SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2006

Metrička ugrađivanja za reprezentiranje osoba u slikama

Marin Kačan

Zagreb, rujan 2019.

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

Zahvaljujem mentoru prof. dr. sc. Siniši Šegviću na pomoći, znanju, savjetima, uviđavnosti i dobroj volji koje je beziznimno pružao od početka naše suradnje. Zahvaljujem asistentima Marinu Oršiću i Josipu Šariću na pomoći u izvedbi zadatka ovog diplomskog rada. Zahvaljujem obitelji, curi, prijateljima i kolegama na potpori tijekom studija.

SADRŽAJ

1. Uvod

Računalni vid područje je računarske znanosti koje se bavi prikupljanjem, obradom i razumijevanjem digitalnih slika i videozapisa. Cilj je izgraditi autonomne sustave koji mogu jednako dobro ili bolje obavljati neke od zadataka koje obavlja ljudski vizualni sustav. [15]

Jedno područje primjene računalnog vida jest praćenje objekata kroz niz slika. Ovaj zadatak podrazumijeva pronalazak putanje gibanja svakog od objekata koje želimo pratiti, a koji se nalazi u nekom dijelu ulaznog niza slika. Praćenje objekata svoju primjenu nalazi u područjima međudjelovanja računala i ljudi, sigurnosti, proširene stvarnosti, obrade videozapisa i drugih.

Jedan od osnovnih elemenata inženjerskog pristupa jest razlaganje početnog problema na potprobleme od kojih su neki već riješeni ranije. O tako razloženom sustavu lakše je razmišljati, štedi se vrijeme i trud potreban za rješavanje problema, a sustav je pouzdaniji budući da se ponovno iskorištavaju dobro prokušana rješenja.

Kod nekih pristupa rješavanju prije spomenutog problema praćenja objekata možemo uočiti taj obrazac. Praćenje objekata može se razložiti na nekoliko neovisnih potproblema - detekcija objekata, procjena gibanja, mjerenje sličnosti dijelova slike, te na kraju optimalno uparivanje objekata iz trenutne slike s identitetima objekata praćenih u prijašnjim slikama. Ti se potproblemi razmatraju u ovom radu. Posebno nam je zanimljiv potproblem mjerenja sličnosti dijelova slike, a potom i potproblem detekcije objekata u slici.

Mjerenje semantičke sličnosti slika može se nadalje svesti na problem ugrađivanja slike u vektorski prostor tako da se semantički slične slike u tom prostoru grupiraju u skupine (cluster). Tako će semantički slične slike u tom novom prostoru po konvencionalnim mjerama udaljenosti (euklidska ili kosinusna udaljenost) biti blizu, dok

će semantički različite slike biti daleko. Dakle, sveli smo problem pronalaska mjere semantičke sličnosti slika na pronalazak adekvatnog ugrađivanja.

Postoje algoritmi strojnog učenja koji omogućavaju učenje takvih ugrađivanja. Ugrađivanje se može naučiti izravno [14], ali i neizravno, tako da algoritam strojnog učenja optimizira neki drugi - ali srodni - cilj, a kao ugrađivanje ulaznog podatka uzima se neki od međurezultata algoritma [33].

U sklopu ovog rada istražen je utjecaj korištenja različitih metrika sličnosti na rezultate postupka praćenja objekata. Da bi se dobio bolji dojam o važnosti i utjecaju koji različite metrike sličnosti imaju na rezultat praćenja, razmatramo i različite algoritme detekcije objekata. Zatim uspoređujemo utjecaj variranja pristupa za rješavanje potproblema pronalaska metrike sličnosti, odnosno potproblema detekcije na rezultate glavnog problema praćenja.

2. Osnovni pojmovi i postojeće metode

2.1. Metrička ugrađivanja

Ugrađivanje je preslikavanje koje sažima ulazne objekte u kontinuirani vektorski prostor koji je često niže dimenzije. Ugrađivanja su nam zanimljiva ako reprezentacije objekata koje nam ona daju čuvaju neka svojstva ulaznih objekata i međusobne odnose objekata koja su važna za naš zadatak, a do neke mjere ignoriraju varijacije ulaznih objekata koje za naš problem nisu važne.

Ugrađivanja se mogu učiti izravno [14], ali često se kao ugrađivanje ulaznog podatka koristi neki od međurezultata nekog drugog zadatka dubokog učenja (npr. zadnji sloj prije izlaznog sloja mreže učene za problem klasifikacije može se koristiti kao reprezentacija ulaznog podatka) [33].

Metričko je ugrađivanje ono koje preslikava ulazne podatke u metrički prostor. [9] Udaljenosti između preslikanih točaka obično mjerimo nekom od uobičajenih metrika (npr. euklidska ili kosinusna udaljenost).

U dubokom učenju, metričko je ugrađivanje parametrizirano i cilj je naučiti parametre tako da ugrađivanje bude što bolje.

Za izravno učenje metričkih ugrađivanja osmišljene su posebne arhitekture i funkcije gubitka. Najčešće se koriste sijamske mreže s kontrastnim [11] ili trojnim gubitkom [19]. Funkcija kontrastnog gubitka nastoji minimizirati udaljenost između primjera iste klase i održati fiksni razmak (marginu) između primjeraka različitih klasa.

Funkcija trojnog gubitka nije toliko stroga, te ne zahtijeva da je udaljenost primjera iste klase minimalna, dok je god udaljenost između primjera različitih klasa veća od udaljenosti između primjera iste klase barem za zadani fiksni iznos (marginu). [33]

Kod ovakvih načina treniranja dobivanje kvalitetnog skupa ulaznih podataka nekad može biti netrivijalno. Naime, treba uzorkovati parove slika istih i različitih klasa za kontrastni gubitak, odnosno trojke od kojih su dvije slike iz iste klase, a treća iz različite za trojni gubitak. [33]

Ugrađivanja se mogu dobiti i treniranjem mreža za rješavanje drugih problema, npr. klasifikacije. Ugrađivanje za reprezentaciju osoba u slikama može se neizravno dobiti učenjem klasifikacijske mreže za zadatak ponovne identifikacije (re-identification) osoba. [33] Jednom kada je mreža naučena, kao ugrađivanje može se uzeti izlaz zadnjeg skrivenog sloja, za koji očekujemo da sadrži sažetu reprezentaciju osobe podatnu za razlikovanje osoba budući da ta informacija treba i izlaznom sloju klasifikacijske mreže za uspješnu ponovnu identifikaciju osobe.

2.2. Praćenje objekata u sceni

Cilj praćenja objekata je lokalizirati objekte zadanih klasa koji se miču u nizu slika te povezati lokacije istih objekata u svim slikama iz niza u kojima se pojavljuju. Time se za svaki objekt klase koja nas zanima dobiva putanja kroz niz slika.

Objekti mogu izaći iz praćene scene i vratiti se u nju, stvoriti se usred scene, biti djelomično prekriveni u dijelu niza slika, mijenjati oblik, orijentaciju, brzinu i smjer kretanja.

Kod praćenja više objekata (multi-object tracking) u svakoj slici možemo imati različitih broj objekata zadanih klasa. Ti se objekti trebaju pratiti kroz niz slika kao pravokutni okviri (bounding boxes) s prepoznatom točnom klasom. [32]

Najveći dio istraživanja praćenja više objekata usmjeren je na praćenje pješaka (pedestrians), te je postupak praćenja koji razmatramo u ovom radu isto primijenjen na taj problem. Pješake nisu kruta tijela, nego stalno mijenjaju oblik dok se kreću, a imaju i drukčiji izgled ovisno o njihovoj orijentaciji u odnosu na kameru. Sve to čini praćenje pješaka zanimljivim i zahtjevnim zadatkom. Uz to, praćenje pješaka ima velik broj praktičnih primjena kao što su vizualni nadzor, virtualna i proširena stvarnost, automatska analiza sportskih događaja itd. [25]

Budući da postupak praćenja mora u nizu slika locirati objekte zadanih klasa (detektirati ih), često se u sklopu rješavanja problema praćenja koristi postupak detekcije objekata, te se algoritam praćenja na to oslanja (tracking by detection), iako to ne mora biti slučaj (detection-free tracking). [25]

Ovdje valja spomenuti i važnost vremenske efikasnosti postupka praćenja. Naime, u nekim primjenama nije nužno da se praćenje odvija u stvarnom vremenu, te se algoritam praćenja može provesti nakon snimanja niza slika (offline). Prednosti tog pristupa su to što se za dodjeljivanje identiteta objektima određena slike niza mogu razmatrati i buduće slike iz tog niza, tj. putanje objekata mogu se dobiti uzimajući u obzir čitav niz. Uz to, ako se u postupku koristi detekcija objekata, nema ograničenje da postupak detekcije objekata mora biti brz, što omogućava korištenje boljih postupaka detekcija. S druge strane, u praksi je često zanimljiviji problem praćenja u stvarnom vremenu. Tada se za svaku sliku niza odluka mora donositi odmah, uzimajući samo prijašnje slika niza u obzir, a postupak detekcije objekata mora trajati kraće od vremenskog razmaka između pristizanja dviju susjednih slika niza. [25, 2]

3. Duboko učenje i konvolucijske neuronske mreže

3.1. Duboko učenje i neuronske mreže

Duboko učenje je grana strojnog učenja koje je proteklih godina bila iznimno uspješna u rješavanju različitih problema, osobito u području računalnog vida. Kod dubokog učenja uče se složene reprezentacije različitih vrsta podataka.

