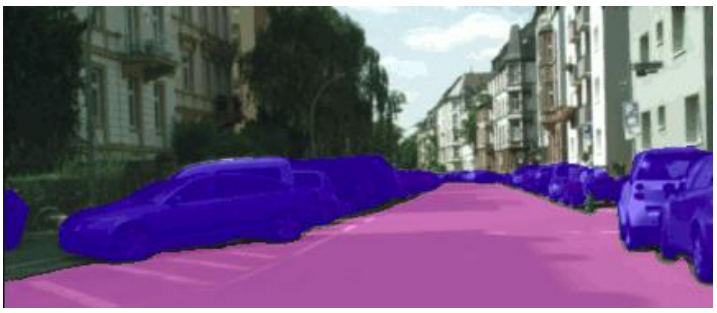
1.1. O semantičkoj segmentaciji

Semantička segmentacija je temeljni zadatak računalnog vida. Cilj semantičke segmentacije je odrediti značenje svakog dijela neke zadane slike. Riječ je o modelu guste predikcije jer se svaki piksel klasificira u njemu odgovarajući razred.



Slika 1-1: Primjer ulazne i izlazne slike (preuzeto iz [12]).

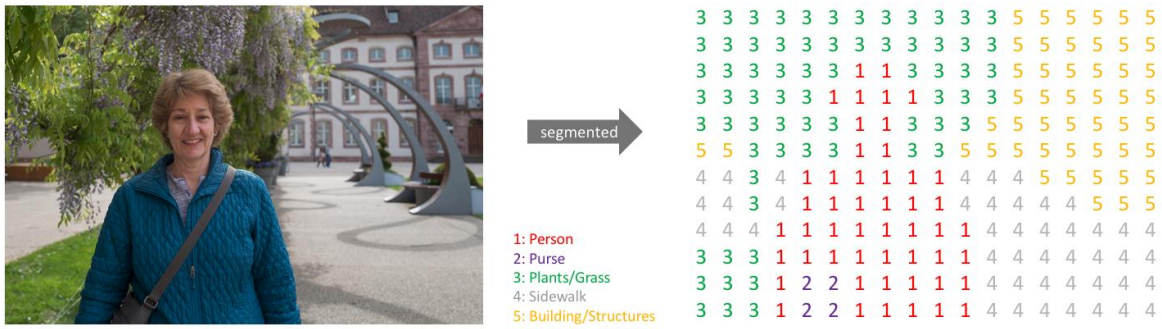
Semantička segmentacija ne razlikuje instance određenog razreda, već sve piksele tog razreda grupira u jednu cjelinu. Da bi to uspjelo piksel je nužno klasificirati (odrediti kojem razredu pripada).



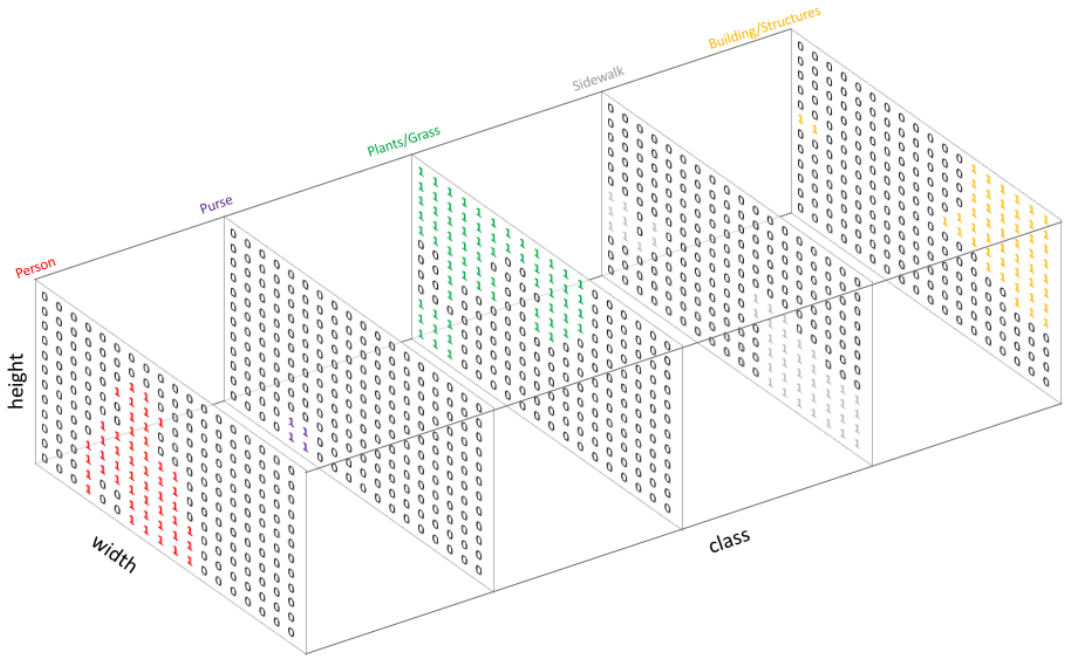
Slika 1-2: Primjer segmentirane slike (preuzeto iz [9]).

Tijekom semantičke segmentacije, iz ulazne slike nastaje više kanala. Svaki kanal je veličine ulazne slike i odgovara točno jednom razredu koji se nalazi na slici. Pojedinačan

piksel nekog kanala ima vrijednost veću od 0 ako i samo ako pripada razredu kojeg taj kanal opisuje. Također, vrijednost piksela odgovara indeksu razreda. Izlazna slika nastaje mapiranjem pobjedničkog indeksa u kod (npr. boju) odgovarajućeg razreda.



Slika 1-3: Pojednostavljen prikaz segmentacije ulazne slike (preuzeto iz [17]).



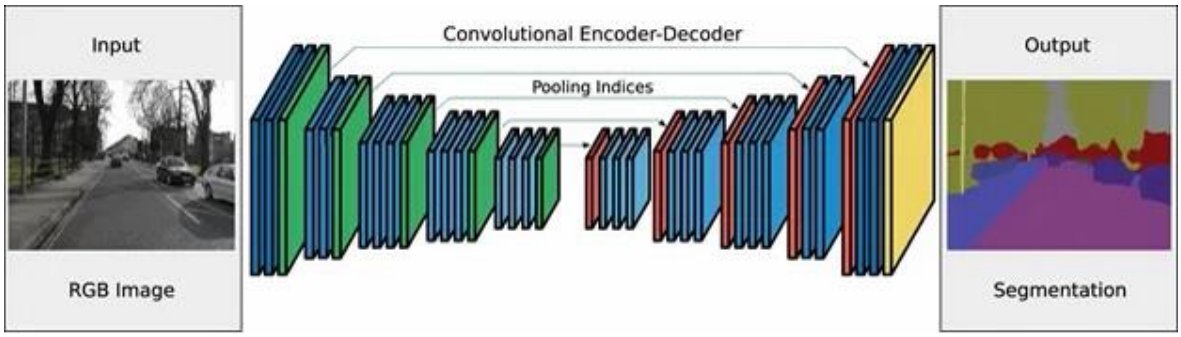
Slika 1-4: Kanali semantičkih razreda (preuzeto iz [17]).

