SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2905

Duboki modeli za detekciju ključnih točaka osoba

Pavo Matanović

Zagreb, rujan 2022.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 11. ožujka 2022.

DIPLOMSKI ZADATAK br. 2905

Pristupnik:	**Pavo Matanović (0036506316)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	prof. dr. sc. Siniša Šegvić

Zadatak: **Duboki modeli za detekciju ključnih točaka osoba**

Opis zadatka:

Detekcija položaja osoba u slikama važno je područje računalnog vida s mnogim zanimljivim primjenama. Ovaj rad usredotočen je na postupke za detekciju ključnih točaka ljudskog tijela poput gležnja, koljena ili kuka. Ti postupci posebno su zanimljivi jer mogu modelirati gotovo sve stupnjeve slobode ljudskog tijela. U okviru rada, potrebno je odabrati okvir za automatsku diferencijaciju te upoznati biblioteke za rukovanje tenzorima i slikama. Proučiti i ukratko opisati postojeće pristupe za detekciju ključnih točaka. Odabrati slobodno dostupni skup slika te oblikovati podskupove za učenje, validaciju i testiranje. Oblikovati algoritme te uhodati učenje, validiranje i zaključivanje. Vrednovati naučene modele te prikazati postignutu točnost i učinkovitost. Predložiti pravce budućeg rada. Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 27. lipnja 2022.

Zahvaljujem se mentoru prof. dr. sc. Siniši Šegviću, na svoj pomoći pri izradi ovog rada i prenesenom znanju tijekom studija. Zahvaljujem se obitelji na potpori tijekom školovanja.

SADRŽAJ

1. Uvod

Detekcija položaja osoba važno je područje računalnog vida s mnogim zanimljivim primjenama. Detekcijom ključnih točaka poput ramena, kuka, koljena ili gležnja možemo modelirati stupnjeve slobode prilikom kretnje ljudskog tijela. Zadatak detekcije ključnih točaka ima zanimljivu primjenu i obradi video sekvenci. Analizirajući položaje ljudskog tijela u nekoliko uzastopnih video okvira, možemo odrediti koju akciju promatrana osoba obavlja. Npr. možemo detektirati da li osoba hoda ili jede.

Postoje dva pristupa u rješavanju problema detekcije ključnih točaka na slikama s više osoba: *od vrha prema dolje* ili *odozdo prema gore*. Oba se pristupa odvijaju u dvije faze. Koristeći pristup *od vrha prema dolje*, originalna slika se prvo stavlja na ulaz modela za detekciju osoba, a zatim se isječak osobe stavlja na ulaz model za detekciju ključnih točaka koji određuje ključne točke jedne osobe. U pristupu *odozdo prema gore*, ulazna slika se odmah stavlja na ulaz modela za detekciju ključnih točaka koji pronalazi sve ključne točke u cijeloj slici. Nakon toga se algoritamski ili na neki drugi način grupiraju ključne točke koje pripadaju jednoj osobi. U našem radu koristimo pristup od vrha prema dolje.

2. Srodni radovi

Detekcija ključnih točaka osoba je aktivno područje istraživanja već desetljećima. Klasični pristupi pristupaju tom problemu koristeći slikovne strukture [9, 1] ili grafičke modele [5]. Svi klasični pristupi [1, 5, 27, 11, 7, 28, 33, 25] formuliraju problem određivanja ključnih točaka kao problem za grafičke modele i modele sa stablastom strukturom, a predviđaju ključne točke koristeći ručno izrađene značajke. Noviji radovi [21, 12, 3, 16, 30, 32] se oslanjaju na konvolucijske modele koji značajno poboljšavaju performanse detekcije položaja osoba. U našem radu koristimo metode s konvolucijskim modelima.

Detekcija položaja više osoba

Zadatak detekcije položaja više osoba na jednoj fotografiji postaje sve popularniji u posljednje vrijeme zbog visoke potražnje za primjenom u stvarnom životu. Problem je vrlo zahtjevan zbog zaklanjanja, raznih položaja pojedinih osoba i nepredvidivih interakcija između različitih osoba. Pristupi rješavanju tog problema uglavnom se dijele u dvije kategorije: pristup od vrha prema dolje i pristup odozdo prema gore.

Pristupi odozdo prema gore

Pristupi odozdo prema gore [16, 4, 22, 26] direktno predviđaju sve ključne točke u prvoj fazi te ih spajaju u potpune položaje svih osoba u drugoj fazi. DeepCut [26] tretira problem razlikovanja osoba u slici kao problem linearnog programiranja (engl. *Integer Linear Program*) i grupira kandidate za ključne točke. Konačna predikcija položaja osobe dobiva se kad se upare grupe kandidata s označenim dijelovima tijela. DeeperCut [16] nadograđuje DeepCut [26] koristeći arhitekturu ResNet [13] i uzima u obzir međusobne udaljenosti ključnih točaka kako bi dobio bolju učinkovitost. Cao *i sur.* [4] određuju odnose između ključnih točaka koristeći PAF (engl. *Part Affinity Fields*). Newell *i sur.* [22] istovremeno koriste mapu sigurnosti predikcije i ugrađivanje na razini piksela kako bi grupirali kandidate za ključne točke u pojedine osobe.

Pristupi od vrha prema dolje

Pristupi od vrha prema dolje [14, 24, 15, 8] koriste dvofazne metode. U prvoj fazi se traže isječci svih osoba sa slike, a onda se rješava problem detekcije položaja jedne osobe na tim isječcima. Papandreou *i sur.* [24] predviđaju toplinske mape (engl. *heatmap*) i udaljenosti točaka na toplinskoj mapi od stvarnih točaka te iz tih informacija dobivaju konačne lokacije ključnih točaka. Mask-RCNN [14] prvo predviđa okvire osoba na slici te koristi isječke mapa značajki koji odgovaraju okviru osobe kako bi dobio ključne točke osobe. Kada koristimo pristup od vrha prema dolje, za dobru učinkovitost jednako nam je bitan i detektor osoba i procjenitelj ključnih točaka jedne osobe.

Detekcija položaja jedne osobe

Toshev *i sur.* prvi predlažu korištenje konvolucijskih modela za problem detekcije položaja osobe u radu DeepPose [29] u kojem koriste kaskadnu arhitekturu konvolucijskih regresora za detekciju položaja. Noviji radovi [21, 30] postižu dobru učinkovitost koristeći duboke konvolucijske modele. Wei *i sur.* [30] predlažu višerazinsku arhitekturu (engl. *multi-stage architecture*) koju koriste na način da prvo generiraju grube rezultate te kontinuirano rafiniraju rezultate u kasnijim razinama. Newell *i sur.* [21] predlažu arhitekturu pješčanog sata (engl. *hourglass architecture*) te slažu nekoliko takvih modula jedan za drugim. Lifshitz *i sur.* [18] koriste glasanje dubokim konsenzusom (engl. *deep consensus voting*) kako bi izglasali najvjerojatniji položaj ključne točke. Gkioxary *i sur.* [12] te Belagiannis i Zisserman [2] primjenjuju povratne modele kako bi slijedno rafinirali rezultate. Yang *i sur.* [32] koriste piramidalne značajke prilikom određivanja položaja osobe te uvode piramidalne rezidualne module (PRM).