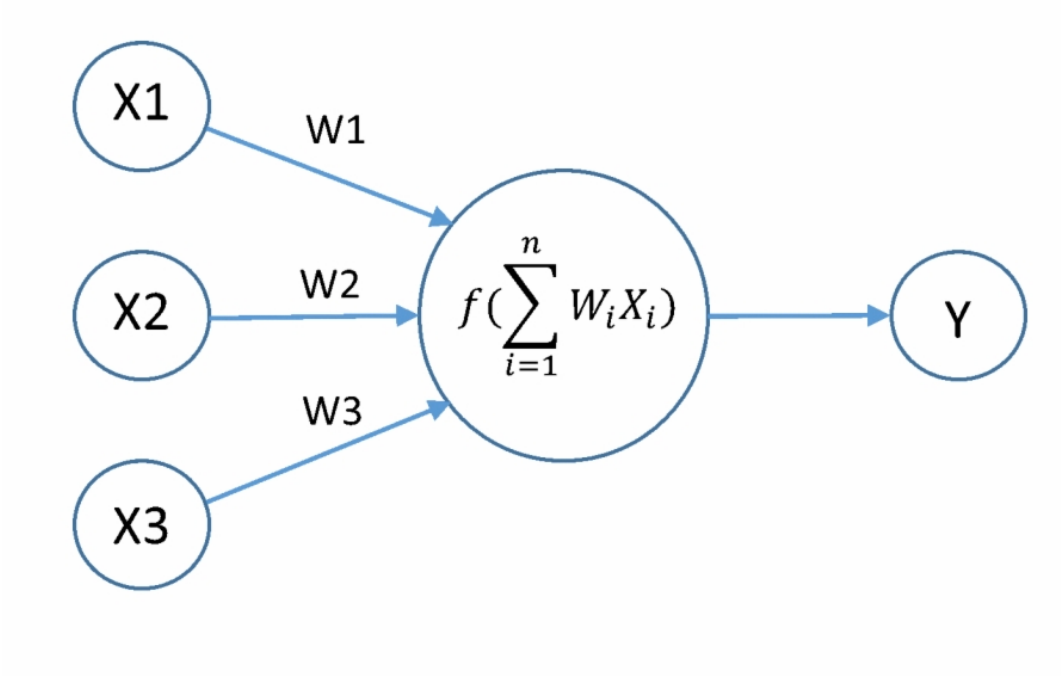
3.1.1. Neuroni i neuronske mreže

Umjetne neuronske mreže su modeli koji se sastoje od međusobno povezanih neurona. Neuron je modul koji radi nelinearnu transformaciju ulaznog podatka. To je matematička funkcija koja ulazni vektor realnih brojeva transformira parametriziranom linearnom transformacijom, primjenjuje nelinearnu aktivacijsku funkciju i na izlazu daje jednu realnu vrijednost.

To se može predstaviti jednadžbom:

$$y = f(\vec{x} \cdot \vec{w} + b) = f\left(\sum_{i=1}^{n} x_i \cdot w_i + b\right) \tag{3.1}$$

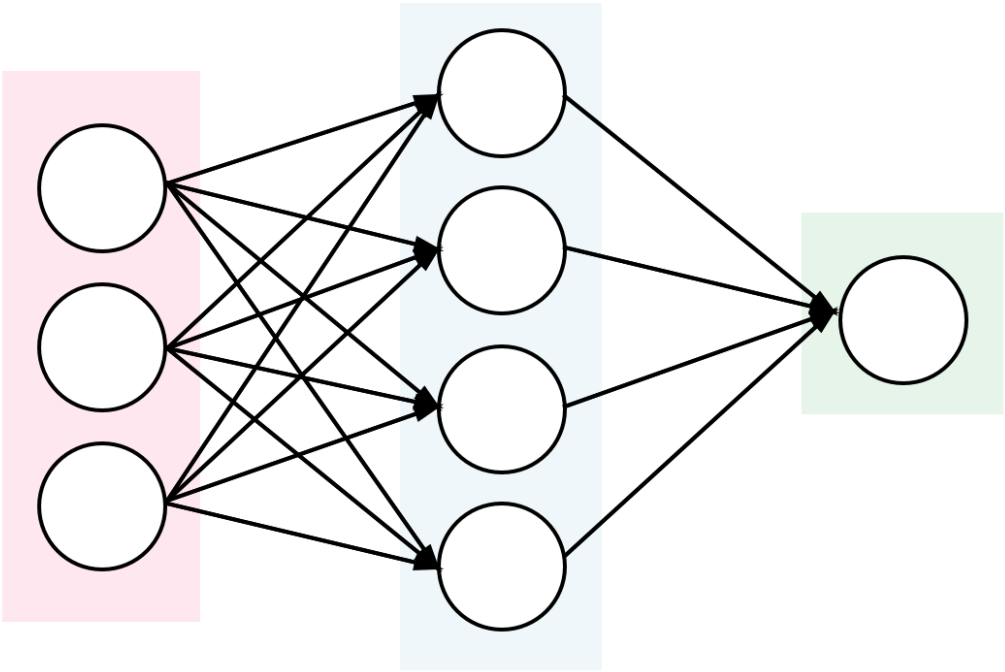
gdje je $\vec{x}$ ulazni vektor, $\vec{w}$ vektor težina, a b prag neurona (sl. 3.1) [10].



Slika 3.1: Struktura umjetnog neurona

Obično se neuronske mreže organiziraju u grupe neurona koje nazivamo slojevima (sl. 3.2). Rekli smo da svaki neuron uzima vektor realnih brojeva i transformira ga u jednu realnu vrijednost. Sloj neuronske mreže kao izlaz dat će vektor realnih brojeva onolikih dimenzija koliko ima neurona. Svaki element izlaznog vektora odgovarat će izlazu jednog od neurona. Ulaz svakog od neurona u istom sloju je isti (cijeli ulazni vektor). Parametri svakog neurona slobodni su i neovisni o parametrima ostalih neurona u sloju. Aktivacijska funkcija je zajednička, tj. svaki neuron u sloju nakon linearne transformacije primjenjuje istu aktivacijsku funkciju.

Slaganjem takvih slojeva neurona dobiva se duboka neuronska mreža koja može naučiti aproksimirati vrlo složene funkcije. Početni slojevi mreže obično se nauče aktivirati na značajke niske razine apstrakcije (npr. u računalnom vidu, detekcija različito orijentiranih rubova). Kasniji slojevi na temelju prethodnih reprezentacija uče reprezentacije više razine apstrakcije. U računalnom vidu, kombiniranjem istovremene detekcije više različitih rubova u nekom međusobnom odnosu detektira se neki složeniji oblik prisutan u slici. [38]



Slika 3.2: Potpuno povezana neuronska mreža s jednim skrivenim slojem

Upravo smo opisali potpuno povezane neuronske mreže. Kod potpuno povezane neuronske mreže slojevi se slažu slijedno. Osim prvog sloja, koji na ulaz dobiva vektor ulaznog podatka, ulaz svakog sljedećeg sloja je izlaz prethodnog sloja. Izlaz mreže je izlaz posljednjeg njezinog sloja. U takvoj mreži, informacija struji u jednom smjeru, od ulaza prema izlazu, bez povratnih veza. Postoje i neke druge inačice dubokih mreža od kojih ćemo kasnije opisati konvolucijske neuronske mreže.

3.1.2. Aktivacijska funkcija

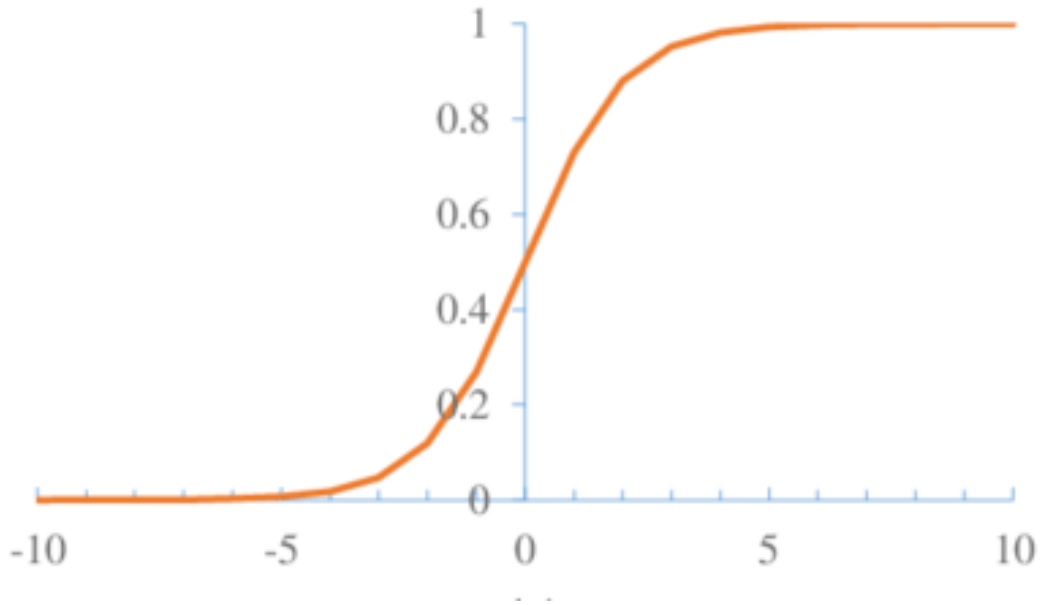
Na izlazima slojeva neuronskih mreža koriste se aktivacijske funkcije da bi se u model unijela nelinearnost. Bez njih bi mreža mogla naučiti samo linearnu granicu između podataka različitih klasa.

