SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2597

SEMANTIČKA SEGMENTACIJA S BRZOM PAŽNJOM U PRESKOČNIM VEZAMA

Lovro Katalinić

Zagreb, lipanj 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2597

SEMANTIČKA SEGMENTACIJA S BRZOM PAŽNJOM U PRESKOČNIM VEZAMA

Lovro Katalinić

Zagreb, lipanj 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 12. ožujka 2021.

DIPLOMSKI ZADATAK br. 2597

Pristupnik:	**Lovro Katalinić (0036498817)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	prof. dr. sc. Siniša Šegvić

Zadatak: **Semantička segmentacija s brzom pažnjom u preskočnim vezama**

Opis zadatka:

Semantička segmentacija prirodnih scena važan je zadatak računalnog vida s mnogim zanimljivim primjenama. U posljednje vrijeme vrlo zanimljive rezultate na tom zadatku postižu konvolucijski modeli s lakom klasifikacijskom osnovom i ljestvičastim naduzorkovanjem. Posebno je zanimljiva izvedba ljestvičastog naduzorkovanja utemeljena na modulu brze pažnje. U okviru rada, potrebno je istražiti postojeće pristupe za semantičku segmentaciju. Posebnu pažnju posvetiti memorijskim zahtjevima naduzorkovanja latentnih reprezentacija. Naučiti i vrednovati model SwiftNet na javno dostupnim skupovima za semantičku segmentaciju. Unijeti brzi sloj pažnje u preskočne veze. Validirati hiperparametre, prikazati i ocijeniti ostvarene rezultate te provesti usporedbu s rezultatima iz literature. Predložiti pravce budućeg razvoja. Radu priložiti izvorni kod razvijenih postupaka uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 28. lipnja 2021.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2597

Semantička segmentacija s brzom pažnjom u preskočnim vezama

Lovro Katalinić

Zagreb, srpanj 2021.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

Zahvaljujem se svojoj obitelji i prijateljima na podršci tijekom cijelog školovanja te mentoru na ukazanoj pomoći i savjetima tijekom izrade rada.

SADRŽAJ

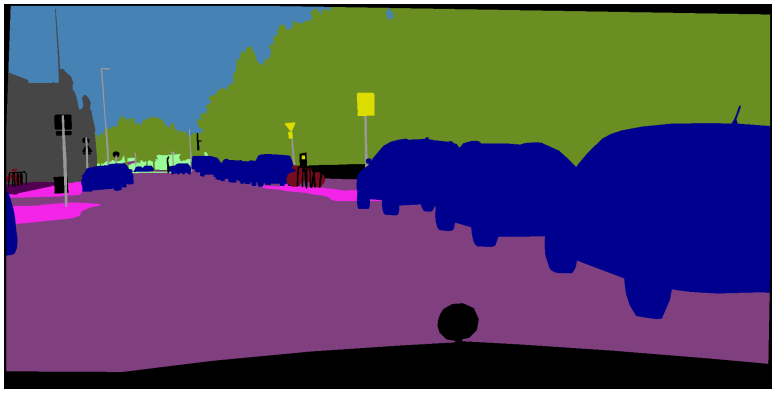
1. Uvod

Autonomna vozila, roboti i medicinski uređaji sve se više oslanjanju na samostalno donošenje zaključaka utemeljenih na interpretaciji slika. Računalni sustavi koji interpretiraju sliku stoga moraju biti pouzdani i precizni. U nekim je primjenama pored preciznosti vrlo bitna i brzina obrade slike.

Analiza i razumijevanja slike područje je kojim se bavi znanstvena disciplina koja se naziva računalni vid. Specifičan zadatak čiji je cilj klasificiranje svake jedinice slike, piksela, naziva se semantička segmentacija [3]. Ona daje informacije o lokacijama objekata na slici koju proučavamo te njihov međusobni odnos. Primjer semantičke segmentacije prirodne scene prikazan je na slici 1.1. Kvalitetna semantička segmentacija medicinske slike može na vrijeme dijagnosticirati postojanje zloćudnih stanica [33]. Segmentacijom više uzastopnih kontinuiranih slika moguće je otkriti smjer kretanja nekog objekta [45]. Na taj način mogu se predvidjeti sudari automobila i brzom reakcijom osigurati putnike na vrijeme.

U posljednje vrijeme, uz napredak procesne moći strojeva i dostupnosti većih skupova podataka, konvolucijski modeli pokazali su se kao najbolja rješenja mnogih problema računalnog vida, uključujući i problem semantičke segmentacije. Složeniji i veći modeli često postižu bolje rezultate i dosežu veću preciznost, ali zahtijevaju više računalnih resursa [4, 6, 33]. Modeli koji nastoje sliku obraditi u stvarnom vremenu stoga moraju žrtvovati složenost kako bi dobili na brzini [10, 16, 23, 26, 27, 29, 30, 32, 36, 41, 42, 43]. Novija istraživanja pokazuju da se preciznost može poboljšati uvođenjem mehanizma pažnje (engl. *self-attention*) u model [37]. Pažnja nastoji izvući kontekstualne informacije koje pomažu pri klasificiranju piksela na način da promatra zavisnost elemenata na različitim pozicijama. Brza pažnja [17] predstavlja reformulaciju tog mehanizma koja je optimizirana za brži izračun.

(a) Originalna slika



(b) Semantička segmentacija slike

Slika 1.1: Semantička segmentacija slike iz skupa podataka Cityscapes [8]. (a) Originalna slika. (b) Semantička segmentacija originalne slike. Svaki element slike pridružen je određenom razredu, a različiti razredi prikazani su drukčijim bojama.

2. Duboki konvolucijski modeli

2.1. Duboki modeli

Duboki modeli odnosno neuronske mreže su nelinearne jednom diferencijabilne funkcije koje predstavljamo računskim grafovima. Mnogi čvorovi računskog grafa odgovaraju skalarnom produktu aktivacija prethodnog sloja $\mathbf{x}$ i parametara modela $\mathbf{w}$ provučenom kroz nelinearnu aktivaciju f: $y = f(\mathbf{w}^{\mathbf{T}}\mathbf{x})$ (slika 2.1a). Aktivacijska funkcija najčešće se nalazi u obliku sigmoide ili zglobnice.

Računski graf dubokog modela možemo podijeliti u slojeve, tako da su izlazi jednog sloja ulazi idućeg sloja. Prvi sloj takve mreže kao ulaz uzima podatke koje analiziramo, a posljednji sloj na izlazu daje vrijednosti na temelju kojih o tim podatcima možemo donijeti zaključak. Kod klasifikacijskih modela, izlaz iz neuronske mreže određuje vjerojatnost pripadnosti ulaznih podataka jednom od jasno definiranih razreda. Između ulaznog i izlaznog sloja nalazi se jedan ili više skrivenih slojeva. Primjer neuronske mreže prikazan je na slici 2.1b. Mrežu koja ima više od jednog skrivenog sloja nazivamo dubokom neuronskom mrežom.

Modelu opisanom neuronskom mrežom nastojimo pronaći skup parametara koji će za problem kojeg rješavamo dati najbolje rezultate. Postupak pronalaska idealnog skupa parametara naziva se učenje. Čest oblik učenja je nadzirano učenje: mreži predajemo primjere parova ulaznih i izlaznih podataka $(\mathbf{x}, \mathbf{y})$ koji opisuju ponašanje koje želimo modelirati. Sličnost izlaza modela i točnih vrijednosti izlaza ocjenjuje se nekom mjerom koju nazivamo gubitkom modela, koji je obično diferencijabilan. Na temelju njegovog iznosa, optimiziramo parametre modela postupkom koji će osigurati da će ta pogreška u idućoj evaluaciji biti manja od trenutne. Cilj optimizacije je postići vrijednost parametara za koje je gubitak minimalan. Duboke modele najčešće optimiramo stohastičkim gradijentnim spustom. Nakon prolaska ulaznog podatka $\mathbf{x}$ kroz model, računa se gubitak modela L i gradijent gubitka s obzirom na parametre modela svakog sloja. Parametri modela W se zatim pomiču za određeni iznos, koji ovisi o stopi učenja η, u smjeru suprotnom od smjera gradijenta koji pokazuje smjer najvećeg

(a) Model neurona **(b)** Neuronska mreža

Slika 2.1: (a) Model neurona. Izlaz neurona je linearna kombinacija njegovih ulaza $x_1, x_2, ..., x_n$ s težinama $w_1, w_2, ..., w_n$, provedena kroz neku nelinearnu aktivacijsku funkciju f. (b) Neuronska mreža s jednim skrivenim slojem.

rasta pogreške:

$$W = W - \eta \frac{\partial L}{\partial W}$$

Najčešći oblik duboke neuronske mreže koja se koristi za obradu nestrukturiranih podataka je unaprijedno potpuno povezana mreža. Slojeve takvih modela možemo opisati afinom transformacijom aktivacija prethodnog sloja s obzirom na parametre modela:

$$x' = W \cdot x + b$$

2.2. Diferencijabilne transformacije za podatke sa strukturom rešetke

Problem potpuno povezanih dubokih mreža kod specifičnih zadataka kao što su računalni vid ili obrada prirodnih jezika je to što one ne uzimaju u obzir strukturu ulaznih podataka. U području računalnog vida podatak koji se obrađuje je slika, koja je specifična zbog svoje strukture rešetke. Svaki piksel u takvoj strukturi povezan je s pikselima koji ga okružuju. Kod obrade prirodnog jezike proučava se rečenica, u kojoj je bitan poredak riječi i njihov odnos.

S obzirom na to da slike predstavljaju relativno velik skup ulaznih podataka (slike iz skupa podataka Cityscapes imaju rezoluciju $3 \times 2048 \times 1024$, što odgovara 6.3 milijuna podataka), ako bismo htjeli naučiti potpuno povezani model da analizira slike, broj parametara kojeg bismo morali čuvati u radnoj memoriji je ogroman. Zbog toga bi

takav model bio sklon prenaučenosti, a učenje i evaluacija bili bi računalno zahtjevni i trajali iznimno dugo.

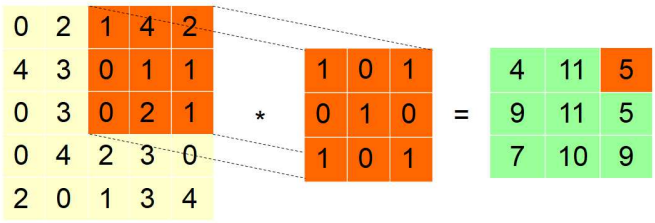
Rješenje navedenim problemima je zamjena potpuno povezanih slojeva konvolucijskim slojevima, koji nastoje lokalizirati interakcije pojedinih dijelova slike. Modeli koji se sastoje od barem jednog konvolucijskog sloja nazivaju se konvolucijskim modelima. Oni se uče na jednak način kao i klasični duboki modeli, optimiranjem parametara ovisno o gradijentu gubitka u ovisnosti o njima. Zbog toga dodatne transformacije koje uvode konvolucijske mreže moraju biti diferencijabilne.

2.2.1. Konvolucija

Operacija konvolucije je također linearna transformacija koja množi parametre s ulaznim podatcima. Matematički precizniji naziv operacije je unakrsna korelacija, ali je u području računalnog vidu naziv konvolucija uvriježen. Za razliku od neurona potpuno povezane mreže, izlazne aktivacije ovise samo o malom dijelu ulaza. Parametri konvolucije definirani su konvolucijskom jezgrom koja je puno manja od ulazne slike, najčešće dimenzija od 3×3 do 7×7. Ona se množi sa svakim isječkom ulazne slike odgovarajućih dimenzija. Konvolucija u dvije dimenzije definira se kao:

$$Y(i,j) = (K * X)(i,j) = \sum_{m=m_{min}}^{m_{max}} \sum_{n=n_{min}}^{n_{max}} K(m,n)X(i+m, j+n)$$

gdje $Y(i,j)$ označava piksel u i-tom retku i j-tom izlaza konvolucije, K jezgru konvolucije veličine $k \times k$, a X ulaznu mapu značajki. Operatori sumacije prolaze kroz susjedne elemente piksela nad kojim se vrši konvolucija, stoga su početne i krajnje vrijednosti sumacija jednake $m_{min} = n_{min} = -\lfloor k/2 \rfloor$ i $m_{max} = n_{max} = \lfloor k/2 \rfloor$.



Slika 2.2: Primjer operacije konvolucije jezgrom veličine 3×3 nad ulaznom mapom značajki dimenzija 5×5. Izlaz je linearna kombinacija okoline piksela koji se promatra i konvolucijske jezgre.

Svrha jezgara konvolucije jest detekcija specifičnih uzoraka na slici. Primjerice,

određeni poznati filteri zaduženi su za detekciju okomitih ili vodoravnih linija. U dubokim konvolucijskim mrežama, tijekom postupka učenja, filteri se specijaliziraju za prepoznavanje određenih uzoraka na slici.