Detekcija osoba

Detekcija osoba se pretežno radi s modelima iz obitelji R-CNN [10, 20, 14]. Te metode uglavnom sastoje od dvije faze. U prvoj se generiraju prijedlozi okvira, a zatim se isječci mapa značajki koriste za fino podešavanje prijedloga kako bi dobili konačni okvir. Detektor koji koristimo radu temelji se na [20, 14].

3. CPN

Model kaskadne piramidalne strukture (engl. *Cascaded Pyramid Network – CPN*) prvi put su uveli Chen *i sur.* u radu [6]. Kao i [14, 24], njihov algoritam koristi pristup od vrha prema dolje.

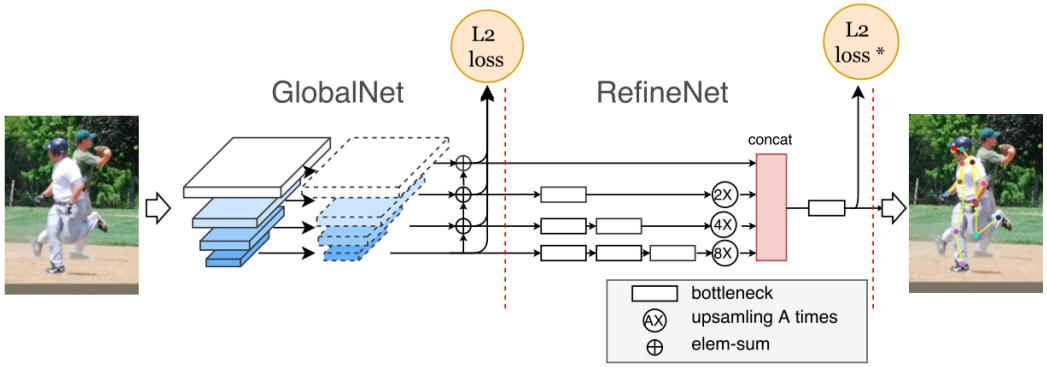
Detektor osoba. Chen *i sur.* radu koriste detektor koji je temeljen na arhitekturi FPN [20]. Umjesto ROIPooling modula koriste ROIAlign modul iz Mask RCNN-a [14]. Prilikom učenja detektora korišteno je svih 80 klasa iz MS COCO [19] skupa, ali se samo okviri osoba koriste prilikom procjene položaja.

Prije nego što objasnimo arhitekturu CPN-a, ukratko ćemo pogledati radove koji su bili inspiracija za dizajniranje arhitekture. Metoda *Stacked hourglass* [21], koja se pokazala dobra za određivanje položaja osoba, slaže osam modula pješčanog sata. Svaki modul radi na način da prvo smanjuje prostornu dimenziju tenzora, a zatim ju povećava. Pokazuje se da strategija slaganja modula pješčanih satova radi, ali se dobre performanse mogu dobiti i koristeći samo 2 modula. [24] koristi arhitekturu ResNet [13] kako bi odredio položaj osobe te se pokazuje da je takav pristup uspješan na skupu podataka MS COCO. Motivirani radovima [21, 24], Chen *i sur.* predlažu arhitekturu CPN koja je primjenjiva za problem određivanja položaja osoba. Na slici 3.1 prikazana je arhitektura CPN-a koja se sastoji od dva modula: GlobalNet i RefineNet.

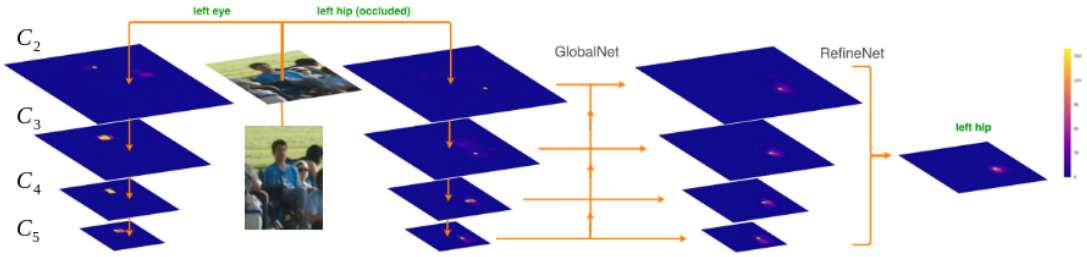
U našem radu zamjenjujemo okosnicu metode te umjesto ResNet-50 koristimo ResNet-18. Ta zamjena nam nameće da promijenimo broj kanala u modulima GlobalNet i RefineNet te rezultira smanjenjem kapaciteta, ali ubrzava treniranje modela.

3.1. GlobalNet

GlobalNet kao ulaze u modul prima mape značajki izlaza ResNet okosnice. Označimo te ulaze C_2, C_3, C_4, C_5. Kako bi generirali toplinske mape ključnih točaka, primjenjujemo konvoluciju s veličinom jezgre 3×3 na $C_2, \ldots, C_5$. Na slici 3.2 možemo vidjeti da plitke značajke poput C_2 i C_3 imaju veliku prostornu dimenziju što nam pomaže prilikom lokalizacije i malu semantičku dimenziju što nam otežava prepoznavanje. S



Slika 3.1: Arhitektura CPN-a [6]. Na ulaz modela se dovode isječci osoba iz detektora. Iz izlaznih značajki ResNet okosnice, pomoću GlobalNet modula, se određuju grube toplinske mape ključnih točaka. Zatim se pomoću RefineNet modula fino ugađaju izlazne toplinske mape te se dobiva konačni položaj ključnih točaka. "L2 loss*" je oznaka za L2 gubitak sa online rudarenjem teških ključnih točaka (engl. *online hard keypoint mining*).



Slika 3.2: Toplinske mape aktivacija metode CPN u pojedinim fazama obrade. Lijevo su aktivacije na izlazu okosnice, u sredini su izlazi GlobalNet-a, a desno je konačna predikcija modela nakon RefineNet-a. [6]

druge strane, duboki slojevi poput C_4 i C_5 imaju više semantičkih informacija, ali nisku prostornu dimenziju zbog konvolucije s korakom i sažimanja. Zbog toga se često koristi struktura u obliku slova U kako bi održali i prostornu dimenziju i semantičku informaciju mapi značajki. Sličnu strukturu koristi i FPN [20], ali razlika je u dodatnoj 1×1 konvoluciji prije sume, a nakon naduzorkovanja mape značajki niže razine.

Analizirajući toplinske mape aktivacija na slici 3.2 vidimo da GlobalNet može dobro locirati ključne točke kao što su oči, ali ne može precizno locirati položaj kukova. Kako bi locirali ključne točke kao što su kukovi, često nam je potrebno više informacija iz konteksta i dubljih slojeva nego što možemo dobiti iz lokalnog susjedstva mape značajki. Postoje mnogi primjeri takvih teških ključnih točaka koje je teško locirati samo s GlobalNet-om.