Najpoznatije aktivacijske funkcije su: sigmoidalna funkcija, tangens hiperbolni, ReLU (engl. rectified linear unit) i LeakyReLU. Izrazi za navedene funkcije i grafovi tih funkcija prikazani su ispod:

Sigmoidalna aktivacijska funkcija [27] (sl. 3.3) na izlazu daje vrijednosti iz intervala od 0 do 1, kao što prikazuje sljedeća jednadžba:

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.2}$$

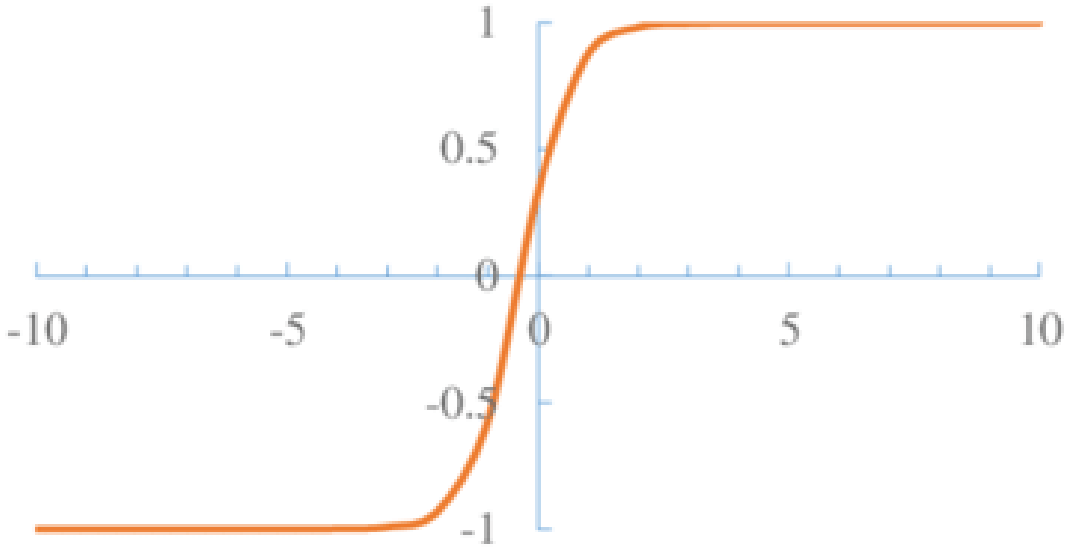
Važan nedostatak sigmoide kao aktivacijske funkcije je zasićenje koje se događa kada nezavisna varijabla (argument funkcije) ima malo veću apsolutnu vrijednost. U tom slučaju je gradijent funkcije blizu nuli, te se radi o problemu nestajućeg gradijenta (engl. vanishing gradient). Neuron koji se nađe u zasićenju jedva propušta gradijent prilikom unatražnog prolaza.



Slika 3.3: Graf sigmoidalne funkcije

Tangens hiperbolni [27] (sl. 3.4) sličan je sigmoidi (funkciju dobivamo tako da skaliramo sigmoidalnu funkciju tako da joj kodomena bude (-1, 1)) i također pati od problema zasićenja i posljedičnog nestajućeg gradijenta, iako se obično ponaša bolje od sigmoide. Za razliku od sigmoide, izlaz tangensa hiperbolnog centriran je oko nule.

$$f(x) = \tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{3.3}$$



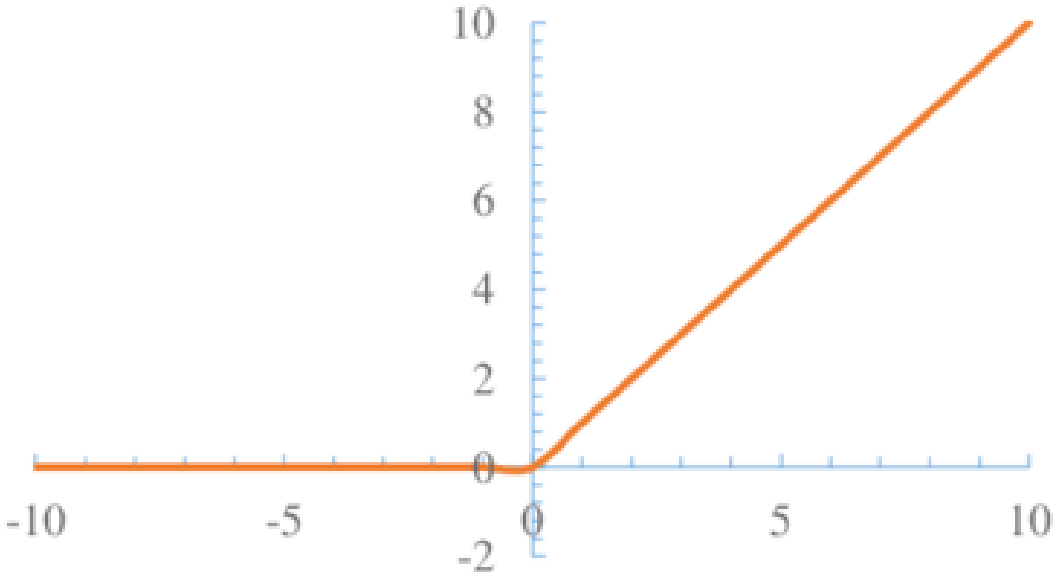
Slika 3.4: Graf funkcije $tanh$

Najčešće korištena aktivacijska funkcija u dubokom učenju je ReLU (engl. Rectified Linear Unit).[27] Istaknimo dva razloga za to. Prije svega, opažena je značajno brža konvergencija modela u slučajevima kada se koristi ReLU u odnosu na prije spomenute aktivacijske funkcije. Budući da ulaze veće od nule samo propušta, u svom pozitivnom dijelu funkcija nema zasićenja.

$$f(x) = ReLU(x) = \max(0, x) \tag{3.4}$$

Problem koji se može javiti kod funkcije ReLU je problem umirućih neurona. Naime, za negativne brojeve na ulazu, izlaz aktivacijske funkcije je nula (sl. 3.5), neuron postaje neaktivan i pri prolasku unatrag gradijent ne teče kroz taj neuron. [24]

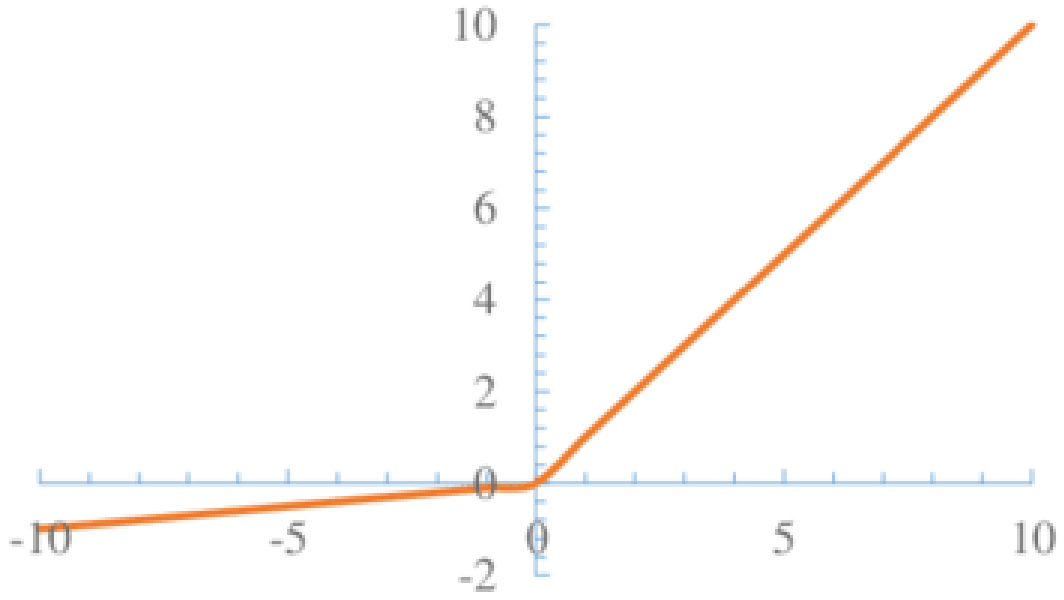
S druge strane, želimo određen broj neaktivnih neurona pri svakom prolasku unaprijed, budući da u protivnom u mreži ne bi bilo linearnosti (isto kao da je aktivacijska funkcija funkcija identiteta).



Slika 3.5: Graf funkcije ReLU

Aktivacijska funkcija može se modificirati tako da se za negativne ulaze ne vraća nula, nego se na tom intervalu aktivacijska funkcija ponaša također kao linearna funkcija, ali s nagibom α manjim od 1 (sl. 3.6). Time se može uhvatiti u koštac s problemom umirućih neurona. Ipak, budući da rezultati nisu uvijek i značajno bolji, aktivacijska funkcija ReLU ostaje i dalje najčešće korištena.

$$f(x) = LeakyReLU(x) = \max(\alpha x, x) \tag{3.5}$$



Slika 3.6: Graf funkcije LeakyReLU

3.1.3. Učenje i optimizacijski postupci

Neuronska mreža diferencijabilnim matematičkim operacijama transformira ulazni podatak. Te su operacije parametrizirane težinama.

U postupku učenja želimo da su izlazi naše mreže što bliži očekivanom izlazu za svaki ulaz iz skupa za učenje. Skup za učenje samo je mali uzorak stvarne distribucije podataka, ali se nadamo da će mreža koja daje točne izlaze za podatke iz skupa za učenje davati točne izlaze i za podatke iz stvarne distribucije neviđene za vrijeme učenje, tj. da će dobro generalizirati.