1.2. Semantička segmentacija i duboki konvolucijski modeli

Iznimni rezultati u semantičkoj segmentaciji postižu se dubokim konvolucijskim modelima. Modeli uče na slikama originalne rezolucije.

Konvolucijska arhitektura sastoji se od nekoliko slojeva među kojima su konvolucijski slojevi i slojevi sažimanja [29]. Slojevi sažimanja smanjuju veličinu latentne slikovne reprezentacije, dok konvolucijski sloj izlučuje značajke iz ulazne reprezentacije u izlaznu mapu značajki. Početni slojevi uče pronalaziti značajke niže razine kao što su boja, rubovi i oblik. Dublji slojevi koriste poduzorkovane reprezentacije iz nižih slojeva da bi naučili značajke više razine (značajke koje se odnose na kompleksnije i veće dijelove slike).

Kod semantičke segmentacije važno je da rezolucija izlazne slike bude kao rezolucija ulazne slike. Umanjena slika koja sadrži bitne značajke više razine, prolazi kroz konvolucijske slojeve i slojeve za naduzorkovanje koji povećavaju njenu rezoluciju. Opisanu funkcionalnost ostvaruje konvolucijska arhitektura koder-dekoder, međutim postoje razna poboljšanja koja poboljšavaju učinkovitost ovakve organizacije.



Slika 1-5: SegNet arhitektura koder-dekoder (preuzeto iz [2]).

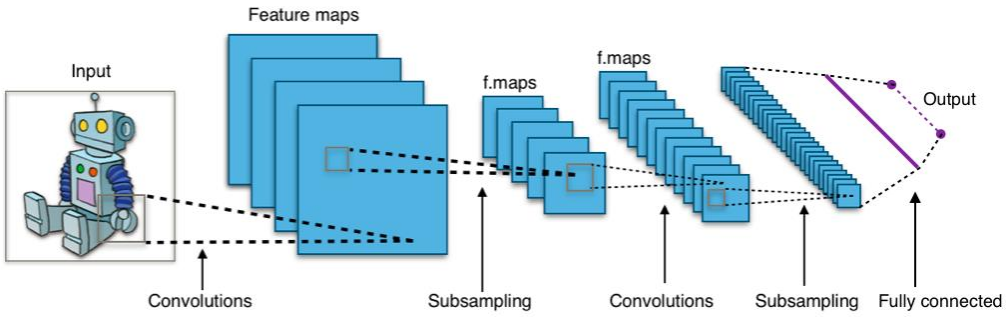
2. Konvolucijske neuronske mreže

Konvolucijska neuronska mreža je vrsta unaprijednih neuronskih mreža čija je svrha filtriranje ulaznih podataka. Konvolucijski model umjesto potpuno povezanog sloja ima najmanje jedan konvolucijski sloj. Takvi modeli su od velike važnosti u računalnom vidu zbog njihove sposobnosti da izluče bitne informacije iz slike.

2.1. Svojstva i arhitektura konvolucijske mreže

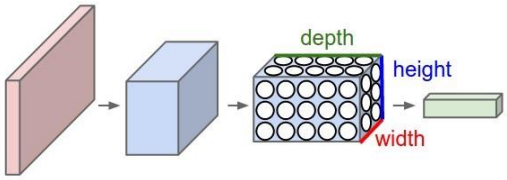
Konvolucijski modeli sastoje se od nekoliko različitih vrsta slojeva. To su konvolucijski sloj, sloj nelinearne aktivacije, sloj sažimanja, normalizacijski sloj te potpuno povezani sloj. Slojevi se mogu kombinirati na razne načine, a slijed slojeva na istoj prostornoj rezoluciji čini jedan blok.

Apstraktnije, konvolucijska mreža je građena od linearnih konvolucijskih slojeva te nelinearne aktivacijske funkcije na izlazu.



Slika 2-1: Primjer arhitekture CNN (preuzeto iz [18]).

Važna razlika između konvolucijskih i potpuno povezanih modela je u načinu slaganja umjetnih neurona. Kod konvolucijskih modela pretpostavlja se da je ulazni podatak slika što omogućuje specifične promjenu u arhitekturi. Aktivacije su posložene u tenzorima trećeg reda koji imaju širinu, visinu i dubinu. Dubina se odnosi na boju te kod RGB slika iznosi 3, a kod crno-bijelih slika iznosi 1. Latentne reprezentacije imaju veće brojeve kanala (npr. 64 ili 512). Sve aktivacije na nekoj prostornoj lokaciji nazivamo vektorom apstraktnih značajki. Izlazi konvolucijskog sloja povezani su samo s malim područjem prethodnog sloja, što nije slučaj kod potpuno povezanih slojeva.



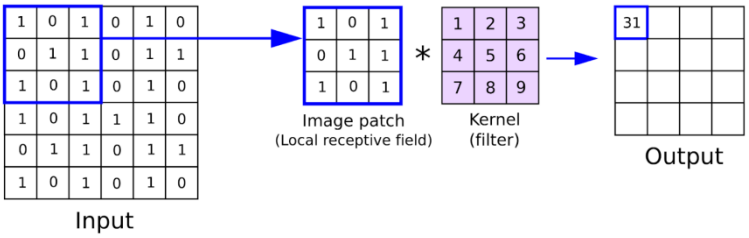
Slika 2-2: Posloženost neurona u konvolucijskoj arhitekturi (preuzeto iz [18]).

Bitna karakteristika koja proizlazi iz opisane posloženosti neurona je znatno manji broj parametara nego kod regularne neuronske mreže. Posljedica ovoga je brže učenje i smanjena mogućnost pojave prenaučenosti.

2.1.1. Konvolucijski sloj

U konvolucijskom sloju filtar (jezgra) se pomiče po svakom dijelu ulazne slike. Filtriranje se izvodi matematičkom operacijom konvolucije.

Filtri su male dimenzije, ali prelaze preko cijele ulazne slike i izlučuju i najmanje značajke. Na jednostavnom primjeru, konvolucija djeluje kao skalarni produkt dijela slike nad kojim se filtar nalazi te samog filtra. Rezultat se sumira i zapisuje u mapu značajki. Operacijom konvolucije smanjuje se visina i širina ulazne reprezentacije osim ako se ne koristi nadopunjavanje.