3.2. RefineNet

RefineNet modul dodajemo kako bi riješili problem teških ključnih točaka. Na ulaz RefineNet-a dovodimo piramidalnu reprezentaciju značajki koja je izlaz GlobalNet modula. Kako bi poboljšali efikasnost i zadržali integritet prilikom prijenosa informacija, RefineNet šalje informacije preko svih razina i integrira sve informacije svih razina pomoću naduzorkovanja i ulančavanja. Za razliku od metode *Stacked hourglass*, RefineNet ulančava sve piramidalne značajke umjesto da koristi naduzorkovane značajke zadnjeg sloja piramide. Dodatno, nad značajkama dubljih slojeva primjenjujemo više *BottleNeck* blokova, pri čemu zbog male prostorne dimenzije dubljih slojeva postižemo dobar omjer između učinkovitosti i brzine.

Prilikom učenja, RefineNet često obraća manje pozornosti na teške i zaklonjene ključne točke. Kako bi to izbjegli i balansirali pažnju između dvaju tipova ključnih točaka, uvodimo online rudarenje ključnih točaka (OHKM). Metoda OHKM radi tako da prilikom učenja odabire teške ključne točke po kriteriju većeg gubitka. Zatim propagira gradijente unazad samo za odabrane točke. Broj ključnih točaka na kojima RefineNet uči reguliramo hiperparametrom.

4. Swiftnet

Model SwiftNet [23] su prvi put predstavili Oršić i Šegvić kako bi rješili problem semantičke segmentacije. To je "lagan" model s malo parametara koji postiže dobru učinkovitost za zadatak semantičke segmentacije unatoč skromnom kapacitetu. Postoje dvije inačice tog modela: jedna koja koristi prostorno piramidalno sažimanje (engl. *Spatial Pyramid Pooling*) i značajke samo na jednoj skali te druga koja koristi piramidalnu fuziju (engl. *Pyramidal fusion*) na slikama različitih skala. Ovaj pristup je učinkovit za guste predikcije na slikama s velikom varijancom zbog regularizacijskog efekta koji se postiže dijeljenjem parametara na različitima skalama piramide. Zbog svoje jednostavnosti, SwiftNet se može koristiti u slučajevima kada je potrebno zaključivati u stvarnom vremenu te na ugradbenim računalima.

Metoda SwiftNet polazi od sljedećih pretpostavki. Enkoderi za raspoznavanje trebaju biti predtrenirani na ImageNet skupu kako bi iskoristili prijenos znanja. Receptivno polje treba se povećati s prikladnim modulom ili arhitekturnim stilom. Rezolucija enkodiranih značajki mora se restaurirati ljestvičastim dekoderom kako bi zadržali razinu detalja. Postupak naduzorkovanja treba biti jednostavan kako bi podržavao zaključivanje u stvarnom vremenu. Potrebno je olakšati tok gradijenata kroz model kako bi osigurali efikasno učenje modela.

Šarić *i sur.* [35] pokazali su da piramidalna fuzija pomaže i za rješavanje zadatka panoptičke segmentacije. U našem radu, prilagodili smo SwiftNet za zadatak detekcije ključnih točaka osoba.

4.1. Osnovni gradivni blokovi

Postoje tri osnovna gradivna bloka modela SwiftNet.

Enkoder za raspoznavanje. Razmatramo kompaktne arhitekture koje imaju dobre performanse i prihvatljive računske zahtjeve. Kao okosnicu koristimo ResNet-18 zato što su javno dostupni predtrenirani parametri koje možemo fino ugađati. Također, možemo ih trenirati i iz nule zbog umjerene dubine i rezidualnih veza. Zbog relativno

niske kompleksnosti, ResNet-18 je kompatibilan s zahtjevom za rad u stvarnom vremenu. Enkoder se sastoji od četiri enkoderska bloka (EB na slici 4.1) čiji međurezultati imaju prostornu dimenziju koja je manja 4, 8, 16 i 32 puta od ulazne slike.

Dekoder za naduzrokovanje. Enkoder za raspoznavanje pretvara ulaznu sliku u semantički bogate značajke. Te značajke imaju malu prostornu dimenziju kako bi uštedjeli memorijski prostor i vrijeme izvođenja. Uloga dekodera je da naduzorkuje te semantički bogate značajke na ulaznu rezoluciju. SwiftNet koristi jednostavan dekoder koji je organiziran kao niz modula za naduzorkovanje (UP na slici 4.1) s lateralnim vezama. Moduli za naduzorkovanje imaju dva ulaza:

1. značajke niske rezolucije iz prethodnog modula i
2. značajke visoke rezolucije iz odgovarajućeg enkoderskog bloka.

Značajke niske rezolucije prvo se naduzorkuju bilinearnom interpolacijom do dimenzije odgovarajućih značajki koje dolaze iz enkodera. Zatim se te značajke zbrajaju po elementima i na kraju se zaglađuju 3×3 konvolucijom.

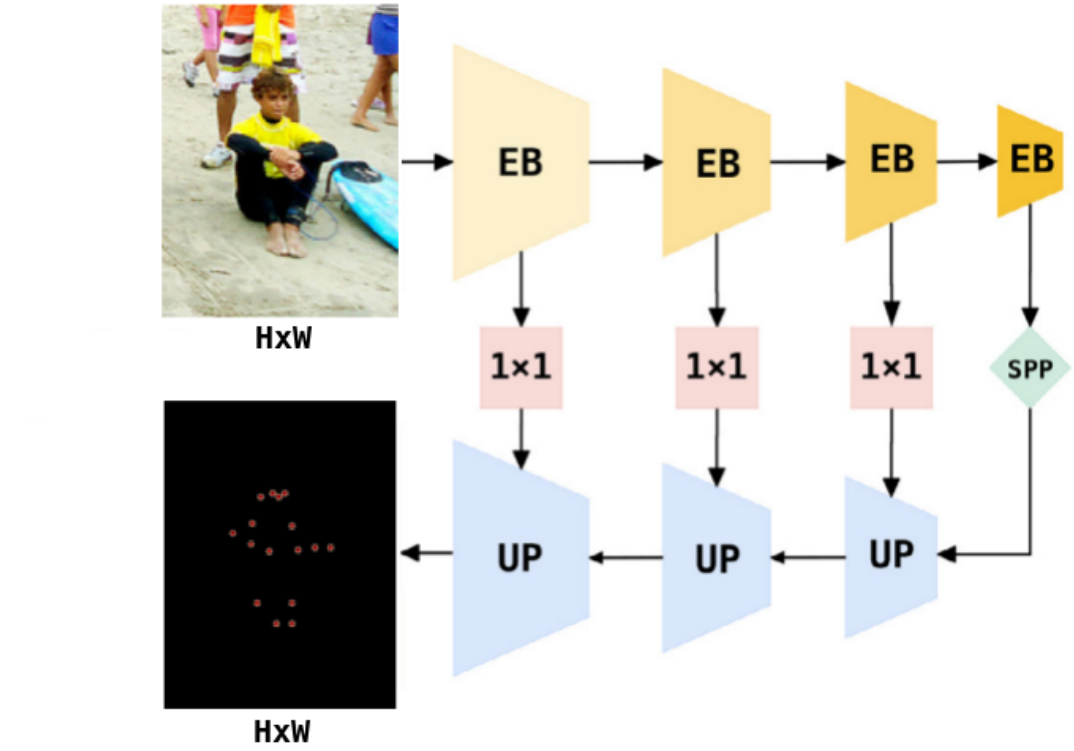
Povećavanje receptivnog polja. Postoje dvije inačice modela SwiftNet, koje se razlikuju u pristupu na koji se povećava receptivno polje:

1. Prostorno piramidalno sažimanje (SPP) i
2. Piramidalna fuzija.

SPP [34, 17] na izlazu daje mape značajki s različitom razinom detalja. To radi na način da obogaćuje enkoderske značajke adaptivnim sažimanjem srednjom vrijednošću na konačne rezolucije veličine 1×1, 2×2, 4×4 i 8×8. Piramidalna fuzija se bazira na reprezentacijama na različitim skalama. Te značajke se spajaju na različitim razinama apstrakcije i tako povećavaju receptivno polje bez žrtvovanja prostorne dimenzije.

4.2. Arhitektura SwiftNet-a s modulom SPP

Na slici 4.1 vidimo arhitekturu modela SwiftNet s jednom skalom koja se sastoji od enkodera za raspoznavanje, modula za prostorno piramidalno sažimanje i jednostavnog dekodera za naduzorkovanje. Žuti trapezi predstavljaju enkoderske blokove (EB), odnosno dijelove okosnice koji na izlazu imaju jednaku prostornu dimenziju. Model se sastoji od četiri takva bloka, prvi od kojih na izlazu daje prostornu dimenziju $H/4 \times W/4$, a svaki sljedeći smanjuje rezoluciju za faktor 2. Dakle, rezolucija na



Slika 4.1: Arhitektura SwiftNet-a s jednom skalom [23]. Žuti trapezi predstavljaju enkoderske blokove koji mogu biti predtrenirani na ImageNet skupu podataka. Zeleni romb predstavlja modul prostornog piramidalnog sažimanja, crveni kvadrati su *bottleneck* projekcije, dok plavi trapezi predstavljaju module za naduzorkovanje. Logite naduzorkujemo do ulazne dimenzije bilinearnom interpolacijom s faktorom x4.

kraju zadnjeg enkoderskog bloka postaje $H/32 \times W/32$. Te značajke se zatim stavljaju na ulaz SPP modula (zeleni romb na slici 4.1) kako bi se povećalo receptivno polje. Izlazni tenzor se onda šalje na ulaz dekodera čiji moduli za naduzorkovanje (UP) su označeni plavom bojom na slici 4.1.

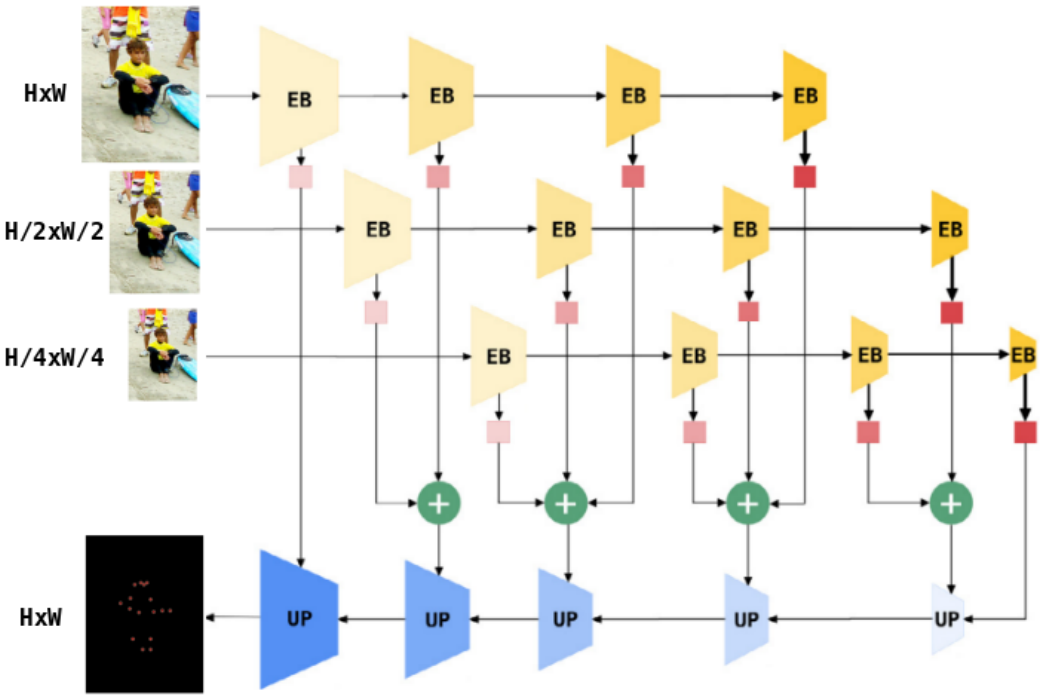
SPP modul je pojednostavljena i blago poboljšana verzija piramidalnog modula za sažimanje (engl. *Pyramid Pooling Module*) (PPM) iz rada PSPNet [34], koja je predložena u [17]. Glavna razlika je u mjestu na koji je postavljen modul. PSPNet ima samo jedan konvolucijski sloj između modula PPM i izlaznih predikcija. Stoga, PPM ima dvije zadaće: da pruži informacije o kontekstu i da pripremi značajke za linearni klasifikator. S druge strane, značajke SPP modula moraju proći kroz dekoder prije nego ih klasificiramo kao što je prikazano na slici 4.1 dolje lijevo. Vidimo da SPP ima samo jednu ulogu, a to je da poveća receptivno polje. Zbog načina postavljanja SPP modula, možemo smanjiti broj mapa značajki na ulazu i izlazu modula te tako "olakšati" model.

Možemo primijetiti da su dekoder i enkoder asimetrični: enkoder ima mnogo 3×3

konvolucija po bloku dok dekoder ima samo jednu 3×3 konvoluciju po bloku. Dodatno, semantička dimenzija enkoderskih značajki se povećava s dubinom, a u dekoderu je konstantna. Razlog tomu je pretpostavka da raspoznavanje treba veći kapacitet od naduzorkovanja koje već posjeduje semantičku informaciju. Moduli za naduzorkovanje rade u tri koraka:

1. reprezentacija niske rezolucije se naduzorkuje bilinearnom interpolacijom
2. naduzorkovana reprezentacija se zbraja po elementima s lateralnom vezom
3. rezultat zbrajanja se zaglađuje 3×3 konvolucijom

4.3. Arhitektura SwiftNet-a s piramidalnom fuzijom



Slika 4.2: Arhitektura SwiftNet-a s piramidalnom fuzijom [23]. Žuti trapezi predstavljaju enkoderske blokove (EB). Crveni kvadrati su projekcije (1×1 konovolucije) koje izjednačavaju semantičke dimenzije značajki prije ulaska u dekoder. Zeleni kružići označavaju zbrajanje po elementima. Rezolucija se restaurira pomoću modula za naduzorkovanje, označenih s plavim trapezima (UP). Jednaka nijansa boje označava dijeljene parametre.