Osim jezgrom, konvolucija se još definira korakom i dilatacijskim faktorom. Korak konvolucije određuje pomak jezgre nad slikom između dvije uzastopne operacije konvolucije. Faktor dilatacije određuje razmak između piksela mape značajki koji se množe sa susjednim elementima jezgre konvolucije.

2.2.2. Sažimanje

Prolaskom kroz mrežu nastojimo ulaznu sliku pretvoriti u sažeti tenzor značajki u kojem konvolucije imaju veće receptivno polje. Uglavnom takav tenzor prolaskom kroz mrežu smanjuje svoju prostornu veličinu, ali povećava dubinu. Slojevi sažimanja zaslužni su za smanjenje prostornih dimenzija slike. Slojevi sažimanja preslikavaju nekoliko bliskih elemenata mapi značajki u jedan element. Najčešće korištene funkcije preslikavanja su statistike ulaznih podataka, primjerice srednja ili maksimalna vrijednost. Uglavnom se slojevi sažimanja primjenjuju nakon provođenja većeg broja konvolucijskih slojeva.

2.2.3. Normalizacija grupe

Učenje dubokih mreža svakom iteracijom ažurira parametre slojeva i radi toga se distribucije ulaznih podataka svakog idućeg sloja mijenjaju. Slojevi se svakoj promjeni ulaznih distribucija moraju prilagođavati i zbog toga učenje modela traje znatno duže. Rješenje koje je omogućilo da se parametri sloja brže prilagode promjenama parametara prethodnih slojeva je normalizacija grupe (engl. *batch normalization*) [19]. Za svaku grupu podataka $X_B = x_1, x_2, ..., x_m$, gdje je m veličini grupe, prije ulaska u određeni sloj provodi se normalizacija distribucije tako da srednja vrijednost bude jednaka 0, a varijanca jednaka 1, i zatim se podatci iz grupe skaliraju i pomiču ovisno o učenim parametrima sloja:

$$\hat{x_i} = \frac{x_i - \mu_B}{\sqrt{{\sigma_B}^2 + \epsilon}}$$
$$y_i = \gamma \hat{x_i} + \beta$$

U gornjem izrazu μ_B predstavlja srednju vrijednost grupe podataka X_B, ${\sigma_B}^2$ njenu varijancu, a ϵ konstantu koja osigurava numeričku stabilnost. Parametri γ i β skaliraju i pomiču normalizirane podatke. S obzirom na to da su transformacije normalizacije po podatcima diferencijabilne, dodatno uvedeni parametri ažuriraju se na jednak

način kao i ostali parametri, propagacijom pogreške unatrag. U konvolucijskim mrežama normalizacija se provodi zasebno za svaki kanal, tako da se srednja vrijednost i varijanca računaju nad tenzorima dimenzija $m \times sirina \times visina$. Normalizacija omogućuje korištenje većih stopa učenja čime mreža brže konvergira te zamjenjuje druge tehnike regularizacije.

2.2.4. Transformacije za naduzorkovanje mapa značajki

U zadatku semantičke segmentacije, nakon ekstrahiranja bitnih značajki prilikom kojeg se prostorne dimenzije mapa značajke postupno smanjuju, potrebno je iste preslikati tako da izlaz mreže bude jednakih dimenzija kao i njen ulaz.

Operacija koja je suprotna operaciji konvolucije naziva se transponirana konvolucija. U normalnoj konvoluciji iz nekoliko susjednih elemenata računamo jedan izlaz, dok u transponiranoj verziji jedan ulaz nastojimo preslikati na više izlaza. Transponiranu operaciju možemo zamisliti tako da normalnu operaciju konvolucije zapišemo u obliku matričnog množenja ulaza izravnatog u vektor X i jezgre konvolucije K pretvorene u rijetku matricu C, čiji su definirani elementi težine jezgre konvolucije, na način $Y = C^T X$ [9]. Primjer jezgre konvolucije K veličine 3×3 i njene prilagodbe za množenje s ulazom veličine 4×4 dan je u nastavku:

$$K = \begin{pmatrix} k_0 & k_1 & k_2 \\ k_3 & k_4 & k_5 \\ k_6 & k_7 & k_8 \end{pmatrix}$$

$$C = \begin{pmatrix} k_0 & k_1 & k_2 & 0 & k_3 & k_4 & k_5 & 0 & k_6 & k_7 & k_8 & 0 & 0 & 0 & 0 & 0 \\ 0 & k_0 & k_1 & k_2 & 0 & k_3 & k_4 & k_5 & 0 & k_6 & k_7 & k_8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & k_0 & k_1 & k_2 & 0 & k_3 & k_4 & k_5 & 0 & k_6 & k_7 & k_8 & 0 \\ 0 & 0 & 0 & 0 & 0 & k_0 & k_1 & k_2 & 0 & k_3 & k_4 & k_5 & 0 & k_6 & k_7 & k_8 \end{pmatrix}$$

Ulazna mapa značajki veličine 4×4 ravna se u vektor dužine 16, množi se matrično s rijetkom matricom C i producira vektor dužine 4 koji se preslaže u matricu dimenzija 2×2. U slučaju da želimo provesti suprotnu operaciju, odnosno povećati dimenzije ulazne matrice i zadržati obrazac operacije konvolucije, možemo ulaz manjih dimenzija pomnožiti transponiranom matricom C.

Druge metoda naduzorkovanja su operacije suprotne sažimanju. Prvo se mapa značajki proširi na željene dimenzije tako da se novi elementi dodaju u okolinu svakog postojećeg piksela, a zatim se novim pikselima određuju vrijednosti. Najčešće se svi

pikseli naduzorkovanog tenzora računaju bilinearnom interpolacijom, a u nekim slučajevima mogu poprimiti i vrijednost najbližeg susjeda ili se njihova vrijednost ona postavlja na nulu.

3. Konvolucijski modeli za gustu predikciju

3.1. Mjere kvalitete modela za gustu predikciju

Najjednostavnija mjera za evaluaciju klasifikacijskih modela za gustu predikciju jest točnost piksela. Definira se kao postotak piksela koje je model ispravno klasificirao:

$$acc = \frac{1}{HW} \sum_{i=1}^{H} \sum_{j=1}^{W} 1\{Y_{ij} == Y'_{ij}\}$$

Parametri H i W predstavljaju visinu, odnosno širinu slike, a Y_{ij} i Y'_{ij} točni, odnosno prediktirani razred piksela u retku i i stupcu j. Problem ove mjere primjećuje se u slučaju neravnomjerne zastupljenosti razreda na slici, koja je u slikama prirodnih scena česta. U slučaju da je zastupljenost nekog razreda na slici puno veća od ostalih, na primjer pozadina u odnosu na automobile, ova mjera će pokazivati vrlo dobar rezultat i u slučaju da su pikseli koji pripadaju većinskom razredu točno, a pikseli manjinskih razreda netočno klasificirani.

Zbog problema neravnomjernosti razreda za ocjenu modela češće se koristi omjer presjeka i unije (engl. *mean intersection over union, mIoU*). Mjera za problem semantičke segmentacije uspoređuje broj piksela koje je model ispravno klasificirao u određeni razred s ukupnim brojem piksela koje je ili model klasificirao u taj razred ili oni stvarno pripadaju tom razredu. Ukupna mjera predstavlja prosjek mjera za pojedine razrede:

$$mIoU = \frac{1}{C} \sum_{c=1}^{C} \frac{|Y_c \cap Y'_c|}{|Y_c \cup Y'_c|}$$

gdje Y_c predstavlja skup piksela na slici čiji je pravi razred c, a Y'_c skup piksela na slici koje je model klasificirao u razred c. Vizualizacija presjeka i unije dostupna je na slici 3.1. Mjera će postići najveću vrijednost za slučaj kad su presjek i unija skupova piksela jednaki, to jest kada su svi pikseli ispravno klasificirani. Ovaj način evaluacije kvalitete modela je precizniji od točnosti piksela jer u obzir uzima i zastupljenost razreda.

Slika 3.1: Mjera omjera presjeka i unije uspoređuje broj ispravno klasificiranih piksela s ukupnim brojem piksela koje je model klasificirao u određeni razred ili njemu stvarno pripadaju. Na lijevoj slici prikazan je presjek dva skupa, dok je na desnoj prikazana njihova unija. Mjera postiže najvišu vrijednost u slučaju kad su skupovi jednaki.

3.2. Arhitekture za klasifikaciju slike

Modeli za semantičku segmentaciju oslanjaju se na modele čiji je zadatak klasifikacija pojedinih dijelova slike. Zbog toga je u narednih nekoliko poglavlja opisan razvoj modela za klasifikaciju. Kronološki su objašnjeni noviteti u mrežama koji su doprinijeli njihovim preciznostima te brzinom učenja i predikcija.

Jedna od prvih konvolucijskih mreža koja je pokazala njihov potencijal u području računalnog vida je LeNet [22] koju su predstavili Yann LeCun i ostali. Metodom prosljeđivanja pogreške unatrag naučili su jednostavnu konvolucijsku mrežu da uspješno prepoznaje ručno pisane znamenke poštanskih kodova. Konvolucijske mreže koje su prethodile mreži LeNet, za razliku od nje, nisu učile jezgre konvolucije, već su koristile ručno odabrane parametre. Mreža se sastoji od dvije konvolucije jezgrama veličine 5×5 praćene slojevima sažimanja srednjom vrijednošću te aktivacijama tangensom hiperbolnim. Izlaz konvolucijskih blokova spaja se na tri potpuno povezana sloja, koji u konačnici daju predikciju znamenke. Iako su konvolucijski modeli godinama značajno napredovali, LeNet-5 je postavio temelje arhitektura koji se i danas koriste: konvolucijski blokovi sastoje se od konvolucije, sažimanja i nelinearne aktivacije, za klasifikaciju slike na konvolucijski sloj spajaju se potpuno povezani slojevi te se za učenje parametara koristi metoda prosljeđivanja pogreške unatrag. U razdoblju koje je uslijedilo nakon predstavljanja LeNet-a 1989., konvolucijske mreže zbog ograničenja računalnih resursa i postojanja manje zahtjevnih alternativa koje su davale slične rezultate nisu bile popularne i njihov je razvoj privremeno stagnirao, sve do 2012. kad je model AlexNet postigao zadivljujuće rezultate i podsjetio na njihov potencijal u problemima obrade slika.

Mreža AlexNet [21] na natjecanju ILSVRC-2012 u problemu klasifikacije postigla

je rezultate koji su bili znatno bolji od suparničkih modela koji nisu koristili učenje s kraja na kraj. Mreža je u odnosu na prethodne radove u arhitekturu uvela novitete koji su ubrzali postupak učenja i poboljšali generalizaciju modela. Umjesto funkcija sigmoide i tangensa hiperbolnog koje su se koristile kao aktivacijske funkcije, AlexNet uvodi funkciju zglobnice koja zbog kvalitetnije propagacije gradijenata ubrzava učenje modela. S obzirom na to da najbolje grafičke jedinice u to vrijeme nisu imale dovoljno memorije za učenje većih mreža (3GB), autori su učenje provodili paralelno na dvije grafičke jedinice. Svaka od njih obrađivala je određeni podskup podataka, a u određenim su slojevima međusobno komunicirale. Time su pokazali način na koji je moguće znatno brže učiti duboke modele, na kojem se baziraju i današnja suvremena rješenja. Omogućavanjem učenja dublje mreže, AlexNet je ukazao na bitnost dubine neuronskih mreža. Arhitektura se sastoji od pet konvolucijskih slojeva s jezgrama veličine do 11×11 i tri potpuno povezana sloja. Tijekom učenja mreže koristila se tehnika povećanja podataka (engl. *data augmentation*), s obzirom da veći broj podataka za učenje smanjuje opasnost od prenaučenosti.

Napretkom u razvoju grafičkih jedinica omogućio se razvoj još dubljih mreža. Mreža VGG [35] postigla je najbolje rezultate u klasifikaciji i lokalizaciji na natjecanju ILSVRC-2014. Za razliku od prethodnih pobjednika natjecanja, koji u prvim slojevima mreže koriste jezgre veličine 7×7 ili 11×11, VGG u svim slojevima koristi manje jezgre veličine 3×3. Spajanje više konvolucijskim blokova s manjim jezgrama omogućuje veća receptivna polja uz manji broj parametara. Više slojeva funkciju odlučivanja čini diskriminativnijom, zbog čega dublji model bolje generalizira. Inačice mreže VGG sastoje se od 11 do 19 konvolucijskih slojeva.

3.2.1. Rezidualne mreže

Povećanjem dubine povezivanjem više slojeva počeo se isticati problem nestajućih gradijenata. S obzirom na to da se mreža uči propagirajući gradijente gubitka unatrag, manji gradijenti u rasponu $[0, 1]$ konstantnim se množenjem smanjuju sve dok ne iščeznu, što onemogućuje ažuriranje parametar plićih slojeva. Uvođenjem normalizirane incijalizacije parametara i normalizacijskih slojeva unutar mreže riješilo je ovaj problem.