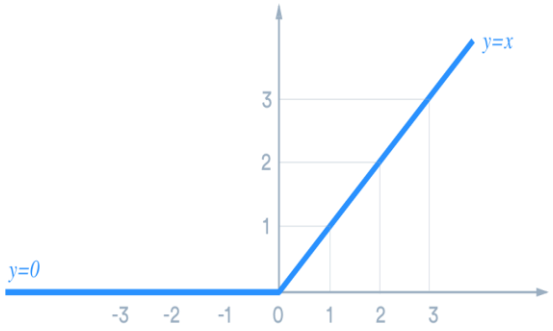
Slika 2-3: Primjer jednog koraka filtriranja (preuzeto iz [18]).

Hiper-parametri filtra koji utječu na oblik izlazne mape značajki su:

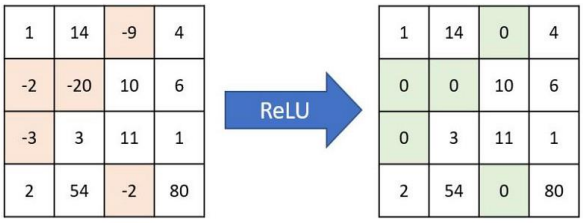
- dubina izlazne mape značajki
- korak
- nadopunjavanje ulazne reprezentacije nulama.

2.1.2. Aktiviranje zglobnicom

U ReLU sloju se primjenjuje ReLU aktivacijska funkcija (zglobnica). Zglobnica svaku negativnu ulaznu vrijednost na izlazu zamijeni s 0. To čini kako bi model imao veći kapacitet.



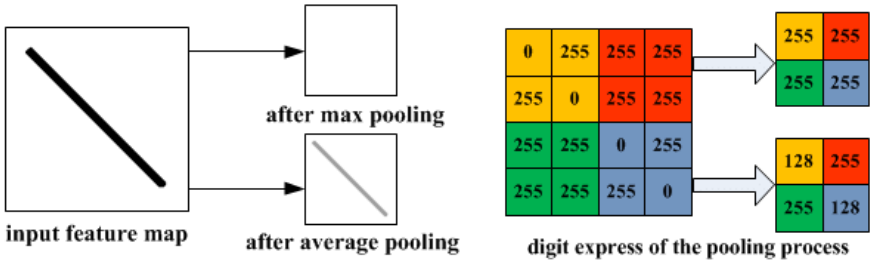
Slika 2-4: ReLU aktivacijska funkcija (preuzeto iz [3]).



Slika 2-5: djelovanje ReLU aktivacijske funkcije (preuzeto iz [3]).

2.1.3. Sloj sažimanja

Slojeve sažimanja smještamo između konvolucijskih slojeva kako bismo smanjili rezoluciju slikovne reprezentacije. Postoji nekoliko vrsta sažimanja, a neki od njih su sažimanje maksimumom, sažimanje minimumom i sažimanje prosječnom vrijednošću. Izlaz iz sloja sažimanja je umanjena slikovna reprezentacija sa značajkama odabranim po kriteriju sažimanja.



Slika 2-6: Primjer sažimanja maksimumom i sažimanja prosječnom vrijednošću (preuzeto iz [18]). Desna slika prikazuje postupke sažimanja brojevima radi lakšeg razumijevanja.

Slika prikazuje kako oba sažimanja utječu na ulaznu mapu značajki. Sažimanje maksimumom odabire maksimalnu vrijednost iz odabranog područja mape značajki. Sažimanje prosječnom vrijednošću računa prosjek vrijednosti iz odabranog područja mape značajki.

2.1.4. Normalizacijski sloj

Normalizacijski [16] sloj normalizira vrijednosti reprezentacije prethodnih slojeva. Tipično se dodaje između konvolucijskog sloja i nelinearne aktivacije kako bi svaki sloj učio samostalnije te kako bismo spriječili prenaučenost modela.

2.1.5. Potpuno povezani sloj

Nakon izlučivanja bitnih značajki, reprezentacija se izravnava (najčešće globalnim sažimanjem) [23] te se potpuno povezanim slojem oblikuje izlaz ovisno o potrebi zadatka. Potpuno povezani sloj najčešći je oblik izlaza u zadacima klasifikacije i regresije. Jedan primjer korištenja potpuno povezanog sloja je klasifikacija znamenaka. U tom primjeru, potpuno povezani sloj daje 10 izlaznih vrijednosti gdje svaka vrijednost predstavlja jednu znamenku. Te vrijednosti aktiviramo Softmaksom te nakon toga interpretiramo kao vjerojatnost pojave znamenke.

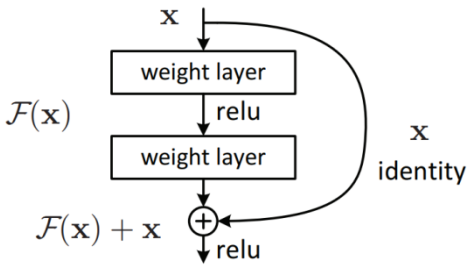
2.2. Primjeri modernih konvolucijskih modela

2.2.1. ResNet

Rezidualni modeli oblikovani su tako da se olakša širenje gradijenta prema ranim slojevima modela [13]. Rezidualne mreže omogućavaju učenje dubokih modela koji se sastoje od preko 20 slojeva. Stohastički gradijentni spust nije u mogućnosti naučiti obične modele koji su dublji od desetak slojeva. Glavno obilježje rezidualnih mreža su aditivne preskočne veze [14]. Arhitektura je građena od takozvanih rezidualnih blokova.

Preskočne veze su uvedene kao odgovor na degradaciju preciznosti tokom učenja. Povećanjem dubine modela, preciznost naglo pada [13]. Unazadnim propagiranjem, izračunati gradijent gubitka se pravilom ulančavanja prenosi od zadnjeg sloja mreže prema prvom. Rezidualni modeli osiguravaju lakšu propagaciju gradijenta gubitka prema ranim slojevima.

Rješenje problema je da se preskočnim vezama na ulaz sljedećeg bloka prenese netransformirani ulaz trenutnog bloka. Prijenos je izveden funkcijom identiteta.



Slika 2-7: Pojednostavljeni rezidualni blok (preuzeto iz [14]).