Arhitektura modela SwiftNet s piramidalnom fuzijom prikazana je na slici 4.2. Sva četiri enkoderska bloka dijele parametre s odgovarajućim blokovima na drugim skalama (označeni jednakom nijansom na slici 4.2). To osigurava kovarijantnost na skalu

unutar samog enkodera (za razliku od modula SPP), što nam omogućava da prepoznajemo objekte različite veličine s istim skupom parametara. Ispreplitanjem heterogenih reprezentacija poboljšavamo propagaciju gradijenata prema ranim slojevima i postiže se ponašanje nalik ansamblu modela.

Žuti trapezi označavaju enkoderske blokove (EB), a nijansom se označava koji blokovi dijele parametre. Crveni kvadrati označavaju projekcije (konvolucije s jezgrom 1×1) koje prilagođavaju broj mapa značajki prema dimenzionalnosti puta naduzorkovanja. Zeleni kružići označavaju zbrajanje po elementima koje spaja sve značajke s istom prostornom dimenzijom. Taj korak se naziva piramidalna fuzija. Spojene značajke se onda prenose u dekoder kao lateralne veze za ljestvičasto naduzorkovanje. Moduli za naduzorkovanje sastoje se od bilinearne interpolacije, zbrajanja po elementima i 3×3 konvolucije. Logite dobivamo bilinearnim naduzorkovanjem s faktorom x4.

5. Programska izvedba i vanjske bibilioteke

U radu koristimo programski jezik Python te uz njega koristimo sljedeće vanjske biblioteke: NumPy, PyTorch, Pycocotools i detectron2.

5.1. NumPy

NumPy je biblioteka za manipuliranje matricama i višedimenzionalnim nizovima koja se koristi u gotovo svakom znanstvenom izračunu. Podržava širok spektar operacija nad višedimenzionalnim nizovima, a brzinu izvođenja postiže tako što su metode implementirane u programskom jeziku C dok ih iz Python-a samo pozivamo putem omotača.

5.2. PyTorch

PyTorch je programski okvir otvorenog koda za automatsku diferencijaciju, kojeg je razvio Facebook. Koristi se za razvoj i učenje modela dubokog učenja. Kao i NumPy, PyTorch podržava širok spektar operacija za rad s višedimenzionalnim tenzorima, uz razliku da PyTorch podržava izvođenje operacija na GPU. Za razliku od konkurentnih radnih okvira za duboko učenje poput TensorFlow-a, PyTorch dinamički stvara graf izvođenja modela te omogućava fleksibilnost i lako računanje gradijenata.

5.3. Pycocotools

Pycocotools je bibilioteka koja se koristi za dohvaćanje primjera iz skupa podataka MS COCO te za ocjenu rezultata. Biblioteka se može koristiti za evaluaciju svih zadataka (detekciju objekata, semantičku segmentaciju, detekciju ključnih točaka, itd.) koji su

definirani na skupu MS COCO.

5.4. Detectron2

Detectron2 [31] je biblioteka koju je razvio Facebook AI tim, a pruža algoritme detekcije objekata i semantičke segmentacije koji su trenutno stanje tehnike. Razvijena s idejom da se brzo može dobiti osnovica za usporedbu novih modela. Pruža mnoštvo modela za različite zadatke računalnog vida. U našem radu koristimo detektor objekata iz ove biblioteke.

5.5. Programska izvedba

Autori CPN-a su uz rad pružili i programsku implementaciju koja koristi biblioteku TensorFlow. S obzirom da mi koristimo PyTorch, pronašli smo reimplementaciju originalnog algoritma koristeći PyTorch[1]. Također, u našem radu koristimo drugačiju okosnicu od one u originalnom radu pa smo morali prilagoditi model. S obzirom da se nova okosnica razlikuje u širini, morali smo prilagoditi konstruktor modela da mu možemo zadati širinu, tj. broj kanala u pojedinim slojevima.

S obzirom da je SwiftNet razvijen za zadatak semantičke segmentacije, morali smo ga prilagoditi za zadatak detekcije ključnih točaka. Prvo smo implementirali svoju verziju skupa podataka MS COCO. Razred smo implementirali prema predlošku drugih skupova koji su korišteni u originalnom radu. Zatim smo implementirali razrede za augmentaciju podataka također prema predlošku. S obzirom da predviđamo toplinske mape ključnih točaka, ne možemo koristiti postojeći gubitak unakrsne entropije nego smo implementirali gubitak L2. Dodali smo i metodu za evaluaciju predviđenih ključnih točaka. Evaluaciju ključnih točaka radimo na način da skaliramo vrijednosti izlaznih logita na raspon $[0, 1]$, zatim poništimo aktivacije koje su ispod definiranog praga. Nakon toga radimo težinsku sumu toplinske mape tako da su vrijednosti koordinate, a težine su vrijednosti iz toplinske mape. Ocjenu predikcije definiramo kao vrijednost toplinske mape na lokaciji koja je najbliža predviđenim koordinatama ključne točke. Nakon što dobijemo konačne predikcije i ocjene ključnih točaka, pozivamo metode razreda `COCOeval` iz biblioteke pycocotools kako bi dobili rezultate predikcija.

[1] https://github.com/GengDavid/pytorch-cpn

6. Eksperimenti

Naša metoda procjene položaja više osoba slijedi pristup od vrha prema dolje. Prvo primjenjujemo detektor osoba kako bi dobili okvire u kojima se nalaze osobe. Za svaki okvir, pretpostavljamo da je samo jedna, glavna osoba u okviru. Zatim šaljemo isječke osoba u model za detekciju položaja osoba koji daje konačne predikcije. U ovom poglavlju ćemo dati više detalja o našim metodama pomoću rezultata eksperimenata.

6.1. Konfiguracija učenja i zaključivanja

Ovdje ćemo dati više informacija o načinu učenja modela, strategiji izrezivanja osoba iz ulaznih slika, način augmentacije podataka te način na koji testiramo naš model.

6.1.1. Strategija izrezivanja osoba

Svaki okvir u kojem je detektirana osoba proširujemo na fiksni omjer visine i širine, npr. u našem radu koristimo omjer visina : širina $= 256 : 192$. Okvir proširujemo tako da se zadržava centar originalnog okvira. Zatim izrezujemo dio slike koji se nalazi unutar okvira bez da iskrivljujemo originalnu sliku i mijenjamo joj omjer stranica. Na kraju skaliramo isječak na fiksnu visinu od 256 piksela i širinu od 192 piksela.

6.1.2. Način augmentacije podataka

Augmentacija podataka važna je da bi spriječili prenaučenost modela. Također, ključna je i za učenje invarijantnosti na skalu i rotaciju. Nakon što dobijemo isječke slika iz okvira, primjenjujemo slučajno skaliranje s vrijednošću iz intervala $[0.7, 1.35]$. Zatim nasumično biramo jednu od sljedećih augmentacija, sve s jednakom vjerojatnošću:

- Vodoravno zrcaljenje,
- Nasumičnu rotaciju za kut iz intervala $[-15°, 15°]$,
- Nasumičnu rotaciju za kut iz intervala $[-45°, 45°]$,

– Identitetu - ne radimo nikakvu augmentaciju.