Da bismo kvantificirali koliko su izlazi mreže blizu očekivanim izlazima za ulazne podatke iz skupa za učenje definiramo funkciju gubitka [17]:

$$L(f(\boldsymbol{x};\boldsymbol{\Theta}),y) = \frac{1}{N}\sum_{i=1}^{N} l\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\Theta}\right),y^{(i)}\right) \tag{3.6}$$

gdje je $l\left(f\left(\boldsymbol{x}^{(i)};\boldsymbol{\Theta}\right),y^{(i)}\right)$ funkcija gubitka izračunata za jedan primjer iz skupa za učenje.

Učenje neuronske mreže postupak je u kojem iterativno mijenjamo slobodne parametre (težine) nastojeći smanjiti ukupni gubitak na skupu za učenje. Budući da je

funkcija gubitka diferencijabilna, a postoji postupak efikasnog računanja gradijenata funkcije gubitka po svim parametara neuronske mreže (algoritam širenja unatrag), najčešće se za optimizaciju parametara mreže koristi postupak gradijentnog spusta.

Intuitivno, budući da gradijent pokazuje u smjeru u kojem funkcija najbrže raste, suprotni je smjer najbržeg pada funkcije. Kako mi želimo minimizirati funkciju, optimizacijski se postupak sastoji od uzastopnih malih pomaka u smjeru suprotnom od smjera gradijenta.

Najčešće se radi ubrzavanja učenja gradijent za jedan pomak ne računa nad cijelim skupom za učenje nego se aproksimira nad malim, nasumično izabranim, podskupom (engl. mini batch) iz skupa za učenje, te tada govorimo o stohastičkom gradijentnom spustu (engl. stochastic gradient descent, SGD).

Pokazuje se da takvim ubrzanjem postupak učenja puno više profitira nego što pati od aproksimacije gradijenta umjesto njegovog točnog izračuna. U prilog tome ide i činjenica da je ovdje riječ o optimizaciji nekonveksne funkcije. U tom slučaju, stohastičnost i šum uvedeni korištenjem mini grupa mogu pomoći optimizacijskom postupku da ne zapne u sedlastom području ili lokalnom minimumu.

Kod uobičajenog postupka SGD parametar kojim skaliramo vektor pomaka (stopa učenja) cijelo je vrijeme isti. U novije vrijeme, popularne su modifikacije osnovnog algoritma SGD koje uključuju promjenjivu stopu učenja. [6, 37]

Uz to, za brže i uspješnije izvlačenje iz lokalnih optimuma, kanjona i sedlastih ploha korisno je u osnovni postupak SGD ubaciti zalet. Moment je skup vrijednosti akumuliranih prošlih vrijednosti gradijenata (s većom važnosti pridanom vrijednostima novijih gradijenata) koji pokazuje generalni smjer pomaka po parametrima i omogućava optimizacijskom postupku da bude otporniji na nagle, šumovite promjene smjera, lokalne optimume i ravna područja u kojima je gradijent malen.

Algoritam Adam koji koristimo prilikom učenja modela korištenih u ovom radu nadograđuje osnovni algoritam stohastičkog gradijentnog spusta tako da koristi obje navedene ideje - promjenjivu stopu učenja i moment.

3.2. Konvolucijske neuronske mreže

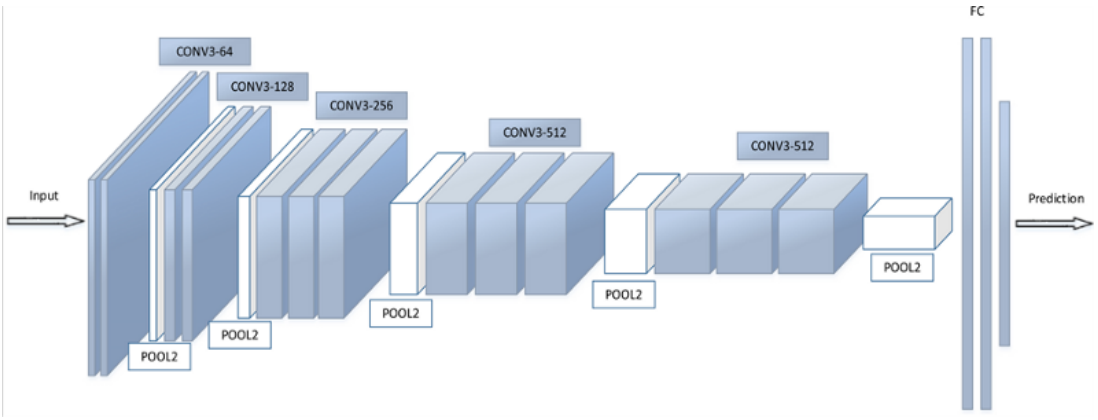
3.2.1. Motivacija

Skriveni slojevi do sada razmatranih mreža isključivo su potpuno povezani slojevi. Međutim, u dubokom učenju postoje i drugi tipovi slojeva među kojima su i konvolucijski slojevi. Konvolucijska neuronska mreža je mreža koja u sebi ima barem jedan konvolucijski sloj. Konvolucijske su mreže svoju najuspješniju primjenu doživjele upravo u računalnom vidu, ali koriste se i u ostalim područjima u kojima je duboko učenje našlo primjenu.

Dijelom inspirirane konvolucijskim filtrima korištenim u klasičnom računalnom vidu (npr. za detekciju rubova), dijelom ljudski vizualnim sustavom, konvolucijske neuronske mreže mogu se promatrati i kao varijanta potpuno povezanih mreža s određenim ograničenjima.

Konvolucijske neuronske mreže pretpostavljaju translacijsku ekvivarijantnost po prostornim dimenzijama ulaznog podatka. [3] To znači da želimo da se translacija nekog objekta prisutnog u ulaznoj slici manifestira kao translacija aktivacije neurona koja odgovara tom objektu u izlaznoj mapi značajki. Uz to, konvolucijske mreže pretpostavljaju lokalnost, tj. pojedini dio izlazne mape značajki ovisi samo o jednom malom prozoru ulazne slike na odgovarajućoj lokaciji. [20] Te dvije pretpostavke implicitno ugrađujemo u model ako potpuno povezani sloj zamijenimo filtrom koji radi kao potpuno povezani sloj, ali samo na malom prozoru ulazne slike, te taj isti filter pomičemo po cijeloj slici kao klizeći prozor i računamo njegov izlaz za svaku lokaciju.

Broj parametara konvolucijskog sloja značajno je manji u odnosu na potpuno povezani sloj upravo zbog toga što je filter malih dimenzija u odnosu na ulaznu sliku i zato što se težine dijele, tj. filter s istim težinama ide po cijeloj slici.



Slika 3.7: Nacrt mreže koja ulaznu sliku transformira naizmjeničnim konvolucijskim slojevima i slojevima sažimanja. Prostore dimenzije transformiranog podatka postepeno se smanjuju, dok semantička dimenzija (dubina) raste. Na kraju, dobivena se mapa značajki izravna u vektor koji se dalje transformira potpuno povezanim slojevima. [5]

3.2.2. Konvolucijski sloj

Konvolucijski sloj dobio je ime po matematičkoj operaciji konvolucije.

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{3.7}$$

Točnije, operacija koju konvolucijski sloj zapravo provodi je diskretna dvodimenzionalna unakrsna korelacija. Budući da su na početku učenja filteri nasumično inicijalizirani i podešavaju se postupkom učenja, ta distinkcija nije odviše važna.

Matematički, operaciju koju provodi taj sloj, tj. element (m, n) izlazne mape značajki zapisujemo:

$$(I * K)(m,n) = \sum_{c} \sum_{i=-(k_1-1)/2}^{(k_1-1)/2} \sum_{j=-(k_2-1)/2}^{(k_2-1)/2} I(m+i, n+j, c)K(m+i, n+j, c) \tag{3.8}$$

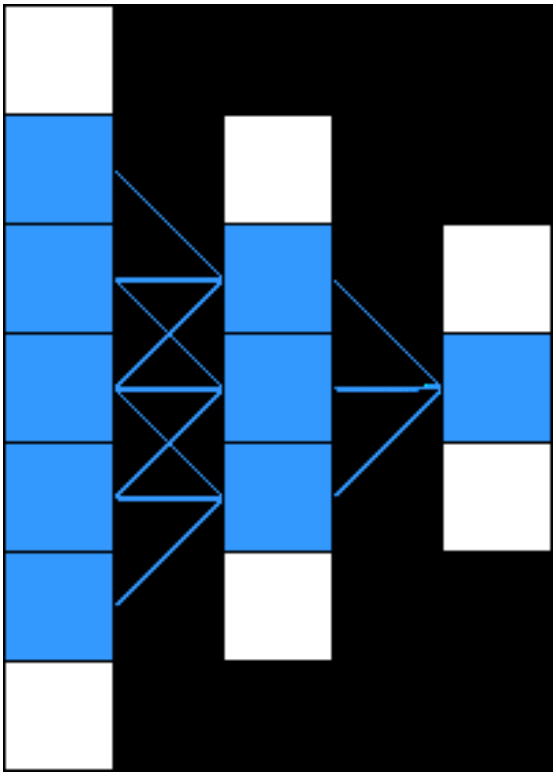
gdje je I ulazna slika ili mapa značajki s dvije prostorne dimenzije (M i N) i jednom semantičkom dimenzijom, dubinom C, a K je konvolucijski filter ili jezgra također s dvije prostorne dimenzije (k_1, k_2) i semantičkom dimenzijom jednake dubine kao i ulazna mapa.