Riješavanjem problema nestajućih gradijenata dublje mreže uspijele su konvergirati, ali počeo se javljati problem zasićenja preciznosti, koji je uputio na to da različiti sustavi imaju različitu zahtjevnost optimizacije. Ideja koju uvode rezidualne mreže [14] je da svakom sloju mreže pored standardnih konvolucijskih blokova dopustimo

da propagira ulazni signal kroz mrežu. Rezidualne neuronske mreže sastoje se od rezidualnih jedinica. Rezidualna konvolucijska jedinica je spoj nekoliko konvolucijskih slojeva $F(x, W)$ čiji se izlaz kombinira s ulazom u jedinicu te se njihova kombinacija provodi kroz funkciju f:

$$y_l = h(x_l) + F(x_l, W_l)$$

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

Veza koja spaja ulaz jedinice i izlaz konvolucija je funkcija h koja se naziva prečica. Najčešće je ona funkcija identiteta, iako se po potrebi mogu koristiti i druge funkcije. Funkcija identiteta daje najbolje rezultate jer ne uvodi dodatne parametre, ne povećava računalnu složenost i ne sputava protok informacija. Osnovnu konvolucijsku jedinicu možemo vidjeti na slici 3.2a. Konvolucijski slojevi unutar jedinice sastoje se od konvolucija praćenih normalizacijama i aktivacijama, a funkcija f definirana je kao nelinearna funkcija zglobnice. U slučaju da se uz konvolucije provodi i sažimanje mapi značajke, potrebno je dimenzije ulaza prečicom transformirati na dimenzije jednake izlazu konvolucija. Autori arhitekture rezidualnih jedinica uspjeli su mrežu naučiti na puno većem broju slojeva od onog kojeg mogu podržati klasične neuronske mreže, s time da su dublje mreže dale bolje rezultate nego pliće. Rezidualne mreže s većim brojem slojeva umjesto temeljnih jedinica koriste konvolucijske jedinice s uskim grlom (engl. *bottleneck*) kako bi ograničili broj parametara mreže koji se uvođenjem novih slojeva povećava. Takvi se jedinice, za razliku od temeljnih koji koriste dvije 3×3 konvolucije, sastoje od jedne takve konvolucije kojoj prethodi i koju slijedi 1×1 konvolucija, od kojih prva smanjuje broj filtera prije konvolucije, a druga ga povećava na broj jednak onom prije konvolucije. Prikaz jedinice s uskim grlom dan je na slici 3.2b. Pored smanjenja broja parametara, dodatni efekt uvođenja jedinica s uskim grlom je regularizacija čime se sprječava prenaučenost modela.

Predaktivacijske konvolucijske jedinice [15] predlažu stvaranje izravnog puta za propagiranje informacija kroz cijelu mrežu, na način da su i prečica h i fukcija f funkcije identiteta. Na taj način prethodnu jednažbu možemo svesti na:

$$x_{l+1} = x_l + F(x_l, W_l)$$

Iz prethodnog izraza slijedi da ulaz u sloj L možemo zapisati rekurzivno kao kombinaciju ulaza u bilo koju pliću jedinicu l i sume svih rezidualnih funkcija od sloja $i = l$ do $i = L - 1$:

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i)$$

U predaktivacijskim jedinicama konvolucijski slojevi sastoje se od konvolucija kojima prethode operacije aktivacije (slika 3.2c).

Nadogradnja rezidualnih mreža predstavljaju guste konvolucijske mreže. Mreža DenseNet [18] prečicama povezuje svaki sloj mreže sa svakim idućim slojem. Umjesto zbrajanja izlaza trenutnog sloja s njegovim ulazom kakvo se odvija u rezidualnim mrežama, gusto povezane mreže konkateniraju izlaz sa svim prethodnim slojevima. Tako se potiče dijeljenje parametara i time smanjuje njihov broj.

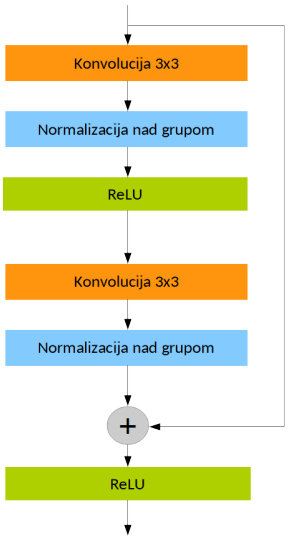
Arhitektura MobileNetV2 [34] koristi inačicu rezidualnih jedinica optimiziranih za mobilne uređaje koja minimizira broj parametara i operacija uz očuvanje visoke preciznosti. Njene se jedinice nazivaju obrnutim rezidualni jedinicama. One, za razliku od standardnih rezidualnih jedinica koje smanjuju broj kanala prije operacije konvolucije, proširuju broj kanala. U obrnutoj rezidualnoj jedinici autori koriste dubinski razdvojivu konvoluciju radi koje broj parametara i operacija ostaju manji nego kod standardnih rezidualnih jedinica. Iako je mreža ResNet značajno kompleksnija od mreže MobileNet V2, oba modela na grafičkim jedinicama imaju podjednaku brzinu zaključivanja, vjerojatno zbog bržeg izvršenja operacija običnih konvolucija naspram dubinski razdvojivih konvolucija.

3.2.2. Optimizacija konvolucije

S obzirom na to da operacije konvolucije predstavljaju značajni dio izračuna koji se provode u konvolucijskim modelima, tijekom godina se pokušavao osmisliti način optimiziranja te operacije. Spomenute rezidualne mreže [14] predstavile su koncept uskog grla. Usko grlo smanjuje broj parametara i broj potrebnih matričnih množenja tako da prije i poslije konvolucije s jezgrom 3×3 smanjuje te povećava broj kanala mape značajki provođenjem konvolucije jezgrom 1×1.

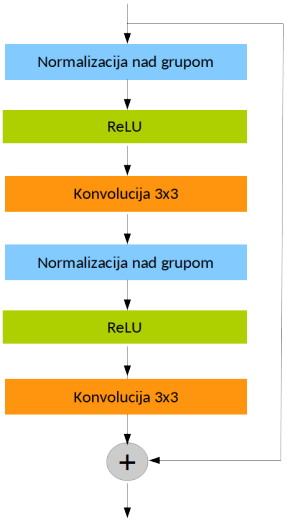
Prostorno razdvojiva konvolucija [20] razlaže standardnu operaciju konvolucije na tri uzastopne konvolucije koje se provode u svakoj od dimenzija mape značajki. Originalna jezgra konvolucije sadrži $C \times k^2$ parametara, gdje je C broj kanala, a $k \times k$ dimenzije jezgre, dok se jezgre prostorno razdvojive konvolucije sastoje od $C + 2k$ parametara. Ovakva dekompozicija ubrzava izračun, ali rijetko se koristi, jer se ne može svaka jezgra razdvojiti u više manjih jezgri. Učenjem dekompozicija se posljedično pretražuje samo podskup vrijednosti težina, koji ne sadrži nužno njihove optimalne vrijednosti.

Drugi oblik dekompozicije konvolucije, koji se češće koristi, jest dubinski razdvojiva konvolucija [38] koja faktorizira operaciju na dvije razine. Prvo se provodi



(a) Temeljna rezidualna jedinica

(b) Rezidualna jedinica s uskim grlom



(c) Predaktivacijska rezidualna jedinica

Slika 3.2: (a) Temeljna rezidualna jedinica sastoji se od dvije 3×3 konvolucije iza kojih slijedi normalizacija nad grupom. Izlaz konvolucija zbraja se s ulazom u jedinicu i provodi kroz nelinearnu funkciju. (b) Rezidualna jedinica s uskim grlom. Za razliku od temeljne jedinice, sastoji se od 3×3 konvolucije kojoj prethodi i slijedi konvolucija 1×1. Ovakva jedinica pokazala se korisnom jer regulira broj parametara i uvodi dodatan efekt regularizacije u model. (c) Predaktivacijska rezidualna jedinica stvara izravni put za propagiranje signala kroz cijelu mrežu. Mijenja redoslijed operacija konvolucija i aktivacija i ne provlači zbroj rezidualne funkcije i prečice kroz nelinearnost.

prostorna konvolucija nad svakim kanalom i zatim se računaju linearne projekcije nad kanalima. Prostorna konvolucija računa se tako da se nad svakim od C_1 kanala ulazne mape značajki provodi dvodimenzionalna konvolucija jezgrom veličine $k \times k$, a rezultirajuće mape značajki ne zbrajaju se kao kod standardne konvolucije. Nad njima se zatim provode konvolucije jezgrom 1×1 sa željenim brojem filtera C_2. Takvom dekompozicijom umjesto transformacije mapi značajki C_2 puta, ona se transformira samo jedanput i zatim se proširuje na C_2 kanala. Omjer broja parametara dubinski razdvojive konvolucije i obične konvolucije jednak je (uz visinu i širinu mape značajki H i W):

$$\frac{HWC_1k^2 + HWC_1C_2}{HWC_1k^2C_2} = \frac{HWC_1(k^2 + C_2)}{HWC_1k^2C_2} = \frac{1}{k^2} + \frac{1}{C_2}$$

Za, primjerice, jezgru veličine $k^2 = 9$ i broj filtera $C_2 = 256$, dubinski razdvojiva konvolucija ima 88% manje parametara od standardne konvolucije. Upotreba dubinski razdvojive inačice značajno ubrzava model, ali zbog redukcije broja parametra posljedično se smanjuje i kapacitet modela [34].

Zbog slabe procesne moći i manje memorije tadašnjih grafičkih procesora, autori mreže AlexNet [21] odlučili su grupirati konvolucije u dvije grane kako bi uspješno i efikasnije mogli naučiti mrežu. Svaka od njih paralelno provodi operacije konvolucije samo na mapama značajki te grane, uz povremene međusobne veze. Time se smanjuje broj operacija i prolaz kroz mrežu postaje efikasniji. Mreža ResNeXt [40] predstavlja nadogradnju mreže ResNet uz uvođenje grupiranih konvolucija u rezidualne blokove. Broj grana koje obrađuju samo dio mapa značajki koja ulazi u rezidualni blok naziva se kardinalitetom i predstavlja novu dimenziju modela pored dubine i širine. Svaka grana predstavlja usko grlo i djeluje tako da odabere podskup značajki konvolucijom jezgre veličine 1×1, zatim taj podskup transformira te vraća njene dimenzije. Autori arhitekture pokazali su da se povećanjem kardinaliteta jača regularizacija i poboljšava preciznosti modela.

3.2.3. Povećanje receptivnog polja

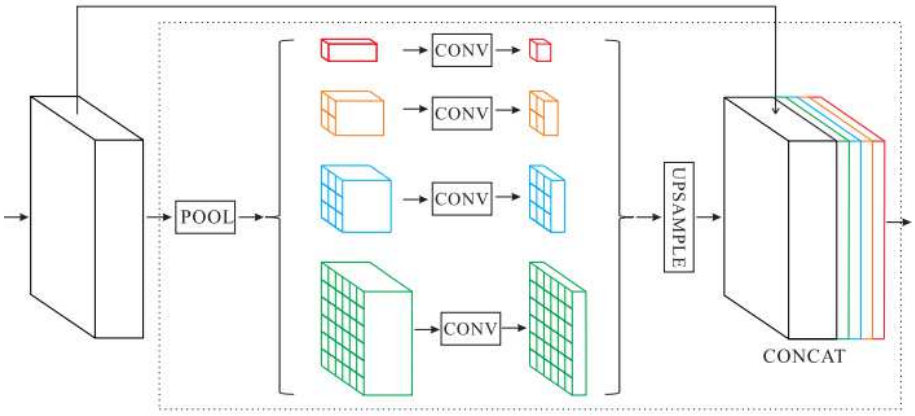
Receptivno polje definira se kao veličina regije ulazne slike koja utječe na određenu značajku u mreži. Veće receptivno polje znači da će se za zaključke donesene nad pojedinim dijelom slike koristiti veći broj njegovih susjednih elemenata, što te zaključke čini preciznijim i pouzdanijim. Slike često sadrže objekte različitih veličina, bilo to zbog njihove fizičke veličine ili njihove udaljenosti od objektiva kamere. Kako bi model uspio obuhvatiti njihov puni opseg, potrebno je da posjeduje veću receptivnu moć.

Pored navedenog, veće receptivno polje modela lokalnim značajkama daje na raspolaganje i globalni kontekst scene.

Jedan od načina povećanja receptivnog polja filtera jest zamjena uobičajene konvolucije dilatiranom konvolucijom [5]. Ona parametre filtera tijekom izvođenja operacije nad nekim pikselom ne množi s njegovim direktnim susjedima, već s onima koji su od njega udaljeni za neku stopu dilatacije. Na taj način širi se receptivno polje, a broj parametara modela ostaje jednak. Iako ovakva nadogradnja konvolucije poboljšava preciznost modela, zbog puno većih procesnih i memorijskih zahtjeva dilatirane inačice, modeli koji slike nastoje obraditi u stvarnom vremenu češće koriste drugačija rješenja.