Rezidualni blok se može opisati kao:

$$y_l = h(x_l) + F(x_l, W_l) \qquad (2.1)$$

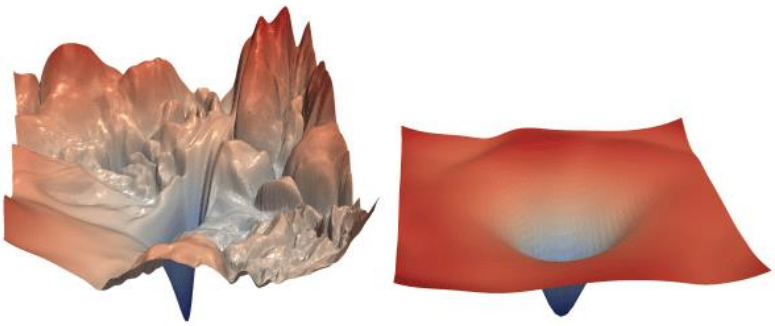
$$x_{l+1} = f(y_l) \qquad (2.2)$$

gdje su x_l i x_{l+1} ulaz i izlaz l −tog bloka, a F rezidualna funkcija.

$h(x_l)$ je funkcija identiteta koja prenosi netransformirani ulaz u rezidualni blok. Rezidualna funkcija F je implementirana kao niz transformacija nad ulaznim podacima. Ulazni podaci se transformiraju prolaskom kroz konvolucijske slojeve bloka.

Posljedice postojanja preskočnih veza je očuvanje gradijenta i očuvanje informacija izgubljenih poduzorkovanjem što rezultira boljom konvergencijom modela.

Eksperimentalno je potvrđeno i da preskočne veze promijene krajolik gubitka pa učenje brže napreduje.



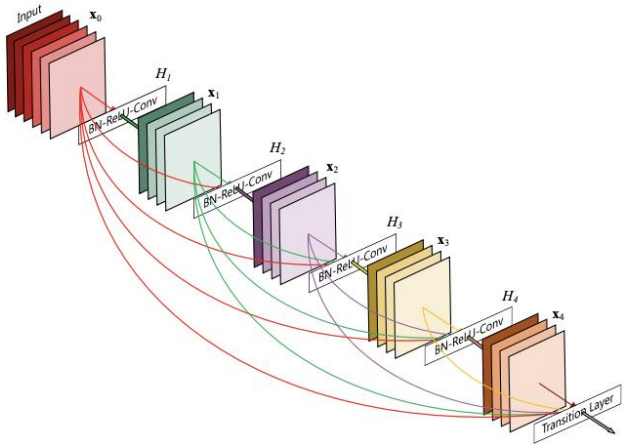
Slika 2-8: Usporedba krajolika gubitka (preuzeto iz [22]).

Lijevo – bez preskočnih veza.

Desno- sa preskočnim vezama.

2.2.2. DenseNet

Gusto povezani konvolucijski modeli također nastoje olakšati širenje gradijenta kao i ResNet. DenseNet koristi preskočne veze konkatenacijski. Svaki sloj je preskočno povezan sa svakim sljedećim slojem. Za svaki sloj, mape značajki svih prethodnih slojeva čine ulaz u taj sloj („globalno znanje“).



Slika 2-9: Organizacija DenseNet-a (preuzeto iz [15]).

Slojevi su povezani svaki sa svakim da bi se osigurao maksimalan protok informacija. Poboljšani protok informacija omogućava lakše treniranje. Ulazi se konkateniraju pa l-ti sloj prima $l-1$ ulaza (mapi značajki). Arhitektura s L slojeva ima $\frac{L(L-1)}{2}$ veza, za razliku od tradicionalnih arhitektura koje imaju L veza.

DenseNet slojevi su vrlo uski (do 48 filtara po sloju) pa samo mali broj mapa značajki pridonosi globalnim informacijama. Preostale mape značajki ostaju nepromijenjene, a izlazni sloj donosi zaključak na temelju svih mapa značajki [15].

Prednosti DenseNet arhitekture su:

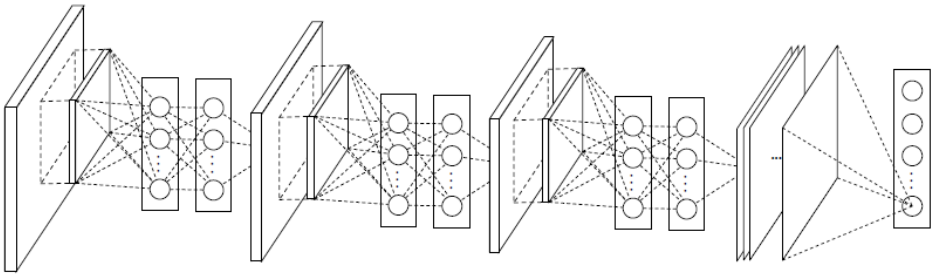
- poboljšani tok informacija i gradijenta kroz mrežu, što omogućuje lakše treniranje
- potaknuto ponovno korištenje značajki
- značajno smanjenje broja parametara mreže

2.2.3. NiN (Network in Network)

NiN (Network in Network) je vrsta konvolucijskih mreža koja drugačije pristupa izlučivanju značajki. Izlučene globalne informacije se temelje na bolje izlučenim

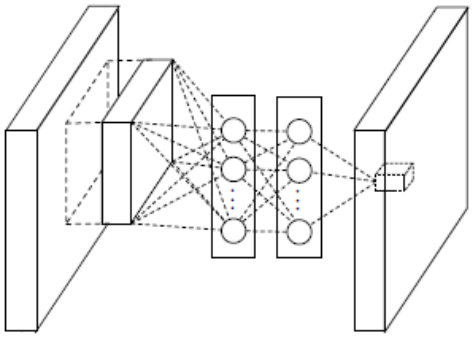
informacijama lokalnog područja. Cilj NiN-a je poboljšati diskriminativnost modela za lokalno područje receptivnog polja. Mreža je pogodna za zadatak klasifikacije.

Za razliku od konvencionalnih konvolucijskih mreža, jedna NiN mreža sadrži više mikro-neuronskih mreža koje izlučuju značajke iz receptivnog polja.



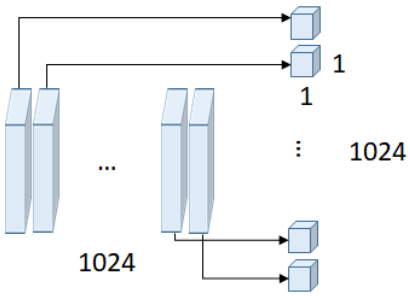
Slika 2-10: Struktura NiN-a (preuzeto iz [28]).

NiN je građen takozvanim mlpconv blokovima. Jedan blok je izveden kao višeslojni perceptron. Blok primjenjuje potpuno povezani sloj (mikro-mrežu) nad svakom lokacijom piksela (za svaku širinu i visinu). Mape značajki se izlučuju pomicanjem mikro-mreža preko ulaza (slično kao pomicanje konvolucijskog filtra) [23].



Slika 2-11: mlpconv blok (preuzeto iz [28]).