Jedan konvolucijski sloj može imati više konvolucijskih filtera (ili jezgri), te svaki od njih na izlazu proizvodi mapu značajki dubine 1. Da bi prostorne dimenzije mape značajki ostale jednake prostornim dimenzijama ulazne mape, često se kao korak pomicanja klizećeg prozora uzima vrijednost 1, a ulazna se mapa nadopunjuje okvirima koji omeđuju mapu, najčešće popunjenim elementima vrijednosti 0. Dodaje se onoliki broj okvira koliko je potrebno, ovisno o dimenziji jezgre. [7] Općenito, prostor dimenzija izlazne mape ovisi o veličini koraka (engl. stride) S, koristi li se nadopunjavanje (engl. padding) P, te o prostornoj dimenziji konvolucijske jezgre K (obično su jezgre kvadratne, tj. jednake veličine u obje prostorne dimenzije). Dimenzije izlazne mape računamo sljedećim izrazom:

$$D_{out} = \frac{D_{in} - K + 2P}{S} + 1 \tag{3.9}$$

Ako se zareda više uzastopnih konvolucijskih slojeva, daljnji konvolucijski slojevi kao ulaz primaju mapu značajki prethodnog sloja, te se tako kao i kod potpuno povezanih slojeva u dubljim slojevima mogu kombinirati detektirani primitivni oblici i tako detektirati elementi slike na višim razinama apstrakcije.

Rekli smo da svaki element izlazne mape značajki ovisi samo o malom prozoru ulazne mape. Dakle, u prvom konvolucijskom sloju mreže, jedan neuron "vidi" samo mali dio ulazne slike, tj. njegovo je receptivno polje maleno. Daljnji slojevi vide isto tako mali dio izlazne mape prethodnog sloja, ali budući da svaki element te ulazne mape vidi jedan dio sloja prije, ukupno receptivno polje u odnosu na ulaznu sliku je veće. Stoga dublji slojevi mogu prepoznavati objekte puno veće od prostornih dimenzija konvolucijske jezgre.



Slika 3.8: Pojednostavljeni prikaz receptivnog polja konvolucijske mreže za jednodimenzionalni ulazni podatak. Prostorna dimenzija jezgre je 3. Izlaz neurona drugog sloja (treći vektor, gledajući s lijeva nadesno) ovisi o tri elementa izlaza prvog sloja. Svaki od izlaza prvog sloja ovise o daljnja tri elementa ulaza. Budući da je korak konvolucije jednak 1, što je manje od prostorne dimenzije jezgre, neki se elementi tih trojki preklapaju. To preklapanje rezultira time da, kao što vidimo na slici, izlaz neurona drugog sloja ne ovisi o 9, nego o 5 elemenata ulaza prvog sloja. Kažemo da je receptivno polje tog neurona veličine 5. Dakle, usprkos tome što je prostorna dimenzija jezgre samo 3, neuroni u kasnijim slojevima mreže "gledaju" na sve veće dijelove ulaza. [35]

3.2.3. Normalizacija po grupi

Normalizacija po grupi postupak je kojim se nastoji ubrzati i poboljšati učenje neuronskih mreža. Opaženo je da kod učenja mreža s više slojeva, u dubljim slojevima postoji problem. Naime, kako se unatražnim prolazom mijenjaju svi parametri, tako se i distribucije izlaza svih slojeva mijenjaju tijekom učenja. Prvom sloju mreže distribucija ulaznih vrijednosti bit će tijekom cijelog učenja ista jer je to distribucija vrijednosti ulaznih podataka koja se ne mijenja kako se parametri mreže mijenjaju. Međutim, distribucija ulaznih vrijednosti nekog dubljeg sloja mijenja se stalno tijekom učenja kako se mijenjaju težine svih slojeva koje mu prethode. To otežava učenje dubljih slojeva

koji se stalno moraju prilagođavati na promjenu distribucije ulaza. U praksi se pokazuje da slojevi lakše uče kada im je distribucija ulaznih podataka normalizirana.

Normalizacija je adaptivna transformacija podataka tijekom učenja duboke mreže. Vrijednosti ulaznih podataka nekog sloja transformiraju se u podatke sa srednjom vrijednošću jednakom nula i varijancom jednakom jedan.

Da bi se mogle izračunati srednja vrijednost i varijanca ulaznih vrijednosti, potrebno je više od jednog ulaznog podatka, te se te statistike računaju se na temelju jedne mini grupe.

Postoji opasnost da se normaliziranjem izgubi ekspresivnost sloja, te se zato nakon normaliziranja, rezultatima dodaje pomak β i skalira ih se s γ, gdje se ti parametri također uče u procesu učenja.

Za mini grupu ulaznih podataka $\left\{x^{(i)}, \ldots, x^{(n)}\right\}$, srednja vrijednost bit će:

$$\mu_B \leftarrow \frac{1}{n} \sum_{i=1}^{n} x_i \tag{3.10}$$

i varijanca će biti:

$$\sigma_B^2 \leftarrow \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu_B)^2 \tag{3.11}$$

Ulazne podatke tada normaliziramo po izrazu:

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \eta}} \tag{3.12}$$

a nakon toga ih skaliramo i pomičemo parametrima γ i β po izrazu:

$$y_i \leftarrow \gamma \hat{x}_i + \beta \tag{3.13}$$

Nakon faze učenja, u fazi testiranja nemamo nužno mini grupe podataka i ne računamo srednju vrijednost i varijancu ulaznih podataka. Umjesto toga, koristimo fiksiranu srednju vrijednost i varijancu koje smo izračunali na svim mini grupama tokom procesa učenja, obično metodom eksponencijalnog pomičnog prosjeka. Izrazi za ažuriranje pomičnog prosjeka srednje vrijednosti i varijance tokom učenja (pomoću srednje vrijednosti i varijance mini grupe) su sljedeći:

$$\mu \leftarrow (1-\alpha)\mu_B + \alpha\mu \tag{3.14}$$

$$\sigma^2 \leftarrow (1-\alpha)\sigma_B^2 + \alpha\sigma^2 \tag{3.15}$$

gdje je α obično ima vrijednost malo manju od 1, npr. 0.999. [16]

Zbog šuma uvedenog računanjem statistika na malom broju podataka (mini grupi) i zbog normalizacije ulaznih podataka, normalizacija po grupi ima i regularizacijski učinak.

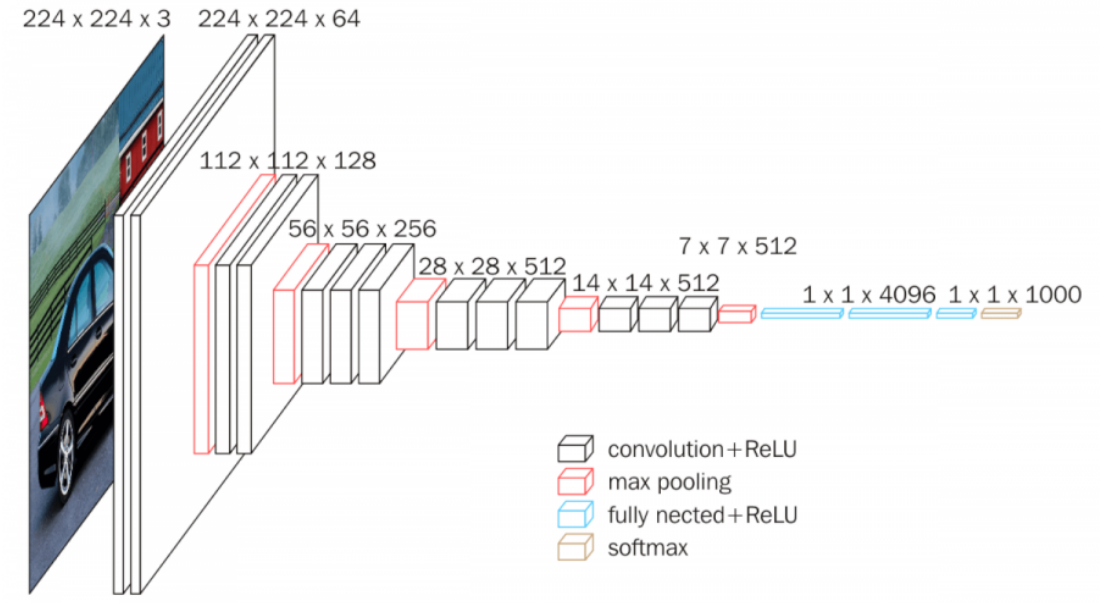
3.2.4. Konvolucijske arhitekture i prijenos znanja

Tijekom godina primjene dubokog učenja u računalnom vidu, iskristalizirale su se određene arhitekture neuronskih mreža koje su se pokazale kao uspješne pri učenju i ekstrakciji korisnih dubokih značajki iz slika. Zbog toga se te mreže često ponovno iskorištavaju tako da ih se nauči na nekom generalnom, zahtjevnom problemu računalnog vida za koji postoji puno podataka. Takva se predtrenirana arhitektura tada može iskoristiti kao prvi dio neke druge neuronske mreže dizajnirane za neki specifičniji zadatak, gdje taj predtrenirani dio služi za ekstrakciju kvalitetnih značajki iz slike. Pritom se i taj prvi predtrenirani dio može fino podešavati na specifičnom problemu, ali često se drži fiksiranim, te se fino podešava samo drugi, manji dio mreže koji kao ulaz prima izlaz predtreniranog modula. Opisana paradigma ima naziv prijenos znanja ili prijenos učenja (engl. transfer learning).

3.2.5. Arhitektura VGG-16

Jedna od poznatijih konvolucijskih arhitektura zove se VGG-16. [31]

Mreža VGG-16 na ulazu prima RGB sliku dimenzija (224, 224, 3). Mreža zatim ima niz konvolucijskih slojeva isprepletenih slojevima sažimanja sve dok izlazna dimenzija ne postane (7, 7, 512). Tada slijedi izravnavanje izlazne mape značajki u vektor, te niz potpuno povezanih slojeva koji na kraju daju izlazni vektor dimenzije broj različitih klasa klasifikacijskog problema o kojem se radi.