Prostorno piramidalno sažimanje (engl. *spatial pyramid pooling, SPP*) [13] je dodatni sloj koji paralelno provodi sažimanje po prostornim rešetkama različite veličine (sažimanje cijele mape značajki, sažimanje četvrtina mape značajki itd.) te tako generira značajke s većim receptivnim poljem. Kod arhitektura za klasifikaciju potpuno povezani slojevi zahtijevaju ulaz konstantne veličine. Prostorno piramidalno sažimanje omogućilo je da ulaz u konvolucijske slojeve može biti proizvoljne veličine, a izlaz iz konvolucijskih slojeva piramidalnim sažimanjem uvijek se pretvara u vektor stalne veličine koji je pogodan za potpuno povezane slojeve [13]. Mreža koja prihvaća ulaz proizvoljne veličine riješila je problem smanjenja preciznosti koji je postojao zbog transformacija originalnih slika u slike jednake veličine. Nadogradnja originalnog SPP-a pogodnija za konvolucijske mreže predstavljena je u mreži PSPNet [42] i prikazana na slici 3.3. Takva inačica, za razliku od SPP-a koji sažete značajke ravna u vektor pogodan za potpuno povezane slojeve, značajke naduzorkuje bilinearnom interpolacijom do veličine ulaznih značajki i s njima ih konkatenira. Rezultanta mapa značajki pored lokalnih informacija sadrži i globalne kontekstualne informacije i koristi se kao ulaz u slojeve za naduzorkovanje. Neki modeli provode piramidalno sažimanja tako da umjesto klasičnih funkcija sažimanja koriste konvoluciju s različitim stopama dilatacije. Takvo se piramidalno sažimanje naziva ASPP (atrous SPP) [7].

Receptivno polje može se proširiti i korištenjem višerezolucijske piramide slika (engl. *multi-scale*) [11]. Višerezolucijski pristup uzima nekoliko različitih rezolucija ulazne slike i provodi ih kroz slojeve koje dijele parametre. Rezultantne mape značajki različitih razina model stapa tako da im prilagodi dimenzije i zatim ih konkatenira ili zbroji. Na taj način broj parametara mreže se ne povećava, a model bolje generalizira.



Slika 3.3: Inačica prostornog piramidalnog sažimanja (SPP-a) predstavljena u mreži PSPNet [42] koja je pogodna za potpuno konvolucijske mreže. Značajke dobivene sažimanjem različitim stopama naduzorkuju se bilinearnom interpolacijom do veličine ulaznih značajki i s njima se konkateniraju.

3.3. Potpuno konvolucijska mreža

Konvolucijske mreže čije posljednje slojeve čine potpuno povezani neuroni na izlazu daju vjerojatnosti da slika pripada određenoj kategoriji. Problem s takvom arhitekturom nastaje ako želimo klasificirati sliku koja se sastoji od nekoliko objekata, koji pri tome mogu pripadati različitim kategorijama. Prvi pokušaji rješavanja ovog problema svodili su se na lokalizaciju objekata, odnosno stvaranja graničnog okvira oko različitih objekata na slici. Nakon toga sadržaj svakog okvira pušten je kroz konvolucijsku mrežu koja ga je klasificirala u jednu od kategorija.

Idući logični korak u razumijevanju slike jest klasifikacija svakog piksela zasebno, odnosno gusta predikcija. Modeli koji daju najbolje rezultate za semantičku segmentaciju su potpuno konvolucijski modeli. Potpuno konvolucijski modeli umjesto potpuno povezanih slojeva koji se koriste prilikom klasifikacije slike sadrže slojeve koji ekstrahirane značajke nastoje preslikati na razinu piksela originalne slike. S obzirom na to da su modeli za semantičku segmentaciju uobičajeno složeniji od modela za klasifikaciju, jer imaju dodatne slojeve koji nastoje vratiti originalnu rezoluciju slike, sporije se i teže uče. Zbog toga je česta praksa učenje započeti s parametrima koji su prethodno naučeni za zadatak klasifikacije. Takva praksa naziva se prijenosom znanja (engl. *transfer learning*) i modelu na početku učenja ugrađuje početno znanje o podatcima kojima se pokušava prilagoditi.

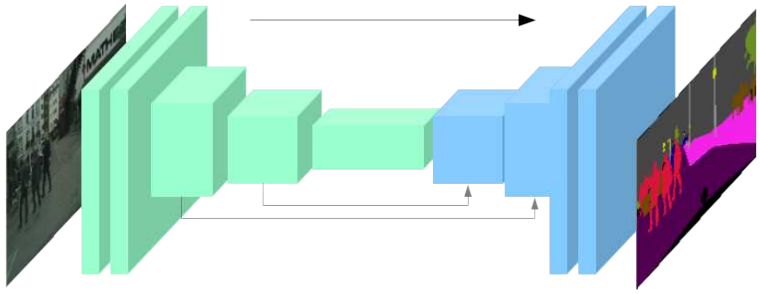
J. Long i E. Shelhamer u svom su radu [25] prilagodili postojeća rješenja za klasifikaciju [21, 35] za zadatak semantičke segmentacije, tako da su potpuno povezane

slojeve zamijenili slojevima za naduzorkovanje koji generiraju semantičku segmentaciju u originalnoj rezoluciji ulazne slike.

3.4. Arhitektura koder-dekoder

Novije arhitekture modela za semantičku segmentaciju najčešće se sastoje od dva dijela: kodera koji sažima prostornu rezoluciju slike i proizvodi mape značajki niske rezolucije, koje sadrže bitne semantičke informacije, te dekodera koji naduzorkuje dobivenu sažetu reprezentaciju u semantičku mapu, koja ima jednaku rezoluciju kao i ulazna slika.

Primjer arhitekture mreže koder-dekoder prikazan je na slici 3.4. Lijevi dio mreže predstavlja koder, dok desni predstavlja dekoder. Unaprijedni prolaz kroz mrežu odvija se u smjeru crne strelice, dok sive strelice predstavljaju preskočne veze između slojeva kodera i dekodera. One prilikom rekonstrukcije originalnih dimenzija mapama značajki koje sadrže apstraktne semantičke informacije daju informacije o prostornom položaju objekata. Na taj način rezultantna slika modela preciznije raspoznaje objekte.



Slika 3.4: Arhitektura mreže koder-dekoder. Lijevi dio mreže predstavlja koder koji sažima sliku i ekstrahira značajke slike. Desni dio mreže predstavlja dekoder koji mapu značajki preslikava na semantičku segmentaciju originalnih dimenzija slike. Crna strelica predstavlja smjer prolaza unaprijed, a sive strelice predstavljaju preskočne veze.

Jedna od prvih mreža koja je uspješno koristila spomenutu arhitekturu za segmentaciju biomedicinskih slika naziva se U-Net [33]. Njezin naziv potječe od simetričnog oblika mreže koji podsjeća na slovo u. Koder se sastoji od nekoliko konvolucijskih

blokova koji postepeno smanjuju rezoluciju slike i povećavaju broj kanala. Dekoder se sastoji od blokova koji dimenzijama odgovaraju blokovima kodera i koji naduzorkuju mapu značajki do pune rezolucije. Između odgovarajućih blokova kodera i dekodera mreža definira preskočne veze koje prenose cijele mape značajki određenog sloja. One, prilikom rekonstrukcije originalnih dimenzija, mapama značajki koje sadrže apstraktne semantičke informacije daju informacije o prostornom položaju objekata.

Mreža SegNet [4] također koristi simetričnu arhitekturu, ali umjesto prijenosa cijelih mapi značajki putem preskočnih veza, prenose se samo pozicije piksela s najvećim vrijednostima iz slojeva sažimanja maksimalnom vrijednošću. Takav pristup smanjuje memorijske troškove, a u dekoder prenosi znanje koje omogućava kvalitetne rezultate.

Prepoznavanje semantičkih razreda predstavlja zahtjevniji i složeniji zadatak od povećanja rezolucije ekstrahiranih značajki na punu rezoluciju ulazne slike uz fino ocrtavanje granica. Zbog toga noviji modeli umjesto simetrične arhitekture koriste asimetričnu, na način je koder snažna osnova za klasifikaciju, a dekoder pojednostavljen ali efikasan. Time se smanjuje broj parametara i operacija te ubrzava učenje i zaključivanje modela.

3.5. Modeli za obradu u stvarnom vremenu

Mnoge primjene semantičke segmentacije zahtijevaju obradu slike u realnom vremenu (engl. *real-time*). Obradom u realnom vremenu smatra se obrada minimalno 30 slika u sekundi (engl. *frames per second, fps*).

Jedna je od prvih arhitektura koja je postigla brzinu zaključivanja u stvarnom vremenu je mreža ENet [30]. Njena preciznost bila je znatno manja od ostalih modela (58.3% mIoU na ispitnom podskupu skupa podataka Cityscapes, učena na punoj rezoluciji), međutim jednostavnijom arhitekturom na grafičkoj jedinici NVIDIA Titan X uspijevala je obraditi 47 slika rezolucije 1280×720 u sekundi. Zbog ovisnosti prostornih dimenzija slike s vremenom obrade, u prvim slojevima mreža nastoji znatno smanjiti njene dimenzije, dok u ostatku mreže minimizira sažimanje radi očuvanja prostornih informacija. Sažimanje slike predstavlja kompresiju redundantnih prostornih informacija u efikasnu reprezentaciju. Za razliku od prethodno spomenutog simetričnog SegNet-a, arhitektura mreže ENet sastoji se od većeg kodera i manjeg dekodera. Ovakav omjer veličina proizlazi iz toga da bi koder trebao odgovarati arhitekturama za klasifikaciju, dok je zadaća dekodera samo fino podešavanje rezultata kodera. Mreža u određenim slojevima koristi prostorno razdvojive konvolucije i vlastite rezidualne blokove te ne koristi povratne veze. ERFNet [32] predlaže arhitekturu sličnu mreži

ENet. Postiže bolju preciznost uz sličnu brzinu obrade koristeći prostorno razdvojive konvolucije s dilatacijom u svakom rezidualnom bloku. Mreža DG2S* [36] predstavlja nadogradnju ERFNet-a koja rezultira s bržim zaključivanjem modela uz blagi pad preciznosti. Posjeduje identičnu arhitekturu kao i ERFNet, međutim umjesto prostorno razdvojivih modula koristi odgovarajuće dubinski razdvojive module. Dodatni iskorak u smanjenju vremena obrade slike mreža postiže uz grupne konvolucije uz miješanje kanala. ESPNet [27] ne poboljšava značajno preciznost u odnosu na ERFNet, ali smanjuje broj parametara za više od 5 puta. To uspijeva uvođenjem efikasne prostorne piramide koja uz puno manji broj parametara i manje memorijske zahtjeve postiže efektivno receptivno polje jednako puno zahtjevnijem ASPP-u. Efektivna prostorna piramida zamjenjuje klasične konvolucijske slojeve, ona reducira broja filtera te primjenjuje konvolucije s različitim dilatacijskim faktorima nad reduciranom mapom značajki. Konvolucije se izvode paralelno i na kraju konkateniraju te zbrajaju s rezidualnom vezom.

Mreža ICNet [43] zaključivanje u stvarnom vremenu s visokom preciznošću postiže višerezolucijskim pristupom. Sastoji se od tri grane, od kojih svaka obrađuje ulaznu sliku različite rezolucije (slika 3.5). Prve dvije grane koriste sliku umanjenu dva odnosno četiri puta, dijele parametre i izvode se istovremeno. Obje grane odgovaraju PSPNet [42] arhitekturi. Treća grana koristi sliku pune rezolucije i sastoji se od puno manjeg broja operacija. Model za razliku od drugih višerezolucijskih pristupa prvo spaja izlaze prve dvije grane te zatim njih s izlazom posljednje grane. GUN [26] također koristi višerezolucijski pristup: sastoji se od dvije grane koje obrađuju različite rezolucije slike. Tijekom naduzorkovanja, umjesto operacija bez parametara poput bilinearne interpolacije ili najbližeg susjeda, predlaže modul za naduzorkovanje koji je definiran parametrima koji se uče s ostatkom modela. Modul za naduzorkovanje koristi prostorne informacije ulazne slike i tako potpomaže fino podešavanje granica između objekata. DFANet [23] predlaže komunikaciju između različitih grana mreže na dva načina. Prvi je komunikacija između razina koja rezultat i-te grane mreže daje na raspolaganje ulazu $i + 1$ grane. Drugi način je komunikacija između jednakih slojeva različitih grana. Model koristi i potpuno povezanu pažnju na kraju razinskih grana kako bi povećao receptivno polje uz zanemariv računalni trošak.

