SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1545

Istovremena lokalizacija i segmentacija objekata

Petra Marče

Zagreb, srpanj 2017.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA DIPLOMSKI RAD PROFILA

Zagreb, 15. ožujka 2017.

DIPLOMSKI ZADATAK br. 1545

Pristupnik: **Petra Marče (0036473653)**
Studij: Računarstvo
Profil: Računarska znanost

Zadatak: **Istovremena lokalizacija i segmentacija objekata**

Opis zadatka:

Detekcija objekata i semantička segmentacija su važni problemi razumijevanja prirodnih scena s mnogim zanimljivim primjenama. U dubokim arhitekturama računalnog vida, semantička segmentacija provodi se konvolucijskom klasifikacijom slikovnih okana, dok se lokalizacija provodi konvolucijskom regresijom okvira objekta. Do sada su se ti zadatci razmatrali odvojeno. Međutim, postojeći skupovi za učenje semantičke segmentacije (PASCAL, Cityscapes) sadrže i informaciju o pripadnosti piksela pojedinačnim objektima. To navodi na ideju da bi se lokalizacija objekata i semantička segmentacija mogli učiti zajedno, istovremenim optimiranjem dvaju gubitaka. Tema ovog rada je istražiti prednosti i nedostatke takvog pristupa za detekciju objekata u prometnim scenama.
U okviru rada, potrebno je proučiti i ukratko opisati postojeće lokalizacijske i segmentacijske pristupe utemeljene na dubokom učenju. Predložiti arhitekturu dubokog modela koji se može učiti s lokalizacijskim i segmentacijskim gubitkom. Uhodati postupke učenja i validiranja hiperparametara. Primijeniti naučene modele na kolekcijama Cityscapes i KITTI. Prikazati i ocijeniti ostvarene rezultate. Predložiti pravce budućeg razvoja.
Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Zadatak uručen pristupniku: 10. ožujka 2017.
Rok za predaju rada: 29. lipnja 2017.

Mentor:
Izv. prof. dr. sc. Siniša Šegvić

Djelovođa:
Doc. dr. sc. Tomislav Hrkać

Predsjednik odbora za diplomski rad profila:
Prof. dr. sc. Siniša Srbljić

SADRŽAJ

1. Uvod

Umjetna inteligencija je grana računarske znanosti koja se bavi rješavanjem kognitivnih problema koji su lagani za ljude, a iznimno teški za računala. Jedno od područja umjetne inteligencije je računalni vid. Računalni vid je znanost kojoj je cilj razviti sustav koji je sposoban vizualnu informaciju interpretirati jednako dobro kao što to čine živa bića. Sustavi računalnog vida "gledaju" svijet pomoću jedne kamere ili više kamera postavljenih u sustav tako da svaka snima istu scenu ali sa različitih pozicija i iz različitih kuteva. Osnovna vizualna informacija, je dakle jedna slika u boji. Cilj je, iz jedne ili više slika, razumjeti scenu, što podrazmijeva prepoznavanje objekata u sceni i njihovih međusobnih odnosa. Primjer gdje je takav sustav potreban su autonomna vozila. Autonomno vozilo je vozilo koje je sposobno upravljati samim sobom bez ljudske interakcije. Takvi sustavi se razvijaju kako bi promet postao sigurniji ali i učinkovitiji, ili kao zamjena za vozača u operacijama koje su opasne za ljude. Takvo vozilo svakako mora imati sofisticirani sustav računalnog vida koji može prepoznati objekte na razini piksela u stvarnom vremenu, kako bi moglo donositi odluke u upravljanju.

Takav sustav između ostalog, treba obavljati zadatak segmentacije te lokalizacije objekata. Segmentacija je postupak pridruživanja kategorije svakom pikselu slike. Na primjer za sliku iz prometne scene, segmentacija će odrediti maske koje određuju piksele koji pripadaju cesti, nebu, okolnim zgradama i ostalim sudionicima u prometu. Kategorije kao što su cesta, nebo, vegetacija koje se ne kreću, i nisu sudionici u prometu, tipično unutar slike nemaju više instanci. S druge strane u realnoj prometnoj sceni, može biti prisutan veliki broj različitih automobila, kamiona, pješaka, biciklista, gdje je svaka od tih instanci važna. Kod tih kategorija koje imaju instance, važno je odrediti masku za svaku instancu zasebno. Detekcija objekata je postupak utvrđivanja prisutnosti objekata određenih kategorija i njihovih točnih pozicija.

U novije vrijeme, najbolji rezultati na zadatcima segmentacije i detekcije objekata postižu se konvolucijskim neuronskim mrežama i postupcima dubokog učenja. Tu se problem ne riješava eksplicitno algoritamski već se uči nadzirano iz označenih podataka. To se radi na način da se definira model s velikim brojem slobodnih parametara koji se podešavaju tako da izlaz modela bude sličan oznakama, odnosno da je pogreška između oznaka i izlaza

modela mala, ali tako da model ima dobre rezultate i na neviđenim ali sličnim podatcima. Iako se takva ideja pojavila i prije, zaživjela je nedavno s porastom računalne moći i količine podataka za učenje.

U okviru ovog rada istražuje se postupak lokalizacije objekata i segmentacije slika promentih scena konvolucijskim neuronskim mrežama. U drugom poglavlju opisane su korištene ideje, modeli te korišteni algoritmi optimizacije. Treće poglavlje govori o detekciji objekata općenito, postojećim pristupima uz detaljniji opis metode SSD (engl. *single shot multibox detector*) korištene u eksperimentima. U trećem poglavlju ukratko su opisani ispitni skupovi korišteni u eksperimentima. U okviru četvrtog poglavlja opisani su eksperimenti s SSD-om na trima različitim ispitnim skupovima. Peto poglavlje u centar stavlja segmentaciju slike klasifikacijom piksela potpuno konvolucijskim neuronskim mrežama. U šestom poglavlju opisan je model za istodobnu segmentaciju i lokalizaciju objekata. Sedmo poglavlje opisuje dva ne toliko uspješna eksperimenta provedena u okviru rada.

2. Korišteni modeli i algoritmi

2.1. Nadzirano učenje

Dvije vrste problema koji se rješavaju postupcima nadziranog učenja su klasifikacija i regresija. **Klasifikacija** ima zadatak klasificirati primjer u jedan od razreda kojemu on pripada. Primjerice, kod zadatka klasifikacije slika u razrede automobil, pješak i biciklist, želimo da klasifikacija pridijeli razred pješak svim slikama koje zaista prikazuju pješaka. Svaka slika kod takve klasifikacije može biti svstana u samo jedan od tri razreda. Klasifikacija dakle primjeru pridružuje jednu od predefiniranih diskretnih vrijednosti. **Regresija** s druge strane primjeru pridružuje realnu kontinuiranu vrijednost. Primjer za regresiju je procijena koordinata centra objekta kojemu piksel slike pripada. Izlaz takve regresije bit će dvije vrijednosti koje predstavljaju y i x koordinatu traženog centra. Razlika između regresije i klasifikacije je u tome, je li ciljna varijabla kontinuirana ili diskretna.

Kod nadziranog učenja na raspolaganju mora biti skup označenih primjera, odnosno skup parova primjer i oznaka. Primjer se definira kao vektor značajki $\boldsymbol{x} = (x_1, x_2, ..., x_n)^\top$, gdje je n dimenzija vektora. Neka je $\mathcal{X}$ skup svih mogućih primjera. Kod klasifikacije slika jedan primjer bili bi pikseli slike. Oznaka primjera y je oznaka razreda za klasifikaciju, odnosno vrijednost funkcije koja se aproksimira u točki primjera za regresiju. Pretpostavka svih algoritama strojnog učenja je da su primjeri uzorkovani nezavisno i iz iste zajedničke distribucije. Skup primjera za učenje $\mathcal{D}$ sastoji se od $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, gdje je N ukupan broj primjera za učenje. Kako je N konačan broj, skup primjera za učenje je podskup skupa svih mogućih primjera.

Zadaća klasifikacijskog algoritma je naučiti **model** ili hipotezu $h : \mathcal{X} \to \{0, 1, .., |C|-1\}$ koja za primjer x određuje njegovu klasu, gdje $|C|$ označava broj klasa. Skup mogućih hipoteza $\mathcal{H}$ koje odabiremo kao skup mogućih rješenja nazivamo prostor hipoteza ili **klasa modela**. Primjer klase modela mogu biti svi pravci u 2D prostoru, a jedan konkretan model iz te klase pravac $y = 3x + 4$.

Kada hipoteza h ispravno klasificira sve primjere iz skupa za učenje kaže se da je konzistentna s primjerima za učenje. Koliko dobro hipoteza h klasificira primjere za učenje pokazuje pogreška učenja ili empirijska pogreška.

$$E(h|\mathcal{D}) = \frac{1}{N}\sum_{i=1}^{N} |h(\boldsymbol{x}^{(i)}) - y^{(i)}| \tag{2.1}$$

Empirijska pogreška klasifikacije (2.1) je udio neispravno klasificiranih primjera. Hipoteza je dakle konzistentna s primjerima za učenje ako je empirijska pogreška jednaka nuli. Ako ne postoji $h \in \mathcal{H}$ takva da je h konzistentna s primjerima za učenje, tada klasa modela $\mathcal{H}$ nije dovoljno velikog kapaciteta ili složenosti da bi naučio klasifikaciju. Kapacitet modela je mjera koja odražava mogućnost prilagodbe podatcima.

Učenje hipoteze, loše je definiran problem jer primjeri iz skupa za učenje nisu dovoljni za jednoznačno induciranje hipoteze h. Ne postoji garancija da će model koji ispravno klasificira primjere iz D, dobro klasificirati i ostale primjere iz $\mathcal{X}$. Sposobnost modela da ispravno klasificira i neviđene primjere naziva se moć generalizacije.

Učenje i generalizacija nisu mogući bez dodatnih pretpostavki, zbog čega se uvodi induktivna pristranost [1] koja omogućuje induktivno učenje. Induktivna pristranost uvodi se odabirom klase modela koji dolaze u obzir te odabirom načina pretraživanja tog skupa. Odabirom skupa hipoteza $\mathcal{H}$ koje uopće dolaze u obzir, uvodi se tzv. pristranost ograničavanjem ili pristranost jezika [1]. Definiranjem načina pretraživanja hipoteza s ciljem pronalaska točno jednog modela h iz $\mathcal{H}$, dajemo prednost nekim hipotezama, što se naziva pristranost pretraživanja ili pristranost preferencijom [1]. Na primjer, u okviru ovog rada pristranost ograničavanjem uvela sam odabirom klase modela neuronske mreže određene arhitekture, a pristranost pretraživanja regularizacijom težina.

Treba reći da i sam skup za učenje nije savršen zbog prisutnosti šuma u podatcima. Šum je dakle neželjena anomalija u podatcima koja može biti prisutna radi pogreški u označavanju, postojanja latentnih varijabli, nejasnih granica klasa te nepreciznosti u mjerenju značajki.

Kod regresije, cilj je na temelju primjera iz $\mathcal{D}$ naučiti nepoznatu funkciju $f : \mathcal{X} \to \mathbb{R}$ takvu da vrijedi $y^{(i)} = f(\boldsymbol{x}^{(i)})$. Zbog prisutnosti šuma u podatcima za učenje, zapravo se uči funkcija $y^{(i)} = f(\boldsymbol{x}^{(i)} + \epsilon)$, gdje je ϵ slučajni šum. Empirijska pogreška hipoteze h koja je aproksimacija funkcije f, na skupu $\mathcal{D}$ može se definirati kao zbroj kvadratnih odstupanja predviđene od stvarne vrijednosti funkcije (2.2).

$$E(k|\mathcal{D}) = \frac{1}{2}\sum_{i=1}^{N} (y^{(i)} - h(\boldsymbol{x}^{(i)}))^2 \tag{2.2}$$

Komponente algoritma nadziranog učenja, koje ujedno uvode induktivnu pristranost su

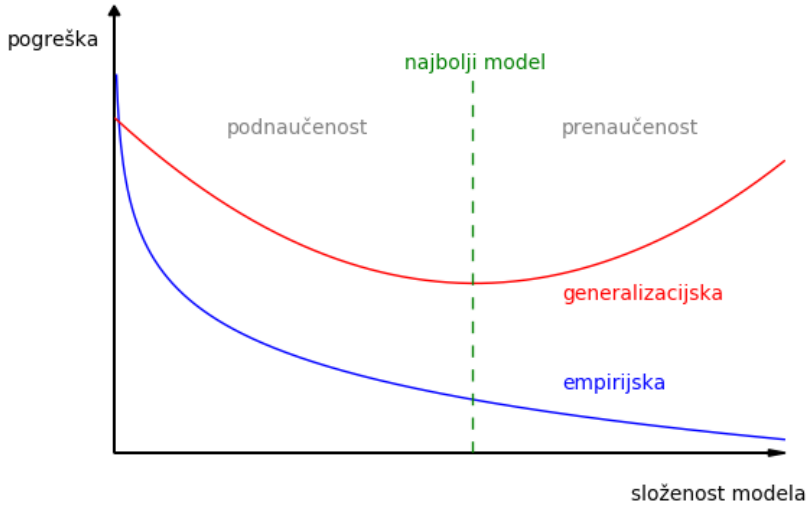
1. klasa modela $\mathcal{H}$ koja je skup hipoteza h parametriziranih s θ
2. funkcija gubitka L koja uz određene parametre θ mjeri odstupanje predikcije modela od ciljnjih vrijednosti

3. Oprimizacijski postupak kojim se pronalaze vrijednosti θ^* za koje je vrijednost funkcije gubitka minimalna, odnosno

$$\theta^* = argmin_\theta L(\theta|\mathcal{D})$$

Pitanje je kako odabrati dobru klasu modela. Što je kapacitet klase modela veći to će empirijska pogreška biti manja, ali željeno svojstvo je generalizacija. Odabere li se klasa modela $\mathcal{H}$ koja je puno složenija od optimalne, hipoteze $h \in \mathcal{H}$ će se moći jako dobro prilagoditi podatcima za učenje. U slučaju klasifikacije moći će "zapamtiti" klasifikacije viđenih primjera, bez da nauče ono bitno pa će se svojstvo generalizacije izgubiti. Kod regresije funkcija aproksimacije neće biti otporna na šum u podatcima. U tom slučaju kaže se da je model **prenaučen** [1]. Prenaučenost je dakle problem kojeg karakterizira mala pogreška na viđenim podatcima uz veliku pogrešku na neviđenim podatcima. Takav model puno varira u ovisnosti o skupu za učenje pa se kaže da složeni modeli imaju visoku varijancu. Ukoliko je pak model $\mathcal{H}$ prejednostavan u odnosu na stvarnu funkciju preslikavanja, hipoteza se ne može dovoljno dobro prilagoditi podatcima iz samog skupa za učenje $\mathcal{D}$. Ako hipoteza loše klasificira viđene primjere, teško je da će dobro klasificirati nove. U tom slučaju model pati od **podnaučenosti** [1]. Podnaučenost je dakle, slučaj u kojem zbog nedovoljne ekspresivnosti modela pogreška na skupu za učenje uvijek ostaje velika. U jednostavan model ugrađeno je više pretpostavki pa se kaže da ima veću pristranost. Slika 2.1 prikazuje ovisnost empirijske i generalizacijske pogreške o složenosti modela. Odabir modela svodi se na traženje modela koji minimizira i pristranost i varijancu čime ostvaruje najbolju generalizaciju.

Unakrsna provjera je jednostavan način da se odaber model koji nije ni podnaučen ni prenaučen. Ideja je da se skup podataka razdvoji na međusbno disjunktne podskupove za učenje, validaciju i ispitivanje (testiranje). Modelu se prikazuju primjeri iz skupa za učenje, a njegove performanse mjere se na neviđenim primjerima iz skupa za validaciju. Odabire se hipoteza čije su performanse na skupu za validaciju najbolje. Iako se na ovaj način primjeri skupa validaciju ne prikazuju modelu u postupku učenja, skup za validaciju koristi se za krajnji odabir hipoteze. Zbog toga se moć generalizacije mjeri na trećem, potpuno netaknutom skupu za ispitivanje. Generalizacijska pogreška je dakle pogreška na skupu za ispitivanje.



Slika 2.1: Pogreška u ovisnosti o složenosti modela

Kada je model toliko jednostavan da se ne može dobro prilagoditi skupu za učenje, empirijska pogreška je velika uz još veću pogrešku generalizacije. U tom slučaju model je podnaučen (lijevi dio grafa). Naspram tome, ako je model presložen, može se podatcima prilagoditi toliko dobro da izgubi svojstvo generalizacije. Kako je model zapamtio podatke za učenje, empirijska pogreška će biti mala, dok će pogreška generalizacije na neviđenim primjerima biti velika (desni dio grafa). Optimalan model nalazi se negdje između te dvije krajnosti.