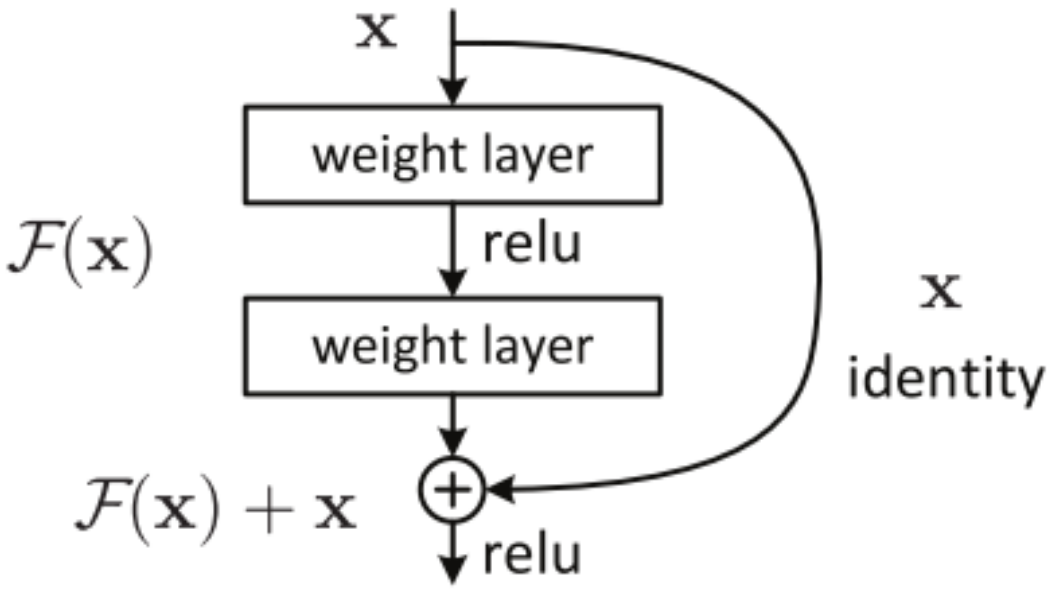
Slika 3.9: Nacrt konvolucijske arhitekture VGG-16. Na ulaz dobiva RGB sliku prostorne dimenzije 224x224. Nizom konvolucijskih i slojeva sažimanja, ulazna slika transformirana je u tenzor dimenzija 7x7x512. Tenzor se tada izravnava u vektor i prosljeđuje potpuno povezanim slojevima koji na kraju daju izlaz dimenzije N, gdje je N broj klasa u konkretnom klasifikacijskom problemu. [31] Slika je preuzeta sa stranice http://www.cs.toronto.edu/ frossard/post/vgg16/.

Jedna od neuronskih mreža za detekciju objekata koja se koristi u ovom radu i koja je bila fino podešena na skupovima za detekciju, oslanja se na arhitekturu VGG-16 predtreniranu na skupu ImageNet.

3.2.6. Arhitektura ResNet

Kako se duboko učenje razvijalo, opažen je generalni trend da su se dodavanjem sve više slojeva u neuronske mreže dobivaju bolji rezultati. Ipak, nakon određenog broja slojeva, rezultati su počeli biti gori. Tome se problemu doskočilo uvođenjem preskočnih ili rezidualnih veza (engl. skip connections) u konvolucijske slojeve. [12] Rezidualne veze funkcijom identiteta preskaču određene slojeve duboke mreže i to tako da se izlazne aktivacije jednog od prijašnjih slojeva izravno zbroje s izlazom linearne transformacije jednog od kasnijih slojeva (prije primjene aktivacije funkcije tog sloja). Na taj način, ako zaključi da su joj slojevi koje rezidualna veza preskače nepotrebni, mreža može puno lakše naučiti funkciju identiteta i to samo tako da postavi

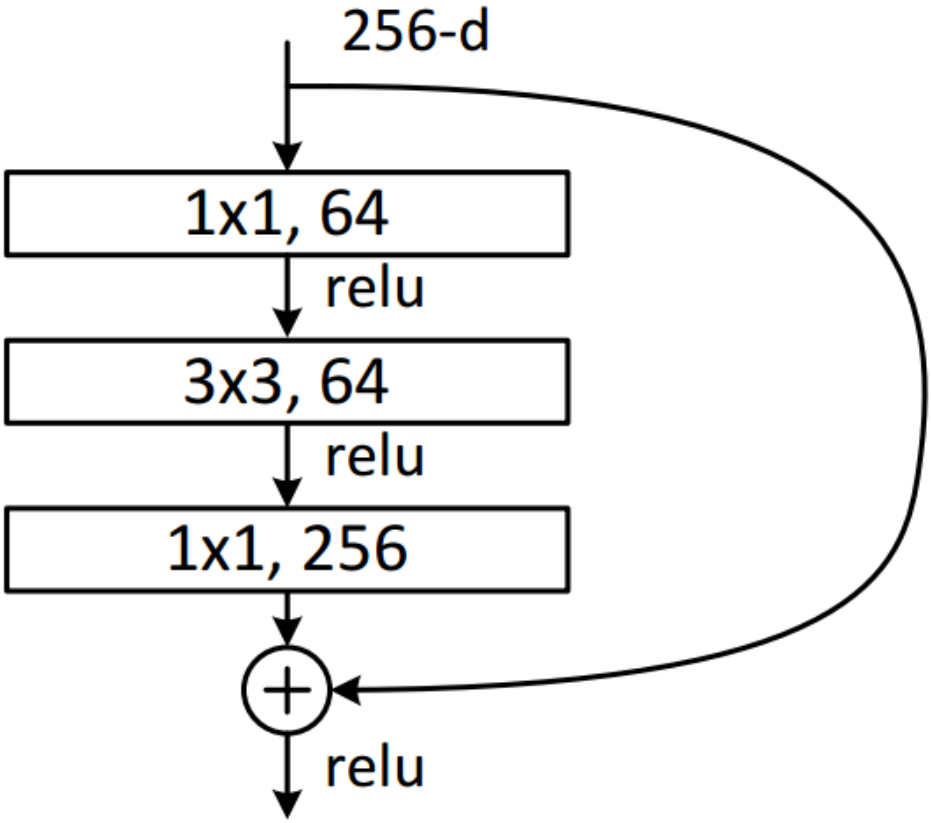
sve težine tih slojeva u nulu. Treba napomenuti da se dimenzije izvornog i odredišnog sloja rezidualne veze moraju poklapati da bi se mogli zbrojiti. Zbog toga se nekad znaju koristiti 1x1 konvolucije koje čine da slojevi koji se zbrajaju budu iste dubine. Sl. 3.10 prikazuje shemu rezidualne jedinice.



Slika 3.10: Klasični rezidualni blok s preskočnom vezom koja provodi preslikavanje funkcijom identiteta i pribraja se izlaznom vektoru nekog od kasnijih slojeva prije aktivacijske funkcije. [12]

Pokazuje se da mreže kojima su dodane rezidualne veze postižu bolje rezultate i brže konvergiraju jer se informacija s kraja mreže lakše propagira unatrag. Također, arhitektura ResNet omogućila je učenje znatno dubljih modela nego je to prije bilo moguće. Ta se arhitektura često se koristi za vađenje značajki u mnogim modelima koji rješavaju probleme iz područja računalnog vida.

Budući da se s arhitekturom ResNet počinju učiti jako duboki modeli, javlja se problem prevelikog broja parametara. Zato se često na početku i kraju rezidualne jedinice koriste 1x1 konvolucije koje na početku smanjuju, a na kraju vraćaju dubinu mape značajki, a između kojih se nalazi konvolucijski sloj čija je jezgra prostorne dimenzije veće od 1x1 (npr. 3x3). To su jedinice s uskim grlom (engl. bottleneck).



.

Slika 3.11: Rezidualna jedinica s uskim grlom koja rješava problem prevelikog broja parametara. Prvi sloj konvolucijom prostornih dimenzija 1x1 smanjuje dubinu mape značajki. Drugi sloj koristi jezgru prostorne dimenzije veće od 1x1 (npr. 3x3). Broj parametara te jezgre manji je nego što bi bio u slučaju da nije bilo smanjenja dubine mape značajki u prvom sloju. Treći sloj 1x1 konvolucijom obnavlja dubinu mape značajki na onu koju je imao ulazni podatak ove rezidualne jedinice. Preskočnom vezom ulazni se podatak prvog sloja zbraja s izlazom trećeg sloja, prije aktivacijske funkcije. [12]

3.3. Detekcija objekata

Detekcija objekata važan je problem računalnog vida i jedan od ključnih dijelova korištenog postupka praćenja objekata u sceni. Zadatak je na slici pronaći sve objekte koji pripadaju klasama koje nas zanimaju tako da za svaki objekt znamo njegovu lokaciju određenu graničnim okvirom (problem lokalizacije) i klasu (problem klasifikacije).

U radu smo koristili mreže FasterRCNN [29] i MaskRCNN [13] koje su među najčešće korištenim pristupima za rješavanje problema detekcije. Ukratko ćemo ih u nas-

tavku opisati.