Poboljšano izlučivanje lokalnih informacija omogućuje provedbu globalnog avgpool-a nad mapama značajki u klasifikacijskom sloju mreže. Za svaki razred klasifikacijskog zadatka se generira mapa značajki u zadnjem mlpconv sloju. Prosjek svake mape značajki se šalje na izlaz [23].



Slika 2-12: Primjer globalnog sažimanja prosječnom vrijednošću (preuzeto iz [28]). Globalno sažimanje prosječnom vrijednošću u zadnjem mlpconv sloju generira po jednu mapu značajki za svaki klasifikacijski razred. Prosječne vrijednosti svake generirane mape čine vektor koji se šalje na izlaz modela [23].

Prednosti klasifikacije globalnim sažimanjem prosječnom vrijednošću nad klasifikacijom potpuno povezanim slojem su:

- globalno sažimanje prosječnom vrijednošću nema nikakve parametre koje treba optimizirati pa je u zadnjem sloju izbjegnuta prenaučenost
- uspostavljen je direktan odnos između značajki i odgovarajućih razreda
- globalni avgpool sumira prostornu informaciju pa je model više otporan na prostorne translacije ulaznih podataka

3. Opis metode

U ovom poglavlju opisujem i objašnjavam korišteno metričko ugrađivanje i korišteni model za semantičku segmentaciju. Opisujem skup podataka nad kojim sam proveo eksperimente te vanjske biblioteke koje sam koristio.

3.1. Vanjske biblioteke

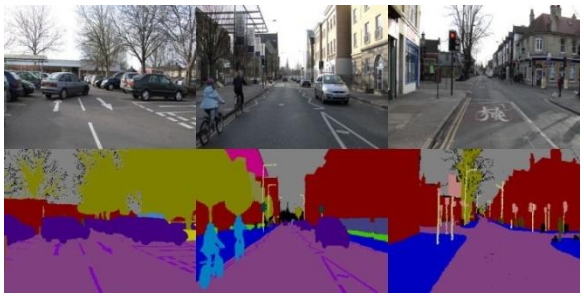
Programska implementacija ostvarena je u programskom jeziku Python 3. Implementacija se temelji na modifikaciji SwiftNet modela [24]. Model je modificiran tako da je na izlaz, prije Softmax gubitka, stavljen modul za predaktivaciju ArcFace. Za treniranje modela korištena je grafička procesna jedinica platforme Google Colab.

Radni okvir PyTorch sam koristio pri pisanju koda za učitavanje skupa podataka CamVid te pisanju implementacije ArcFace modula. Slike su učitane kao struktura podataka Tensor koju definira PyTorch. PyTorch je radni okvir baziran na Torch biblioteci i primjenjuje se u domeni dubokog učenja. Omogućuje automatsku diferencijaciju (unazadno propagiranje izračunatih gradijenta) što je vrlo korisno za treniranje dubokih modela. Također, nudi transparentno izvođenje na GPU ili CPU sklopovlju, razne optimizacijske postupke, funkcije gubitka te slojeve za izgradnju arhitekture mreže.

Skup podataka CamVid pripremio sam uz pomoć biblioteke pillow koja omogućuje učitavanje i rad na slikama. Originalne RGB slike sam pretvorio u crno-bijele slike. Smanjio sam broj razreda s 32 na 11. Vrijednost svakog piksela segmentacijskih mapa sam promijenio u indeks razreda kojem taj piksel pripada.

3.2. Skup podataka CamVid

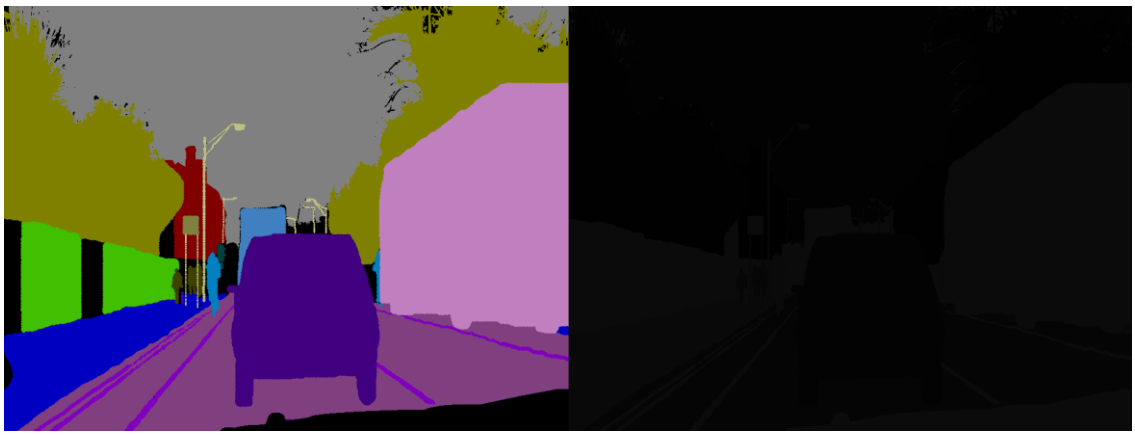
Cambridge-driving Labeled Video Database (CamVid) je skup podataka korišten za treniranje i validaciju SwiftNet modela [2]. CamVid sadrži slike koje su slikane iz perspektive vozila. Slike su u RGB formatu, a veličine su 720x960 piksela. Za svaku sliku postoji odgovarajuća datoteka sa semantičkim oznakama. Svaki piksel slike označen je u jedan od 32 semantička razreda. Svaki semantički razred je predstavljen jednom bojom radi lakšeg prikaza.



Slika 3-1: Primjeri CamVid slika i njihovih segmentacijskih mapa (preuzeto iz [7]).

Za potrebe ovog rada odabrao sam 11 originalnih klasa (zgrada, biciklist, stablo, stup, automobil, ograda, pješak, trotoar, prometni znak, cesta, nebo) dok sam preostale klase predstavio dodatnom klasom „void“. Klasa „void“ se ignorira i pri učenju i pri vrednovanju modela. Skup podataka sastoji se od 701 slike i podijeljen je u skup za učenje, skup za validaciju i skup za testiranje. Pošto radim s relativno malim skupom podataka, novi skup za učenje sam oblikovao spajanjem originalnih skupova za učenje i validaciju (468 slika). Također, novi skup za validaciju je sada originalni skup za testiranje (233 slike).

Eksperimente sam proveo na jednorazinskom modelu SwiftNet pa je bilo potrebno prilagoditi skup podataka. Jednorazinski model SwiftNet namijenjen je za rad s crno-bijelim slikama. Koristeći biblioteku pillow, transformirao sam RGB segmentacijske mape u format sivih tonova. U RGB formatu je svaki razred bio označen jednom bojom, a u formatu sivih tonova svaki razred ima indeks od 0 do 11 (11 je korišten za void).



Slika 3-2: Primjer originalne i transformirane segmentacijske mape (preuzeto iz [7]).