2.2. Duboki konvolucijski modeli

2.2.1. Neuronske mreže

Konvolucijska neuronska mreža je unaprijedna neuronska mreža prilagođena za rad sa slikama. Neuronske mreže općenito su modeli koji se sastoje od skupa međusobno povezanih osnovnih jedinica - neurona. Izlaz jednog neurona y funkcija je ulaza i parametara odnosno težina neurona. Težina w_0 naziva se prag neurona i na taj ulaz se uvijek dovodi jedinica.

$$y = f(\vec{w} \cdot \vec{x}) = f(\sum_i w_i x_i) \tag{2.3}$$

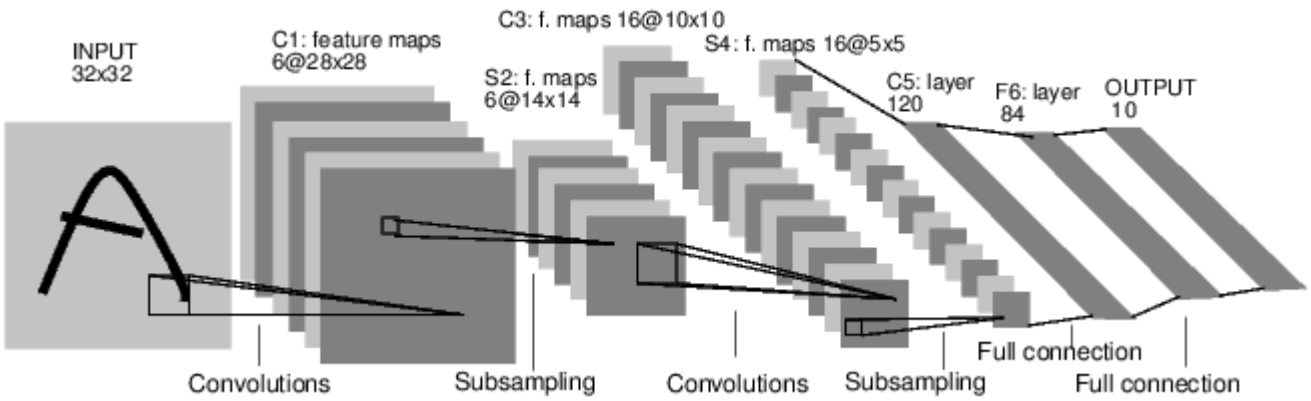
Neuronska mreža je struktura koja nastaje povezivanjem više slojeva neurona. Sloj koji predstavlja ulazni signal naziva se ulazni sloj, a izlaz mreže izlazni sloj. Između ulaznog i izlaznog sloja može se nalaziti jedan ili više skriveni sloj. Funkcija f naziva se prijenosna ili aktivacijska funkcija. Argument aktivacijske funkcije je linearna kombinacija ulaza i težina. Kad bi aktivacijska funkcija također bila linearna cijeli neuron bi obavljao linearnu funkciju. S obzirom da su problemi umjetne inteligencije koji se mrežama nastoje naučiti redom nelinearni, potrebno je u mrežu unijeti nelinearnost. To je uloga aktivacijske funkcije. Neuronske mreže uče se algoritimima koji zahtijevaju izračun gradijenata funkcije gubitka po svim parametrima mreže. Zbog toga je nužno da svaka aktivacijska funkcija bude derivabilna. Uobičajeno korištene aktivacijske funkcije su sigmoidalna funkcija $\sigma(x)$ i linearna rektifikacijska funkcija $relu(x)$

$$\sigma(x) = \frac{1}{1 - e^{-\alpha x}} \tag{2.4}$$

$$relu(x) = max(0, x) \tag{2.5}$$

Neuronska mreža nastaje međusobnim povezivanjem više slojeva neurona. Mreža u kojoj se ulazni signal propagira jednosmjerno, od ulaza prema izlazu naziva se unaprijednom. Ulazni sloj predstavlja ulazni signal a izlazni izlaz mreže. Svi slojevi između, nazivaju se skriveni slojevi. Broj neurona u svakom skrivenom sloju i broj skrivenih slojeva određuje složenost mreže. Broj izlaza mreže ovisi o problemu. Tako kod zadatka klasifikacije ulaza u jednu od K kategorija imamo K izlaza od kojih svaki predstavlja vjerojatnost da ulaz pripada toj kategoriji. U tom slučaju se u izlaznom sloju koristi aktivacijska funkcija softmax (2.6). Softmax je funkcija čiji je ulaz vektor, a izlaz vektor jednakih dimenzija čiji elementi u zbroju daju 1 pa ih se može interpretirati kao vjerojatnosnu razdiobu diskretne slučajne varijable.

$$softmax(x_i) = \frac{e_i^x}{\sum_{j=1}^{N} e_j^x}, \quad x_i \in \vec{x} \tag{2.6}$$



Slika 2.2: Primjer arhitekture konvolucijskog modela: LeNet-5 za klasifikaciju slika rukom pisanih znamenki. (slika preuzeta iz članka [2])

Ulaz u model je crno bijela slika dimenzija 32×32. Prvi konvolucijski sloj C1 obavlja konvoluciju jezgrom 5×5 bez korištenja nadopune s korakom 1, pa su dimenzije mapa prema formuli (2.7) tog sloja jednake $\frac{32-5+0}{1} + 1 = 28$. Broj mapa značajki prvog sloja je 6. Sloj S2 je sloj nepreklapajućeg sažimanja veličine bloka 2×2. Dimenzije mapa sloja S2 su stoga dvostruko manje nego prethodnog sloja, odnosno 14×14. Sloj sažimanja ne smanjuje broj značajki pa broj mapa u sloju S2 ostaje 6. Konvolucijski sloj C3 obavlja konvoluciju na isti način kao sloj C1 pa su dimenzije mapa sloja $14 - 5 + 1 = 10$. Broj značajki u sloju je jednak 16. Sloj sažimanja S4 isti je kao $S2$ pa sadrži 16 mapa značajki dimenzija 5×5. Slijede dva potpuno povezana sloja sa po 120 i 84 neurona, te izlazni klasifikacijski sloj koji ima 10 neurona gdje svaki od njih predstavlja jedan razred za jednu od deset znamenki.

Sloj neuronske mreže čiji neuroni na ulaz dobivaju izlaz svih neurona prethodnog sloja naziva se potpuno povezani sloj. Broj parametara jednog neurona potpuno povezanog sloja je dakle jednak dimenzionalnosti vektora ulaza. Kada bi sliku doveli na ulaz potpuno povezanog sloja, svi bi neuroni sloja vidjeli svaki piksel slike. Problemi kod takvog pristupa su što ne koristi prostornu informaciju te prokletstvo dimenzionalnosti.

2.2.2. Struktura i posebnosti dubokih konvolucijskih modela

Konvolucijske neuronske mreže koriste implicitnu pretpostavku da su ulazi slike, pa su u skladu s tim njeni neuroni imaju topologiju rešetke. Ulazi svih slojeva konvolucijskog modela općenito su tenzori. Jedan takav tenzor dimenzija $M_h \times M_w \times D$, može se promatrati kao D rešetki rezolucije $M_h \times M_w$ naslaganih jedna iza druge. Jedan neuron konvolucijskog modela je jedan element jedne takve rešetke koja se naziva mapa značajki. Konvolucijski duboki model sastoji se od niza konvolucijskih slojeva ispresjecanih slojevima sažimanja. Posljednji slojevi konvolucijskog modela mogu biti potpuno povezani slojevi. Primjer jedne takve arhitekture prikazan je slikom 2.2. Mreža se naziva konvolucijskom jer konvolucijski slojevi u kojima su pohranjene težine, umjesto skalarnog produkta ulaza i težina sada ko-

riste operaciju konvolucije. Težine konvolucijskih slojeva organizirane su u matrice koje se nazivaju konvolucijskim jezgrama.

Konvolucijski sloj kao što je već rečeno, obavlja operaciju konvolucije ulaza i jezgre u kojoj su pohranjene težine. Ulaz u sloj su mape prethodnog sloja. Na ulazu prvog konvolucijskog sloja mreže može biti monokromatska ili slika u boji. Dimenzije jegre $K_h \times K_w$ su hiperparametar sloja. U modernim konvolucijskim dubokim modelima najćešće se koriste kvadratne jezgre dimenzije K gdje je K neparan. Konvolucija se vrši na sljedeći način. Unutar ulazne mape definira se prozor dimenzija jezgre koji se naziva receptivno polje. Elementi ulazne mape unutar definiranog receptivnog polja se množe s odgovarajućim elementima jezgre. Rezultat se sprema u pripadnu lokaciju izlazne mape. Dimenzija jezgre, odnosno veličina receptivnog polja određuje broj elemenata ulazne mape koji sudjeluju u izračunu jedne lokacije mape idućeg sloja. Sve vrijednosti izlazne mapa dobivaju se prolazom receptivnog polja kroz cijelu ulaznu mapu. Konvolucijska jezgra kao da svojim prolazom filtrira sliku, pa se osim naziva konvolucijska jezgra koristi i termin konvolucijski filtar.

Neka je M^l mapa l-tog sloja, a M^{l-1} mapa prethodnog sloja. Neka su dimenzije mape $M_h \times M_w$. Isto tako, neka se konvolucija vrši kvadratnom jezgrom dimenzije K. Neka je S korak pomaka jezgre po širini odnosno visini unutar ulazne mape prilikom konvolucije, a P broj redaka nadopune nulama uzduž granica mape. Tada je dimenzija mape značajki u nekom sloju dana izrazom (2.7). Veličina nadopune P najčešće se bira tako da dimenzije mape nakon konvolucije ostanu iste.

$$M_h^l = \frac{M_h^{l-1} - K + 2P}{S} + 1 \quad M_w^l = \frac{M_w^{l-1} - K + 2P}{S} + 1 \tag{2.7}$$

Opisani postupak prikazuje vezu između mape određenog sloja i jedne mape njemu prethodnog sloja. U potpuno povezanim konvolucijskim neuronskim mrežama u izračunu svake mape značajki sudjeluju sve mape prethodnog sloja. Neka je broj mapa u prethodnom sloju sloja kojeg promatramo jednak D. Za izračun mape značajki promatranog sloja potrebno je obaviti D konvolucija mape i njoj pridružene jezgre dimenzija $K \times K$. Tako dobivenih D mapa zajedno se sumira u jednu mapu. Svakoj lokaciji mape dodaje se prag, nakon čega se primjenjuje aktivacijska funkcija. Taj postupak formalnije je definiran izrazom (2.8).

$$x_{k'}^l(m,n) = f\left[\left(\sum_{k \in M^{l-1}} \sum_{\substack{0<i<=K^k-1 \\ 0<j<=K^k-1}} x_k^{l-1}(m+i, n+j) w_k^l(i,j)\right) + b_k^l\right] \tag{2.8}$$

$x_{k'}^l$ je element mape k' sloja l na poziciji (m,n). Unutarnja suma u izrazu predstavlja rezultat konvolucije *k-te* mape sloja $l-1$ s jezgrom w_k na poziciji (m,n) izlazne mape. Vanjska suma ide po svim mapama sloja $l-1$. b_k^l predstavlja jedan zajednički prag, a f je aktivacijska funkcija.

S obzirom da je za svaku izlaznu mapu potrebno imati jezgara koliko je mapa prethodnog sloja, dimenzije jezgre jedne mape konvolucijskog sloja često se označavaju kao tenzor $K \times K \times D$, gdje je K širina i visina kvadratne jezgre a D broj mapa prethodnog sloja. Broj parametara jednog konvolucijskog sloja koji ima F mapa, koristi jezgru dimenzija $K \times K$, a njemu prethodni sloj ima D mapa bit će $F(K \times K \times D + 1)$

Dilatirane konvolucije ili konvolucije s rupama (engl.*atrous convolution*) su poopćenje operatora konvolucije dano formulom (2.9) gdje se klasična konvolucija dobiva za $r = 1$.

$$(x *_d w)(m, n) = \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} x(m + r \cdot i, n + r \cdot j) w(i, j) \tag{2.9}$$

Isti izlaz dobio bi se klasičnom konvolucijom ulaza i jezgre nadopunjene sa $r - 1$ redaka nula između svakog retka i $r - 1$ stupaca nula između svakog stupca u jezgra, zbog čega se naziva konvolucijom s rupama. U nastavku je prikazana tako proširena jezgra dimenzija 3×3 za $r = 2$.

$$\begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \\ w_{20} & w_{21} & w_{22} \end{bmatrix} \rightarrow \begin{bmatrix} w_{00} & 0 & w_{01} & 0 & w_{02} \\ 0 & 0 & 0 & 0 & 0 \\ w_{10} & 0 & w_{11} & 0 & w_{12} \\ 0 & 0 & 0 & 0 & 0 \\ w_{20} & 0 & w_{21} & 0 & w_{22} \end{bmatrix}$$

Autori članka [3] pokazuju kako se primjenom eksponencionalno rastuće stope dilatacije r u uzastopnim slojevima koji koriste istu dimenziju jezgre, ostvaruje eksponencijalni rast receptivnog polja. Pri tome se prostorne dimenzije mapa značajki ne smanjuju kao što je to slučaj kod sažimanja. Neka su $F_0, F_1, ..., F_{n-1}$ n uzastopnih slojeva, u kojima se koriste jezgre $k_0, k_1, ..., k_{n-2}$, dimenzija 3×3. Neka se u tim slojevima vrše konvolucije sa eksponencijalno rastućom dilatacijom:

$$F_{i+1} = F_i *_{2^i} k_i, \quad i = 0, 1, ..., i - 2 \tag{2.10}$$

Receptivno polje elementa p u sloju F_{i+1} je broj elemenata iz F_0 koji utječu na $F_{i+1}(p)$. Receptivno polje svakog elementa iz F_{i+1} iznosi:

$$(2^{i+2} - 1) \times (2^{i+2} - 1) \tag{2.11}$$

Sloj sažimanja Sloj sažimanja smanjuje rezoluciju ulaznih mapa na način da prozor veličine $S_w \times S_h$ ulazne mape predstavlja jednom vrijednošću iz tog bloka. Najčešće se uzima

maksimalna ili srednja vrijednost u bloku. Tada govorimo o sažimanju maksimumom odnosno sažimanju srednjim vrijednošću. Cijela izlazna mapa dobiva se pomicanjem tog prozora kroz cijelu sliku slično kao kod konvolucije. Najćešće se sažimanje obavlja kvadratnim prozorom dimenzija $S \times S$ s korakom S čime se obe prostorne dimenzije ulazne mape smanjuju za taj faktor. Ukoliko se uzme korak manji od veličine bloka dobiva se preklapajuće sažimanje. Slojevi sažimanja efektivno povećavaju receptivno polje idućih slojeva i zbog smanjenja prostornih dimenzija ulaza štede memoriju za pohranu skrivenih mapa značajki. Slojevi sažimanja ne smanjuju broj mapa sloja i ne uče nikakve težine.

Invarijantnost na pomak Motivacija iza slojeva sažimanja osim smanjenja dimenzija mapa kroz mrežu je povećanje invarijantnosti na pomak. Invarijantnost na pomak ili translaciju je svojstvo otpornosti značajke na male varijacije položaja unutar ulazne mape. To je svojstvo mreže poželjno kada pretpostavljamo da je za problem važnije detektirati prisutnost određene značajke nego njenu točnu lokaciju.

Dijeljenje težina U konvolucijskim slojevima mreže, jedna konvolucijska jezgra predstavlja vezu između jedne mape prethodnog sloja i jedne izlazne mape. Izlazna mapa dobiva se konvolucijom ulazne mape i jezgre. Svi neuroni te izlazne mape posljedično dijele istih $K \times K$ težina i isti prag. Postupkom učenja se jezgre specijaliziraju za određene funkcije. Dijeljenjem težina postiže se mogućnost prepoznavanja naučene značajke u bilo kojem dijelu slike.

2.3. Optimizacija dubokih modela

Dubokim modelima nastojimo aproksimirati parametarsku funkciju koja mapira ulaz na željeni izlaz. Učenjem dubokog modela tražimo podskup težina i pragova za koji je odstupanje izlaza mreže od stvarnog izlaza minimalno. Funkcija gubitka odražava nezadovoljstvo tim odstupanjem. Dobre parametre modela naći ćemo pronalaskom minimuma funkcije gubitka. Kako rješenje za optimalne parametre mreže nije moguće pronaći u zatvorenoj formi, poseže se za gradijentnim metodama. Ideja gradijentnog spusta je izračunati gradijent funkcije u nekoj točki te se pomaknuti za mali pomak u njegovom negativnom smjeru, te ponavljati postupak dok gradijent ne postane jednak nuli kada je dosegnut minimum. Učenje dubokih modela zasniva se na tom principu, zbog čega funkcija gubitka kao i aktivacijske funkcije korištene u modelu moraju biti derivabilne. Tako primjerice kod zadatka klasifikacije kao funkciju gubitka ne možemo odabrati pogrešku klasifikacije, jer ne zadovoljava taj uvjet. Stoga se za taj zadatak kao funkcija gubtika uzima negativna log-izglednost točnog razreda primjera. Popularni algoritam učenja parametara dubokih modela naziva se algoritam unazadne propagacije pogreške.

2.3.1. Algoritam unazadne propagacije pogreške

je popularan, računski nezahtjevan algoritam optimizacije parametara neuronskih mreža općenito, pa tako i dubokih modela. Naziva se unazadnim jer se greška propagira unatrag, od izlaznog, preko skrivenih slojeva do ulaznog sloja mreže. Ideja koja se koristi kod učenja neuronskih mreža je sljedeća. Na početku se parametri mreže inicijaliziraju na slučajno odabrane vrijednosti. Kroz mrežu se zatim propuštaju jedan primjer, te izračuna gradijent odabrane funkcije gubitka po svim parametrima mreže. Gradijent funkcije gubitka se propagira unazad na način da se prvo izračuna gradijent gubitka izlaznog sloja. Zatim se za prethodne slojeve određuje njihov utjecaj na slojeve ispred njega koji o njemu ovise, te se na temelju toga određuju njihovi gradijenti. Na kraju se određuju gradijenti gubitka po svim težinama koje te slojeve povezuju te se njihove vrijednosti ažuriraju. Neka je promatrana neuronska mreža potpuno povezana unaprijedna mreža. Promatrani sloj neka je izlazni sloj mreže,označen sa j a njemu prethodni sloj sa i. Funkcija gubitka koja se koristi neka je srednja kvadratna pogreška, gdje je t_j očekivana vrijednost izlaza a y_j je stvarni izlaz j-tog neurona.

$$L = \frac{1}{2}\sum_j (t_j - y_j)^2 \tag{2.12}$$

Gradijent funkcije gubitka po ulazu u posljednji sloj j se računa sljedećom formulom.

$$\frac{\partial L}{\partial z_j} = \frac{\partial L}{y_j}\frac{\partial y_j}{\partial z_j} \tag{2.13}$$

Prvi dio izraza (2.13) predstavlja parcijalnu derivaciju funkcije gubitka po izlazu neurona, a drugi dio parcijalnu derivaciju izlaza po ulazu neurona. Parcijalna derivacija gubitka po izlazu nerona tada je:

$$\frac{\partial L}{\partial y_j} = \frac{1}{2}\frac{\partial}{\partial y_j}(t_j - y_j)^2 = -(t_j - y_j) \tag{2.14}$$

Desni dio izraza (2.13) ($\frac{\partial y_j}{\partial z_j}$) ovisi o konkretnoj aktivacijskoj funkciji koja se koristi. U slučaju logističke sigmoidalne funckije, kao što je opisano algoritmom 3 imamo:

$$\frac{\partial y_j}{\partial z_j} = \frac{\partial}{\partial z_j}(1 + e^{-z_j})^{-1} = \frac{1}{1 + e^{-z_j}}\frac{e^{-z_j}}{1 + e^{-z_j}} = y_j\frac{e^{-z_j}}{1 + e^{-z_j}} = y_j(1 - y_j) \tag{2.15}$$

Nakon što se izračuna gradijent gubitka trenutnog sloja prema (2.13), on se propagira na prethodni sloj sumiranjem utjecaja neurona i na sve neurone trenutnog sloja j.

$$\frac{\partial L}{\partial y_i} = \sum_j \frac{\partial L}{\partial z_j}\frac{\partial z_j}{\partial y_j} = \sum_j w_{ij}\frac{\partial L}{\partial z_j} \tag{2.16}$$

Na kraju se određuju parcijalne derivacije gubitka po težinama:

$$\frac{\partial L}{\partial w_{ij}} = \frac{\partial L}{\partial z_j}\frac{\partial z_j}{\partial w_{ij}} = \frac{\partial L}{\partial z_j}y_j \tag{2.17}$$

Nakon toga se ažuriraju vrijednosti težina prema:

$$w_{ij} \leftarrow w_{ij} - \eta\frac{\partial L}{\partial w_{ij}} \tag{2.18}$$

Algoritam 1 Algoritam backpropagation

procedure STOHASTIČKI GRADIJENTNI SPUST (D, η)
inicijalziriaj sve težine na male slučajne vrijednosti
    while nije ispunjen uvjet zaustavljanja **do**
        for $(\vec{x}, \vec{t}) \in D$ **do**
            //proslijedi ulazne vrijednosti kroz mrežu
            izračunaj izlaz y_u za svaku jedinicu u za ulaz x
            //prosljedi gradijent gubitka unatrag kroz mrežu
            for izlazni čvor c **do**
                izračunaj gradijent gubitka δ_c:
                $\delta_c = \frac{\partial L}{\partial z_c} \leftarrow y_c(1 - y_c)(t_c - y_c)$
            for svaku skrivenu jedinicu i **do**
                //izračunaj gradijent gubitka δ_i:
                $\delta_i = \frac{\partial L}{\partial z_i} =\leftarrow y_i(1 - y_i) \sum_j w_{ij}\delta_j$
            //ugodi svaki težinski faktor w_{ij}
            $w_{ij} \leftarrow w_{ij} - \eta\delta_j y_j$

Pseudokod algoritma dan je odsječkom 1 pri čemu je korištena sigmoidalna logistička funkcija. korištene sljedeće oznake: y_i je izlaz neurona i, y_j izlaz neurona j, w_{ij} je odgovarajuća težina, a δ_n pogreška izlaza jedinice n. Korištena aktivacijska funkcija je sigmoidalna logistička funkcija. Algoritam prima skup za učenje D i stopu učenja η. Mreži se predočavaju primjeri za učenje u obliku para $(\vec{x}, \vec{t})$ gdje je $\vec{x}$ vektor ulaznih a $\vec{t}$ vektor ciljnih izlaznih vrijednosti. Algoritam nakon inicijalizacije težina ponavlja predstavljanje svih primjera mreži dok nije ispunjen uvjet zaustavljanja. Uvjet zaustavljanja može biti smanjivanje pogreške ispod određene vrijednosti ili broj iteracija algoritma. Konvolucijske neuronske mreže za razliku od klasičnih imaju svojstvo dijeljenja težina u konvolucijskim slojevima gdje konvolucijska jezgra utječe na sve neurone u izlaznoj mapi. Zbog toga je prilikom provođenja unazadne propagacije pogreške potrebno izračunati korekcije težina po svim neuronima te sumirati njihov utjecaj.