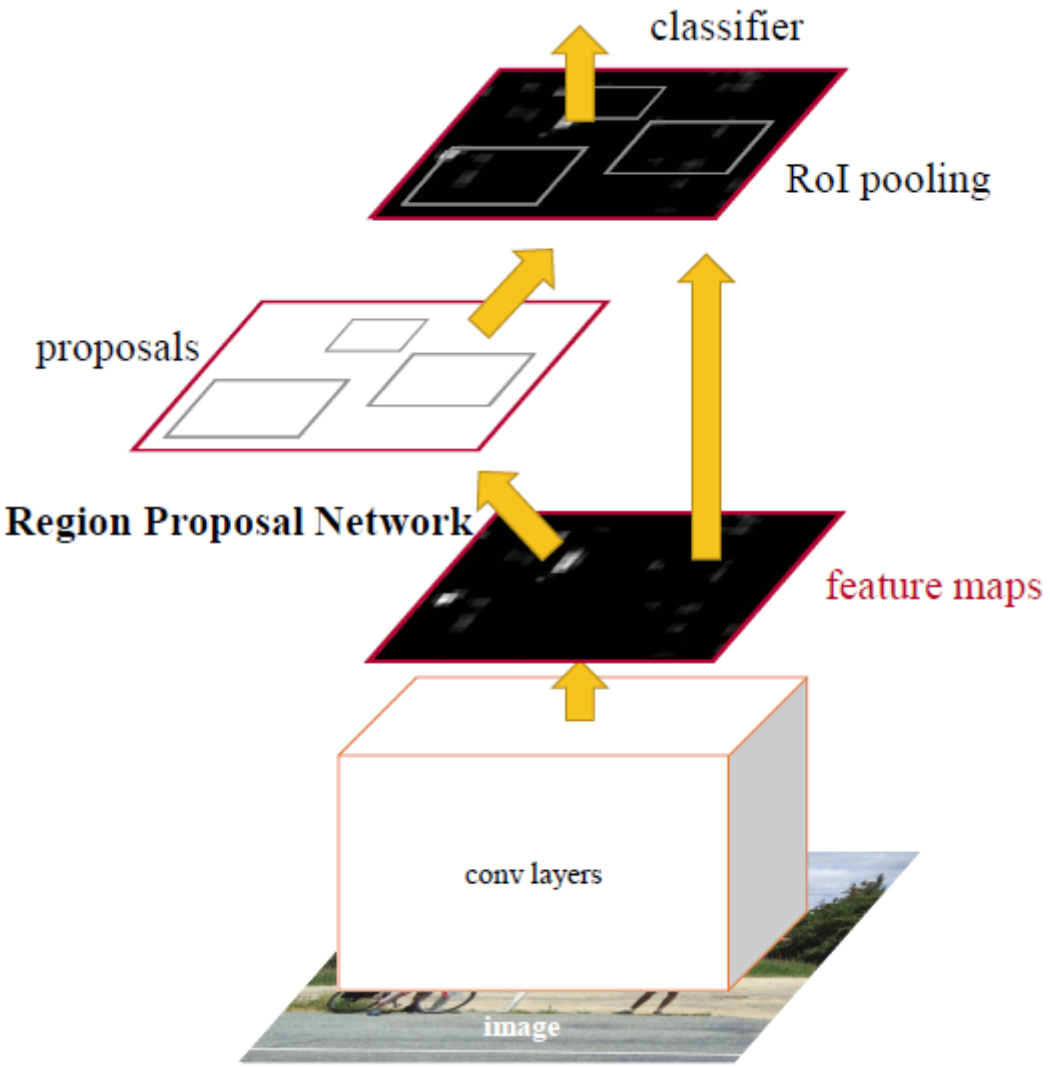
3.3.1. Mreže FasterRCNN i MaskRCNN

Kako je zadatak detekcije objekata dvojak - sastoji se od lokalizacije i klasifikacije tako će se i mreža FasterRCNN granati u dva smjera za svaki od tih zadataka, ali tek nakon zajedničkog prvog dijela mreže koji proizvodi mapu značajki.

Dakle, ulazna slika prvo prolazi kroz niz konvolucijskih slojeva i slojeva sažimanja. Najčešće je riječ o nekoj od poznatih konvolucijskih arhitektura (npr. VGG-16).

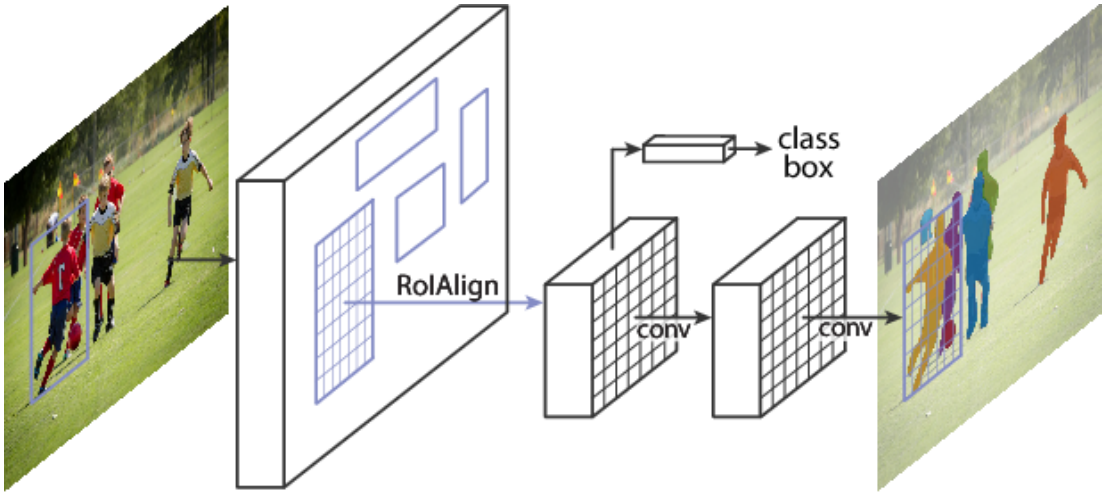
Nakon toga se iz dobivene mape značajki treba dobiti prijedloge regija gdje bi se mogli nalaziti objekti koji nas zanimaju. To radi mreža za predlaganje regija (engl. region proposal network, RPN). Ona može predlagati granične okvire različitih veličina i omjera dimenzija (tzv. sidra) za svaki djelić slike.

Nakon što su prijedlozi regija dobiveni, oni se izrezuju iz mape značajki, te se sažimanjem regije interesa (engl. *RoI pooling*) oblikuju u vektore iste fiksne veličine. Takvi vektori idu u potpuno povezane slojeve koji obavljaju klasifikaciju prijedloga tih objekata (klasa objekta ili pozadina) i regresiju pomaka (korekcije) graničnog okvira prijedloga tako da granični okvir bolje obuhvaća detektirani objekt. Skica mreže dana je u nastavku:



Slika 3.12: Nacrt mreže Faster R-CNN. Faster R-CNN objedinjuje dvije faze u postupku detekcije objekata - predlaganje regija u kojima se nalaze objekti i detekciju objekata (klasifikacija i pomak okvira) unutar tih regija. [29]

Mreža MaskRCNN nasljednik je mreže FasterRCNN, te ona uz detekciju provodi i segmentaciju detektiranih objekata. Jedno od najvažnijih unaprijeđenja je zamjena sloja sažimanja regije interesa slojem poravnatog sažimanja regije interesa (engl. *RoI align*). Taj sloj korištenjem bilinearne interpolacije izbjegava gubljenje preciznosti prilikom zaokruživanja na cijele brojeve koje se događa kada se realne koordinate graničnog okvira preslikavaju na mapu značajki i kada se takav dobiveni okvir dijeli na dijelove koji se sažimaju, što je bio problem kod sloja *RoI pooling*. Skica mreže MaskRCNN dana je u nastavku:



Slika 3.13: Nacrt mreže Mask R-CNN. Mask R-CNN dodaje modul za predviđanje segmentacijske maske objekta. [13]

3.4. Metrička ugrađivanja

3.4.1. Softmax sloj i gubitak unakrsne entropije

Kod problema višeklasne klasifikacije, najčešće se kao aktivacijska funkcija izlaznog sloja mreže koristi funkcija *softmax*.[27] Ona ulazni vektor realnih brojeva preslikava u vektor koji predstavlja vjerojatnosnu distribuciju (zbroj elemenata izlaznog vektora je 1) gdje najveću vjerojatnost ima element koji je u ulaznom vektoru imao najveću vrijednost, a najmanju vjerojatnost onaj koji je u ulaznom sloju imao najmanju vrijednost.

Funkcija koju provodi izlazni sloj mreže je:

$$p(y = k|\boldsymbol{s}) = \frac{\exp(\boldsymbol{s}_k)}{\sum_{n=1}^{C} \exp(\boldsymbol{s}_n)} \tag{3.16}$$

Ovdje vektor $\boldsymbol{s}$ možemo promatrati kao ugrađivanje, tj. kao sažetu reprezentaciju ulaznog podatka, na temelju kojeg izlazni sloj radi klasifikaciju.