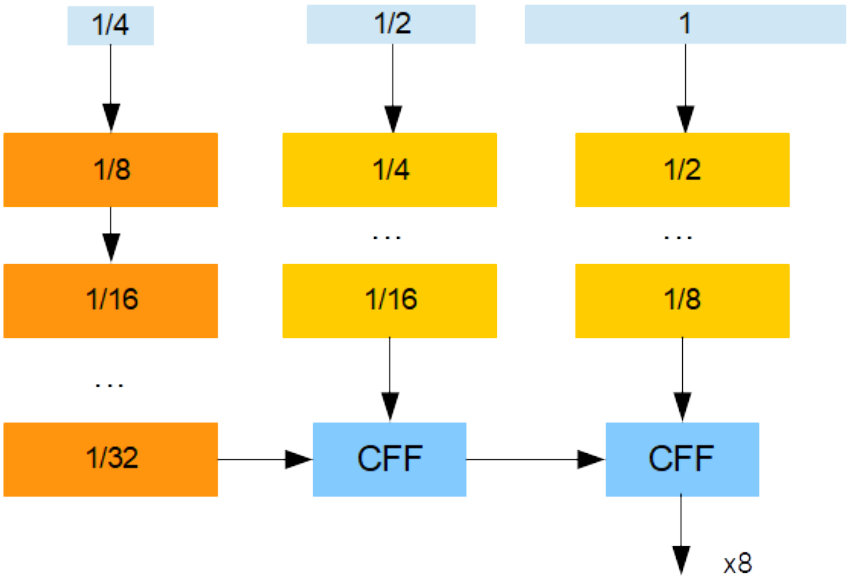
Neki modeli nastoje spriječiti gubitak prostornih informacija i smanjenje receptivnog polja tako da mrežu strukturiraju u dvije grane: grana koja izlučuje bogate prostorne informacije slike te grana koja smanjuje mapu značajki i ekstrahira globalne kontekstualne informacije. Arhitektura BiSeNet [41] je strukturirana u dvije grane koje međusobno ne dijele informacije već se konkateniraju u kasnijim slojevima. Prije

konkatenacija rezultat grane koja ekstrahira kontekst globalno se sažima i računa se vektor pažnje koji pospješuje učenje modela. Mreža DDRNet [16] nadograđuje strukturu tako da grane dijele ranije slojeve, a u kasnijim slojevima međusobno razmjenjuju informacije (slika 3.6). Umjesto korištenja pažnje, kontekst se upotpunjuje piramidalnim sažimanjem. Po mojim saznanjima, u trenutku pisanja rada, ovaj model ostvaruje najbolji omjer preciznosti segmentacija i brzine inferencije. Na skupu podataka Cityscapes, model učen na slikama pune rezolucije postiže 77.4% mIoU. Brzina obrade slika iz skupa Cityscapes pune rezolucije, na grafičkoj jedinici GTX 2080Ti, iznosi 108.8 okvira u sekundi. Usporedbe radi, model BiSeNet učen na slikama veličine $3/4$ od pune rezolucije na istom podskupu postiže preciznost od 74.7% mIoU.

Autori STDC [10] mreže smatraju da korištenje osnove za klasifikaciju nije efikasno za segmentaciju s obzirom na to da su one specifične samo za taj zadatak te da je dodavanje nove grane za mapiranje prostornih informacija redundantno i usporava model. Zbog toga se njihova arhitektura sastoji samo od jedne grane čiji su raniji slojevi zaduženi za prostorne informacije i detalje, a kasniji slojevi za apstraktne značajke. Model tijekom treniranja nastoji specijalizirati ranije slojeve da pronalaze detalje na slici pomoću gubitka detalja. On se računa usporedbom izlaza ranijih slojeva s točnom segmentacijskom mapom (engl. *ground-truth*) koja je konvoluirana s Laplaceovom jezgrom koja generira detaljne granice objekata na slici. Kasniji slojevi, zaduženi za pronalazak konteksta, sastoje se od blokova koji konkateniraju značajke različitih razina i tako povećavaju receptivno polje.

DF-Seg [24] mreže postižu vrlo dobar omjer preciznosti i vremena obrade slike. Njihove arhitekture dobivene su pretragom prostora različitih konvolucijskih arhitektura uz određena ograničenja koja optimiziraju omjer preciznosti i vremena obrade metodom rezanja ravnine. Autori HyperSeg-M [28] predlažu arhitekturu hipermreža za semantičku segmentaciju. Hipermreža se sastoji od kodera čije se težine optimiziraju tijekom učenja, ali tijekom predikcije su stalne, te od dekodera čije se težine ne uče već dinamički određuju tijekom predikcije. ShelfNet [44] uvodi strukturu mreže koja podsjeća na police (engl. *shelf*), a sastoji se od nekoliko umreženih stupaca koji naizmjenično imaju ulogu kodera i dekodera. Zbog dobre povezanosti i boljeg protoka informacija mreža djeluje kao ansambl više mreža.

Usporedba izvedbi spomenutih modela za semantičku segmentaciju u stvarnom vremenu učenih na skupu podataka Cityscapes prikazana je u tablici 3.1. Za svaki model dostupna je godina objave, veličina slike koju model prima na ulazu, preciznost, brzina obrade te broj parametara.



Slika 3.5: Višerezolucijska arhitektura mreže ICNet [43]. Obrađuje sliku različitih rezolucija od kojih slika najmanje rezolucije prolazi kroz složene konvolucijske slojeve, a slike veće rezolucije kroz jednostavne slojeve kako bi se izračun pojednostavnio. Rezultati prolaska kroz različite grane stapaju se modulom za fuziju kaskadnih značajki (engl. *cascade feature fusion, CFF*)

3.6. Brza pažnja u preskočnim vezama

Operacija konvolucije računa se u lokalnom susjedstvu jedinice slike, zbog čega je za prepoznavanje relacija između udaljenijih dijelova slike potrebno primijeniti nekoliko uzastopnih operacija. Povećanje receptivnog polja uzastopnim konvolucijama je računalno zahtjevno i stvara optimizacijske poteškoće zbog propagacije podataka kroz veliki broj slojeva. Efikasnija alternativa ovakvom pristupu su globalne operacije [39] koje za određenu poziciju računaju odziv kao težinsku sumu značajki sa svih pozicija mape značajki. Općenita globalna operacija u dubokim neuronskim mrežama u vektorskom obliku definira se kao [39]:

$$\boldsymbol{y}_i = \frac{1}{N(\boldsymbol{x})} \sum_j f(\boldsymbol{x}_i, \boldsymbol{x}_j) g(\boldsymbol{x}_j)$$

gdje $\boldsymbol{y}_i$ predstavlja poziciju i izlaza globalne operacije koji se računa pomoću svih mogućih pozicija ulaza j. Vektor $\boldsymbol{x}$ predstavlja značajke dok funkcija f računa stupanj relacije između pojedinih elemenata ulaza. Funkcija g računa reprezentaciju značajke

Tablica 3.1: Izvedbe modela za semantičku segmentaciju u stvarnom vremenu učenih na skupu podataka Cityscapes. Prikaz je naziv modela, godina objave članka, rezolucija ulaznih slika tijekom eksperimenata (izražen kao postotak pune rezolucije 1024×2048), dosegnuta preciznost na ispitnom podskupu (mIoU), brzina (fps) te broj milijuna parametara. Brzina svih modela prikazana je za grafičku jedinicu GTX 1080Ti, osim za (a) FANet-34 čija je brzina izmjerena na jedinici Titan X te (b) DDRNet-23-Slim čija je brzina izmjerena na jedinici GTX 2080Ti.

Model	Godina	Rez.	mIoU (%)	Brzina (fps)	Param. (mil.)
ENet	2016	1	58.3	35.4	1.4
ERFNet	2017	1/2	68.0	18.4	20.0
ESPNet	2018	1/2	60.3	108.7	0.4
ICNet	2018	1	69.5	49.7	-
GUN	2018	1/2	70.4	33.3	-
DG2S	2018	1/2	70.6	-	1.2
BiSeNet	2018	1	74.7	65.5	49.0
DFANet A	2019	1	71.3	89.3	7.8
ShelfNet18-lw	2019	1	74.8	36.9	-
DF2-Seg2	2019	1	75.3	56.3	-
SwiftNetRN-18pyr	2020	1	76.4	34.0	12.0
FANet-34	2020	1	75.5	58.0	-
HyperSeg-M	2021	1/2	75.8	36.9	10.1
STDC2-Seg75	2021	3/4	76.8	97.0	-
DDRNet-23-Slim	2021	1	77.4	108.8	5.7

na poziciji j, a cijeli je izraz normaliziran funkcijom $N(\boldsymbol{x})$. Za razliku od potpuno povezanih slojeva, globalne operacije ne koriste učene parametre već samo različite lokacije mapi značajki te ne zahtijevaju stalnu veličinu ulaza.

Mehanizam pažnje (engl. *self-attention*) [37, 39] je specifičan slučaj globalne operacije koji se pokazao uspješnim za razne zadatke računalnog vida. Njegova mogućnost da izdvoji globalni kontekst iz mape značajki poboljšala je preciznost modela koji ga koriste. Definiran je mapama upita $Q \in \mathbb{R}^{n \times c'}$ te skupom parova ključeva $K \in \mathbb{R}^{n \times c'}$ i vrijednosti $V \in \mathbb{R}^{n \times c}$ koji se dobivaju projekcijom ulazne mape značajki $X \in \mathbb{R}^{n \times c}$ pomoću konvolucije jezgrom veličine 1×1. Nakon izračuna tri spomenute mape značajki, samopažnja se računa kao težinska suma vrijednosti čije se težine računaju funkcijom relacije upita s odgovarajućim ključem: $Y = f(Q, K) \cdot V$. Funkcija f naziva se afinitet i modelira povezanost svakog para piksela. Najčešće se afinitet nalazi

u obliku funkcije softmaks te se u tom slučaju samopažnja računa pomoću jednadžbe:

$$Y = \text{softmaks}(Q \cdot K^T) \cdot V$$

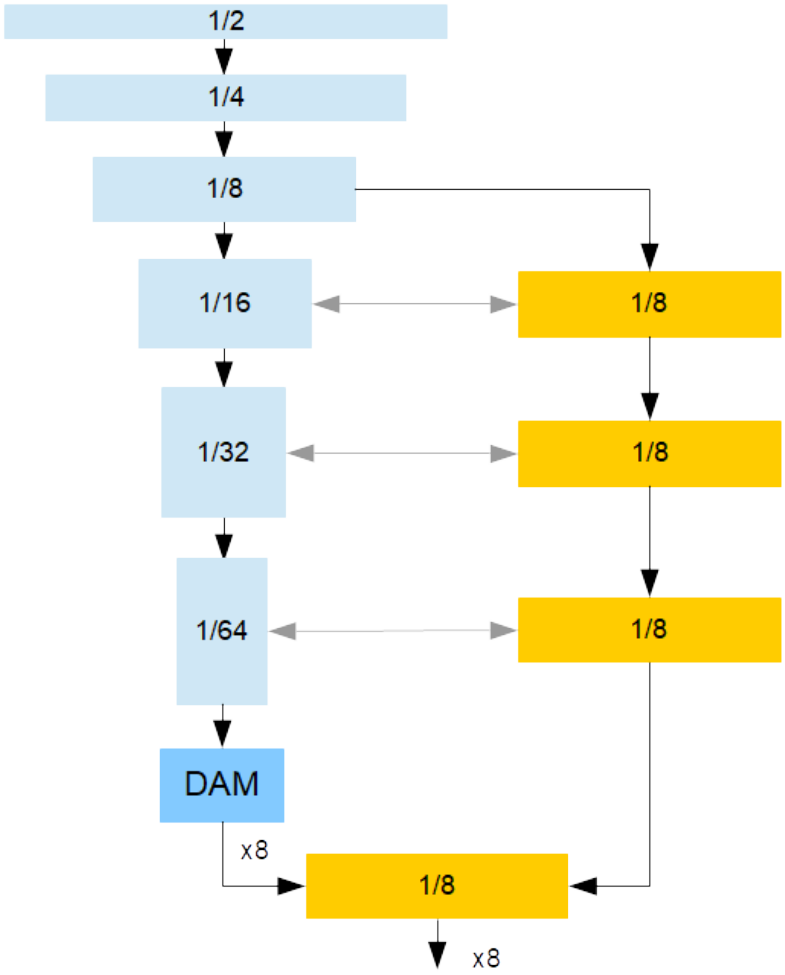
Složenost algoritma ovisna je o prostornim dimenzijama mape značajki $n = sirina \times visina$ te o broju kanala c. Zbog postojanja normalizacijskog člana u funkciji softmaks, za izračun samopažnje potrebno je prvo matrično pomnožiti unutarnje elemente Q i K te nakon toga rezultat pomnožiti s mapom vrijednosti V. Takav redoslijed operacija uzrokuje složenost $\mathcal{O}(n^2c)$. S obzirom na to da je veličina slike uobičajeno dosta velika, ovakva formulacija mehanizma je skupa i značajno utječe na vrijeme obrade slike, koje je za zadatak semantičke segmentacije u stvarnom vremenu izrazito bitno. Konvolucijska mreža FANet [17] nastoji smanjiti računalnu složenost tako da zamijeni matematičke operacije mehanizma, čime postiže složenost od $\mathcal{O}(nc^2)$. Takvu preinaku pažnje autori su nazvali brzom pažnjom (engl. *fast attention*), jer je brža čak $\frac{n}{c}$ puta, gdje je $n \gg c$. Autori mreže FANet pokazali su da zamjenom pažnje brzom pažnjom na skupu podataka Cityscapes preciznost ostaje jednaka, a brzina obrade slike se poveća čak 9 puta.