2.3.2. Pojedinačno, grupno i učenje s mini-grupama

Ukoliko algoritam radi izračun gradijenta funkcije gubitka i korekcije težina za svaki primjer, jedan po jedan, naziva se pojedinačni ili stohastički. Prethodno dani pseudokod algoritma unazadne propagacije pogreške 1 koristi pojedinačno učenje. Optimizacijski postupci koji pak koriste cijeli skup primjera za izračun gradijenta nazivaju se grupni. Kako je cilj optimizacijskog algoritma minimizirati empirijsku pogrešku na cijelom skupu, kod grupnog načina

procjena gradijenta je točnija u odnosu na pojedinačni jer u obzir uzima sve primjere. No, u slučaju velikog broja primjera grupni način može biti preskup. Zato se koristi metoda koja se nalazi negdje između dvije krajnosti, a to je procjena gradijenta na temelju m uzoraka iz skupa. Taj način naziva se učenje s mini-grupama. Što je veća veličina mini-grupe m to je gradijent precizniji. Ipak, članak [4] pokazuje da optimizacija dubokih modela koja prilikom učenja koristi jako velike mini-skupine konvergira u oštre minimume (engl. *sharp minimizers*), a optimizacija koja koristi manju mini-skupinu vodi u ravne minimume (engl. *flat minimizers*). Unutar malog susjedstva oko točke oštrog minimuma vrijednost funkcije gubitka puno više varira nego oko ravnog minimuma. Eksperimenti iz članka [4] pokazuju da iako je pogreška na skupu za učenje u oba slučaja sumjerljiva, pogreška na skupu za ispitivanje manja je kod ravnih nego oštrih minimuma. Drugim riječima duboki modeli optimirani s manjom mini-skupinom bolje generaliziraju. Treba naglasiti da su u članku [4] kao jako veliku mini-skupinu uzimali 10% cijelog skupa za učenje (6000 primjera ako se radi o skupu CIFAR-10), dok je malu mini-skupinu činilo 256 primjera za učenje. Na temelju tog razmatranja može se zaključiti da će grupne inačice optimizacijskih algoritama, koje za procjenu gradijenta funkcije gubitka koriste sve podatke voditi do modela koji slabije generaliziraju u odnosu na one dobivene učenjem s mini-skupinama.

Kako bi odredili što nepristraniju ocijenu gradijenta, uzorci koji ulaze u mini-grupu bi trebali biti odabrani slučajnim uzorkovanjem. Osim toga, dvije uzastopne procijene gradijenta bi također trebale biti nezavisne. Često se to rješava na način da se prije svake epohe napravi slučajna permutacija skupa za učenje,i tim se redosljedom rade mini grupe. Osim takvog načina grupa se svaki puta može uzimati potpuno slučajno iz skupa za učenje.

Jedna iteracija stohastčkog gradijentnog spusta s mini grupama radi se na sljedeći način.

1. slučajno se iz $\mathcal{D}$ odabere mini-skupina veličine m:

$$\{(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}), ..., (\boldsymbol{x}^{(m)}, \boldsymbol{y}^{(m)})\}$$

2. izračuna se procjena gradijenta po parametrima $\boldsymbol{\theta}$

$$\tilde{\boldsymbol{g}} \leftarrow \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_i L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}), \boldsymbol{y}^{(i)}) \tag{2.19}$$

3. radi se korekcija svih parametara prema

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \tilde{\boldsymbol{g}} \tag{2.20}$$

2.3.3. Učenje s momentom

Učenje s momentum poboljšana je varijanta standardnog algortima stohastičkog gradijentnog spusta koja se primjenjuje za ubrzavanje učenja. Metoda je posebno pogodna u slučaju

kad su gradijenti mali ali konzistentni ili je u procjeni gradijenta prisutan šum. Algoritam uvodi novu varijablu momenta v. Ta varijabla čuva eksponencijalno umanjujući prosjek iznosa negativnog gradijenta. U odnosu na prethodno opisani stohastički gradijentni spust razlikuje se samo korak korekcije parametara koji se radi prema.

$$\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} - \eta \boldsymbol{g} \tag{2.21}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{v} \tag{2.22}$$

Vrijednost parametra α određuje utjecaj prethodno izračunatih gradijenata na smjer ažuriranja. Što je on veći u odnosu na stopu učenja to je veći utjecaj prethodnih gradijenata.

2.3.4. Optimizacijski algoritam ADAM

Optimizacijski algoritam Adam [5] od (engl. *Adaptive Moments*) također koristi ideju momenta - on je implicitno ugrađen u praćenje kretanja gradijenta. Adam prati kretanje momenta prvog i drugog reda, uz eksponencionalno zaboravljanje. Pseudokod algoritma je dan odsječkom 2, gdje je $\odot$ oznaka za množenje po elementima vektora. $\rho_1, \rho_2 \in [0, 1)$ određuju utjecaj prethodno izračunatih gradijenata odnosno kvadrata gradijenta.

Algoritam 2 ADAM

procedure ADAM (D, η, $\rho_1 = 0.9$,$\rho_2 = 0.99$,$\delta = 10^{-8}$)
  inicijaliziraj parametare $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta_0}$
  inicijaliziraj varijable prvog i drugog momenta i vremenski korak:
  $\boldsymbol{s} = \boldsymbol{0}, \boldsymbol{v} = \boldsymbol{0}, t = 0$
  while nije ispunjen uvjet zaustavljanja **do**
    odaberi mini-skupinu $\{(\boldsymbol{x}^{(1)}, \boldsymbol{y}^{(1)}), ..., (\boldsymbol{x}^{(m)}, \boldsymbol{y}^{(m)})\} \in \mathcal{D}$
    izračunaj procjenu gradijenta $\tilde{\boldsymbol{g}} \leftarrow \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_i L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}), \boldsymbol{y}^{(i)})$
    ažuriraj vremenski korak $t \leftarrow t + 1$
    ažuriraj prvi moment $\boldsymbol{s} \leftarrow \frac{\rho_1}{1-\rho_1^t} \boldsymbol{s} + \frac{1-\rho_1}{1-\rho_1^t} \tilde{\boldsymbol{g}}$
    ažuriraj drugi moment $\boldsymbol{r} \leftarrow \frac{\rho_2}{1-\rho_2^t} \boldsymbol{r} + \frac{1-\rho_2}{1-\rho_2^t} \tilde{\boldsymbol{g}} \odot \tilde{\boldsymbol{g}}$
    izračunaj korekciju $\Delta\boldsymbol{\theta} = -\eta \frac{\boldsymbol{s}}{\sqrt{\boldsymbol{r}} + \delta}$
    primijeni korekciju $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \Delta\boldsymbol{\theta}$

2.3.5. Regularizacija

Kao što je već spomenuto, izazov strojnog učenja je pronaći model koji ima malu varijancu i ne preveliku pristranost jer se od takvog modela očekuje dobra generalizacija. Uzme li

se za određeni problem model dovoljnog kapaciteta, nema opasnosti od podnaučenosti, ali prenaučenost je i dalje problematična. Prenaučenost se može rješiti postupkom odabira modela metodom unakrsne provjere. Alternativa tomu su tehnike regularizacije koje omogućuju povećanje pristranosti modela uz smanjenje varijance.

Regularizacija normom vektora parametara , složenost modela eksplicitno ugrađuje u funkciju gubitka (2.23).

$$\widetilde{J}(\boldsymbol{\theta}; \mathcal{D}) = J(\boldsymbol{\theta}; \mathcal{D}) + \alpha\Omega(\boldsymbol{\theta}) \tag{2.23}$$

gdje je Ω regularizacijski član a $\alpha \in [0, \infty)$ određuje relativni doprinos čiste funkcije gubitka i regularizatora. Regularizacijski član općenito glasi

$$\Omega(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{n} |w_i|^q \tag{2.24}$$

Za $q = 1$ dobiva se L1 regularizacija a za $q = 2$ L2 regularizacija. L1 regularizacija vodi na rijetke modele, gdje su neke vrijednosti parametara 0, pa L1 regularizacija efektivno obavlja selekciju varijabli. L2 regularizacija ne rezultira rijetkim modelima ali je analitički pogodnija. Pragovi ne sudjeluju u regularizacijskom članu jer njihove vrijednosti ne želimo pritezati prema ishodištu. Ovakvom regularizacijom uvodi se pristranost preferencijom jer se preferiraju modeli čije su težine manje.

Regularizacija ranim zaustavljanjem koristi se kod iterativnih optimizacijskih metoda, kao što je učenje neuronskih mreža. Ideja je odvojiti skup za validaciju, te povremeno kroz učenje, na njemu evaluirati model. U trenutku kad pogreška na skupu za validaciju prestane padati, učenje se zaustavlja. U okviru rada u svakom eksperimentu korištena je L2 i regularizacija ranim zaustavljanjem.

Osim navedene dvije metode, regularizacijski efekt ima i proširivanje skupa za učenje. Osim dodavanjem novih primjera u skup, koriste se i tehnike umjetnog proširivanja skupa. Kad su primjeri slike, kao u ovom radu to mogu biti transformacije u prostoru boja, zrcaljenje slike po horizontali, izrezivanje dijela slike i slično. Osim toga dijeljenje parametara između više različitih zadataka može imati regularizacijski efekt na pojedine zadatke.

2.3.6. Normalizacija nad grupom

Normalizacija nad grupom (engl. *batch normalization*) je metoda za ubrzavanje učenja predstavljena u članku [6]. Učenje mreže, podrazumjeva postepene promjene vrijednosti težina mreže kako bi se smanjila ukupna pogreška. Te promjene parametara sloja povlače promjenu

distribucija ulaznih mapa značajki idućeg sloja. Pojava da se distribucija ulaza pojedinog sloja mjenja kroz postupak učenja mreže naziva se unutarnji kovarijacijski pomak. Pojava kovarijacijskog pomaka zahtijeva niže stope učenja i pažljivije odabrane metode inicijalizacije parametara, što usporava proces učenja. Problem se rješava normalizacijom ulaznih mapa značajki svakog sloja između različitih primjera unutar grupe. Normalizira se dakle jedna značajka među svim primjerima unutar grupe, za svako sloj zasebno. Svaka mini-grupa estimira srednju vrijednost i varijancu svake aktivacije zasebno.

Razmotrimo mini-podskup B koji sadrži m ulaznih mapa. S obzirom da se normalizacija provodi za svaku aktivaciju zasebno, izdvojimo jednu od njih $x^{(k)}$ i ispustimo k radi preglednosti. Postoji m vrijednosti u tom mini-podskupu $B = x_{1...m}$. Njihove normalizirane vrijednosti neka su $\hat{x}_{1...m}$ i pripadne linearne transformacije $y_{1...m}$. Tada sljedeća transformacija predstavlja normalizaciju mapa značajki:

$$BN_{\gamma,\beta} : x_{1..m} \rightarrow y_{1...m} \tag{2.25}$$

gdje su parametri γ i β parametri koje mreža treba naučiti. Treba primjetiti da $BN_{\gamma,\beta}(x)$ transformacija ovisi o svim primjerima unutar mini-skupa a ne samo o jednom primjeru za učenje.

Algoritam 3 Normalizacija mapa značajki

Input: Vrijednosti mape x unutar podskupa za učenje $B = \{x_{1...m}\}$;

Output: Parametri γ, β koji se uče;

$\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i)$

$\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$

$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$

$y_i \leftarrow \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$

Linearna transformacija koja se primjenjuje nakon centriranja, radi se kako bi se mogla rekonstruirati početna distribucija značajke. Naime, čista normalizacija smanjuje ekspresivnu moć mreže. Ako je određenja značajka prirodno imala određenu srednju vrijednost i odstupanje, normalizacija ih uklanja. Uvođenjem parametara skaliranja γ i pomaka β to se nadoknađuje. Primjena normalizacije nad grupom u fazi učenja tehnički mijenja unaprijedni prolaz. Unaprijedni prolaz ne može se organizarati kao m slijednih prolaza kroz cijelu mrežu. Potrebno je prvo napraviti m prolaza do ulaza u prvi sloj na kojemu se primjenjuje normalizacija. Tada se podatci normaliziraju, transformiraju i propuštaju dalje kroz mrežu do idućeg sloja koji se normalizira gdje se postupak ponavlja. Algoritam uči sve parametre kao i bez normalizacije $\boldsymbol{\theta}$ uz dodatak γ i β za svaku značajku koja se normalizira.

Primjena ovakve normalizacije omogućuje brže učenje mreže, mogućnost korištenja većih stopa učenja te povećanje otpornosti na lošu incijalizaciju, a pokazuje i efekte regularizacije.

3. Ispitni skupovi

3.1. PASCAL VOC

Pascal Visual Object Classes Challenge je godišnje natjecanje koje se održava od 2005. godine. Dva glavna zadatka natjecanja su klasifikacija slika i detekcija objekata. Pascal VOC skup podataka sastoji se od označenih slika sakupljenih sa flickr web stranice za dijeljenje slika. Svake se godine u sklopu natjecanja objavljuje novi skup slika. Svakoj slici u skupu pridružena je jedna datoteka oznaka koja sadrži položaj i oznaku razreda za svaki objekt od interesa unutar slike. Položaj je opisan oknom koji uokviruje objekt (engl. bounding box). Označeni objekti pripadaju jednom od sljedećih dvadeset razreda: aeroplane, bird, bicycle, boat, bottle, bus, car, cat, chair, cow, dining table, dog, horse, motorbike, person, potted plant, sheep, sofa, train, tv/monitor. Skup se sastoji od dva podskupa za učenje i testiranje sustava. VOC2007 [7], skup iz 2007. godine sadrži 5011 označenih slika za učenje i validaciju te 4952 slika za testiranje. Oznake skupa za testiranje javno su dostupne pa se taj skup koristi za usporedbu performansi različitih modela. VOC2012 [8]sadrži 11540 označenih slika za učenje i validaciju, te skup za testiranje čije oznake nisu javno dostupne. U okviru rada za zadatak detekcije objekata korištena je unija skupova VOC2007 i VOC2012 za učenje i validaciju kao skup za učenje, a skup VOC2007 je korišten kao skup za validaciju.

3.2. Cityscapes

Cityscapes[9] je skup slika visoke rezolucije kojeg čini raznolik skup slika iz video sekvenci snimljenih na ulicama 50 njemačkih gradova tokom različitih godišnjih doba u dobrim do srednje dobrim vremenskim uvjetima. Ovaj skup s oznakama na razini piskela, namjenjen je zadatcima semantičke segmentacije i označavanja instanci s ciljem boljeg razumijevanja urbanih scena. Sadrži 5000 fino označenih slika. Primjer fino označene slike prikazan je na slici 3.1. Različite boje predstavljaju različite kategorije objekata od interesa. Svaka instanca razreda koji ih imaju označena je zasebno. Osim toga skup sadrži i 20 000 ugrubo označenih slika koje mogu koristiti istraživanjima čiji je cilj iskoristiti veće količine označenih podataka, primjerice za treniranje duboke neuronske mreže. Označeni objekti dolaze

Slika 3.1: Fino označena slika u skupu Cityscapes

Različitim bojama označene su različite kategorije. Kod kategorija unutar kojih razlikujemo različite primjerke kao što su čovjek, bicikl i slično svaki primjerak je označen zasebno. Skup Cityscapes sadrži 5000 ovako fino označenih slika.

iz 30 različitih kategorija razvrstanih u 8 većih grupa. Tako se primjerice u grupi *vehicle* nalaze kategorije *car, truck, bus, bicycle* itd. a u grupi *nature* se nalaze kategorije *vegetation* i *terrain*. Uz same oznake dostupne su i dodatne informacije o prethodnoj i idućoj slici iz video sekvence, kako bi se omogućili zadatci poput analize optičkog toka i praćenje objekata. Osim toga za svaku sliku dostupna je i njoj odgovarajuća desna slika kako bi se mogla koristiti stereo informacija. Dostupne su i GPS koordinate i podatci o kretanju vozila. U okviru rada korištene su fino označene slike. 5000 slika razvrstano je prema gradovima u kojima su snimane. Skup za učenje sadrži slike 18 gradova, njih ukupno 2975, skup za validaciju 500 slika iz ukupno tri grada. Ostatak slika snimljeno je u 6 gradova i pripada skupu za testiranje.

3.3. KITTI

KITTI Vision Benchmark je ispitni skup napravljen u suradnji instituta Karlsruhe Institute of Technology i Toyota Technological Institute at Chicago. Snimljen je s opremom pričvršćenom na krov autonomnog vozila. Oprema se sastojala od stereo sustava od dvije sive i dvije kamere u boji visoke rezolucije te Velodyne laser skenera i GPS sustava lokalizacije. Rezultat je veliki skup slika prometnih scena gradske i izvangradske vožnje namjenjen različitim zadatcima: stereo, optički tok, vizualna odometrija, 3D detekcija objekata i praćenje. Ispitni skup **KITTI object** [10] namjenjen je sustavima za detekciju i estimaciju orijentacije objekata. Sastoji od oko 15,000 slika od kojih je pola namjenjeno za učenje a pola za testiranje sustava. Oznake dostupne za skup za treniranje dane su u obliku koordinata okvira oko objekata. Objekti na slikama mogu pripadati jednom od sljedećih sedam razreda: Car, Van,

Truck, Pedestrian, Person sitting, Cyclist, Tram, Misc or DontCare. U okviru rada za zadatak detekcije objekata korišteno je ukupno 7481 slika za učenje podijeljeno u skup za učenje od 6500 slika i skup za validaciju od 981 slika, a detekcija je provođena nad objektima koji pripadaju razredima Car, Cyclist i Pedestrian.