3.3. ArcFace gubitak (Additive Angular Margin Loss)

ArcFace ili Additive Angular Margin Loss je funkcija gubitka napravljena da unaprijedi diskriminativnost modela [10]. Izveden je iz Softmaxa normalizacijom težina i značajki te

njihovim skaliranjem. ArcFace se koristi kao predaktivacija izlaza koje uči gubitak unakrsne entropije. ArcFace predaktivacija dolazi prije Softmax aktivacije. ArcFace preslika naučena središta razreda u kutni prostor te penalizira kutnu marginu između značajki izlaznog sloja i odgovarajućeg razreda.

ArcFace gubitak je izražen kao:

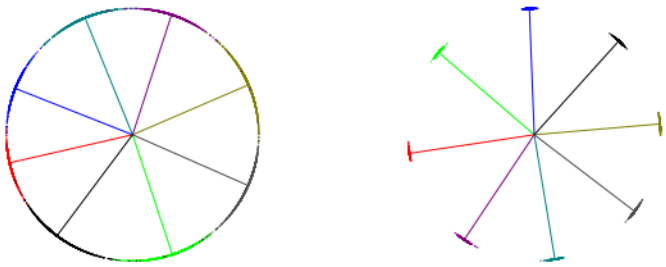
$$L = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s*\cos(\theta_{y_i}+m)}}{e^{s*\cos(\theta_{y_i}+m)}+\sum_{j=1,j\neq y_i}^{n}e^{s*\cos\theta_j}} \qquad (3.1)$$

gdje je N broj piksela u grupi za učenje, n broj klasa, a m umetnuta kutna margina koja povećava unutarklasnu kompaktnost i međuklasnu različitost. Vrijednost margine m je proizvoljno odabrana, a eksperimentalno je utvrđeno da najbolje rezultate daje vrijednost 0.5 [10]. θ_j je izveden iz transformiranih značajki Softmaxa:

$$W_j^T x_i = \left||W_j|\right| * ||x_i|| * \cos\theta_j \qquad (3.2)$$

θ_j predstavlja kut između težina W_j i značajki x_i.Težine W_j smatramo naučenim središtima razreda. Izraz $s * \cos(\theta_{y_i} + m)$ predstavlja skalirani zapis izlaznog sloja u kutnom prostoru (s predstavlja proizvoljnu skalu) [10].

Važna karakteristika ArcFace gubitka je da potiče veću sličnost između primjera istog razreda te veću različitost između primjera različitih razreda. Iz te karakteristike proizlazi svojstvo da korištenje ArcFace predaktivacije pridonosi boljem grupiranju značajki u odnosu na klasični gubitak unakrsne entropije uz Softmax aktivaciju.



Slika 3-3: Usporeba učenja gubitkom unakrsne entropije uz Softmax aktivaciju bez (lijevo) i sa (desno) ArcFace predaktivacijom.
Linije predstavljaju središta razreda, točke predstavljaju primjerke nekog razreda (preuzeto iz [10]).

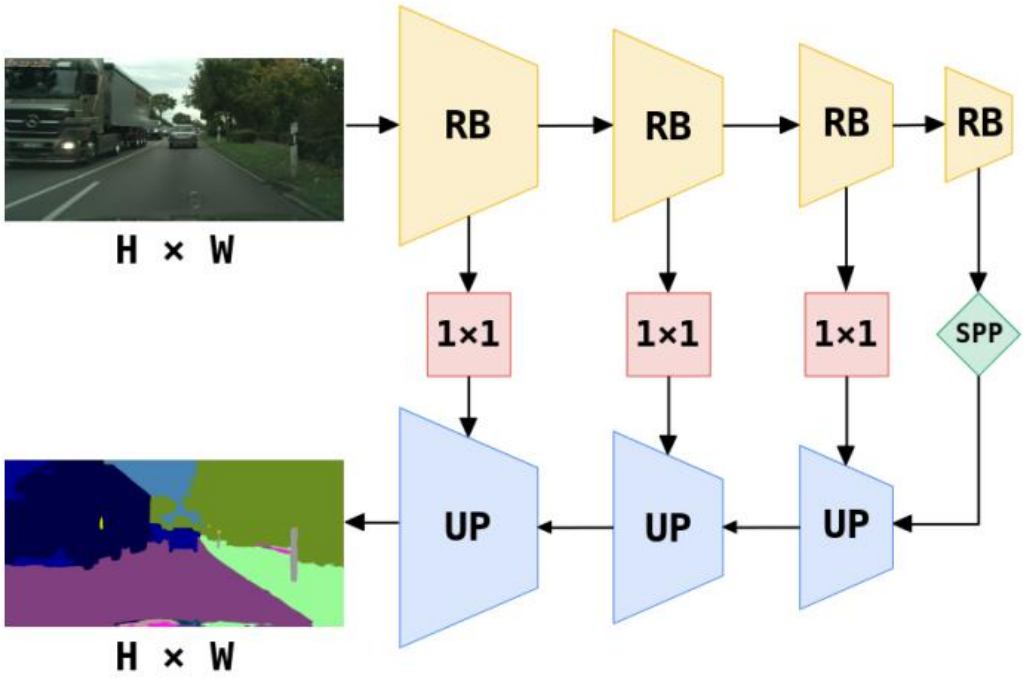
3.4. SwiftNet model

SwiftNet model zasniva se na asimetričnoj koder-dekoder arhitekturi pogodnoj za semantičku segmentaciju. Koder ima funkciju raspoznavanja, tj. izlučivanja značajki i ostvaren je klasifikacijskom okosnicom predtreniranom na ImageNetu. Koriste se ResNet-18 i MobileNet V2 jer postoje javno dostupni parametri, podržavaju učinkovito učenje i zaključivanje i imaju rezidualnu strukturu. Dekoder ima funkciju naduzorkovanja i izveden je što jednostavnije jer je empirijski pokazano da naduzorkovanje zahtijeva manji kapacitet od raspoznavanja [19]. Dvije su izvedbe SwiftNet modela: jednorazinski i višerazinski model. Razlika između izvedba je u tome da višerazinska izvedba raspoznavanje provodi usporedno na slikama različite veličine. Koder jednorazinske izvedbe ima nedovoljnu veličinu receptivnog polja te zbog toga sadrži modul za piramidalno prostorno sažimanje [32]. U ovom radu koristim i opisujem jednorazinski SwiftNet model [25].

3.4.1. Jednorazinski SwiftNet model

Jednorazinski model koristi koder za raspoznavanje i dekoder za naduzorkovanje kako bi ulaznu sliku transformirao u guste semantičke predikcije. Model prima ulaznu sliku u originalnoj rezoluciji.