Kako bi se riješio problem složenosti matematičkih operacija samopažnje, brza pažnja mijenja afinitet iz softmaksa u jednostavno matrično množenje mapi upita i ključeva. Zbog svojstva asocijativnosti matričnog množenja, redoslijed izračuna može se zamijeniti i tako složenost smanjiti na $\mathcal{O}(nc^2)$:

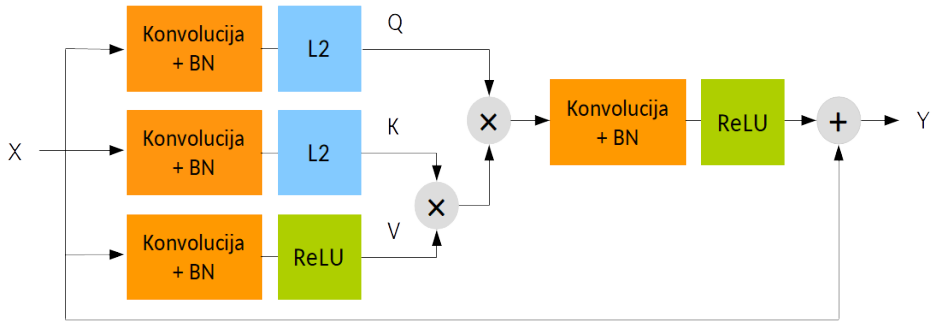
$$Y = (Q \cdot K^T) \cdot V = Q \cdot (K^T \cdot V)$$

S obzirom na to da mape upita i ključeva nastoje prikazati korelaciju između piksela, htjeli bismo da njihove vrijednosti ostanu u rasponu smislenih brojeva. Stoga ih je prije matričnog množenja potrebno normalizirati na raspon $[-1, 1]$, što postižemo L2 normalizacijom uzduž dimenzije kanala. Također, kako bi očuvali negativne vrijednosti, nakon konvolucije kojima dobivamo ove dvije mape, ne provodimo aktivaciju zglobnicom. Prikaz modula brze pažnje nalazi se na slici 3.7.

Modul slične složenosti kao brza pažnja, predstavljen u [12], naziva se kanalna pažnja (engl. *channel attention*). Mapa pažnje $A \in \mathbb{R}^{c \times c}$ računa se tako da se ulaz $X \in \mathbb{R}^{n \times c}$ matrično pomnoži sa svojom transponiranom inačicom te se rezultat provodi kroz funkciju softmaks. Zatim se mapa pažnje množi s ulazom X, skalira te pribraja ponovno ulazu X. Složenost operacija jednaka je $\mathcal{O}(nc^2)$, ali kanalna pažnja za razliku od brze pažnje za svaki piksel agregira samo međuzavisnosti između različitih kanala, ne i prostorne međuzavisnosti.



Slika 3.6: Dvograna arhitektura mreže DDRNet [16]. Sastoji se od dvije grane koje dijele ranije slojeve, zatim se dijele na granu konteksta i prostornih informacije te međusobno razmjenjuju informacije. Na kraju grane konteksta nalazi se modul za izvlačenje konteksta (engl. *deep aggregation pyramid pooling module, DAM*).



Slika 3.7: Struktura modula brze pažnje. Nad mapom značajki X provode se konvolucije s jezgrom veličine 1×1 i grupne normalizacije (BN) koje daju mape upita Q, ključeva K i vrijednosti V. Mape upita i ključeva se normaliziraju kako bi bile u rasponu od -1 do 1. Ovakva struktura omogućuje efikasniji izračun pažnje sa složenošću od $\mathcal{O}(nc^2)$.

4. Uvođenje brze pažnje u model SwiftNet

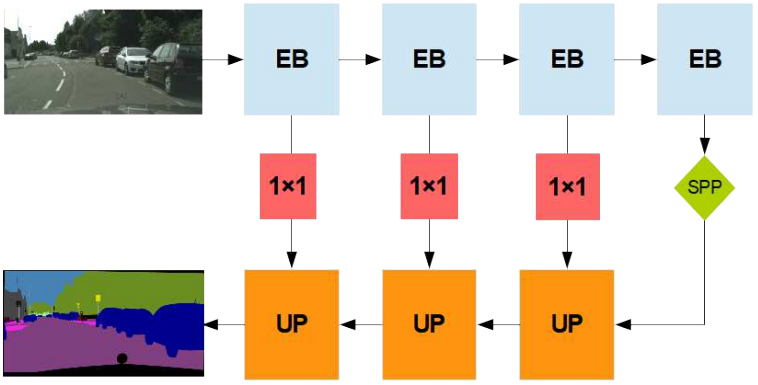
4.1. Model SwiftNet

Model SwiftNet-RN18pyr [29] svojedobno je postigao najbolju preciznost semantičke segmentacije u stvarnom vremenu na skupu podataka Cityscapes, postigavši 76.4% mIoU na ispitnom podskupu. Jednostavna arhitektura modela čini ga pogodnim za korištenje i na ugradbenim računalnim sustavima. Postoje dvije varijante modela koje se razlikuju po broju različitih razina slike koja se obrađuje. Prvi je jednorazinski, a drugi višerazinski model.

4.1.1. Jednorazinski model

Jednorazinski model SwiftNet koristi samo originalnu rezoluciju slike kao ulaz modela. Sastoji se od kodera za raspoznavanje (engl. *recognition encoder*) i dekodera za naduzorkovanje (engl. *upsampling decoder*). Koder za raspoznavanje koristi arhitekture ResNet-18 i MobileNet V2 za koje postoje javno dostupni parametri dobiveni učenjem nad velikim brojem slika iz različitih skupova podataka. Obje mreže umjereno su duboke i sastoje se od rezidualnih blokova, zbog čega su pogodne za probleme obrade slike u stvarnom vremenu. Koder se sastoji od četiri bloka koji postepeno smanjuju rezoluciju i ekstrahiraju bitne značajke. Svaki od njih sastoji se od nekoliko rezidualnih konvolucijskih blokova. Izlaz iz svakog bloka predstavlja ujedno ulaz u idući blok i preskočnu vezu prema sloju za naduzorkovanje. Dekoder za naduzorkovanje sastoji se od sekvence blokova koji provode tri koraka: reprezentacija niske rezolucije se bilinearno naduzorkuje, zatim se zbraja s preskočnim vezama te rezultat stapa provođenjem konvolucije. Svi blokovi za naduzorkovanje dijele jednak broj filtera. Izlaz iz posljednjeg bloka kodera prolazi kroz modul prostornog piramidalnog sažimanja (SPP). Struktura jednorazinskog modela prikazana je na slici 4.1. Zbog razlike u složenosti

zadataka koji rješavaju koder i dekoder, arhitektura modela je asimetrična: blokovi kodera sastoje se od četiri (RN18), osam (RN34) ili više konvolucija dok se blokovi dekodera sastoje samo od jedne.



Slika 4.1: Struktura jednorazinskog modela SwiftNet. Sastoji se od kodera za raspoznavanje koji se sastoji od konvolucijskih blokova (EB), te jednostavnog dekodera za naduzorkovanje, koji se sastoji od blokova koji naduzorkuju mapu značajki te je stapaju s preskočnim vezama (UP). Rezultat posljednjeg bloka kodera prolazi kroz modul prostornog piramidalnog sažimanja (SPP).

.

4.1.2. Višerezolucijski model

Višerezolucijski SwiftNet model izvlači semantičke značajke iz nekoliko različitih razina rezolucijske piramide. Svaka razina piramide koristi sliku rezolucije dvostruko manju u odnosu na prethodnu razinu. Kako bi se smanjila mogućnost prenaučenosti i postigla djelomična ekvivarijantnost na mjerilo, sve razine piramide dijele iste parametre. Broj dodatnih izračuna koje višerezolucijski model zahtjeva ne prelazi 33% ($\frac{1}{4} + \frac{1}{16} + \frac{1}{64} + ... = \frac{1/4}{1-3/4} = 1/3$). Mape značajki različitih razina piramide zbrajaju se, zbog čega model djeluje kao ansambl manjih, nezavisnih modela.

Višerezolucijski model se sastoji od kodera i dekodera, jednako kao i njegova jednorazinska varijanta. Koder za raspoznavanje svaku razinu provodi kroz zasebne blokove koji dijele parametre, čiji se izlazi odgovarajućih dimenzija zbrajaju međusobno i naposljetku s odgovarajućim blokom dekodera. Zbrajanje mapa značajki različitih razina i izlaza bloka naduzorkovanja naziva se piramidalnom fuzijom (engl. *pyramid*

fusion). Za razliku od jednorazinske inačice, ovaj model ne koristi modul SPP jer veće receptivno polje postiže višerezolucijskim pristupom.

S obzirom na to da dimenzije reprezentacije manjih slika rezolucijske piramide prolaskom kroz koder za raspoznavanje postaju sve manje, stvara se mogućnost neidentificiranja manjih objekata na slici. Višerezolucijski SwiftNet model pokušava riješiti taj problem prioritiziranjem piksela koji se nalaze na semantičkim granicama i uvodi gubitak svjestan granica (engl. *boundary-aware loss*). Takav gubitak za svaki piksel množi negativnu log-izglednost s parametrom koji je obrnuto proporcionalan udaljenosti tog piksela od semantičke granice:

$$L^{ij} = -\alpha^{ij} e^{\gamma(1-p_t^{ij})} \log p_t^{ij}$$

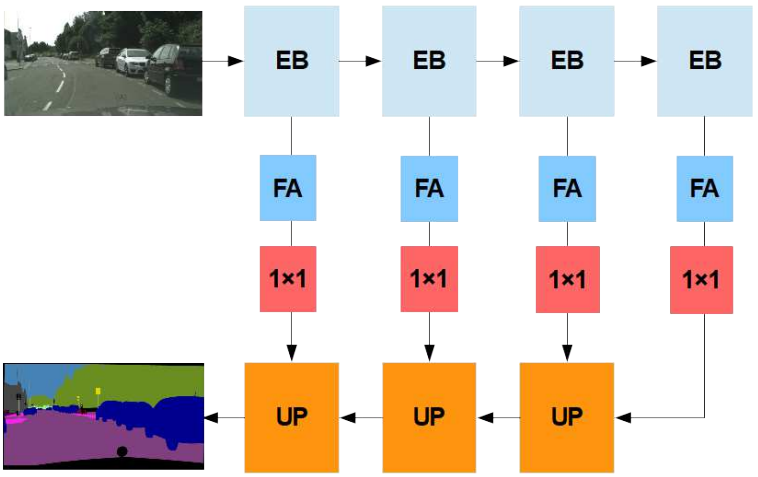
U prethodnom izrazu L^{ij} predstavlja gubitak piksela slike na poziciji (i, j), parametar α^{ij} za određeni piksel regulira jačinu gubitka ovisno o njegovoj udaljenosti od granice objekta kojem pripada, p_t^{ij} predstavlja vjerojatnost pripadanja piksela stvarnom razredu t koju je dao model, a faktor $e^{\gamma(1-p_t^{ij})}$ nastoji pojačati vrijednost gubitka za loše klasificirane piksele.

4.2. Implementacija brze pažnje u model SwiftNet

Model SwiftNet implementiran je korištenjem programskog okvira PyTorch, stoga je i nadogradnja koju razmatra ovaj rad također pisana u istom okviru. PyTorch [31] je okvir koji nudi funkcionalnosti za implementaciju algoritama strojnog učenja, često korišten u računalnom vidu i obradi prirodnog jezika. Okvir nudi mogućnost ubrzavanja složenih računskih operacija s matricama korištenjem grafičkog procesora. Za pohranu višedimenzionalnih polja podataka, PyTorch definira razred koji se naziva Tensor. Njegova struktura i funkcionalnost slična je NumPy-jevom nizu, a njega je moguće koristiti u operacijama na platformi CUDA. Za razliku od NumPy-jevh polja, tenzori PyTorch-a podržavaju automatizirane diferencijacije. Metoda koristi skup tehnika za numerički izračun derivacija složenih funkcija, a temelji se na njihov rastav na slijed elementarnih funkcija čije su derivacije poznate i primjeni pravila ulančavanja kod određivanja derivacije kompozicije funkcija.

Moduli brze pažnje u model SwiftNet uvedeni su na preskočne veze između kodera i dekodera, prije prolaska kroz projekcije jezgrama 1×1 koje smanjuju dimenzionalnost mapi značajki. Piramidalno prostorno sažimanje na kraju kodera za prepoznavanje također je zamijenjeno modulom brze pažnje. Preinaka strukture jednorazinskog

modela prikazana je na slici 4.2.