4. Detekcija objekata

Detekcija objekata jedan je od zadataka računalnog vida. Cilj detekcije je odrediti prisutnost određenog objekta i njegovu točnu lokaciju. Ti objekti općenito pripadaju različitim kategorijama, te je cilj u idealnom slučaju detektirati svaki pojedinačni primjerak svake kategorije koja je u slici prisutna. Ti primjerci u realnim scenama mogu biti različitih skala, prekriveni drugim objektima ili pak odsječeni granicom slike, kada detekcija postaje izazov.

U ispitnim skupovima koji se koriste za učenje detekcije objekata svakoj slici pridružen je skup parova pozicija-kategorija. Pozicija je najčešće određena kao okno odnosno pravokutnik koji uokviruje određeni primjerak. Taj pravokutnik zadan je sa vrijednosti koje predstavljaju koordinate gornjeg lijevog i donjeg desnog kuta pravokutnika, ili koordinate centra, visinu i širinu pravokutnika. Osim takvog zapisa pozicije objekta, oznake mogu biti dostupne i u obliku maski na razini piksela za svaki primjerak objekta. Zapis u obliku maski je precizniji, te je iz njega moguće dobiti zapis u obliku pravokutnika.

Pod pretpostavkom prvog zapisa, modeli za detekciju predviđaju isto - za ulaznu sliku određuju skup pravokutnika i njihovih oznaka, gdje broj objekata nije poznat. Detekcija se smatra pozitivnom ukoliko se pravokutnik detekcije i oznake poklapaju i ako je predviđena točna kategorija objekta. Kao mjera poklapanja uzima se omjer presjeka i unije dvaju pravokutnika (engl. *intersection over union*) ili drugim nazivom *Jaccardov* indeks. Ako je Jaccardov indeks između detekcije i oznake veći od nekog praga, imamo poklapanje i detekcija se smatra točno pozitivnom (engl. *true positive*) za razred kojemu objekt pripada. Prag koji se najčešće uzima je 0.5 a nekad i 0.7 za veće objekte.

Osim točno pozitivnih detekcija, drugi scenariji su da detektor nije prepoznao neki od označenih objekata, pa je to pogrešno negativna (engl. *false negative*) detekcija razreda kojemu pripada oznaka. Suprotno tome, u slučaju da detektor uoči postojanje objekta za kojeg ne postoji oznaka, detekcija se smatra lažno pozitivnom (engl. *false positive*) za razred kojeg je sustav pogrešno previdio. U nastavku su prikazane formule preciznosti (engl. *precision*) i odziva (engl. *recall*) detekcije definirane preko broja točno pozitivnih (TP), broja pogrešno pozitivnih (FP) i broja pogrešno negativnih (FN) detekcija određenog razreda k.

$$R_k = \frac{TP_k}{TP_k + FN_k} \quad R_{mean} = \frac{1}{K} \sum_{i=0}^{K-1} R_i \tag{4.1}$$

$$P_k = \frac{TP_k}{TP_k + FP_k} \quad P_{mean} = \frac{1}{K}\sum_{i=0}^{K-1} P_i \tag{4.2}$$

Odziv je mjera koja govori koliki udio označenih objekata je detektiran, a preciznost koliki udio predviđenih detekcija je relevantan. Mjera koja povezuje te dvije naziva se srednja preciznost (engl. *average precision*). Srednja preciznost računa se na način da se uzme neki parametar koji kontrolira odziv detektora. Taj parametar najčešće je sigurnost detektora u prisutnost detekcije, odnosno vjerojatnost da pripada predviđenom razredu.

Uzme li se kao prag sigurnost detektora od 0.9, detektor će predvidjeti određeni broj detekcija. Uzme li se kao prag, manji broj, primjerice 0.5, detektor će ih predvidjeti više: sve one koje je previdio kad je prag bio 0.9, uz sve one za koje je siguran manje od 0.9 i više od 0.5. Smanjenje praga sigurnosti dakle povećava broj pozitiva detektora. Kako je mjera odziva udio detektiranih od označenih objekata, sa smanjenjem praga očekivan je porast odziva. No kako detektor sa smanjenjem praga, manje pažljivo bira pozitive, veliki dio tih pozitiva bit će lažno pozitivne detekcije, pa će se preciznost smanjiti.

Krivulja preciznost-odziv (engl. *precision recall curve*) prikazuje odnos preciznosti i odziva. Primjer takve krivulje prikazan je slikom 4.1. Na x osi je prikazan iznos odziva, a na y preciznsosti. Plave točke označavaju koliki postotak detekcija je odbačen u odnosu na broj detekcija kada je sigurnost minimalna.

Prosječna preciznost (engl. *average precision*) je mjera koja jednim brojem opisuje krivulju preciznost-odziv. To je preciznost uprosječena za sve vrijednosti odziva između 0 i 1, odnosno površina ispod krivulje.

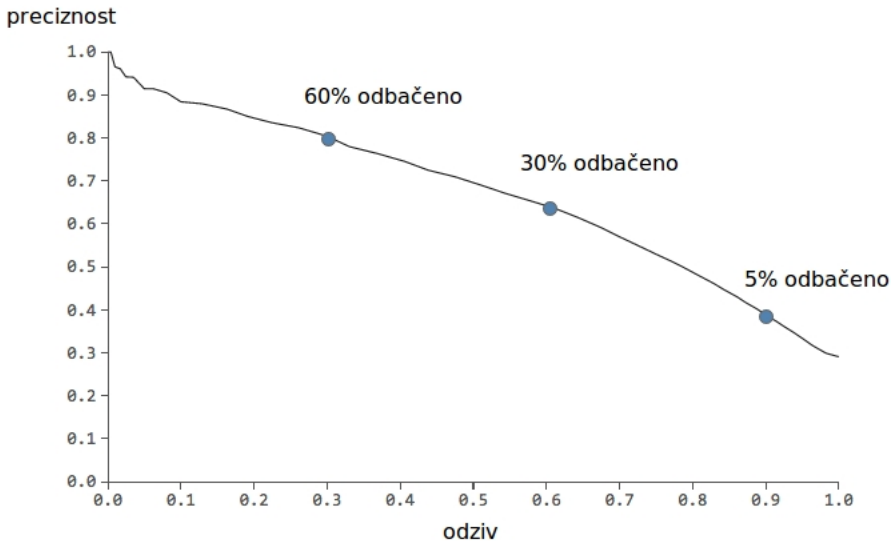
$$AP = \int_0^1 p(r)dr \tag{4.3}$$

Integral se aproksimira sumom na sljedeći način. Uzme se N jednako razmaknutih vrijednosti odziva između 0 i 1 te se izračuna preciznost za svaku od tih točaka, sumira i podijeli s brojem točaka 4.4.

$$AP = \frac{1}{N}\sum_{k=0}^{N-1} P[r = \frac{k}{N-1}] \tag{4.4}$$

Tzv. Interpolirana prosječna preciznost, koja se koristi kao mjera za evaluaciju detekcije u natjecanju Pascal VOC i u okviru ovog rada, malo je drugačija od prethodno definirane prosječne preciznosti 4.5. Umjesto da se uzima vrijednost preciznosti u promatranoj točki odziva, uzima se maksimalna među preciznostima za odziv veći od promatrane točke.

$$AP_{voc07} = \frac{1}{11}\sum_{k=0}^{10} \max(P[r \geq \frac{k}{10}]) \tag{4.5}$$



Slika 4.1: Primjer krivlje preciznost-odziv

Za dobivanje ovakve krivulje koristi se parametar koji kontrolira odziv sustava. Kod detekcije objekata to može biti pouzdanost klasifikacije pronađenog objekta u razred u koji je svrstan. Plave točke na krivulji pokazuju koliko je detekcija odbačeno u odnosu na slučaj kada je pouzdanost klasifikacije minimalna. Kada je odbačen veći dio detekcija odziv je manji, ali je veća preciznost. Naspram tome, kada se pozitivima smatraju sve detekcije bez obzira na pouzdanost, odziv će biti velik, ali preciznost manja. Mjera koja jednom brojčanom vrijednošću opisuje krivulju preciznost-odziv je površina ispod krivulje odnosno srednja preciznost.

Srednja prosječna preciznost (engl *mean average precision*) je prosječna preciznost usrednjena po svim razredima 4.6. U nastavku rada pod nazivom srednja prosječna preciznost mAP pretpostavljat će se interpolirana verzija.

$$mAP = \sum_{i=0}^{K-1} AP_i \tag{4.6}$$

4.1. Fast, Faster R-CNN

R-CNN što je kratica od engl. regions with convolutional neural network features, je sustav za detekciju objekata, inspiriran uspjehom konvolucijskih neuronskih mreža u klasifikaciji slika, predstavljen u okviru članka [11]. Ta metoda ima više faza. Prva faza je iz ulazne slike izdvojiti regije unutar kojih će se tražiti objekti. U originalnom radu to se radilo metodom selektivnog pretraživanja (engl. *selective search*) koja izdvaja regije neovisne o razredima objekata koje prikazuju. Temelji se na hijerathijskom grupiranju regija koje su slične po boji, teksturi, veličini i komplementarne po obliku. Neke od tako dobivenih regija zaista će uokvirivati objekt pa će biti označene kao pozitivi onoga razreda s čijom se oznakom poklapaju. Ostale regije koje nisu uspjele uhvatiti objekte smatraju se negativima. Nakon izdvajanja

potencijalnih regija i pridruživanja oznaka, za svaku regiju računa se vektor konvolucijskih značajki. Vektor se dobiva provlačenjem regije kroz duboku konvolucijsku mrežu predtreniranu na velikom skupu za klasifikaciju slika. Treća faza je klasifikacija tako dobivenih vektora u jednu od kategorija strojevima potpornih vektora. Uz navedeno model radi i regresiju okvira, gdje se uči pomak koordinata regije do okvira stvarne oznake, kako bi se što preciznije ukovirili objekti.

Iako je ovakav pristup dao dobre rezultate u odnosu na prethodne vodeće detektore bio je relativno spor. Prvo je potrebno izdvojiti regije, zatim svaka takva regija zahtjeva jedan prolaz kroz mrežu za izračun vektora značajki, pa se tek onda klasificira. Kako se istaknute regije mogu međusobno preklapati, dosta izračuna se bespotrebno ponavlja. Osim toga, tri modela se uče zasebno: konvolucijska neuronska mreža, linearni strojevi potpornih vektora te regresija okvira.

Fast R-CNN [12] je poboljšanje RCNN-a koje zaobilazi te probleme. Za svaku sliku radi se samo jedan prolaz kroz duboku neuronsku mrežu čime se dobiju značajke dijeljene između svih regija interesa. Tada se za svaku regiju iz zajedničke mape određuje dio značajki koji joj pripada, koji se zatim sažima maksimumom tako da rezultat bude unaprijed određenih dimenzija. Osim toga linearne SVM-ove zamjenjuje softmax slojem na izlazu mreže, te u paralelu dodaje sloj za regresiju okvira. Na ovaj način, sve potrebne izlaze daje ista mreža. No ipak, postupak i dalje zahtijeva prvotnu selekciju istaknutih regija, što je nakon opisanih promjena postalo usko grlo postupka. Unaprijedni prolaz ove metode za jednu sliku trajao je oko dvije sekunde (0.5 fps).

Faster R-CNN [13] izbacuje potrebu za algoritmom izdvajanja istaknutih zanačajki uvodeći mrežu za predlaganje regija (engl. *region proposal network*). Ta mreža predlaže regije koristeći tzv. referentne okvire (engl. *anchor boxes*), koji su određeni površinom i omjerom stranica. Faster R-CNN koristi devet različitih referentnih okvira. RPN centrira svaki od tih devet okvira u svaku od lokacija ulazne mape značajki. Svaku tako dobivenu regiju klasificira u jedan od dva razreda: objekt i ne-objekt, te radi regresiju pomaka referentnog okvira do stvarne oznake kako bi izlazni okvir bolje uokvirivao objekt. Daljnji postupak dobivanja detekcija isti je kao kod Fast R-CNN-a. Vrijeme tranjanja unaprijednog prolaza smanjilo se na 200ms za jednu sliku (5 fps). Tablica 4.1 prikazuje usporedbu performansi tri opisana modela, učena na skupu Pascal VOC-2007 za učenje i validaciju. Srednja prosječna preciznost mjerena je na skupu Pascal VOC 2007 za testiranje.

4.2. Single shot multibox detector

Metoda SSD predstavljena u [14] za detekciju objekata koristi jedinstvenu duboku neuronsku mrežu. Pristup pretpostavlja da se potencijalni objekti mogu nalaziti unutar naprijed odre-

VOC2007-test	R-CNN	Fast R-CNN	Faster R-CNN
mAP [%]	66.0	66.9	67.6

Tablica 4.1: Usporedba metoda na Pascal 2007 test skupu (podatci preuzeti iz članka [13])

đenih pravokutnika odnosno okvira različitih veličina i omjera stranica centriranih u svakoj lokaciji mape značajki. Mreža za svaki okvir određuje mjeru sigurnosti u prisutnost svake od mogućih kategorija te prilagođava poziciju okna tako da bolje uokviruje objekt. Kako bi prirodno nadvladala probleme koji proizlaze iz razlike u veličinama objekata različitih kategorija, mreža odluke donosi kombinirajući predikcije iz više mapa značajki različitih dimezija.

Dotadašnji state-of-the-art sustavi za detekciju objekata pristupali su problemu na način da prvo predvide okna na kojima se mogu nalaziti objekti, zatim iz tog okna izlučuju značajke pomoću kojih klasificiraju okvir u neki od razreda. Brzina detekcije takvih sustava se često mjeri kao broj sekundi po slici. Najbrži od njih Faster R-CNN obrađuje samo 7 slika u sekundi. Rad [14] predstavlja prvu metodu za detekciju objekata temeljenu na dubokoj mreži koja izostavlja fazu predlaganja okvira i izlučivanja značajki, bez gubitka točnosti. Rezultat je veliko poboljšanje u brzini detekcije. Iako autori rada nisu prvi koji su pokušali takvo što, niz poboljšanja koji su unjeli značajno je povećao točnost u odnosu na prethodne slične metode. Ta poboljšanja uključuju korištenje malih konvolucijskih jezgri za predviđanje kategorija i pomaka okvira od pretpostavljene pozicije, korištenje različitih jezgara za detekcije različitih skala u više slojeva mreže.

4.2.1. Model

SSD pristup temelji se na unaprijednoj konvolucijskoj mreži. Izlaz mreže je kolekcija fiksnog broja okvira i njima pridruženih vektora vjerojatnosti klasifikacije. Dimenzionalnost tog vektora jednaka je broju kategorija. Svaki njegov element predstavlja vjerojatnost da okvir zaista uokviruje objekt koji pripada promatranom razredu. Prvi slojevi mreže temelje se na standardnim arhitekturama za visokokvalitetnu klasifikaciju slika. Eksperimenti iz članka koristili su model zasnovan na VGG-16 arhitekturi [15]. Na slojeve odabrane bazne arhitekture, dodaju se novi slojevi, kako bi se dobila struktura sa sljedećim karakteristikama.

Mape različitih skala dimenzije mapa značajki postepeneno se smanjuju kako napredujemo kroz mrežu. S obzirom da su objekti u slikama različitih veličina na različitim skalama, uspješna detekcija ostvaruje se kombiniranjem značajki iz različitih slojeva merže. Pri tome se za značajke iz različitih slojeva koriste različiti konvolucijski modeli za detekciju.

Konvolucijski prediktori detekcija Svaki dodani sloj ili opcionalno postojeći sloj bazne mreže može sudjelovati u detekciji. Iznad svakog odabranog sloja dolaze dva izlazna konvolucijska sloja, jedan za regresiju pozicija okvira, jedan za predikciju kategorije okvira. U oba izlazna konvolucijska sloja koriste se male jezgre prostornih dimenzija 3×3.

Pretpostavljeni okviri Svakom izlaznom sloju mreže koji se koristi za detekciju, pridruženo je određeni broj referentnih okvira (engl. *anchor boxes*). Ti okviri, odnosno pravokutnici određeni su svojom površinom i omjerom stranica. Takvi okviri centriraju se u svakoj od pozicija mape značajki sloja kojemu pripadaju, slično kao tehnike klizućeg prozora, čime se dobiva skup pretpostavljenih okvira. Za svaki takav pretpostavljeni okvir mreža predviđa kategoriju i pomak od pretpostavljene pozicije. Točnije, za svaki referentni okvir od njih k na određenoj lokaciji, računa se c vjerojatnosti pripadnosti svakoj od kategorija i 4 pomaka relativna u odnosu na originalnu poziciju okvira. To znači da je potrebno koristiti $(c + 4)k$ konvolucijskih jezgara koje se primjenjuju u svakoj lokaciji mape, što rezultira s $(c+4)kmn$ izlaza za mapu značajki dimenzija $m \times n$.

Arhitektura modela za detekciju SSD300/VGG16 prikazana je na slici 4.2.

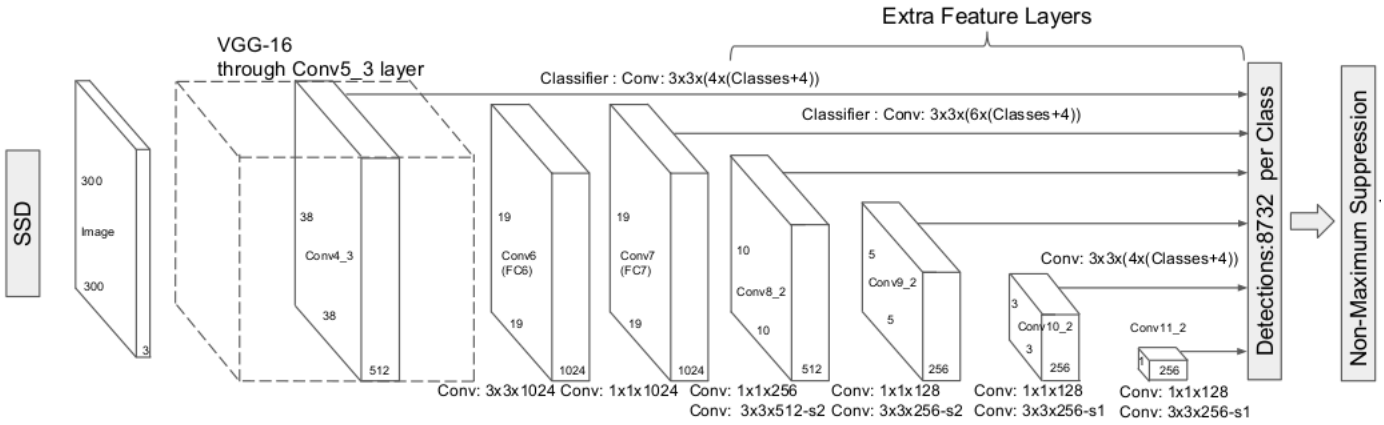
4.2.2. Postupak učenja

Ključna razlika između učenja SSD-a i tipičnog detektora koji koristi predložene regije je u tome što informacija o stvarnim oznakama ispitnog skupa mora biti pridružena odgovarajućim pretpostavljenim okvirima. Kad se to pridruživanje ustanovi, funkcija gubitka i unazadna propagacija pogreške primjenjuju se s kraja na kraj. Učenje također uključuje odabir tog skupa predpostavljenih okvira i skala za detekciju, te strategije proširivanja skupa za učenje i miniranja teških negativa.