Nakon toga mijenjamo boje pikselima tako da uzmemo 3 različita faktora iz intervala $[0.8, 1.2]$ te pomnožimo vrijednosti svakog kanala zasebno s odgovarajućim faktorom. S obzirom da nakon ove augmentacije možemo dobiti vrijednosti koje su izvan očekivanog raspona, onda sve vrijednosti koje su izašle iz raspona postavljamo na najveću (ili najmanju) vrijednost unutar raspona. Konačno, normaliziramo sliku tako da joj oduzmemo srednju vrijednost i podijelimo sa standardnom devijacijom.

Prilikom obavljanja gore navedenih transformacija ulaznih podataka, te iste transformacije radimo i na priloženim oznakama. Dodatno, s obzirom da naši modeli predviđaju toplinske mape ključnih točaka, moramo pretvoriti oznake iz koordinata u toplinske mape. Za oznake koje nisu prisutne (ili nisu vidljive) u skupu podataka, vraćamo tenzor nula. Za ostale oznake generiramo toplinsku mapu na način da stavimo vrijednost 1 na transformirane koordinate ključnih točaka, zatim primjenjujemo Gaussov filter s veličinom jezgre 7×7. Na kraju skaliramo toplinsku mapu tako da podijelimo s maksimalnom vrijednošću i pomnožimo s 255. Dobili smo toplinsku mapu u kojoj na pravom mjestu ključne točke imamo najveći intenzitet (255), dok u okolini ključne točke intenzitet opada što se dalje odmičemo od točnih koordinata.

6.1.3. Način učenja modela

Za sve metode koristimo gubitak prosječne kvadratne udaljenosti (MSE), odnosno L2 gubitak. U svim metodama koristimo optimizator Adam. U metodi CPN početna stopa učenja iznosi 5×10^{-4} te se smanjuje za pola svakih 6 epoha. U metodi SwiftNet početna stopa učenja iznosi 4×10^{-4} te se smanjuje kosinusnim kaljenjem do minimalne stope od 1×10^{-6} u zadnjoj epohi. Prilikom učenja modela CPN postavljamo faktor propadanja težina na 1×10^{-5}, a veličinu minigrupe na 32. Metoda SwiftNet koristi faktor propadanja težina na 1×10^{-4}, a veličinu minigrupe postavili smo na 100. Za parametre SwiftNet-a koje fino ugađamo koristimo stopu učenja i faktor propadanja težina koji su manji 4 puta. Broj epoha na kojem učimo CPN i SwiftNet s modulom SPP je 32, dok smo SwiftNet s piramidalnom fuzijom učili 100 epoha. Koristimo normalizaciju po minigrupama u obje metode, s tim da se u modelu SwiftNet s piramidalnom fuzijom statistike za normalizaciju računaju odvojeno za svaku skalu. Sve modele učimo na grafičkoj kartici NVIDIA GeForce GTX 1070.

6.1.4. Način testiranja modela

Kako bi minimizirali varijancu predikcije, primjenjujemo Gaussov filter na izlazne toplinske mape. Metoda CPN na ulaz modela stavlja originalnu i horizontalno zrcalnu sliku te uzima prosjek predviđenih toplinskih mapa. Kao konačnu koordinatu uzima točku koja ima najveći intenzitet te ju pomiče u smjeru točke drugog najvećeg intenziteta za četvrtinu udaljenosti između njih. Također, metoda CPN koristi strategiju ponovnog ocjenjivanja tako da pomnoži ocjenu detektiranog okvira osobe s prosječnom ocjenom svih ključnih točaka.

Kako bi dobili konačnu poziciju ključnih točaka, u metodi SwiftNet prvo skaliramo izlazne značajke na interval $[0, 1]$. Zatim postavljamo sve aktivacije koje su manje od praga na nulu. U našim eksperimentima postavljamo prag na vrijednost 0.5. Konačnu točku dobivamo težinskom sumom koordinata, gdje su težine jednake skaliranim aktivacijama. Konačna ocjena predikcije jednaka je prosječnoj ocjeni svih ključnih točaka.

6.2. MS COCO

Naše modele učimo na skupu podataka MS COCO [19]. MS COCO se sastoji od skupa za učenje, skupa za validaciju i skupa za testiranje. Skup za učenje sadrži 118000 slika na kojima se nalazi ~ 150000 pojedinačnih osoba. Skup za validaciju sadrži 5000 slika na kojima se nalazi ~ 6300 osoba. Skup za testiranje sadrži 41000 slika, ali ga u radu ne koristimo jer nam oznake nisu javno dostupne za preuzimanje. Sve rezultate testiranja iskazujemo na skupu za validaciju.

Za svaku osobu imamo oznake ključnih točaka u formatu $[x_1, y_1, v_1, \ldots, x_k, y_k, v_k]$, gdje x_i, y_i označavaju lokacije ključnih točaka, a v_i je zastavica vidljivosti koja ima vrijednosti

$$v_i = \begin{cases} 0 & \text{nema oznake} \\ 1 & \text{oznaka nije vidljiva} \\ 2 & \text{oznaka je vidljiva} \end{cases}$$

Kao metriku za evaluaciju, skup MS COCO definira novu mjeru sličnosti ključnih točaka objekta (engl. *object keypoint similarity, OKS*). Ključna ideja koja se koristila pri definiranju metrike je napraviti mjeru koja će biti nalik na mjere koje se koriste prilikom detekcije objekata (IoU). OKS se definira kao:

$$\text{OKS} = \frac{\sum_i [\exp(-d_i^2/2s^2\sigma_i^2)\delta(v_i > 0)]}{\sum_i \delta(v_i > 0)} \tag{6.1}$$

gdje d_i označavaju euklidske udaljenosti između predviđenih i stvarnih ključnih točaka, a v_i su zastavice vidljivosti točnih oznaka. Kako bi izračunali OKS, uvrštavamo d_i u nenormaliziranu Gaussovu distribuciju sa standardnom devijacijom $s\sigma_i$, gdje je s^2 površina okvira ključnih točaka, a σ_i je konstantna definirana za svaku ključnu točku koja kontrolira širinu distribucije. Za svaku ključnu točku OKS daje mjeru sličnosti između 0 i 1. Ključne točke koje nisu označene ($v_i = 0$) ne utječu na OKS. Savršene predikcije će imati OKS $= 1$, dok će predikcije koje su udaljenije od nekoliko standardnih devijacija $s\sigma_i$ imati OKS ~ 0.

6.3. Rezultati detektora osoba

S obzirom da obje metode koriste pristup od vrha prema dolje, potreban nam je detektor osoba koji će dobro prepoznavati osobe na slici. Za tu svrhu ne treniramo svoj detektor nego koristimo predtrenirane modele iz radnog okvira detectron2 [31]. Ti modeli su učeni na skupu podataka MS COCO. Rezultati su prikazani u tablici 6.1. Najbolji rezultat ima model ResNet-101-FPN koji je učen na 400 epoha i njega odabiremo za korištenje prilikom evaluacije modela.