Slika 4.2: Model SwiftNet s uvedenom brzom pažnjom na preskočne veze. Moduli brze pažnje (FA) prethode projekcijama konvolucijama 1×1 čiji su izlazi spojeni s blokovima za naduzorkovanje. Brza pažnja također mijenja modul SPP iz originalne strukture SwiftNet-a.

.

U višerezolucijskom modelu moduli brze pažnje ukomponirani su na jednak način kao i u jednorazinskom, s time da različite razine dijele iste parametre. U isječku koda 4.1 prikazan je implementacija modula brze pažnje čija je funkcionalnost opisana u poglavlju 3.6. Instanca modula definirana je brojem ulaznih kanala c koja je jednaka broju izlaznih kanala te broju unutarnjih kanala c' koji se koriste kod izračuna mapa ključeva K i upita Q.

Isječak koda 4.1: Implementacija modula brze pažnje.

```
class FastAttention(nn.Module):
  def __init__(self, in_channels, fa_channels=32):
    super(FastAttention, self)._init_()
    self.channels = fa_channels
    self.qs = _ConvBNRelu(in_channels, self.channels,
        k=1, batch_norm=True, activation=False)
    self.ks = _ConvBNRelu(in_channels, self.channels,
        k=1, batch_norm=True, activation=False)
    self.vs = _ConvBNRelu(in_channels, in_channels,
        k=1, batch_norm=True, activation=True)
    self.smooth = _ConvBNRelu(in_channels, in_channels,
        k=1, batch_norm=True, activation=True)

  def forward(self, x):
    query = self.qs(x)
    key = self.ks(x)
    value = self.vs(x)

    N, C, H, W = x.size()

    query_plain = query.view(N, self.channels,-1)
                       .permute(0, 2, 1)
    query = F.normalize(query_plain, p=2, dim=2, eps=1e-12)

    key_plain = key.view(N, self.channels,-1)
    key = F.normalize(key_plain, p=2, dim=1, eps=1e-12)

    value = value.view(N,C,-1).permute(0, 2, 1)

    f = torch.matmul(key, value)
    y = torch.matmul(query, f)
    y = y.permute(0, 2, 1).contiguous()

    y = y.view(N, C, H, W)
    smoothed = self.smooth(y)

    return x + smoothed
```

5. Eksperimenti

5.1. Google Colaboratory

Google Colaboratory [2] je platforma u oblaku (engl. *cloud platform*) koja dopušta pisanje i izvršavanje Python koda unutar preglednika, s besplatno dostupnim grafičkim jedinicama. Dopušta sinkronizaciju i interakciju s dokumentima na platformi Google Drive, što omogućuje jednostavno korištenje većih skupova podataka tijekom izvršavanja koda. Colaboratory korisnicima dodijeljuje jednu od dostupnih grafičkih jedinica: Nvidia K80, T4, P4, V100 ili P100. U svim eksperimentima u ovom radu korištena je spomenuta platforma.

5.2. Učenje na skupu podataka Cityscapes

Skup podataka Cityscapes [8] predstavlja kolekciju slika urbanih uličnih scena. Snimljene su kroz 50 različitih njemačkih i švicarskih gradova iz perspektive vozača, kroz različita godišnja razdoblja i razne vremenske prilike. Na njima se nalazi velik broj dinamičnih objekata s raznim pozadinama. Podskup koji je pogodan za učenje modela za semantičku segmentaciju, sastoji se od 2975 precizno označenih slika za učenje, 500 za validaciju te 1525 za testiranje. Slike su veličine 1024×2048 piksela, a svaki od piksela razvrstan je u jedan od 19 mogućih razreda.

Parametri modela najčešće se zapisuju u obliku IEEE formata jednostruke preciznosti FP32. Numerički zapisi u formatima manje preciznosti zahtijevaju manje računalnih resursa što omogućuje učenje većih neuronskih mreža. Pored toga, prijenos podataka kroz model i matematičke operacije su znatno brže. Učenje u miješanoj preciznosti [1] omogućuje ubrzanje izvršavanjem matematičkih operacija u polovičnoj preciznosti FP16, dok se težine čuvaju u FP32 formatu kako se ne bi izgubila preciznost ključnih dijelova mreže. S obzirom da format polovične preciznosti FP16 obuhvaća manji spektar brojeva, problem nastaje kod vrlo malenih vrijednosti gradijenata

koje lako iščeznu prolaskom unatrag kroz mrežu. Kako se vrijednost gradijenta ne bi izgubila, gubitak se skalira određenom vrijednošću i zatim se prije ažuriranja težina skalira recipročnom vrijednošću, kako bi se povratio točan iznos gradijenta. Radni okvir PyTorch omogućuje učenje s miješanom preciznošću u paketu torch.cuda.amp, uz automatski odabir prikladnog faktora skaliranja gubitka. Eksperimenti su prvo provedeni učenjem u jednostrukoj preciznosti, a nakon toga je model s brzom pažnjom učen u miješanoj preciznosti s različitim hiperparametrima.

5.2.1. Učenje s jednostrukom preciznošću

Nad skupom podataka Cityscapes isprobali smo reproducirati rezultate iz rada [17] na način da brzu pažnju uvedemo u preskočne veze jednorazinskog modela SwiftNetRN-18 [29], kako je objašnjeno u prethodnog poglavlju. Isprobali smo učenje s hiperparametrima koji su definirani u radu, međutim neki ključni parametri (broj epoha, veličina isječka, faktor finog podešavanja) nisu bili definirani stoga su odabrane vrijednosti koje su se činile pogodnima. Pokušaji reproduciranja prikazani su u tablici 5.1. U prvom su retku prikazani rezultati iz članka [17], a u ostalim retcima rezultati eksperimenata iz ovog rada. S obzirom na to da učenje s hiperparametrima spomenutima u članku nije dalo dobre rezultate, idući eksperimenti koristili su različite stope učenja, broj epoha, optimizacijski algoritam te planer stope učenja. Model prikazan u posljednjem retku tablice dao je najbolju preciznost koja iznosi 73.57% mIoU.

S hiperparametrima kojima je dobiven najprecizniji model učen je i trorazinski SwiftNetRN-18 pyr model s brzom pažnjom u preskočnim vezama. Pored njega, usporedbe radi, učeni su i jednorazinski i trorazinski SwifNet modeli bez brze pažnje. Svi modeli učeni su pomoću optimizatora Adam. Tijekom eksperimentiranja, pokazalo se da modeli s brzom pažnjom sporije uče, stoga je stopa učenja za osnovne modele inicijalno postavljena na $lr_{max} = 4 \times 10^{-4}$, dok je za modele s brzom pažnjom postavljena na $lr_{max} = 7 \times 10^{-4}$. Stopa učenja lr_t postepeno opada metodom kosinusovog kaljenja (engl. *cosine annealing*) bez korištenja ponovnog pokretanja:

$$lr_t = lr_{min} + \frac{1}{2}(lr_{max} - lr_{min})(1 + \cos{(\frac{epoch_t}{epoch_{max}}\pi)})$$

Vrijednost stope učenja pada sve dok ne dosegne vrijednost od $lr_{min} = 10^{-6}$. Parametri kodera za raspoznavanje koji su prednaučeni na skupu ImageNet za ažuriranje parametara u svakom trenutku koriste 4 puta manju stopu od onih koju koriste nasumično inicijalizirani parametri. Tijekom cijelog učenja faktor L2 regularizacije postavljen je na $5 \cdot 10^{-4}$. Osnovni modeli uče kroz 250, dok nadogradnje s brzom pažnjom uče kroz

Tablica 5.1: Eksperimenti s brzom pažnjom u jednorazinskom SwiftNet modelu. U prvom retku prikazani su rezultati opisani u članku [17]. U drugom retku prikazani su rezultati reproduciranja opisanog eksperimenta, uz pretpostavljene vrijednosti nepoznatih hiperparametara. U ostalim eksperimentima, koji su rezultirali boljim modelima, korištene su različite početne stope učenja, optimizacijski algoritam te broj epoha. U tablici stupac *grupa* predstavlja veličinu grupe, *optim* optimizacijski algoritam, *lr init* početnu stopu učenja, *ft* faktor finog podešavanja, *lr planer* planer stope učenja. Korišteni planeri stope učenja su multiplikativni planer (multi) te kosinusovo kaljenje (kos). Model SwiftNetRN-18 FA zapisan je u kraćem obliku kao SNRN18FA.

Model	grupa	optim	lr init	ft	lr planer	isječak	epohe	mIoU (%)
FANet	16	SGD	1e-2	-	multi	-	-	75.00
SNRN18FA	16	SGD	1e-2	1/4	multi	768	250	65.59
SNRN18FA	14	Adam	5e-3	1/4	kos	768	350	65.60
SNRN18FA	14	Adam	4e-4	1/4	kos	768	250	72.61
SNRN18FA	14	Adam	7e-4	1/4	kos	768	350	**73.57**

350 epoha. Učenje se odvija nad kvadratnim isječcima slika veličine 768×768 s veličinom mini-grupe postavljenom na 14. Isječci slika prije ulaska u model nasumično se zrcale i skaliraju kako bi se pojačala regularizacija modela.

U modulima brze pažnje broj unutarnjih kanala postavljen je na $c' = 32$ jer je taj broj autorima modela FANet dao optimalnu preciznost uz manji broj parametara. Rezultati evaluacije modela nad podskupom za validaciju prikazani su u tablici 5.2. Modeli bez brze pažnje dali su bolje rezultate od modela s brzom pažnjom u preskočnim vezama. Pretpostavka je da bi modeli s brzom pažnjom trebali proizvesti preciznije rezultate, ali da su zahtjevniji za naučiti.

5.2.2. Učenje s miješanom preciznošću

S obzirom na to da se učenje modela s brzom pažnjom pokazalo kompliciranim, u idućem eksperimentu isprobana je tehnika učenja s miješanom preciznošću. Petlja učenja upotpunjena je tako da se konfiguracijski može odrediti hoće li se učenje odvijati u jednostrukoj ili miješanoj preciznosti. U isječku koda 5.1 prikazana je razlika u učenju s dvije vrste preciznosti. Učenje u miješanoj preciznosti omogućilo je povećanje veličine mini-grupe te veličine isječka.

U tablici 5.3 prikazani su rezultati dobiveni učenjem modela s različitim hiperpa-

Tablica 5.2: Izvedbe modela SwiftNet sa i bez brze pažnje učenih na skupu podataka Cityscapes. Prikazan je naziv modela, dosegnuta preciznost na validacijskom podskupu (mIoU), brzina obrade slike (fps) te broj milijuna parametara. Svi modeli su trenirani nad slikama pune rezolucije te koriste parametre kodera koji su predtrenirani na skupu podataka ImageNet. Brzina je izmjerena na grafičkoj jedinici Tesla P100-PCIE-16GB.

Model	mIoU (%)	Brzina (fps)	Parametri (mil.)
SwiftNetRN-18	**75.31**	34.4	11.8
SwiftNetRN-18 pyr	**75.61**	26.0	12.1
SwiftNetRN-18 FA	73.57	27.6	12.7
SwiftNetRN-18 FA pyr	70.95	20.0	12.8

rametrima u miješanoj preciznosti. U usporedbi s modelima učenim u jednostrukoj preciznosti FP32, modeli s istim hiperparametrima učeni u miješanoj preciznosti proizveli su lošije rezultate. Povećanje veličine isječka doprinijelo je preciznosti modela s brzom pažnjom te je najveća preciznost postignuta za veličinu grupe 14 te isječak veličine 1024×1024 i iznosi 74.59% mIoU. Zanimljivo je primijetiti povećanje brzine prilikom smanjenja preciznosti brojeva s pomičnim zarezom. Model s brzom pažnjom učen u miješanoj preciznosti s istim hiperparametrima postigao je brzinu od $12.69 fps$ u odnosu na brzinu referentnog modela od $7.11 fps$, što predstavlja povećanje za 78%.

Isječak koda 5.1: Prilagodba za učenje s miješanom preciznošću.