Slika 4.2 prikazuje SSD 300×300 sa baznom arhitekturom VGG-16. Slojevi koji sudjeluju u detekciji su sloj označeni kao conv4_3, conv7, conv 8_2, conv9_2, conv10_2 i conv11_2. Svakom od tih slojeva pridruženo je više referentnih okvira (engl. *anchor boxes*) unutar kojih se traže objekti u tom sloju. Svi pretpostavljeni okviri sloja dobiju se primjenom referentnih okvira na sve lokacije mape značajki promatranog sloja. Tako će broj mogućih okvira u određenom sloju biti jednak broju referentnih okvira sloja $\times$ broj lokacija mape značajki sloja.

4.2.3. Odabir referentnih okvira

Svaki referentni okvir (engl. *anchor*) određen je svojom skalom i omjerom visine i širine. Skala referentnih okvira koja će se koristiti u datom sloju određuje se prema formuli (4.7),



Slika 4.2: Prikaz arhitekture VGG-16/SSD 300×300 za detekciju objekata (slika preuzeta iz članka [14])

Ulaz u model je slika u boji fiksnih dimenzija 300×300. Kvadar označen isprekidanom linijom označava slojeve koji pripadaju prvih pet konvolucijskih blokova mreže VGG-16. Slojevi koji sudjeluju u detekciji su sloj označeni kao conv4_3, conv7, conv 8_2, conv9_2, conv10_2 i conv11_2. Ti slojevi su na slici strelicama spojeni s izlazom mreže. Iznad svakog od tih slojeva dodaju se po dva konvolucijska sloja s veličinom jezgre 3×3 od kojih jedan uči regresiju pozicija okvira oko objekta, a drugi klasifikaciju okvira u jednu od kategorija. Završni skup detekcija je unija detekcija na svim slojevima, filtrirana korakom eliminacije nemaksimalnih elemenata.

gdje je m broj mapa koje se koriste u predikciji, a s_{min} iznosi 0.2 i s_{max} 0.9, uz pretpostavku da su slojevi poredani od najnižeg prema najvišem.

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m - 1}(k - 1), \qquad k \in [1, m] \tag{4.7}$$

To znači da će pretpostavljeni okviri najnižeg sloja obuhvatiti 0.2 površine ulazne slike a najviši 0.9, a slojevi između su jednoliko razmaknuti. Omjer visine i širine referentnog okvira a_r odabire se iz $\{1, 2, \frac{1}{2}, 3, \frac{1}{3}\}$. Relativna širina i visina svakog referentnog okvira u odnosu na veličinu ulazne slike sada je.

$$\begin{aligned} w_k^a &= s_k\sqrt{a_r} \\ h_k^a &= s_k\sqrt{a_r} \end{aligned} \tag{4.8}$$

Za omjer $a_r = 1$ dodaje se još jedan referentni okvir skale $s'_k = \sqrt{s_k s_{k+1}}$, čime se dobiva ukupno 6 različitih okvira po lokaciji mape značajki. Centar svih okvira pomiče se u $(\frac{i+0.5}{f_k}, \frac{j+0.5}{f_k})$ gdje je f_k dimenzija kvadratne mape značajki i $i, j \in [0, f_k)$.

Ova distribucija oblika prepostavljenih okvira može se odrediti tako da najbolje obuhvaća određeni skup. Tako su u primjeru sa slike 4.2 i za ispitni skup PASCAL VOC2007 korišteni su omjeri skale prikazani u tablici 4.7.

Kombinacijom predikcija mreže za sve pretpostavljene okvire na svim lokacijama mapa značajki koje se koriste u detekciji dobiva se raznolik skup detekcija koji pokriva objekte

sloj	s_k	a_r	dimenzije sloja	broj okvira
conv4_3	0.2	$\{1, 2, \frac{1}{2}\}$	38×38	5776
conv7	0.34	$\{1, 2, \frac{1}{2}, 3, \frac{1}{3}\}$	19×19	2166
conv8_2	0.48	$\{1, 2, \frac{1}{2}, 3, \frac{1}{3}\}$	10×10	600
conv9_2	0.62	$\{1, 2, \frac{1}{2}, 3, \frac{1}{3}\}$	5×5	150
conv10_2	0.76	$\{1, 2, \frac{1}{2}\}$	3×3	36
conv11_2	0.9	$\{1, 2, \frac{1}{2}\}$	1×1	4
$\sum$				8732

Tablica 4.2: Skale i omjeri stranica referentnih okvira modela VGG-16/SSD 300×300

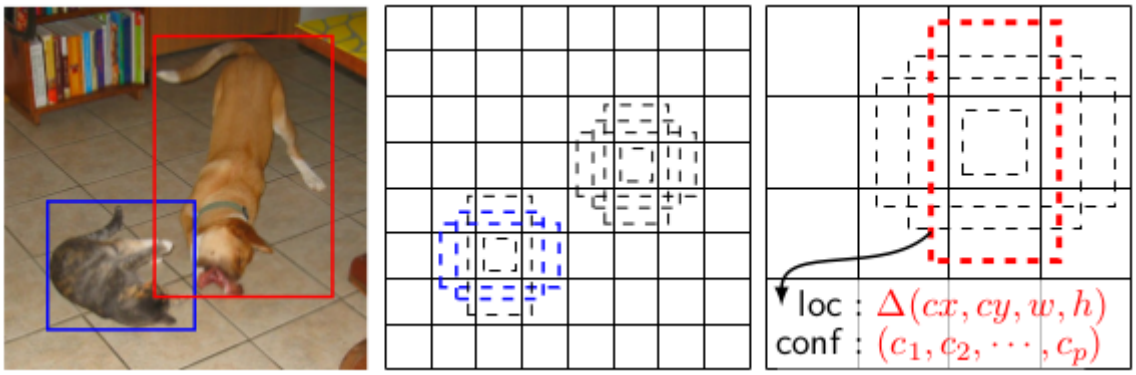
različitih veličina i oblika. Tako npr. slika 4.3 prikazuje situaciju u kojoj se jedan pretpostavljeni okvir u sloju veličine 4×4 podudara s oznakom psa, dok u sloju 8×8 to nije slučaj. To je zato što referentni okviri korišteni u slojevima veće rezolucije imaju manju skalu i u mogućnosti su detektirati manje objekte kao što je mačka prikazana na slici.

Podudaranje stvarnih oznaka s pretpostavljenim okvirima

Jedan primjerak za učenje u metodi SSD zapravo je jedan pretpostavljeni okvir mreže. Metoda ne radi sa skupom stvarnih oznaka skupa izravno. Tijekom učenja je dakle potrebno svkom primjeru iz skupa pretpostavljenih okvira pridružiti određenu oznaku. Jedan način za to napraviti bio bi za svaku stvarnu oznaku pronaći okvir koji se s njim ima najveće Jaccardovo podudaranje, te tome najbolje podudarajućem okviru pridružiti odgovarajuću oznaku. Drugi način, koji se koristi u ovoj metodi je sljedeći. Za svaki pretpostavljeni okvir koji se s nekom stvarnom oznakom skupa podudara više od 0.5 mjere Jaccard, pridružuje se ta oznaka. Time se se mreži dopušta da predvidi oznaku za više okvira koji se dobro poklapaju s oznakom skupa, umjesto da se odabere samo jedan najbolji, što pojednostavljuje problem učenja. Za sve okvire koji se podudaraju s nekom oznakom ispitnom skupa može se reći da su pozitivi u smislu da pokrivaju neki objekt od interesa. Prepostavljenim okvirima koji se ne podudaraju ni sa jednom oznakom ispitnog skupa na opisani način, pridružuje se oznaka pozadine. Za okvire kojima je pridružena oznaka pozadine može se reći da su negativi jer ne predstavljaju niti jedan od objekata koje je zadatak prepoznati.

4.2.4. Funkcija gubitka

Neka je $x_{ij}^p = \{0, 1\}$ indikatorska varijabla podudaranja i-tog pretpostavljenog okvira sa j-tom oznakom skupa koja pripada razredu p. Prema prethodno opisanom načinu podudaranja pretpostavljenih okvira i oznaka, može vrijediti $\sum_i x_{ij}^p \geq 1$. Funkcija gubitka (4.9) je težin-



Slika 4.3: Podudaranje pretpostavljenih okvira iz dva različita sloja mreže s oznakama (slika preuzeta iz članka [14])

Prva sličica prikazuje ulaznu sliku na kojoj je označen po jedan primjerak razreda mačka i pas. Druga sličica prikazuje mapu značajki dimenzija 8×8, a treća mapu dimenzija 4×4. Plavom crtkanom linijom na drugoj sličici označeni su dva pretpostavljena okviri koja su se podudarila s oznakom mačke. Crvenom crtkanom linijom unutar mape 4×4 označen je pretpostavljeni okvir koji se podudario s oznakom psa. Dva označena objekta različitih veličina podudarena su s okvirima iz različitih slojeva mreže.

ska suma gubitka lokalizacije i gubitka pouzdanosti klasifikacije u točnu kategoriju, gdje je N) broj pozitivnih okvira, koji su podudareni s nekim od objekata više od 0.5 mjere Jaccard. Ukoliko je $N = 0$ onda $L(x, c, l, g) = 0$.

$$L(x, c, l, g) = \frac{1}{N}[L_{conf}(x, c) + \alpha L_{loc}(x, l, g)] \tag{4.9}$$

Gubitak lokalizacije L_{loc} je zaglađeni L1 gubitak između parametara okvira kojeg je predvidila mreža (l) i stvarnih parametara okvira oko objekta (g) (4.10), gdje c_x, c_y, w i h predstavljaju koordinate centra odnosno širinu i visinu pravokutnika.

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \left(\sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} smooth_{L1}(l_i^m - \hat{g}_j{}^m) \right) \tag{4.10}$$

Varijable koje se uče su dane formulama (4.11), gdje su $\hat{g}_j^{cx}$ i $\hat{g}_j^{cy}$ predstavljaju pomak koordinata centra pretpostavljenog okvira d_i i njemu pridruženog okvira stvarne oznake g_j, u odnosu na širinu odnosno visinu promatranog pretpostavljenog okvira. $\hat{g}_j^w$ predstavlja logaritmirani odnos širine pravokutnika stvarne oznake pridružene pretpostavljenom okviru d_i i njegove širine. Analogno, $\hat{g}_j^h$ je logaritmirani odnos visine pravokutnika stvarne oznake pridružene pretpostavljenom okviru d_i i njegove visine.

$$\hat{g}_j^{cx} = \frac{(g_j^{cx} - d_i^{cx})}{d_i^w} \qquad \hat{g}_j^{yx} = \frac{(g_j^{cy} - d_i^{cy})}{d_i^h} \qquad \hat{g}_j^w = \log(\frac{g_j^w}{d_i^w}) \qquad \hat{g}_j^h = \log(\frac{g_j^h}{d_i^h}) \tag{4.11}$$

Funkcija zaglađenog L1 gubitka $smooth_{L1}$

$$smooth_{L1}(x) = \begin{cases} \frac{1}{2}x^2, & \text{za} \quad |x| < 1 \\ |x| - \frac{1}{2}, & \text{inače} \end{cases} \tag{4.12}$$

Gubitak pouzdanosti klasifikacije (4.13) je negativna log izglednost klasifikacije okvira u točnu kategoriju. Prva suma u izrazu (4.13) odnosi se na pretpostavljene okvire. Tim okvirima pridružena je oznaka p razreda s čijom se stvarnom oznakom podudaraju. $\hat{c}_i^p$ predstavlja vjerojatnost da pretpostavljeni okvir i pripada razredu p. Druga suma odnosi se na negativne pretpostavljene okvire, gdje $\hat{c}_i^p$ predstavlja vjerojatnost da pretpostavljeni okvir pripada kategoriji pozadine.

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Neg} \log(\hat{c}_i^0) \quad \text{gdje je} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_k \exp(c_i^k)} \tag{4.13}$$

Težina uz lokalizacijski član funckije gubitka odabrana je kros validacijom i postavljena na $\alpha = 1$.

4.2.5. Traženje teških negativa

Prema opisanom načinu podudaranja pretpostavljenih okvira sa stvarnim oznakama, većina okvira prilikom treniranja su proglašeni negativima, odnosno predstavljaju pozadinu. Ako bismo u obzir prilikom izračuna gradijenta funkcije gubitka u jednoj mini-skupini primjera za učenje uzeli sve primjere, učenje bi bilo otežano zbog neuravnoteženosti skupine. Zbog toga se pri izračunu gradijenta kod metode SSD uzima samo dio negativa, i to tako da je u svakoj mini-skupini broj negativa n puta veći od broja pozitiva. Postupak miniranja teških negativa je odabir onih negativnih primjera koji su za mrežu teški, odnosno teško ih razlikuje od pozitiva, s ciljem smanjenja broja lažno pozitivnih detekcija. SSD provodi traženje teških negativa na način da uzima one negative za koje je pouzdanost klasifikacije L_{conf} najmanja. Odnos broja negativa i pozitiva je jednak 3.

4.2.6. Povećanje skupa za učenje

Umjetno povećanje skupa za učenje koristi se kako bi se smanjio efekt prenaučenosti mreže. Kako bi se model učinio robusnijim na varijacije u veličinama i oblicima metoda SSD obrađuje ulaznu sliku na jedan nasumično odabran način:

- koristi cijelu ulaznu sliku
- uzima dio ulazne slike na način da je minimalno Jaccardovo preklapanje s objektima 0.1, 0.3, 0.5, 0.7 ili 0.9

– nasumično odabire dio ulazne slike

Veličina svakog odabranog dijela slike je $[0.1, 1]$ veličine ulazne slike a omjer visine i širine između $\frac{1}{2}$ i 2. U slučaju kad se kao ulaz uzima dio slike, moguće je da će neki od označenih objekata biti djelomično ili potpuno odsječeni. Ukoliko se centar pravokutnika originalne oznake ne nalazi unutar odabranog dijela slike oznaka se odbacuje, inače se zadržava njen preklapajući dio. Odabranom ulazu se mjenja veličina na fiksnu 300×300 ili 512×512, ovisno o modelu. Slika se zrcali po horizontali s vjerojatnošću 0.5 uz dodatne fotometrijske distorzije.

4.2.7. Rezultati

Svi eksperimenti u radu SSD baziraju se na već spomenutoj arhitekturi VGG16 prethodno učenoj za zadatak klasifikacije slika. Potpuno povezani slojevi fc6 i fc7 pretvoreni su u konvolucijske, peti sloj sažimanja pool5 veličine 2×2 sa korakom dva transformiran je u pool5 veličine 3×3 s korakom jedan. Svi dropout slojevi i fc8 su odbačeni. Tako dobivena arhitektura učena je stohastičkim gradijentnim spustom inicijalne stope učenja $10^{-}3$ momentom 0.9, 0.0005 faktorom l2 regularizacije težina i veličinom mini-skupine za učenje od 32 slike.

PASCAL VOC2007 Na ovom skupu uspoređuju se rezultati modela SSD sa metodama Fast i Faster R-CNN. Sve tri metode fino podešavaju težine iste pretreniranje mreže VGG16. Slojevi koji se koriste za predikciju su Slojevi koji se koriste za predikciju kategorija i pomaka okvira su conv4_3, conv7, conv 8_2, conv9_2, conv10_2 i conv11_2. Skala pretpostavljenih okvira na sloju conv4_3 postavljena je na 0.1. SSD512 model korsti i sedmu mapu conv12_2 za predikciju, skala u sloju conv4_3 je 0.07 a s_min postavljen na 0.15.

Težine svih nadodanih slojeva inicijaliziraju se xavier metodom [16]. Sloj conv4_3 ima različit raspon vrijednosti težina u odnosu na kasnije slojeve. Zbog toga se ulaz u prediktorski sloj iznad sloja conv4_3 $L2$ normalizira tako da norma po kanalima u svakoj lokaciji mape značajki bude 20 te se uči skala za svaki od kanala. Takva normalizacija koristi se kada se kao završni izlaz modela kombiniraju izlazi slojeva koji imaju različite skale vrijednosti kao što je pokazano u članku [17]. Usporedba performansi modela prikazana je tablicom 4.3.

Autori članka analizirali su model koristeći alat za analizu detekcije objekata i zaključili sljedeće.

– SSD model može detektirati razne kategorije objekata i većina pouzdanih detekcija su točne

– SSD se može zbuniti kod lokalizacije sličnih kategorija kao što su životinje, djelomično zbog toga što se sloj za lokalizaciju dijeli među kategorijama

skup za učenje	Fast R-CNN	Faster R-CNN	SSD300	SSD512
07	66.9	69.9	68.0	71.6
07+12	70.0	73.2	74.3	76.8
07+12+COCO	-	78.8	79.6	81.6

Tablica 4.3: Srednja prosječna preciznost mAP[%] na Pascal 2007 test skupu (podatci preuzeti iz članka [14]))

– SSD je dosta osjetljiv na veličinu okvira oko objekta odnosno ima puno lošije performanse na malim objektima u odnosu na veće objekte. Ističu da to nije iznenađujuće s obzirom da se u višim slojevima informacija o malim objektima gotovo potpuno gubi. Povećanje dimenzije ulaza pomaže.

4.3. Eksperimenti

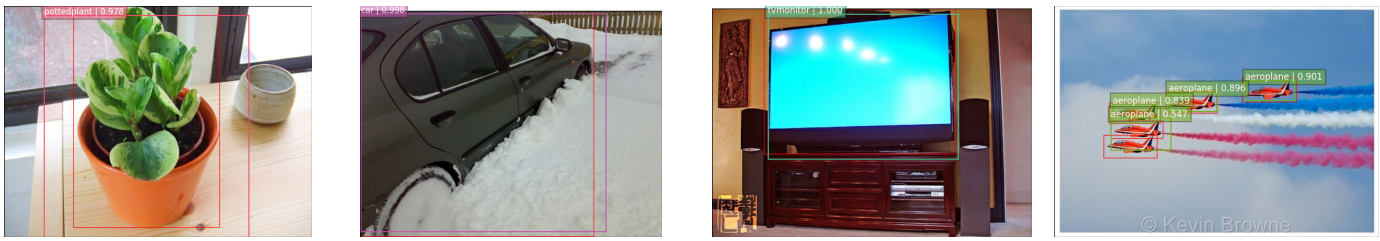
Metoda SSD isprobana je na skupovima Pascal VOC2007, Cityscapes i KITTI object. U svim eksperimentima korištena je arhitektura SSD300 jer je u odnosu na SSD512 iako manje precizna, brža u fazi učenja i evaluacije. U eksperimentima je korištenslužbeni kod temeljen na caffe-u [18]. Kod svih skupova korišten je isti skup pretpostavljenih okvira. Skala pretpostavljenih okvira u prvom sloju za predikciju s_{min} postavljena je na 0.2, a s_{max} u posljednjem na 0.9. Tu skalu uzela sam jer je predložena u okviru github repozitorija rada opisanog u članku. U prvom, predzanjem i zadnjem sloju korišteni su omjeri stranica okvira $\{1, 2, \frac{1}{2}\}$, a u ostalima $\{1, 2, \frac{1}{2}, 3, \frac{1}{3}\}$ što rezultira u ukupno 8732 okvira po jednoj slici. Faktor regularizacije u svim eksperimentima je 0.0005, a metoda optimizacije je učenje s momentom 0.9.