Tablica 6.1: Rezultati detektora osoba. AP(all) je prosječna preciznost na svih 80 COCO klasa. AP(H) je preciznost na klasi 'osoba'.

Model	# epoha učenja	AP(all)	AP(H)
ResNet-101-FPN	37	42.928	56.562
ResNet-101-FPN	400	48.882	60.571
ResNeXt-101-FPN	37	44.277	57.651

6.4. Rezultati

Modele smo testirali na MS COCO validacijskom skupu i rezultate smo prikazali u tablici 6.2. Model CPN18 postiže prosječnu preciznost od 67.1 postotna boda (pb), odnosno 65.4 pb kada ga testiramo s točnim okvirima osoba, odnosno s detektiranim okvirima. U originalnom radu [6] CPN koristi ResNet-50 što smo u tablici označili kao CPN50. CPN50 postiže prosječnu preciznost od 72.1 pb s jednim modelom, odnosno

73.0 pb s ansamblom modela. S obzirom na slabiji kapacitet CPN18 naspram CPN50, preciznost je očekivano lošija, ali nije drastično.

Rezultati metode SwiftNet nisu zadovoljavajući. SwiftNet s modulom SPP postiže preciznost od 27.4 pb, dok SwiftNet s piramidalnom fuzijom postiže preciznost od 31.2 pb. Autori u svom radu [23] navode da arhitektura postiže loše rezultate na malim objektima, što može biti razlog loših rezultata u našoj primjeni za detekciju ključnih točaka. Drugi razlog loših rezultata može biti u arhitekturi modela. Drugi radovi [21, 30] za detekciju ključnih točaka imaju arhitekturu koja u prvoj fazi predviđa grubu lokaciju ključnih točaka, zatim ih u sljedećim fazama postupno ugađaju kako bi dobili bolje predikcije. U metodi SwiftNet se lokacije ključnih točaka određuju samo u jednoj fazi što može biti pretežak zadatak za takvu arhitekturu. Primjere predikcija modela možemo vidjeti na slici 6.1.

Tablica 6.2: Rezultati detekcije ključnih točaka na MS COCO skupu. * označava ansambl modela. † označava evaluaciju na točnim okvirima osoba.

Metoda	AP	$AP_{@.5}$	$AP_{@.75}$	AP_m	AP_l	AR	$AR_{@.5}$	$AR_{@.75}$	AR_m	AR_l
CMU-Pose [4]	61.8	84.9	67.5	57.1	68.2	66.5	87.2	71.8	60.6	74.6
Mask-RCNN [14]	63.1	87.3	68.7	57.8	71.4	-	-	-	-	-
CPN50 [6]	72.1	91.4	80.0	68.7	77.2	78.5	95.1	85.3	74.2	84.3
CPN50* [6]	73.0	91.7	80.9	69.5	78.1	79.0	95.1	85.9	74.8	84.7
CPN18 †	67.1	89.4	74.9	65.0	70.7	70.6	90.8	77.3	67.7	75.0
CPN18	65.4	87.3	72.4	62.3	70.8	70.3	90.3	77.0	66.7	75.5
SwiftNet (SPP) †	27.4	65.3	17.5	26.2	29.7	33.3	68.2	28.0	30.2	37.6
SwiftNet (pir) †	31.2	68.6	24.2	29.6	33.8	38.5	72.4	35.5	35.2	43.2

Slika 6.1: Primjeri predikcije modela. U prvom redu su predikcije modela CPN18, u drugom modela SwiftNet s modulom SPP, a u trećem modela SwiftNet s piramidalnom fuzijom.

7. Zaključak

U ovom radu učili smo modele za detekciju ključnih točaka osoba koristeći dvije metode: CPN i SwiftNet. Objasnili smo dva različita pristupa problemu detekcije položaja više osoba na jednoj slici. Pristup od vrha prema dolje radi na način da prvo detektira pojedinačne osobe na slici, a zatim detektira ključne točke jedne osobe. Pristup odozdo prema gore detektira sve ključne točke svih osoba na slici, pa ih dodatnim postupkom grupira za pojedinu osobu. Opisali smo arhitekturu modela CPN koji se sastoji od dva modula: GlobalNet i RefineNet. GlobalNet služi za grubu procjenu ključnih točaka, nakon čega se procjena rafinira pomoću modula RefineNet. Zatim smo opisali arhitekturu modela SwiftNet, tj. njegove dvije inačice. Jedna koristi značajke samo na jednoj skali, a receptivno polje povećava posebnim modulom za prostorno piramidalno sažimanje (SPP). Dok druga koristi piramidalnu fuziju kako bi spojila informacije različitih skala. Piramidalna fuzija spaja značajke koje imaju dobru lokacijsku informaciju s značajkama koje imaju bogato semantičko znanje. Pokazuje se da se takva arhitektura ponaša kao ansambl modela. Modele smo učili i testirali na skupu podataka MS COCO. Metoda CPN s okosnicom ResNet-18 postiže prosječnu preciznost od 67.1, SwiftNet s modulom SPP postiže preciznost 27.4, a SwiftNet s piramidalnim sažimanjem postiže preciznost 31.2. Rezultati metode CPN su zadovoljavajući, dok rezultati SwiftNet-a nisu. U budućem radu potrebno je provjeriti zbog čega metoda SwiftNet daje loše rezultate.

LITERATURA

[1] Mykhaylo Andriluka, Stefan Roth, i Bernt Schiele. Pictorial structures revisited: People detection and articulated pose estimation. U *2009 IEEE Conference on Computer Vision and Pattern Recognition*, stranice 1014–1021, 2009. doi: 10.1109/CVPR.2009.5206754. 2

[2] Vasileios Belagiannis i Andrew Zisserman. Recurrent human pose estimation. U *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*. IEEE, may 2017. doi: 10.1109/fg.2017.64. 2

[3] Adrian Bulat i Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. U *Computer Vision – ECCV 2016*, stranice 717–732. Springer International Publishing, 2016. doi: 10.1007/978-3-319-46478-7_44. 2

[4] Zhe Cao, Tomas Simon, Shih-En Wei, i Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 7291–7299, 2017. 2, 6.2

[5] Xianjie Chen i Alan L Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. U Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, i K.Q. Weinberger, urednici, *Advances in Neural Information Processing Systems*, svezak 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/8b6dd7db9af49e67306feb59a8bdc52c-Paper.pdf. 2

[6] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, i Jian Sun. Cascaded Pyramid Network for Multi-Person Pose Estimation. U *CVPR*, stranice 7103–7112, 2018. 3, 3.1, 3.2, 6.4, 6.2

[7] Matthias Dantone, Juergen Gall, Christian Leistner, i Luc Van Gool. Human pose estimation using body parts dependent joint regressors. U *2013 IEEE Conference*

on Computer Vision and Pattern Recognition. IEEE, jun 2013. doi: 10.1109/cvpr.2013.391. 2