Koder za raspoznavanje se sastoji od četiri konvolucijske grupe, a svaka od tih grupa izlučuje značajke i smanjuje rezoluciju slike. Iz svake konvolucijske grupe postoji preskočna veza prema sloju za naduzorkovanje. Veličina značajke je 32 puta manja od veličine ulaza po obje dimenzije. Značajke provodimo kroz modul za prostorno piramidalno sažimanje radi povećanja receptivnog polja modela čiji izlaz vodimo na dekoder za naduzorkovanje. Za razliku od kodera koji ima nekoliko konvolucijskih slojeva po konvolucijskoj grupi, dekoder ima samo jedan konvolucijski sloj po modulu za naduzorkovanje. Modul za naduzorkovanje obavlja tri koraka: 1) bilinearno naduzorkovanje reprezentacije niske rezolucije, 2) zbrajanje dobivene reprezentacije sa preskočnim vezama, 3) stapanje rezultata konvolucijom 3x3 [25].



Slika 3-4: Organizacija jednorazinskog SwiftNet modela (preuzeto iz [25]).

4. Eksperimenti

Osnovni jednorazinski SwiftNet model sam modificirao ArcFace modulom. Proveo sam eksperimente nad osnovnim i modificiranim modelom. Modele uspoređujem omjerom presjeka i unije te prikazujem rezultate u sljedećim potpoglavljima. Zanima me kako će korištenje implementacije ArcFace modula utjecati na semantičku izvedbu jednorazinskog SwiftNet modela.

4.1. Metrika

Mjere kojima evaluiram model su srednji omjer presjeka i unije te točnost piksela.

Omjer presjeka i unije (IoU) [11] za klasu C definiran je kao:

$$IoU_C = \frac{|X \cap Y|}{|X \cup Y|} \qquad (4.1)$$

X predstavlja skup piksela klase C, dok Y predstavlja skup piksela klasificiranih u klasu C.

Omjer presjeka i unije služi za definiranje srednjeg omjera presjeka i unije ($mIoU$). To je mjera koja predstavlja sumu omjera presjeka i unije za svaku klasu, podijeljenu s brojem klasa:

$$mIoU = \frac{1}{C} \sum_{i=0}^{C} \frac{TP_i}{TP_i + FP_i + FN_i} \qquad (4.2)$$

C predstavlja broj klasa. TP_i predstavlja broj točno pozitivno klasificiranih piksela, FP_i predstavlja broj netočno pozitivno klasificiranih piksela, a FN_i predstavlja broj netočno negativno klasificiranih piksela za klasu i.

Točnost piksela ($pixelAccuracy$) je mjera koja predstavlja omjer točno klasificiranih piksela za svaku klasu i ukupan broj piksela. Jednostavno se računa sljedećim izrazom:

$$pixelAccuracy = \frac{\sum_{i=0}^{C} n_{ii}}{\sum_{i=0}^{C} t_i} \qquad (4.3)$$

n_{ii} predstavlja broj točno klasificiranih piksela klase i. Broj svih piksela klase i izražava se kao $t_i = \sum_{j=0}^{C} n_{ij}$.

4.2. Rezultati eksperimenta s osnovnim SwiftNet modelom

Prvo sam uhodao osnovni jednorazinski SwiftNet model na transformiranom skupu podataka CamVid. Ulazne slike su dimenzije 720 x 960. Cilj je bio reproducirati rezultate iz originalnog SwiftNet rada [25]. Učenje je provedeno kroz 400 epoha i korišteno je po 12 primjera u svakoj grupi. Veličina slučajnog isječka postavljena je na 448. Korak učenja je postavljen na $1 \cdot 10^{-4}$, a korišten je optimizator Adam.

Tablica 1 - rezultati eksperimenta s osnovnim SwiftNet modelom

Preciznost piksela	84.91 %
Najbolji mIoU	63.33 %

Tablica 2 – rezultati eksperimenta s osnovnim SwiftNet modelom za svaki semantički razred

Razred	**IoU preciznost**
zgrada	80.06 %
stablo	76.43 %
nebo	93.51 %
automobil	70.37 %
znak	30.52 %
cesta	87.53 %
pješak	42.27 %
ograda	45.68 %
stup	32.32 %
trotoar	75.05 %
biciklist	62.93 %

4.3. Rezultati eksperimenta s modificiranim SwiftNet modelom

Uhodao sam jednorazinski SwiftNet model modificiran ArcFace gubitkom na transformiranom skupu podataka CamVid. Cilj je bio postići bolju semantičku izvedbu od osnovnog modela. Učenje je provedeno kroz 400 epoha. Korištenje implementacije ArcFace modula je memorijski zahtjevno za Google Colab platformu pa je veličina grupe ograničena na 1. Veličina slučajnog isječka je postavljena na 448. Korak učenja je postavljen na $1 \cdot 10^{-4}$, a korišten je optimizator Adam.

Tablica 3 – rezultati eksperimenta s modificiranim SwiftNet modelom

Preciznost piksela	87.1 %
Najbolji mIoU	68.13 %

Tablica 4 - rezultati eksperimenta s modificiranim SwiftNet modelom za svaki semantički razred

Razred	**IoU preciznost**
zgrada	70.01 %
stablo	85.05 %
nebo	81.0 %
automobil	83.66 %
znak	2.73 %
cesta	77.38 %
pješak	66.22 %
ograda	78.53 %
stup	56.2 %
trotoar	65.92 %
biciklist	82.73 %

Zaključak

U radu sam se bavio jednim od glavnih zadataka računalnog vida, semantičkom segmentacijom. Proučavao sam način za poboljšanje semantičke izvedbe modela. Bolja semantička izvedba je željena u slučaju izvandistribucijskih ulaza.

U teoretskom dijelu rada sam predstavio semantičku segmentaciju, duboke konvolucijske modele i njihovu povezanost. U dijelu o programskoj implementaciji sam predstavio ArcFace modul kao jedan od načina za poboljšanje semantičke izvedbe. Također sam predstavio jednorazinski SwiftNet model za semantičku segmentaciju. Opisao sam provedene eksperimente u svrhu poboljšanja semantičke izvedbe i prikazao njihove rezultate.

Jednorazinski SwiftNet model modificiran ArcFace modulom je postigao malo bolje vrijednosti definiranih metrika. U točnosti piksela, modificirani model je bolji od osnovnog za 2.19 postotnih bodova (odsad pa na dalje p.b.). U najboljem srednjem omjeru presjeka i unije, modificirani model je bolji od osnovnog za 4.8 p.b. Modificirani model je ostvario veću točnost omjera presjeka i unije za pojedine semantičke razrede (stablo: +8.62 p.b., automobil: +13.29 p.b., pješak: +23.95 p.b., ograda: +32.85 p.b., stup: +23.88 p.b., biciklist: +19.8 p.b.). Za pojedine semantičke razrede je ostvario nešto lošiju metriku (zgrada: -10.05 p.b., nebo: -12.51 p.b., cesta: -10.15 p.b., trotoar: -9.13 p.b.). Dok je za semantički razred znak ostvario znatno lošiju metriku u odnosnu na osnovni model (znak: -27.79 p.b.).