4.3.1. PASCAL VOC2007

Kao podatci za učenje korištena je unija skupova VOC2007 i VOC2012 za učenje i validaciju što je ukupno oko 11 tisuća slika, a performanse su mjerene na skupu VOC2007 za testiranje od 4952 slike. Provedeno je $120k$ iteracija odnosno 232 epohe učenja. Inicijalna stopa učenja postavljena je na 0.001 i smanjivana za faktor 10 u 80k-toj i 100k toj iteraciji. Dobiveni su rezultati prikazani tablicom 4.4. Najuspješnije kategorije, konj, automobil, autobus i vlak predstavljaju objekte koji su generalno veći, dok su najmanje uspješne kategorije biljka i boca, objekti koji češće zauzimaju manji dio slike. Slika 4.4 prikazuje 4 slike iz testnog skupa u kojima su pronađeni svi označeni objekti. Objekti se unutar tih slika ističu u odnosu na pozadinu. Slika ispod 4.5 prikazuje uspješnost SSD-a u izazovnijim situacijama.

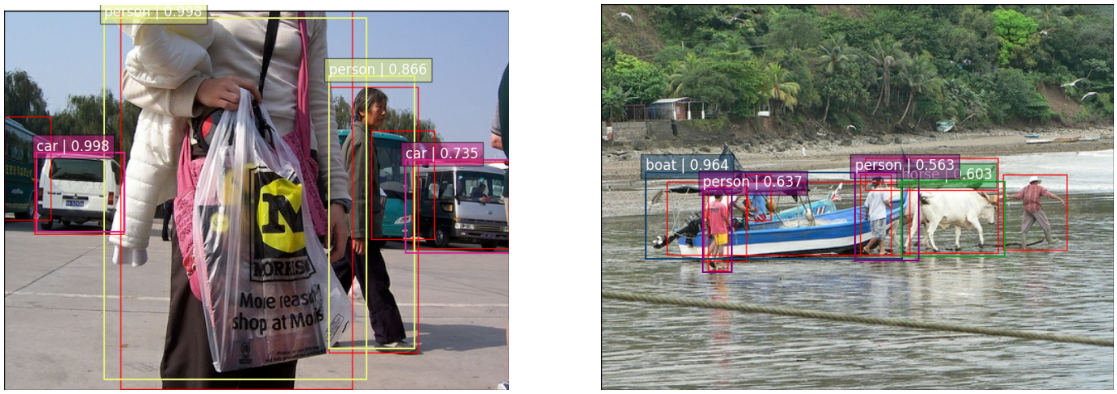
	$AP_i[\%]$		$AP_i[\%]$
aeroplane	80.1	diningtable	73.85
bicycle	82.6	dog	85.78
bird	75.34	horse	86.79
boat	71.78	motorbike	84.9
bottle	50.56	person	79.3
bus	86.21	pottedplant	50.9
car	85.76	sheep	78.54
cat	87.56	sofa	72.39
chair	53.85	train	86.7
cow	81.42	tvmonitor	75.29
mAP	**76.5%**		

Tablica 4.4: VOC2007 test: SSD300 AP



Slika 4.4: Prikaz slika s uspješno detektiranim objektima

Slika iznad prikazuje uspješne detekcije SSD-a na PASCAL VOC 2007 skupu za testiranje. Na slikama su crvenim okvirom označene stvarne oznake. SSD je u mogućnosti detektirati objekte različitih kategorija kao što su biljka, automobil, tv ekran i avion.



Slika 4.5: Detekcije u složenijim scenama

U slici lijevo SSD uspjeva detektirati obje osobe, iako su različitih skala, a jedna je i djelomično odsječena, ali ne i autobuse od kojih je jedan odsječen a jedan prekriven. U desnoj slici zamjenjuje kravu za konja, prepoznaje brod, i dvoje od troje ljudi.

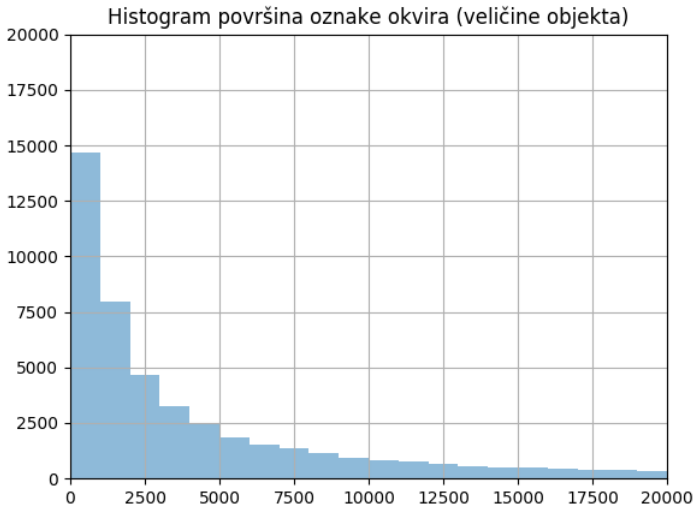
4.3.2. Cityscapes

Učenje SSD-a na skupu cityscapes provedeno je na isti način kao za pascal. Inicijalna stopa učenja od 0.001 na ovom je skupu bila prevelika, što je zaključeno na temelju oscilacija iznosa funkcije gubitka. Zbog toga je učenje započeto deset puta manjom stopom od 0.0001 i provedeno nekoliko epoha učenja. Kako konvergencija ne bi bila prespora, učenje je zaustavljeno, i nastavljeno sa stopom 0.001, nakon čega se stopa smanjivala na isti način kao kod ispitnog skupa PASCAL. Provedeno je ukupno 120k iteracija odnosno oko 1290 epoha učenja. Rezultati na skupu cityscapes prikazani su tablicom 4.5. Cityscapes je skup koji sadrži realne prometne scene, gdje su u odnosu na pascal objekti manji, i nema fotografske pristranosti. Neke od stvarnih oznaka uokviruju objekte koje je teško vidjeti i na originalnoj rezoluciji. Slika 4.7 prikazuje ulaz i izlaz mreže za dvije slike skupa za validaciju.

	$AP[\%]$
person	23
rider	26.58
car	49.58
truck	34.58
bus	46.98
train	52.49
motorcycle	18.92
bicycle	22.66
mAP	**34.34**

Tablica 4.5: Cityscapes skup za validaciju: SSD300 AP

Veliki dio oznaka skupa Cityscapes uokviruje jako male objekte. To je ilustrirano slikom 4.6 koja prikazuje histogram veličina oznaka objekata Cityscapesa. Detekcija malih objekata teža je nego velikih, dobro vidljivih objekata. Ako se uzme u obzir da metoda SSD na ulazu smanjuje sliku na veličinu 300×300 pitanje je koliki se dio informacije o malim objektima izgubi. Zbog toga su u nastavku dane tablice 4.6 koje prikazuju srednju preciznost zasebno za manje i veće objekte.



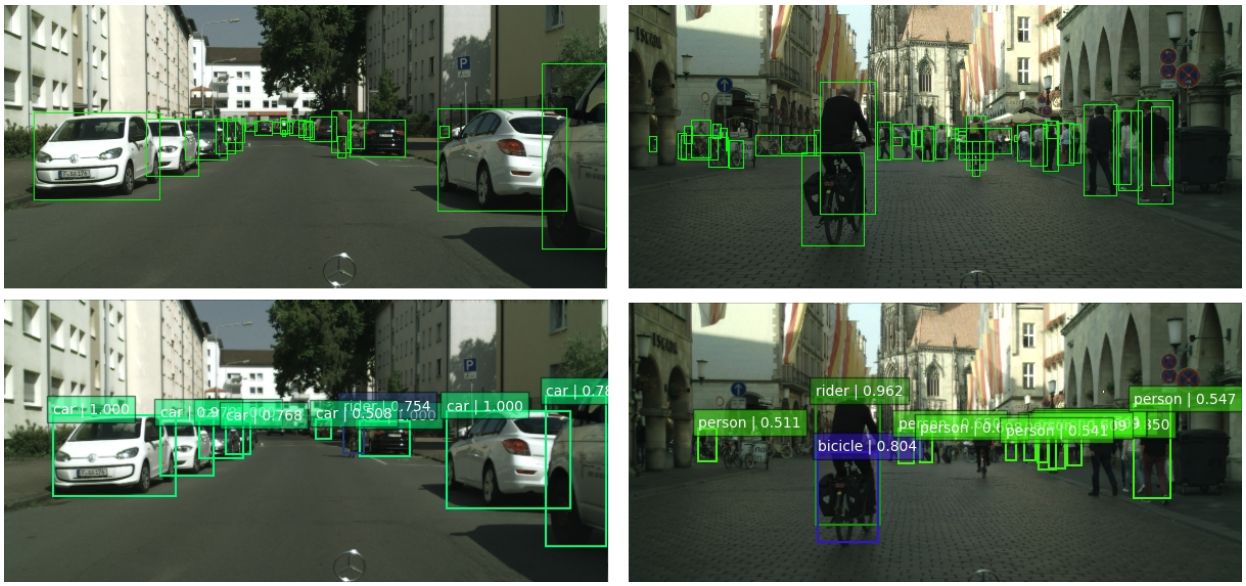
Slika 4.6: Histogram veličina okvira oko objekata skupa Ciytscapes

Iz histograma se vidi da većina oznaka pripada malim objektima. Broj objekata čija je površina oznake do 5000 je 33 tisuće, dok je broj objekata veličine čija je površina veća od 5000 i manja od 20000 piksela jednak 12 tisuća. S obzirom da slike Cityscapesa imaju veličinu 1024×2048, objekt veličine 5000 zauzima oko 0.2% cijele slike.

objekti>5000	$AP[\%]$
person	37.63
rider	50.68
car	75.32
truck	50.9
bus	58.03
train	53.03
motorcycle	33.11
bicycle	39.94
mAP	**49.8**

objekti<5000	$AP[\%]$
person	3.3
rider	1.1
car	6.8
truck	0.0
bus	0.0
train	0.0
motorcycle	0.46
bicycle	0.64
mAP	**1.53**

Tablica 4.6: Cityscapes skup za validaciju: **Tablica lijevo**: slučaj kad se prilikom evaluacije ignoriraju oznake objekata površinom manje od 5000. Performanse su bolje nego u slučaju kad se sve oznake uzimaju u obzir kao u tablici 4.5. Ta razlika je drastičnija kod kategorija manjih objekata kao što su *person* i *rider* nego primjerice *bus* i *train*. **Tablica desno**: slučaj kada se ignoriraju oznake objekata površinom veće od 5000. Performanse detekcije na malim objektima značajno su lošije nego na velikima.



Slika 4.7: Usporedba stvarnih oznaka (gore) i izlaza modela za dvije slike skupa Cityscapes

Gornje slike prikazuju stvarne oznake skupa. Neke od oznaka uokviruju objekte koji su mali i jedva vidljivi. Donje slike prikazuju detekcije metode SSD300. Sustav teže detektira i manje je siguran u detekcije objekta koji su udaljeniji od kamere i time manje površine.

4.3.3. KITTI

Kitti object skup za učenje i validaciju od 7481 podijeljen je na skup za učenje od 6500 i validaciju od 981 slika. Početna stopa učenja bila je 10^{-4} i u 100k-toj iteraciji smanjena je na $10^{-}5$. Mreža je učena ukupno 120k iteracija odnosno oko 590 epoha. Tablica 4.7 prikazuje performanse SSD-a na skupu KITTI.

	AP [%]
Car	84.7
Cyclist	56
Pedestrian	41.3
mAP	**60.7**

Tablica 4.7: KITTI skup za validaciju: SSD300 AP

Performanse su bolje nego na skupu Cityscapes jer je Cityscapes bolje označen: mali objekti koji u KITTI skupu nisu označeni, u Cityscapesu bi vjerojatno bili.

Slika 4.8: KITTI: Usporedba stvarnih oznaka (zeleno) i detekcija (crveno)
I na ovom skupu lakše se detektiraju veći objekti.

5. Segmentacija slika

Semantička segmentacija slika je postupak pridruživanja semantičke oznake svakom pikselu slike ili drugim riječima klasifikacija piksela u semantičke razrede. Isto kao što je opisano u prethodnom poglavlju, ovisno o točnosti klasifikacije, izlazni piksel može biti točno pozitivan, pogrešno pozitivan, pogrešno negativan i točno negativan. Točno negativan primjer je onaj koji ne pripada određenom razredu i u njega nije ni klasificiran. Kvaliteta segmentacije evaluira se sljedećim mjerama. Točnost piksela (engl. *pixel accuracy*) je udio točno klasificiranih piksela u ukupnom broju piksela 5.1, gdje je K broj kategorija, a j označava proizvoljno odabran razred.

$$Acc_{pixel} = \frac{\sum_{i=0}^{K-1} TP_i}{TP_j + FP_j + FN_j + TN_j} \tag{5.1}$$

Odziv i preciznost razreda i:

$$R_i = \frac{TP_i}{TP_i + FN_i} \quad P_i = \frac{TP_i}{TP_i + FP_i} \tag{5.2}$$

IoU (engl. *intersection over union*) razreda i je udio ispravno klasificiranih piksela razreda u skupu svih piksela koji tom razredu pripadaju ili su u njega svrstani 5.3.

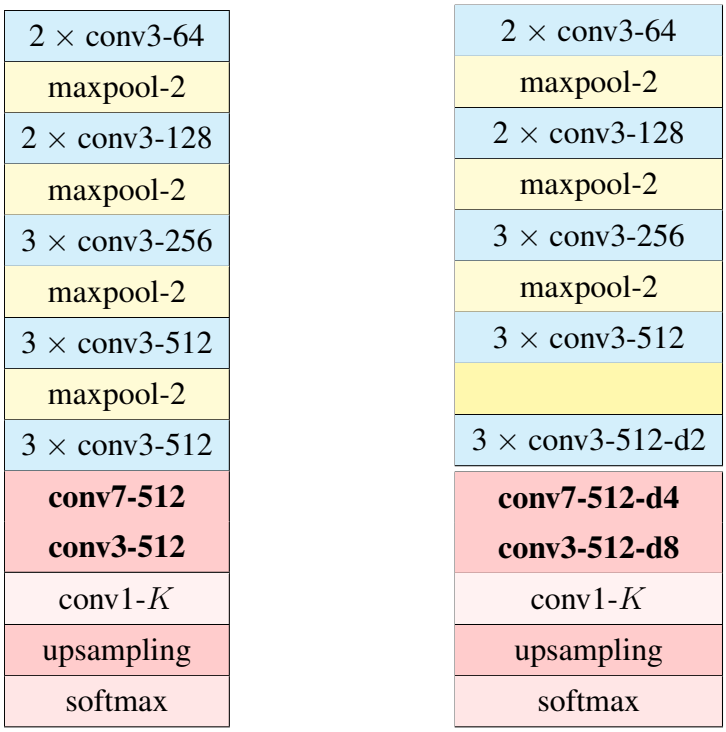
$$\text{IoU}_i = \frac{TP_i}{TP_i + FP_i + FN_i} \tag{5.3}$$

Odziv,preciznost i IoU se usrednjavaju po svim razredima.

$$R_{mean} = \frac{1}{K}\sum_{i=0}^{K-1} R_i \quad P_{mean} = \frac{1}{K}\sum_{i=0}^{K-1} P_i \quad IoU_{mean} = \frac{1}{K}\sum_{i=0}^{K-1} IoU_i \tag{5.4}$$

5.1. Potpuno konvolucijske mreže za segmentaciju

U okviru rada segmentacija se radi potpuno kovnolucijskom neuronskom mrežom. Prvi slojevi mreže za segmentaciju slika tipično će pripadati nekoj dubokoj arhitekturi prethodno učenoj za neki drugi zadatak, čime se dobivaju kvalitetne značajke za segmentaciju. Na to se dodaje nekoliko dodatnih konvolucijskih slojeva, zadnji sloj mreže imat će onoliko mapa značajki koliko je semantičkih kategorija. Ako mreža smanjuje prostorne dimenzije ulazne



Slika 5.1: Dvije potpuno konvolucijske mreže

Slojevi označeni plavom i žutom bojom pripadaju VGG mreži koja se sastoji od pet blokova konvolucijskih slojeva između kojih se nalaze slojevi sažimanja maksimumom. $n\times$ conv$k$-m-dr označava blok od n uzastopnih konvolucijskih slojeva u kojima se koristi kvadratna jezgra dimenzije k i dilatacija r, a broj mapa značajki u sloju je m. Kod oznake conv1-K, K označava broj kategorija. Oznaka maxpool-s označava nepreklapajuće sažimanje maksimumom, kvadratnim blokom veličine s. Podebljanim slovima označeni su slojevi u kojima se koristi normalizacija nad grupom.

mape, mapa završnog sloja interpolacijom se povećava na veličinu ulazne mape. Nad tim zadnjim slojem primjenjuje se funkcija softmax po kanalima čime se dobivaju mape vjerojatnosti za svaku od kategorija. Jedna takva mapa pokazuje vjerojatnost da piksel na određenoj poziciji pripada kategoriji promatrane mape. Završna segmentacija dobiva se na način da se pikselu izlaza pridruži ona oznaka čija vjerojatnosna mapa ima najveću vrijednost.

5.1.1. Eksperiment

Modul za semantičku segmentaciju korišten u okviru rada napisan je u Pythonu3 koristeći biblioteke numpy i tensorflow.