[8] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, i Cewu Lu. Rmpe: Regional multi-person pose estimation. U *Proceedings of the IEEE international conference on computer vision*, stranice 2334–2343, 2017. 2

[9] M.A. Fischler i R.A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, C-22(1):67–92, 1973. doi: 10.1109/T-C.1973.223602. 2

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, i Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 580–587, 2014. 2

[11] Georgia Gkioxari, Pablo Arbelaez, Lubomir Bourdev, i Jitendra Malik. Articulated pose estimation using discriminative armlet classifiers. U *2013 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2013. doi: 10.1109/cvpr.2013.429. 2

[12] Georgia Gkioxari, Alexander Toshev, i Navdeep Jaitly. Chained predictions using convolutional neural networks. U *European Conference on Computer Vision*, stranice 728–743. Springer, 2016. 2, 2

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. Deep residual learning for image recognition. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 770–778, 2016. 2, 3

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, i Ross Girshick. Mask r-cnn. U *Proceedings of the IEEE international conference on computer vision*, stranice 2961–2969, 2017. 2, 2, 3, 6.2

[15] Shaoli Huang, Mingming Gong, i Dacheng Tao. A coarse-fine network for keypoint localization. U *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, oct 2017. doi: 10.1109/iccv.2017.329. 2

[16] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, i Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. U *European conference on computer vision*, stranice 34–50. Springer, 2016. 2, 2

[17] Ivan Krešo, Josip Krapac, i Siniša Šegvić. Efficient ladder-style densenets for semantic segmentation of large images. *IEEE Transactions on Intelligent Transportation Systems*, 22(8):4951–4961, 2020. 4.1, 4.2

[18] Ita Lifshitz, Ethan Fetaya, i Shimon Ullman. Human pose estimation using deep consensus voting. U *European Conference on Computer Vision*, stranice 246–260. Springer, 2016. 2

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, i C Lawrence Zitnick. Microsoft coco: Common objects in context. U *European conference on computer vision*, stranice 740–755. Springer, 2014. 3, 6.2

[20] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, i Serge Belongie. Feature pyramid networks for object detection. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 2117–2125, 2017. 2, 3, 3.1

[21] Alejandro Newell, Kaiyu Yang, i Jia Deng. Stacked hourglass networks for human pose estimation. U *European conference on computer vision*, stranice 483–499. Springer, 2016. 2, 2, 3, 6.4

[22] Alejandro Newell, Zhiao Huang, i Jia Deng. Associative embedding: End-to-end learning for joint detection and grouping. *Advances in neural information processing systems*, 30, 2017. 2

[23] Marin Oršić i Siniša Šegvić. Efficient semantic segmentation with pyramidal fusion. *Pattern Recognition*, 110:107611, feb 2021. doi: 10.1016/j.patcog.2020.107611. 4, 4.1, 4.2, 6.4

[24] George Papandreou, Tyler Zhu, Nori Kanazawa, Alexander Toshev, Jonathan Tompson, Chris Bregler, i Kevin Murphy. Towards accurate multi-person pose estimation in the wild. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 4903–4911, 2017. 2, 3

[25] Leonid Pishchulin, Mykhaylo Andriluka, Peter Gehler, i Bernt Schiele. Poselet conditioned pictorial structures. U *2013 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2013. doi: 10.1109/cvpr.2013.82. 2

[26] Leonid Pishchulin, Eldar Insafutdinov, Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, Peter V Gehler, i Bernt Schiele. Deepcut: Joint subset partition and labeling for multi person pose estimation. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 4929–4937, 2016. 2

[27] Ben Sapp i Ben Taskar. MODEC: Multimodal decomposable models for human pose estimation. U *2013 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2013. doi: 10.1109/cvpr.2013.471. 2

[28] Benjamin Sapp, Chris Jordan, i Ben Taskar. Adaptive pose priors for pictorial structures. U *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, jun 2010. doi: 10.1109/cvpr.2010.5540182. 2

[29] Alexander Toshev i Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 1653–1660, 2014. 2

[30] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, i Yaser Sheikh. Convolutional pose machines. U *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, stranice 4724–4732, 2016. 2, 2, 6.4

[31] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, i Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 5.4, 6.3

[32] Wei Yang, Shuang Li, Wanli Ouyang, Hongsheng Li, i Xiaogang Wang. Learning feature pyramids for human pose estimation. U *proceedings of the IEEE international conference on computer vision*, stranice 1281–1290, 2017. 2, 2

[33] Yi Yang i Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. U *CVPR 2011*. IEEE, jun 2011. doi: 10.1109/cvpr.2011.5995741. 2

[34] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, i Jiaya Jia. Pyramid scene parsing network. U *Proceedings of the IEEE conference on computer vision and pattern recognition*, stranice 2881–2890, 2017. 4.1, 4.2

[35] Josip Šarić, Marin Oršić, i Siniša Šegvić. Panoptic swiftnet: Pyramidal fusion for real-time panoptic segmentation. Ožujak 2022. 4

Duboki modeli za detekciju ključnih točaka osoba

Sažetak

Detekcija ključnih točaka osoba važno je područje računalnog vida sa zanimljivim primjenama. Opisanim postupkom moguće je modelirati gotovo sve stupnjeve slobode čovjeka. Prvi dio rada opisuje model *Cascaded Pyramid Network* koji u prvoj fazi daje grube predikcije ključnih točaka te ih u drugoj fazi rafinira. Drugi dio rada opisuje model SwiftNet i njegove dvije inačice: SwiftNet s modulom SPP i SwiftNet s piramidalnom fuzijom. Ideja je bila napraviti model koji ima male memorijske zahtjeve da ga možemo trenirati na dostupnom GPU, a da ima što bolje performanse. Oba modela koriste pristup od vrha prema dolje te je uz njih potrebno koristiti i detektor osoba u slučaju detekcije položaja više osoba na jednoj slici. Treniranje i evaluaciju provodili smo na skupu MS COCO 2017. Na ispitnom skupu model CPN postiže točnost od 67.1%. SwiftNet s modulom SPP postiže točnost od 27.4%, dok SwiftNet s piramidalnom fuzijom postiže točnost od 31.2.

Ključne riječi: Detekcija ključnih točaka osoba, CPN, SwiftNet, MS COCO

Deep models for human key-point detection

Abstract

Human key-point detection is important field of computer vision with interesting applications. Using the described method it is possible to model almost all degrees of freedom on human. The first part of paper describes two-phased model named Cascaded Pyramid Network, where the first phase predicts coarse key-points while the second phase refines detected key-points. The second part of paper describes SwiftNet model for key-point detection and its two variants: SwiftNet with SPP and SwiftNet with pyramidal fusion. Main idea was to make a model with small memory footprint that can be trained on available GPU, while trying to achieve good performance. Both models use top-down approach so we need to employ human detector in case of multi-person pose estimation. Training and evaluation was conducted on MS COCO 2017 dataset. On test dataset CPN achieves accuracy of 67.1%. SwiftNet with SPP achieves 27.4% accuracy, while SwiftNet with pyramidal fusion achieves 31.2 accuracy.

Keywords: Human key-point detection, CPN, SwiftNet, MS COCO