Modificirani model je postigao malo bolju generalnu metriku (točnost piksela i srednji omjer presjeka i unije) i višu razinu točnosti omjera presjeka i unije za 10 od 11 semantičkih razreda (u rasponu od 56.2 p.b. do 85.05 p.b.). Međutim, važno je napomenuti da je modificirani model postigao prikazane rezultate s veličinom grupe 1. Nije tipično da model postigne takve rezultate s tako malom veličinom grupe pa ne mogu sa sigurnošću zaključiti da ArcFace modul poboljšava semantičku izvedbu jednorazinskog SwiftNet modela.

Literatura

1. Babić, I. (2020). Autentikacija osoba analizom izgleda lica, Diplomski rad.
2. Badrinarayanan, V., Kendall, A., & Cipolla, R. (2015). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation.
3. Baheti, P. (2022). *https://www.v7labs.com/blog/convolutional-neural-networks-guide#cnn-architecture*. Dohvaćeno iz https://www.v7labs.com.
4. Bevandić, P., Krešo, I., Oršić, M., & Šegvić, S. (2021). Dense open-set recognition based on training with noisy negative images.
5. Bratulić, J. (2020). Semantička segmentacija kolničkih trakova, Završni rad.
6. Brownlee, J. (2019). *https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/*. Dohvaćeno iz https://machinelearningmastery.com.
7. Cambridge, U. o. (n.d.). *http://mi.eng.cam.ac.uk/research/projects/VideoRec/CamVid/*. Dohvaćeno iz http://mi.eng.cam.ac.uk.
8. Courville, A., Goodfellow, I., & Bengio, Y. (2015). *Deep Learning.*
9. datahacker.rs. (2021). *https://datahacker.rs/020-overview-of-semantic-segmentation-methods/*. Dohvaćeno iz https://datahacker.rs.
10. Deng, J., Guo, J., & Xue, N. (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition.
11. Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2009). The Pascal Visual Object Classes (VOC) Challenge.
12. Gupta, D. (2019). *https://divamgupta.com/image-segmentation/2019/06/06/deep-learning-semantic-segmentation-keras.html*. Dohvaćeno iz https://divamgupta.com.
13. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition.

14. He, K., Zheng, X., Ren, S., & Sun, J. (2016). Identity Mappings in Deep Residual Networks.

15. Huang, G., Liu, Z., & van der Maaten, L. (2017). Densely Connected Convolutional Networks.

16. Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

17. Jordan, J. (2018). *https://www.jeremyjordan.me/semantic-segmentation/*. Dohvaćeno iz https://www.jeremyjordan.me.

18. Kone, C. (2019). *https://towardsdatascience.com/introducing-convolutional-neural-networks-in-deep-learning-400f9c3ad5e9*. Dohvaćeno iz https://towardsdatascience.com.

19. Krešo, I., Krapac, J., & Šegvić, S. (2015). Efficient Ladder-style DenseNets for Semantic Segmentation of Large Images.

20. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (n.d.). ImageNet Classification with Deep Convolutional Neural Networks.

21. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998).

22. Li, H., Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2017). Visualizing the Loss Landscape of Neural Nets.

23. Lin, M., Chen, Q., & Yan, S. (2014). Network In Network.

24. Oršić, M., & Šegvić, S. (2020). Efficient semantic segmentation with pyramidal fusion.

25. Oršić, M., Šegvić, S., Bevandić, P., & Krešo, I. (2019). In Defense of Pre-trained ImageNet Architectures for Real-time Semantic Segmentation of Road-driving Images.

26. Shelhamer, E., Long, J., & Darrell, T. (n.d.). Fully Convolutional Networks for Semantic Segmentation.

27. Šegvić, S., Bevandić, P., Šarić, J., Oršić, M., & Grubišić, I. (n.d.). *http://www.zemris.fer.hr/~ssegvic/du*. Dohvaćeno iz http://www.zemris.fer.hr/~ssegvic.

28. Tsang, S.-H. (2019). *https://towardsdatascience.com/review-nin-network-in-network-image-classification-69e271e499ee*. Dohvaćeno iz https://towardsdatascience.com.

29. vision.stanford.edu. (n.d.). *https://cs231n.github.io/convolutional-networks/*. Dohvaćeno iz https://cs231n.github.io.

30. Wang, H. (n.d.). *https://www.researchgate.net/figure/Toy-example-illustrating-the-drawbacks-of-max-pooling-and-average-pooling_fig2_300020038*. Dohvaćeno iz https://www.researchgate.net.

31. Wang, J., Sun, K., Cheng, T., & Borui, J. (2019). Deep High-Resolution Representation Learning for Visual Recognition.

32. Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid Scene Parsing Network.

Semantička segmentacija s naučenim središtima razreda

Sažetak

Semantička segmentacija je temeljni zadatak računalnog vida s raznovrsnim primjenama u industriji i znanosti. Moderni modeli postižu odlične rezultate u semantičkoj segmentaciji, međutim problem nije riješen u slučaju izvandistribucijskih ulaza. U ovom radu proučavam je li moguće ostvariti bolje rezultate ako su značajke više grupirane, a razredi jasno odvojivi. Kako bi ostvario navedeno, proučavao sam i isprobao ArcFace gubitak na modelu za semantičku segmentaciju SwiftNet. Opisao sam korišteni ArcFace gubitak i SwiftNet model. Prikazao sam rezultate kroz definiranu metriku i donio konačni zaključak. Uz to su opisane i osnove semantičke segmentacije, dubokih konvolucijskih modela te same programske implementacije.

Ključne riječi: semantička segmentacija, računalni vid, ArcFace, SwiftNet, duboki konvolucijski modeli

Semantic segmentation with learned class centers

Summary

Semantic segmentation is a fundamental task of computer vision with a variety of applications in industry and science. Modern models achieve excellent results in semantic segmentation, however the problem is not solved in the case of non-distribution inputs. In this paper, I study whether it is possible to achieve better results if the features are more clustered and the classes are clearly separable. To accomplish this, I studied and tested the ArcFace loss on the SwiftNet semantic segmentation model. I described the used ArcFace loss and SwiftNet model. I presented the results through defined metrics and came to a final conclusion. In addition, the basics of semantic segmentation, deep convolutional models and the program implementation itself are described.

Keywords: semantic segmentation, computer vision, ArcFace, SwiftNet, deep convolutional models