Model potpuno konvolucijske mreže temeljen je na arhitekturi VGG16 kao što je prikazano slikom 5.1. Težine slojeva koje pripadaju VGG-u, inicijalizirane su iz modela učenog za zadatak klasifikacije slika na skupu ImageNet. Aktivacijska funkcija u svim konvolu-

[%]	IoU_{mean}	Acc_{pixel}	R_{mean}	P_{mean}
model 5.1a	55.04	88.85	69.95	70.41
model 5.1b	59.03	90.52	74.17	72.57

Tablica 5.1: Performanse mreža na skupu od 500 slika za validaciju skupa Cityscapes

	$IoU[\%]$		$IoU[\%]$
cesta	95.06	nebo	87.43
pločnik	71.31	osoba	58.17
zgrada	83.95	vozač	38.33
zid	42.71	auto	86.60
ograda	39.94	kamion	43.61
stup	32.94	autobus	65.35
svjetlo	40.22	vlak	57.85
znak	48.93	motocikl	37.16
vegetacija	84.65	bicikl	58.82
teren	84.65	**avg**	**59.03**

Tablica 5.2: IoU segmentacije dilatiranim modelom (5.1b)na validacijskom skupu Cityscapes

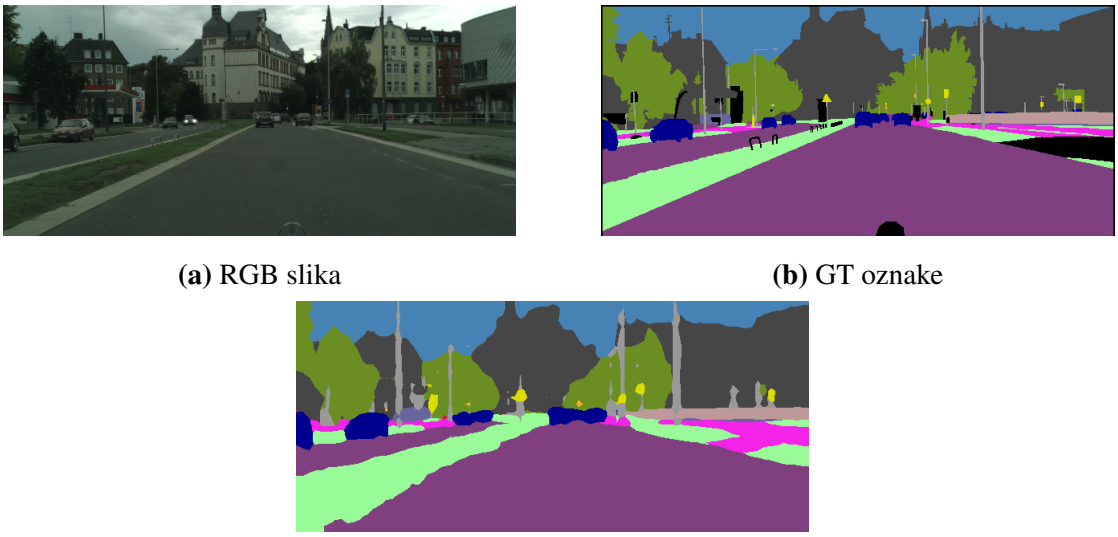
cijskim slojevima osim posljednjeg je $relu$. U konvolucijskim slojevima dodanima nakon slojeva VGG-a, osim posljednjeg sloja conv1-K koristi se normalizacija nad grupom.

Funkcija gubitka Funkcija gubitka koja se minimizira je gubitak unakrsne entropije odnosno negativna log izglednost pripadnosti piksela u stvarnu kategoriju (5.5), gdje je N broj piksela u mini-skupini, a $\boldsymbol{\theta}$ predstavlja sve parametre mreže.

$$L_{CE} = -\sum_{i=0}^{N} \log p(\hat{y}_i = y_i | x_i, \boldsymbol{\theta}) \tag{5.5}$$

Učenje mreže Model je učen na 2975 fino označenih slika skupa cityscapes smanjenih na rezoluciju 640×288 i veličinom mini grupe od 5 slika. Evaluacija je provođena na preostalih 500 slika iz skupa za učenje i validaciju. Broj kategorija skupa je $K = 19$. Korišteni optimizacijski algoritam je Adam. Inicijalna stopa učenja bila je 10^{-4}, a smanjivala se svake 3 epohe dijelenjem s 2. U svim slojevima osim posljednjeg korištena je regularizacija $L2$ normom vektora parametara s faktorom regularizacije 0.0005.

Eksperimentalni rezultati Nakon 15 epoha učenja opisana dva modela, dobiveni su rezultati prikazani tablicom 5.1. Rezultati potvrđuju da već odbacivanje jednog sloja sažimanja



(a) RGB slika **(b)** GT oznake

(c) Izlaz mreže

Slika 5.2: Prikaz ulaza i izlaza naučenog modela

i dilatacija konvolucija koje slijede. poboljšavaju performanse. Razlog je taj što iako povećavaju receptivno polje, slojevi sažimanja smanjuju rezoluciju mapa značajki pa se gubi informacija. Korištenjem dilatiranih konvolucija mogu se dobiti velika receptivna polja, bez uvođenja novih parametara i smanjenja rezolucije mapa. Iou mjera za svaki od razreda Cityscapesa, izmjerena na skupu za validaciju dana je tablicom 5.2. Rezultat segmentacije naučenim modelom za jednu sliku iz skupa za validaciju prikazan je slikom 5.2.

6. Istovremena segmentacija i lokalizacija

U okviru rada, zasebno gledajući zadatak segmentacije i detekcije objekata pronađene su metode koje daju zadovoljavajuće rezulte: potpuno konvolucijska mreža za segmentaciju i metoda SSD za detekciju objekata, no cilj rada je razviti model koji istodobno segmentira sliku i pronalazi pojedinačne primjerke objekata. Takav model učio bi oba zadatka istovremeno minimizacijom dva gubitka. Model koji istodobno rješava više zadataka smanjit će količinu izračuna i potrebne memorije i potencijalno omogućiti izvođenje u stvarnom vremenu. Ideja kako model za segmentaciju slika, opisan u prethodnom poglavlju, proširiti tako da pronalazi primjerke potječe iz članka [19] koji govori o višezadaćnom učenju u domeni razumijevanja scena. Članak predstavlja način otežavanja pojedinih komponenti gubitka koje pripadaju različitim zadatcima, koji se temelji na homogenosti varijance nesigurnosti zadataka. Osim toga autori članka razvili su model za razumijevanje scene koji istodobno obavlja zadatke semantičke segmentacije, segmentacije na razini primjerka (engl. *instance segmentation*) te regresije dubine. Na tom su modelu demonstrirali važnost odgovarajućeg otežavanja gubitka svakog zadatka za dobivanje najboljih performansi.

Višezadačno učenje je problem optimiranja modela s obzirom na više različitih ciljnih funkcija. Naivni pristup kombiniranju gubitaka različitih zadataka bio bi formulirati ukupni gubitak kao linearnu kombinaciju pojedinih gubitaka.

$$L_{total} = \sum_i w_i L_i \tag{6.1}$$

Kod ovakve formulacije gubitka, performanse modela mogu biti dosta osjetljive o izboru težina w_i, a pronalaženje optimalnih vrijednosti može biti skupo. Autori predlažu sljedeći gubitak gdje indeks i označava pojedini zadatak, a parametri σ_i se uče.

$$L(\boldsymbol{W}, \sigma_1, \sigma_2, ..., \sigma_i) = \sum_i \frac{1}{2\sigma_i^2} L_i(\boldsymbol{W}) + \log(\sigma_i^2) \tag{6.2}$$

Model razumijevanja scene iz članka [19] ima arhitekturu enkoder-dekoder. Enkoder je temeljen na ResNet-101 mreži i koristi dilatirane konvolucije. Enkoder izlučuje 2048 značajki koje se dijele između tri dekodera za zadatak segmentacije, označavanja primjeraka i rekonstrukcije dubine. Dimenzije izlaza enkodera su 8 puta manje od dimenzija ulaza.

Svaki dekoder sastoji se od tri konvolucijska sloja s jezgrama dimenzija 3×3, 1×1, 1×1 respektivno. Broj značajki u prva dva sloja je 512 a posljednja konvolucija ima broj mapa jednak broju dimenzija izlaza promatranog zadatka.

6.1. Regresija vektora do centroida objekta

Iz opisanog rada preuzeta je reprezentacija primjeraka objekata. Oni su zadatak označavanja primjeraka parametrizirali na način da su svakom pikselu ulazne slike pridružili vektor do centroida objekta kojemu promatrani piksel pripada. Ukoliko određeni piksel pripada kategoriji unutar koje ne razlikujemo pojedinačne objekate, pridružuje mu se nul-vektor. Učenje primjeraka svodi se na učenje vektora $\hat{x}_n$, za svaku koordinatu c_n koji pokazuje prema centroidu objekta i_n takvom da $i_n = \hat{x}_n + c_n$. Primjer tako generiranih oznaka prikazan je na slici 6.1. Ovu regresiju učili su koristeći L1 gubitak uprosječen preko svih označenih piksela u mini-skupini.

$$L_{instance} = \frac{1}{|N|} \sum_{N_I} ||x_n - \hat{x}_n||_1 \tag{6.3}$$

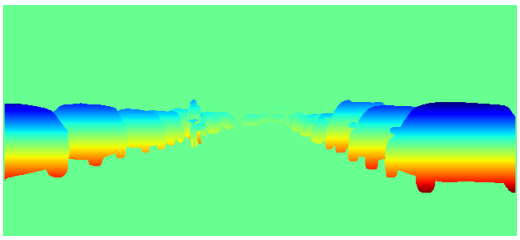
Da bi se dobile završne maske primjeraka, estimirani centri promatraju se kao glasovi u Houghovu parametarskom prostoru i grupiraju u grupe koje predstavljaju tražene primjerke.

6.1.1. Eksperiment

Kako bih proširila postojeću potpuno konvolucijsku mrežu za segmentaciju opisanu u prethodnom poglavlju za istodobnu segmentaciju i regresiju vektora do centroida, kao dijeljenu reprezentaciju između dva zadatka uzela sam izlaz iz posljednjeg sloja VGG-mreže. Na taj sloj nastavljaju se dvije grane, jedna za segmentaciju, jedna za označavanje primjeraka. Dio mreže za regresiju vektora sadrži tri konvolucijska sloja s veličinom jezgre 3×3, 1×1, 1×1 i brojem mapa 512, 512 i 2, kao što je opisano u članku. Izlaz regresije su dvije mape dimenzija ulazne slike. Prva mapa predstavlja x koordinatu, a druga y koordinatu vektora do centroida. Dio mreže za segmentaciju ostao je isti kao prethodno. Takav model prikazan je slikom 6.2, gdje podebljana slova označavaju da se u tom sloju koristi normalizacija nad grupom.

Gubitak segmentacije je negativna log izglednost klasifikacije piksela u točan razred, sumirana za sve piksele unutar mini-skupine (6.4). Gubitak regresije je $L1$ udaljenost stvarnog vektora do predviđenog vektora do centroida $[x_c, y_c]$, sumirano preko svih piksela u mini-skupini (6.5).

(a) x komponenta vektora



(b) y komponenta vektora

(c) RGB slika

Slika 6.1: Prikaz oznaka vektora do centroida objekta

Komponente vektora do centroida objekta kojemu piksel pripada za svaki piksel slike. Pikselima koji pripadaju kategorijama unutar kojih ne razlikujemo primjerke objekata kao što su kategorije cesta, zgrada, nebo, vegetacija, pridružen je nul vektor, što se na slikama vidi kao zelena boja. Učenje maski primjeraka objekata svodi se na regresiju vektora do centroida. Naučeni centri grupiraju se u grupe koje predstavljaju nađene primjerke.

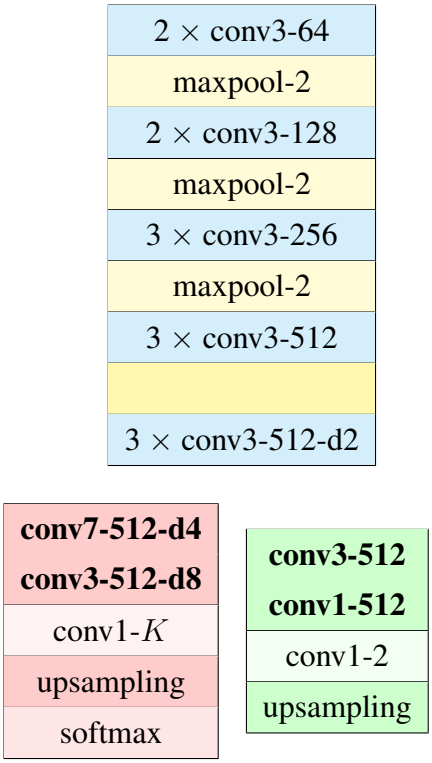
$$L_{CLS} = -\sum_{i=0}^{N} \log p(\hat{y}_i = y_i | x_i, \boldsymbol{\theta}) \tag{6.4}$$

$$L_{REG} = \sum_{i=0}^{N} |x_{c_i} - \hat{x}_{c_i}| + |y_{c_i} - \hat{y}_{c_i}| \tag{6.5}$$

Ukupni gubitak dan je sljedećom formulom (6.6).

$$L_{TOTAL} = \frac{1}{2\sigma_1^2} L_{CLS} + \log(\sigma_1^2) + \frac{1}{2\sigma_2^2} L_{REG} + \log(\sigma_2^2) \tag{6.6}$$

Opisani model učen je na skupu Cityscapes. Dijeljeni slojevi, i slojevi koji pripadaju segmentaciji inicijalizirani su iz modela prethodno naučenog za segmentaciju. Sve težine regulariziraju se $L2$ normom uz faktor regularizacije 0.001. Učenje je provedeno u 50 epoha optimizacijskim algoritmom Adam s početnom stopom učenja 0.0001 koja se svako dvije epohe smanjivala množenjem s faktorom 0.9. Početne vrijednosti parametara $\log(\sigma_i^2)$ za oba zadatka su inicijalizirani na vrijednost 3.5. Učenje je prekinuto kada je utvrđeno da greška segmentacije na validacijskom skupu počinje rasti. Performanse naučenog modela za zadatak segmentacije prikazane su u drugom retku tablice 6.1. Prvi redak u tablici prikazuje model koji je istovjetan ali naučen minimizirajući isključivo gubitak segmentacije.



Slika 6.2: Arhitektura za istovremenu segmentaciju i detekciju

Zajednički slojevi temelje se na prvih pet blokova konvolucijskih slojeva arhitekture VGG-16, gdje se u petom bloku koriste konvolucije dilatirane za faktor 2. Slojevi za semantičku segmentaciju sastoje se od dva dodana konvolucijska sloja i posljednjeg sloja čiji je izlaz K mapa vjerojatnosti pripadnosti piksela promatranoj kategoriji. Dio mreže za označavanje primjeraka čine dva dodana konvolucijska sloja i izlazni sloj s dvije mape koje predstavljaju x iy koordinatu vektora do centroida objekta kojemu promatrani piksel pripada. Semantička segmentacija uči se kao klasifikacija piksela, a označavanje primjeraka kao regresija koordinata vektora do centroida objekata.

[%]	IoU_{mean}	Acc_{pixel}	R_{mean}	P_{mean}
model 5.1b	59.03	90.52	74.17	72.57
model_multi	59.15	92.11	67.78	78.81

Tablica 6.1: Usporedba performansi mreža na zadatku semantičke segmentacije slika

Prvi model učen je isključivo za zadatak semantiče segmentacije, dok je drugi model učen minimizirajući dva gubitka: semantičke segmentacije i označavanja primjeraka

Kao mjera evaluacije regresije korištena je mjera korijena srednje kvadratne pogreške (6.7) gdje je N je ukupan broj svih piksela u skupu. Iznos pogreške na skupovima za učenje i validaciju dan je tablicom (6.2).

$$RMSE_x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_{c_i} - \hat{x}_{c_i})^2} \quad RMSE_y = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_{c_i} - \hat{y}_{c_i})^2} \tag{6.7}$$

	MSE_x	MSE_y	MSE_{mean}
train	4.75	3.00	3.875
validation	6.18	3.78	4.98

Tablica 6.2: Mjera korijena srednje kvadratne pogreške na skupu Cityscapes za učenje i validaciju

Za grupiranje centara u maske koje predstavljaju pronađene primjerke, dobiveni centri transformirani su u Houghov parametarski prostor i grupirani koristeći DBSCAN algoritam grupiranja [20]. Korištena je gotova implementacija iz pythonske biblioteke *scikit-learn*. U funkciju grupiranja ne predaju se sve točke već samo one koje je semantička segmentacija svrstala u neku od kategorija unutar koje se razlikuju pojedinačni primjerci. Kad algoritam grupiranja odredi grupe, svakoj grupi pridjeljuje se oznaka kategorije. To se radi na način da se pogleda izlaz semantičke segmentacije u svim pikselima koji pripadaju određenoj grupi, i uzme najčešća. Slikama 6.3 i 6.4 dani su izlazi modela za dvije slike skupa Cityscapes. U tablici 6.3 prikazana je srednja preciznost po kategorijama skupa. Za potrebe evaluacije svakoj grupi pridružena je pouzdanost detekcije kao nasumičan broj iz intervala $[0, 1]$. Kako bi se pokazala peformansa na većim objektima, filtrirani su svi primjerci objekata koji zauzimaju manje od 5000 piksela unutar slike. Ti rezultati prikazani su u tablici 6.4. Vrijeme potrebno za obradu jedne slike ovim modelom je oko 1.6 sekundi.

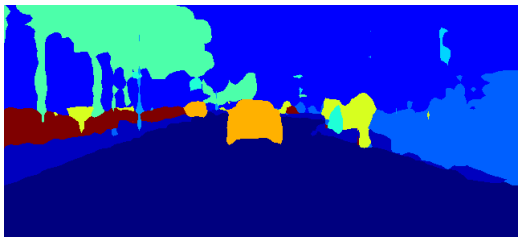
[%]	AP	$AP\_50\%$
person	2.3	9.3
rider	1	1
car	15.3	36.8
truck	13.1	24.3
bus	15.5	23.6
train	5.5	14.8
motorcycle	0.8	4.2
bicycle	0.3	1.9
mAP	**6.6**	**14.5**

Tablica 6.3: Cityscapes skup za validaciju: model za označavanje primjeraka
Stupac označen sa $AP\_50\%$ predstavlja mjeru srednje preciznosti kada se pozitivima smatraju detekcije koje se s određenom stvarnom oznakom preklapaju 0.5 mjere Jaccard. Tu mjeru možemo uspoređivati sa srednjim preciznostima danim u tablicama 4.5 i 4.6. Stupac označen sa AP je srednja preciznost uprosječena za slučajeve kad je minimalno preklapanje s oznakom u intervalu $[0.5, 0.95]$ s korakom 0.05.

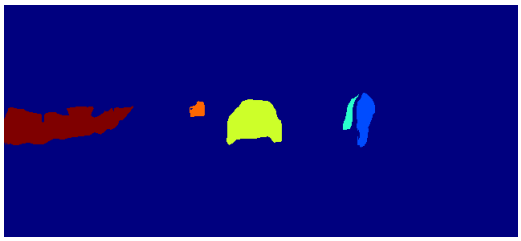
[%]	AP	$AP\_50\%$
person	4.3	14.5
rider	0	0
car	45.3	69.8
truck	34	57.6
bus	37.5	50.5
train	5.8	16.4
motorcycle	23	61.3
bicycle	0.8	4
mAP	**18.8**	**34.3**

Tablica 6.4: Cityscapes skup za validaciju: model za označavanje primjeraka
Srednje preciznosti na većim objektima. Većim objektima se smatraju primjerci čija maska oznake sadrži više od 5000 piksela.