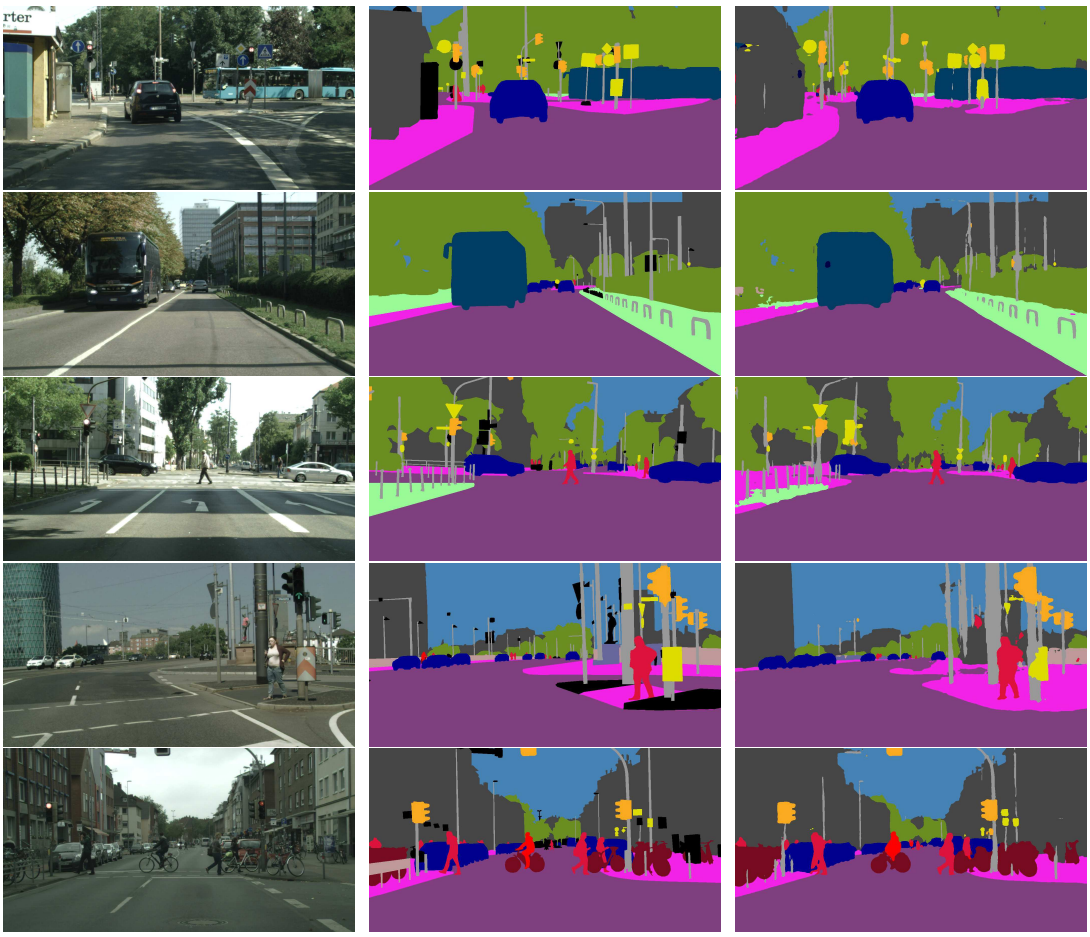
```
for step, batch in batch_iterator:
    if self.scaler is not None:
        with torch.cuda.amp.autocast():
            loss = self.model.loss(batch)
        self.scaler.scale(loss).backward()
        self.scaler.step(self.optimizer)
        self.scaler.update()
    else:
        loss = self.model.loss(batch)
        loss.backward()
        self.optimizer.step()
```

Na slici 5.1 prikazano je nekoliko slika iz validacijskog podskupa skupa podataka Cityscapes, njihove ispravne (engl. *ground-truth*) semantičke segmentacije te segmentacije koje je proizveo jednorazinski model SwiftNet s brzom pažnjom u preskočnim

Tablica 5.3: Izvedbe jednorazinskih modela SwiftNet sa i bez brze pažnje učenih na skupu podataka Cityscapes s različitim preciznostima, veličinama grupe i veličinama isječaka. Za svaki model prikazana je dobivena preciznost na validacijskom podskupu te brzina učenja. Modeli s brzom pažnjom učeni su na grafičkoj jedinici Tesla V100, dok su modeli bez brze oažnje učeni na jedinici Tesla P100.

			SwiftNetRN-18 FA (Tesla V100)		SwiftNetRN-18 (Tesla P100)	
Preciznost	Grupa	Isječak	mIoU(%)	Brzina (fps)	mIoU(%)	Brzina (fps)
FP32	14	768	73.57	7.11	**75.31**	7.05
miješana	14	768	70.93	**12.69**	74.45	**12.67**
miješana	14	1024	**74.59**	10.98	75.25	9.97
miješana	18	1024	74.37	9.71	75.23	9.66

vezama, koji je u prethodnom eksperimentu pokazao najbolje rezultate. Različite boje na slikama segmentacija označuju različite semantičke razrede na slikama.



Slika 5.1: Prikaz nekoliko slika iz validacijskog podskupa Cityscapes-a, njihovih ispravnih segmentacija te segmentacija dobivenih jednorazinskim SwiftNet modelom s brzom pažnjom u preskočnim vezama.

6. Zaključak

Semantička segmentacija slike u stvarnom vremenu zahtjevan je zadatak računalnog vida kome se u posljednje vrijeme pridaje velika pažnja zbog njegove široke primjene. U okviru ovog rada predstavljen je koncept dubokih konvolucijskih modela na kojem se temelje najuspješnija rješenja spomenutog zadatka. Prikazan je razvoj okosnica za klasifikaciju te su objašnjene najčešće arhitekture pogodne za semantičku segmentaciju. Opisana su rješenja koja nastoje pojednostavniti mrežu kako bi se obrada slike mogla izvršiti u realnom vremenu.

U model SwiftNet [29], u preskočne veze između kodera i dekodera, implementiran je mehanizam brze pažnje. Uvođenje mehanizma je autorima mreže FANet [17] dalo značajno poboljšanje u preciznosti uz malen dodatni broj parametara. U sklopu rada, u usporedbi osnovnog modela SwiftNet-a i modela s brzom pažnjom na skupu podataka Cityscapes [8], osnovni model je rezultirao boljom preciznošću. Razlog tome je vjerojatno složenost brze pažnje koja zahtijeva složenije učenje. U ovom radu također smo istražili i učenje modela za gustu predikciju u miješanoj preciznosti. Učenje u miješanoj preciznosti omogućilo je korištenje veće veličine isječaka (1024×1024) bez smanjenja veličine grupa, što je dovelo do istovremenog poboljšanja preciznosti (74.59% mIoU) i brzine (10.98 fps).

S obzirom na to da originalan članak o bržoj pažnji upućuje na to da uvođenjem brze pažnje model bolje generalizira, u budućem radu trebalo bi isprobati učenje s različitim hiperparametrima koji mogu pospješiti i ubrzati učenje. U ovom radu nismo uspjeli reproducirati eksperimente iz originalnog članka zbog nedostatka ključnih informacija o hiperparametrima tijekom učenja.

LITERATURA

[1] Mixed precision training. https://docs.nvidia.com/deeplearning/performance/mixed-precision-training/index.html. Pristupljeno: 02-07-2021.

[2] Google colaboratory. https://colab.research.google.com/. Pristupljeno: 01-07-2021.

[3] Image segmentation. https://en.wikipedia.org/wiki/Image_segmentation. Pristupljeno: 07-07-2021.

[4] Vijay Badrinarayanan, Alex Kendall, i Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation, 2016.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, i Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, 2017.

[6] Liang-Chieh Chen, George Papandreou, Florian Schroff, i Hartwig Adam. Rethinking atrous convolution for semantic image segmentation, 2017.

[7] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, i Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation, 2018.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, i Bernt Schiele. The cityscapes dataset for semantic urban scene understanding, 2016.

[9] Vincent Dumoulin i Francesco Visin. A guide to convolution arithmetic for deep learning, 2018.

[10] Mingyuan Fan, Shenqi Lai, Junshi Huang, Xiaoming Wei, Zhenhua Chai, Junfeng Luo, i Xiaolin Wei. Rethinking bisenet for real-time semantic segmentation, 2021.

[11] Clement Farabet, Camille Couprie, Laurent Najman, i Yann Lecun. Learning hierarchical features for scene labeling, 2013.

[12] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, i Hanqing Lu. Dual attention network for scene segmentation, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition, 2014.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Deep residual learning for image recognition, 2015.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Identity mappings in deep residual networks, 2016.

[16] Yuanduo Hong, Huihui Pan, Weichao Sun, Senior Member, IEEE, i Yisong Jia. Deep dual-resolution networks for real-time and accurate semantic segmentation of road scenes, 2021.

[17] Ping Hu, Federico Perazzi, Fabian Caba Heilbron, Oliver Wang, Zhe Lin, Kate Saenko, i Stan Sclaroff. Real-time semantic segmentation with fast attention, 2020.

[18] Gao Huang, Zhuang Liu, Laurens van der Maaten, i Kilian Q. Weinberger. Densely connected convolutional networks, 2018.

[19] Sergey Ioffe i Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

[20] Jonghoon Jin, Aysegul Dundar, i Eugenio Culurciello. Flattened convolutional neural networks for feedforward acceleration, 2015.

[21] Alex Krizhevsky, Ilya Sutskever, i Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks, 2012.

[22] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, i L. D. Jackel. Backpropagation applied to handwritten zip code recognition, 1989.

[23] Hanchao Li, Pengfei Xiong, Haoqiang Fan, i Jian Sun. Dfanet: Deep feature aggregation for real-time semantic segmentation, 2019.

[24] Xin Li, Yiming Zhou, Zheng Pan, i Jiashi Feng. Partial order pruning: for best speed/accuracy trade-off in neural architecture search, 2019.

[25] Jonathan Long, Evan Shelhamer, i Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015.

[26] Davide Mazzini. Guided upsampling network for real-time semantic segmentation, 2018.

[27] Sachin Mehta, Mohammad Rastegari, Anat Caspi, Linda Shapiro, i Hannaneh Hajishirzi. Espnet: Efficient spatial pyramid of dilated convolutions for semantic segmentation, 2018.

[28] Yuval Nirkin, Lior Wolf, i Tal Hassner. Hyperseg: Patch-wise hypernetwork for real-time semantic segmentation, 2021.

[29] Marin Oršić i Siniša Šegvić. Efficient semantic segmentation with pyramidal fusion, 2021.

[30] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, i Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation, 2016.

[31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, i Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. U H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, i R. Garnett, urednici, *Advances in Neural Information Processing Systems 32*, stranice 8024–8035. Curran Associates, Inc., 2019.

[32] Eduardo Romera, José M. Álvarez, Luis M. Bergasa, i Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2018. doi: 10.1109/TITS.2017.2750080.

[33] Olaf Ronneberger, Philipp Fischer, i Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[34] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, i Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019.

[35] Karen Simonyan i Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[36] Nikitha Vallurupalli, Sriharsha Annamaneni, Girish Varma, C V Jawahar, Manu Mathew, i Soyeb Nagori. Efficient semantic segmentation using gradual grouping, 2018.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, i Illia Polosukhin. Attention is all you need, 2017.

[38] Min Wang, Baoyuan Liu, i Hassan Foroosh. Factorized convolutional neural networks. U *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, stranice 545–553, 2017. doi: 10.1109/ICCVW.2017.71.

[39] Xiaolong Wang, Ross Girshick, Abhinav Gupta, i Kaiming He. Non-local neural networks, 2018.

[40] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, i Kaiming He. Aggregated residual transformations for deep neural networks, 2017.

[41] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, i Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation, 2018.

[42] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, i Jiaya Jia. Pyramid scene parsing network, 2017.

[43] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, i Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images, 2018.

[44] Juntang Zhuang, Junlin Yang, Lin Gu, i Nicha Dvornek. Shelfnet for fast semantic segmentation, 2019.

[45] Çağrı Kaymak i Ayşegül Uçar. A brief survey and an application of semantic image segmentation for autonomous driving, 2018.

Semantička segmentacija s brzom pažnjom u preskočnim vezama

Sažetak

Semantička segmentacija je zadatak računalnog vida koji elemente slike razvrstava u različite semantičke razrede. U posljednje vrijeme kvalitetna rješenja zadatka daju konvolucijski modeli koji se sastoje od osnove za klasifikaciju i ljevičastog naduzorkovanja. Modeli za obradu slike u stvarnom vremenu moraju žrtvovati svoju kompleksnost kako bi dobili na brzini. Mehanizam brze pažnje prilagođen je za obradu u stvarnom vremenu i poboljšava preciznost modela. U radu su u preskočne veze modela SwiftNet uvedeni moduli brze pažnje. Dobiveni model evaluiran je nad skupom podataka Cityscapes i uspoređen s osnovnim modelom.

Ključne riječi: semantička segmentacija, zaključivanje u stvarnom vremenu, brza pažnja, koder-dekoder, SwiftNet, FANet, preskočne veze

Semantic segmentation with fast attention in skip connections

Abstract

Semantic segmentation is a computer vision task which classifies pixels of an image into different semantic classes. Recently convolutional models based on classification backbone and encoder-decoder architecture have given quality solutions for the task. Models for real-time inference have to sacrifice their complexity in order to increase their speed. Fast attention mechanism is adapted for real-time inference and it improves model accuracy. In this thesis, fast attention modules have been introduced into lateral connections of SwiftNet model. The resulting model was evaluated over the Cityscapes dataset and compared with the baseline model.

Keywords: semantic segmentation, real-time inference, fast attention, encoder-decoder, SwiftNet, FANet, skip connections