(a) RGB slika



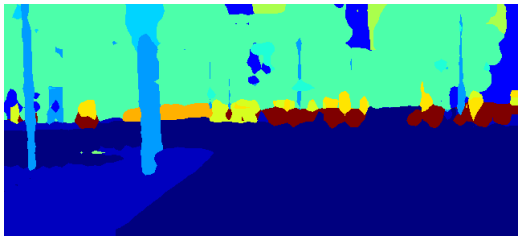
(b) Segmentacija slike



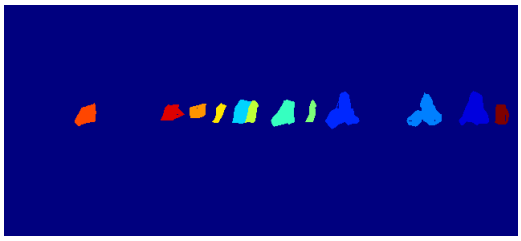
(c) Pronađeni primjerci

Slika 6.3: Prikaz kvalitativnih rezultata modela za istovremenu segmentaciju i detekciju na jednoj slici Cityscapes skupa za validaciju. Detektirana su dva automobila na cesti označena žutom i narančastom bojom na donjoj slici. Primjerak crnog automobila vidljiv u desnom središnjem dijelu ulazne slicke u boji nije detektiran jer piksele koji mu pripadaju semantička segmentacija nije ispravno klasificirala. Razlog bi mogao biti u prekrivenosti objekta ogradom.

(a) RGB slika



(b) Segmentacija slike



(c) Pronađeni primjerci

Slika 6.4: Prikaz kvalitativnih rezultata modela za istovremenu segmentaciju i detekciju na jednoj slici skupa za validaciju Slika prikazuje scenu sa puno sudionika u prometu. Neki pješaci i biciklisti se međusobno prekrivaju. U tim slučajevima često dolazi do pogreške grupiranja više primjeraka objekata u jedan.

7. Neuspjeli eksperimenti

7.1. Tensorflow implementacija metode SSD

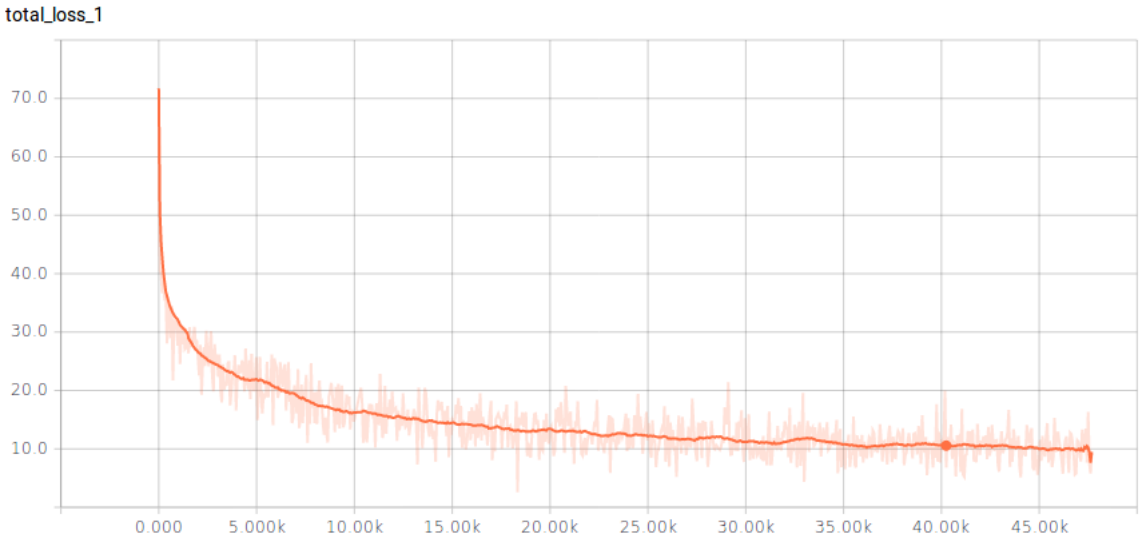
Modul za semantičku segmentaciju slika ispitnog skupa Cityscapes napisan je u tensorflow-u još u okviru kolegija Diplomski projekt. U okviru diplomskog rada najveći cilj je bio predložiti arhitekturu za istodobnu lokalizaciju objekata i segmentaciju slika. Kako bi se lokalizacijski modul mogao nadograditi na postojeću semantičku segmentaciju i učiti zajedno, bilo je poželjno da i on bude napisan u tensorflow-u. Kako je službeni kod metode SSD napisan koristeći biblioteku caffe, trebalo je pronaći (ili napisati) reimplementaciju metode koristeći tensorflow. U tu svrhu korišten je kod kojeg je na githubu objavio korisnik **balancap** na sljedećoj poveznici. Rezultati koje je autor prijavio koristeći ovu implementaciju prikazani su u tablici 7.1. Tablica pokazuje rezultate koji su čak bolji od rezultata iz originalnog članka [14]. S obzirom da su navedeni rezultati bili dobri, eksperimentiranje s metodom SSD započeto je koristeći kod iz ovog repozitorija.

model	skup za učenje	testni skup	mAP[%]
SSD300/VGG	VOC 07+12 trainval	VOC07 test	77.8
SSD300/VGG	VOC 07+12+COCO trainval	VOC07 test	81.7
SSD512/VGG	VOC 07+12+COCO trainval	VOC07 test	83.7

Tablica 7.1: Rezultati koje je prijavio **balancap** na skupu PASCAL VOC 2007 (preuzeto sa stranice https://github.com/balancap/SSD-Tensorflow)

7.1.1. Prvi pokušaj

Prvi eksperiment kao podatke za učenje koristio je isključivo slike iz skupa PASCAL VOC 2007 za učenje i validaciju, od ukupno 5011 slika. Inicijalne težine inicijalizirane su iz mreže VGG16 prethodno učene za zadatak klasifikacije slika na ImageNet skupu. Iz mreže VGG16 uzete su težine prvih 5 konvolucijskih blokova. Ostali slojevi conv_6-conv_11 kao i slojevi za regresiju pozicija okvira i klasifikaciju okvira inicijalizirani su xavier metodom. Skala pretpostavljenih okvira uzeta je iz raspona $[0.15, 0.9]$. Za optimizaciju je korišten algoritam



Slika 7.1: Kretanje funkcije gubitka tijekom optimizacije parametara modela SSD300/VGG tijekom prvih 50k iteracija učenja na PASCAL VOC 2007 skupu. Nakon ovih 50k iteracija mAP na skupu za učenje je bio 40% a na skupu za ispitivanje 32%.

Adam uz početnu stopu učenja 0.0005. Stopa učenja smanjivana je eksponencijalno svako 3 epohe množenjem s faktorom 0.98. Faktor L2 regularizacije težina bio je 0.0005. Nakon 50k iteracija što je oko 120 epoha učenja, prosječna srednja preciznost na skupu za učenje bila je 40% a na skupu za testiranje 32%. Učenje je nastavljeno do čak 350k iteracija što je otprilike 840 epoha. Prosječna srednja preciznost bila je 65% na skupu za učenje, a 44% na skupu za testiranje. U jednom trenutku gubitak je počeo stagnirati oko iznosa 6.0.

7.1.2. Uključenje skupa VOC2012

U tom trenutku u skup za učenje dodane su slike skupa za učenje i validaciju VOC2012. Osim toga, eksperimenti su preseljeni na novi stroj gdje je bilo moguće koristiti veću mini-skupinu od 26 slika. Učenje je nastavljeno s većom mini-skupinom i više podataka (ukupno 16551 slika) još 40k iteracija (62 epohe) što je rezultiralo poboljšanjem performansi: na skupu za učenje mAP je porastao sa 65% na 77%, a na skupu za testiranje sa 44% na 51%. Ovo je poboljšanje protumačeno kao pomak iz lokalnog minimuma zbog dodavanja novih primjera za učenje. Učenje je nastavljeno dalje no funkcija gubitka je ponovno počela stagnirati.

7.1.3. Usporedba s caffe kodom

Za usporedbu, prilikom učenja metode SSD300/VGG sa caffe kodom, već nakon 10000 iteracija, što je oko 20 epoha učenja, na testnom skupu prosječna srednja preciznost bila je

64%. Primjećene razlike između caffe i tensorflow eksperimenta su u sljedećem:

- **skup referentnih okvira**
  oba eksperimenta koristila su isti broj referentnih okvira s tim da je skala kod tensorflow-a bila $[0.15, 0.9]$ a kod caffe-a bila $[0.15, 0.9]$
- **normalizator funkcije gubitka**
  kod caffe koda gubitak mini-skupine dijeli se sa brojem pozitivnih okvira (broj N u formuli 4.9) dok je kod tensorflow koda normalizator bio broj slika u mini-skupini
- **inicijalizacija težina**
  Tensorflow kod na početku težine sloja conv_6 i conv_7 inicijalizira slučajno, dok ih caffe kod uzorkuje iz istih slojeva VGG16
- **optimizacijski postupak** Caffe eksperiment za optimizaciju koristi učenje s momentom, dok su eksperimenti u tensorflow-u koristili Adam

Razmotrivši razlike, zaključila sam da bi najveći problem mogao biti u razlici funkcija gubitka te inicijalizacije težina. Slučajna inicijalizacija slojeva conv_6 i conv_7 kod eksperimenta s tensorflow-om mogla bi objasniti sporost učenja. Razlika u skupovima referentnih okvira ne predstavlja problem što mogu zaključiti na temelju dobrih rezultata u članku SSD [14] koji koristi isti skup okvira kao tensorflow kod. Optimizacijski postupak također ne bi trebao igrati preveliku ulogu.

7.1.4. Imitacija caffe eksperimenta

Nakon što su utvrđene prethodno navedene razlike provela sam još jedan eksperiment s balancapovim tensorflow kodom. Cilj mi je bio što više imitirati eksperiment s originalnim caffe kodom. Koristila sam iste podatke za učenje, dakle uniju skupova PASCAL 2007 i 2012 za učenje i validaciju. Koristila sam isti optimizacijski algoritam dakle učenje s Momentom s veličinom mini skupine od 32 slike, s istom stopom učenja i politikom njenog smanjivanja te istu regularizaciju težina. Normalizator funkcije gubitka sam korigirala da bude jednak kao u caffe kodu. Da bih osigurala što sličniju inicijalizaciju težina napravila sam sljedeće. Ponovo sam pokrenula caffe kod i snimila težine nakon jedne jedine iteracije algoritma učenja. Tako dobivene težine koristila sam za inicijalizaciju modela u tensorflow-u. Rezultati dobiveni nakon završetka učenja prikazani su u tablici 7.2. Vidi se da su rezultati dosta lošiji nego kod čistog caffe modela. Razlog nažalost nisam uspjela utvrditi, ali zaključujem da ta dva modela nisu potpuno ekvivalentna.

	$AP_i[\%]$			AP_i
aeroplane	70.7		diningtable	54.0
bicycle	67.8		dog	80
bird	57.9		horse	75.25
boat	48.9		motorbike	73.25
bottle	33.1		person	66.26
bus	69		pottedplant	39.65
car	68.2		sheep	57.97
cat	80.9		sofa	62.32
chair	42		train	77.24
cow	66.9		tvmonitor	65.29
mAP	**62.76**			

Tablica 7.2: VOC2007 test: SSD300 AP

Ova se tablica može direktno uspoređivati s tablicom 4.4 koja prikazuje performanse caffe modela nakon učenja. Caffe model postigao je mAP od 76.5% što je značajno više od 62.7% koliko je postigao tensorflow model.

7.1.5. Kombinacija SSD-a i semantičke segmentacije

Unatoč tome što sam primjetila da balancap-ov kod u tensorflow-u nije ekvivalentan caffe-ovom, nisam potpuno odbacila tu reimplementaciju. Naime, htjela sam ukomponirati semantičku segmentaciju s detekcijom objekata. Zamislila sam to na sljedeći način. Kako SSD i modul za semantičku segmentaciju opisan u okviru petog poglavlja ovog rada koriste iste početne slojeve temeljene na mreži VGG16, te bi slojeve mogli dijeliti. Na izlazu takvog modela imat ću dva izlaza: izlaz semantičke segmentacije i okvire oko objekata s pridruženom oznakom razreda. Maske primjeraka objekata mogu dobiti kombinacijom ta dva izlaza. Na primjer, ako je metoda SSD određeni okvir klasificirala u razred automobil, onda ću uzimanjem onih piksela koje je semantička segmentacija unutar tog okvira svrstala u razred automobil, dobiti masku traženog objekta. Takvo označavanje primjeraka ovisilo bi o točnosti obje metode.

Eksperiment sam provela na sljedeći način. Iznad sloja conv_4 metode SSD dodala dva konvolucijska sloja za semantičku segmentaciju. Veličina jezgre prvog sloja bila je 7×7, a drugog 3×3. Broj mapa značajki u oba sloja bio je 512. Na ta dva sloja dodala sam završni izlazni sloj s brojem mapa jednakom broju razreda. Slična arhitektura završnih slojeva korištena je u modulu za semantičku segmentaciju iz poglavlja 5. Na početku sam naišla na problem. Metoda SSD300, smanjuje cijelu ulaznu sliku na veličinu 300×300.

Osim toga prilikom učenja izrezuje dijelove slike kako bi proširila skup za učenje i postigla veću robusnost. Drugi problem je rješen na način da se ulazna slika ne izrezuje već samo smanjuje na veličinu 300×300. Provela sam učenje u 50 tisuća iteracija optimizacijskim algoritmom Adam sa stopom učenja 0.0001 i veličinom mini-skupine od 16 slika. U eksperimentu su učeni samo dodani slojevi za semantičku segmentaciju, dok su slojevi za detekciju i zajednički slojevi ostali zamrznuti. Tako sam napravila kako ne bih izgubila performanse detekcije. Nakon procesa učenja izlaz semantičke segmentacije na skupu za validaciju mjera IoU usrednjena preko svih kategorija bila je samo 2.25%. Najbolje su se naučili razredi vegetacija (9.78%), nebo(8.56%), pločnik (8.28%), automobil (6%) i cesta (5.13%). Ostale kategorije imale su mjeru IoU manju od 1%. Ovaj način doživio je neuspjeh zbog dva razloga. Prvi razlog je preveliko smanjenje slike zbog metode SSD. Drugi razlog može biti što parametri zajedničkih slojeva metode SSD i semantičke segmentacije nisu optimirani. Zbog toga se težine prvih slojeva ne mogu prilagoditi zadatku semantičke segmentacije.

Kao ideju za budući rad predlažem da se metoda SSD prilagodi za rad sa cityscapesom na način da ulazna slika ne bude kvadratnih dimenzija, već da omjer visine i širine ulaza bude 1:2. Osim toga, model koji kombinira SSD i segmentaciju može se probati učiti koristeći otežavanje pojedinih gubitaka kao što je to napravljeno u modelu za istovremenu segemtaciju i lokalizaciju opisanom u prethodnom poglavlju ovog rada.

8. Zaključak

U okviru rada razmatrani su metode za detekciju objekata i segmentaciju slika. Metoda SSD pokazala se uspješnijom na PASCAL VOC2007 skupu nego na skupovima KITTI i Cityscapes koji sadrže slike iz prometa koje imaju manje objekte i složenije scene. Osim toga, model SSD300 prilikom faze testiranja na ulaz prima cijelu sliku kojoj prethodno mijenja veličinu na 300×300. Dimenzije slika u skupu PASCAL puno su bliže tim dimenzijama nego dimenzije slika iz druga dva skupa. Kao ideja za budući rad, predlažem testiranje modela SSD512 na skupu Cityscapes. Uz to, za pojedine skupove mogla bi se napraviti analiza veličina i omjera stranica njihovih oznaka kako bi se skup referentnih okvira SSD-a prilagodio određenom skupu. Osim toga, predlažem i razmatranje prilagodbe SSD-a slikama Cityscapesa na način da omjer visine i širine ulaza ne bude 1:1 već 1:2. Za te eksperimente mogao bi se koristiti nedavno objavljeni Tensorflow Object Detection API.

Semantička segmentacija potpuno konvolucijskiom mrežom zasnovanom na arhitekturi VGG16, klasifikacijom piksela pokazala se uspješnom. Uočena su poboljšanja u peformansama kada se umjesto slojeva sažimanja koriste slojevi s dilatiranim konvolucijama. Model za semantičku segmentaciju proširen je za zadatak segmentacije instanci. Označavanje instanci obavlja se učenjem regresije vektora promatranog piksela do centroida objekta kojemu pripada, nakon čega slijedi grupiranje naučenih centara u grupe. Da bi ovaj pristup mogao raditi u stvarnom vremenu, nužno je da korak postprocesiranja odnosno algoritam grupiranja bude učinkovit. Implementacija koju sam koristila u radu, nije dovoljno brza, pa bi svakako trebalo pronaći alternativu. Osim toga, za istodobnu segmentaciju i lokalizaciju mogao bi se isprobati neki moćniji model kao što je ResNet 101. Drugačiji pristup istodobnoj segmentaciji i lokalizaciji bilo bi proširivanje, detekcije objekata predikcijom maske objekta. U skladu s tim predlažem eksperimentiranje s metodom označavanja instanci Mask-RCNN koja je proširenje metode Faster-RCNN. Kako uspješno ukomponirati semantičku segmentaciji s metodom detekcije objekata SSD ostaje otvoreno pitanje.

LITERATURA

[1] Jan Šnajder i Bojana Dalbelo Bašić. Strojno učenje - skripta, 2014.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Intelligent Signal Processing*, pages 306–351. IEEE Press, 2001.

[3] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *CoRR*, abs/1511.07122, 2015.

[4] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836, 2016.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[12] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[13] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[14] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015.

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[16] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10). Society for Artificial Intelligence and Statistics*, 2010.

[17] Wei Liu, Andrew Rabinovich, and Alexander C. Berg. Parsenet: Looking wider to see better. *CoRR*, abs/1506.04579, 2015.

[18] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[19] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. *CoRR*, abs/1705.07115, 2017.

[20] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

[21] Petra Marče. Diplomski seminar: Detekcija objekata konvolucijskim neuronskim mrežama, 2016.

[22] J. Krapac M. Čupić, S. Šegvić. Duboko učenje - nastavni materijali, 2016.

Istovremena lokalizacija i segmentacija objekata

Sažetak

U okviru magistarskog rada proučavaju se duboke konvolucijske neuronske mreže za zadatke detekcije objekata i segmentacije slika. Metoda detekcije objekata SSD detaljno je opisana i isprobana na ispitnim skupovima PASCAL VOC2007, Cityscapes i KITTI. Za semantičku segmentaciju slika ispitnog skupa Cityscapes korištena je potpuno konvolucijska mreža s dilatiranim konvolucijama. Segmentacija instanci objekata ostvarena je kombiniranjem semantičke segmentacije i regresije vektora centroida objekta.

Ključne riječi: konvolucijske neuronske mreže, duboko učenje, klasifikacija, regresija, VGG16, detekcija objekata, semantička segmentacija, SSD, višezadaćno učenje, Cityscapes

Simultaneous object localization and segmentation

Abstract

This master thesis studies approach to object detection and image segmentation using deep convolutional neural networks. The SSD object detection approach is thoroughly described and tested on PASCAL VOC2007, Cityscapes and KITTI datasets. For semantic segmentation on Cityscapes dataset, fully convolutional network with *atrous* convolutions is used. Instance segmentation is achieved combining semantic segmentation and object centroid vector regression.

Keywords: convolutional neural networks, deep learning, classification, regression, VGG16, object detection, semantic segmentation, multi-task learning, Cityscapes