Funkcija gubitka kojom se najčešće trenira mreža kojoj je izlazni sloj *softmax* jest ona funkcija koja se dobije ako se pokuša maksimizirati izglednost (zapravo minimizirati negativnu log-izglednost) točne klase. To je unakrsna entropija između prave distribucije oznaka (*one-hot* vektor s jedinicom na mjestu točne klase) i distribucije koje je na izlazu dala mreža:

$$\mathcal{L}(\mathcal{D}) = -\sum_{k=1}^{C} \mathbb{1}\{y_{true} = k\} \log p\left(y = k|\boldsymbol{s}\right) \tag{3.17}$$

ili, jednostavnije:

$$\mathcal{L}(\mathcal{D}) = -\log p\left(y = y_{true}|\boldsymbol{s}\right) \tag{3.18}$$

3.4.2. Cosine softmax

Već smo rekli da izlaz zadnjeg sloja mreže prije aktivacijske funkcije *softmax* možemo promatrati kao sažetu reprezentaciju ulaznog podatka. Kada se takva mreža trenira gubitkom unakrsne entropije, težine se mreže mijenjaju tako da postepeno guraju reprezentacije primjera neke klase što dalje od decizijske granice klasifikatora. S druge strane, taj proces neće nužno reprezentacije svih primjera istog razreda gurati u jednu kompaktnu grupu (cluster). [33]

Uz nekoliko izmjena standardnog *softmax* klasifikatora, Wojke i Bewley [33] tvrde da se može dobiti mreža kod koje će primjeri istog razreda biti blizu jedni drugima po kosinusnoj sličnosti. Da bi se to postiglo, najprije se na izlaze posljednjeg skrivenog sloja primijeni L2-normalizacija tako da je duljina reprezentacijskog vektora jednaka 1:

$$\boldsymbol{r} = \frac{\boldsymbol{s}}{\sqrt{\sum_i \boldsymbol{s}_i^2}} \tag{3.19}$$

Također, vektor težina za svaku klasu normalizira se na duljinu 1:

$$\tilde{\boldsymbol{w}}_{\boldsymbol{k}} = \frac{\boldsymbol{w}_{\boldsymbol{k}}}{\sqrt{\sum_i \boldsymbol{w}_{\boldsymbol{k}i}^2}} \tag{3.20}$$

Tada izraz za kosinusni *softmax* klasifikator glasi:

$$p(y = k|\boldsymbol{r}) = \frac{\exp\left(\kappa \cdot \tilde{\boldsymbol{w}}_k^{\mathrm{T}} \boldsymbol{r}\right)}{\sum_{n=1}^{C} \exp\left(\kappa \cdot \tilde{\boldsymbol{w}}_n^{\mathrm{T}} \boldsymbol{r}\right)} \tag{3.21}$$

gdje je κ novi slobodni parametar koji služi za skaliranje. Gubitak unakrsne entropije ostaje isti.

Da bismo razumijeli zašto ovaj postupak potiče veliku kosinusnu sličnost između reprezentacija primjera istog razreda, treba primijetiti da je dobivena log-vjerojatnost nekog razreda proporcionalna upravo kosinusnoj sličnosti (skalarnom produktu dva normalizirana vektora) normalizirane reprezentacije ulaznog podatka ($\boldsymbol{r}$) i normaliziranog vektora težina koji odgovara tom razredu ($\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$). Maksimiziranjem te log-vjerojatnosti, maksimizirati će se i kosinusna sličnost spomenuta dva vektora.

Pokušajmo još malo približiti ovaj koncept. Razmotrimo sve primjere razreda k. Normaliziranu reprezentaciju i-tog primjera razreda k označimo s $\boldsymbol{r}_{\boldsymbol{k}i}$. Normalizirani vektor težina izlaznog sloja koji odgovara razredu k označimo s $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$.

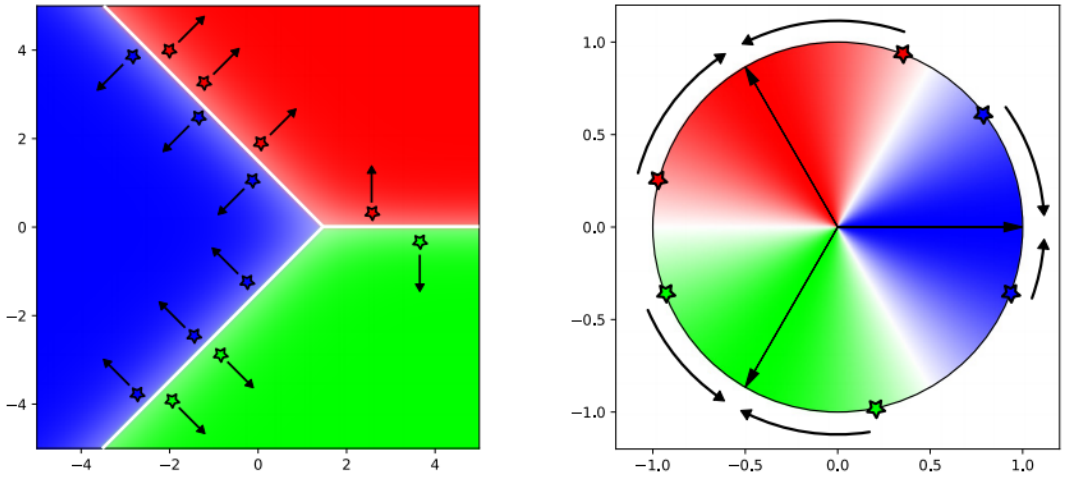
Zbog normalizacije, svi $\boldsymbol{r}_{\boldsymbol{k}i}$, neovisno o parametrima prethodnih slojeva, nalazit će se na kružnici polumjera 1. Vektor $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$ također se zbog normalizacije nalazi na istoj kružnici.

Parametri svih slojeva mreže osim izlaznog sloja određuju gdje će se na toj kružnici nalaziti svi vektori $\boldsymbol{r}_{\boldsymbol{k}i}$, tj. mijenjanjem tih parametara, ti se vektori "šeću" po toj kružnici. Također, mijenjanjem parametara vektora $\boldsymbol{w}_{\boldsymbol{k}}$ izlaznog sloja, vektor se $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$ također "šeće" po toj kružnici.

Gledajući isključivo primjere razreda k, najmanji ukupni gubitak mreža će postići u slučaju kada se svi $\boldsymbol{r}_{\boldsymbol{k}i}$ i vektor $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$ nalaze na istoj točki kružnice. Naime, tada će kosinusne sličnosti svih $\boldsymbol{r}_{\boldsymbol{k}i}$ s vektorom $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$ biti maksimalne. Dakle, mreža će nastojati na jediničnoj kružnici približiti sve $\boldsymbol{r}_{\boldsymbol{k}i}$ vektoru $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$, podešavajući težine svih slojeva prije izlaznog. Podešavajući parametare vektora $\boldsymbol{w}_{\boldsymbol{k}}$ izlaznog sloja, mreža će nastojati na jediničnoj kružnici vektor $\tilde{\boldsymbol{w}}_{\boldsymbol{k}}$ približiti svim vektorima $\boldsymbol{r}_i$. Ovo rezultira grupiranjem svih vektora $\boldsymbol{r}_{\boldsymbol{k}i}$ oko iste točke jedinične kružnice. Samim time, međusobne će kosinusne sličnosti vektora $\boldsymbol{r}_{\boldsymbol{k}i}$ biti visoke, što nam je bio cilj.

Na slici 3.14 prikazan je opisani fenomen, uz posteriorne vjerojatnosti klasa naznačene bojama. Prikazani su i smjerovi u kojima se guraju reprezentacije hipotetskih primjera u slučaju korištenja standardnog softmaxa, odnosno kosinusnog softmaxa.

Ovakvo ponašanje kosinusnog softmax klasifikatora moglo bi biti korisno ako želimo da nam posljednji sloj predstavlja metričko ugrađivanje ulaznih podataka.



Slika 3.14: Prikaz fenomena opisanog u odjeljku 3.4.2. Kod klasičnog *softmaxa*, reprezentacije se primjera udaljavaju od decizijske granice. Kod kosinusnog *softmaxa*, zbog normalizacije, reprezentacije primjera i vektori težina izlaznog sloja za svaki razred nalaze se na jediničnoj kružnici. Što su primjeri nekog razreda bliži vektoru težina izlaznog sloja za taj razred, gubitak će biti manji. Zato će se reprezentacije primjera istog razreda na kružnici grupirati oko vektora težina izlaznog sloja tog razreda.

3.4.3. Trojni gubitak

Uz navedene klasifikatore, isprobali smo i trojni gubitak (engl. *triplet loss*) [19] koji se često koristi za učenje korespondencijskih metrika.

Trojni je gubitak definiran za trojke $\boldsymbol{r}_a$, $\boldsymbol{r}_p$ i $\boldsymbol{r}_n$, gdje prva dva elementa pripadaju istom razredu (pozitivan par), dok treći element pripada nekom drugom razredu (prvi i treći element su negativan par). Trojni je gubitak definiran izrazom:

$$\mathcal{L}_{\mathrm{t}}\left(\boldsymbol{r}_a, \boldsymbol{r}_p, \boldsymbol{r}_n\right) = max(\|\boldsymbol{r}_a - \boldsymbol{r}_n\|_2 - \|\boldsymbol{r}_a - \boldsymbol{r}_p\|_2 + m, 0) \tag{3.22}$$

gdje je m fiksni pozitivan broj koji označava marginu za koju udaljenost između negativnog para mora biti veća od udaljenosti između pozitivnog para da bi gubitak bio nula. Ovo je standardna formulacija s takozvanom funkcijom zglobnice, dok smo mi zamijenili zglobnicu glatkom funkcijom *softplus*, tako da umjesto $max(x + m, 0)$ koristimo $log(1 + exp(x))$.

3.4.4. Magnet loss

Posljednji oblik gubitka koji smo isprobali jest varijanta magnetnog gubitka [30]. Taj gubitak uzima u obzir udaljenosti primjera od srednjeg vektora svojeg razreda i svih ostalih razreda, te računanjem omjera jednog i drugog tjera primjer bliže ostalim primjerima svog razreda i dalje od primjera drugih razreda.

Točan izraz za gubitak koji koristimo je:

$$\mathcal{L}_{\mathrm{m}}(y, \boldsymbol{r}) = max(-\log \frac{\mathrm{e}^{-\frac{1}{2\hat{\sigma}^2}\left\|\boldsymbol{r}-\hat{\boldsymbol{\mu}}_y\right\|_2^2 - m}}{\sum_{k \in \overline{\mathcal{C}}(y)} \mathrm{e}^{-\frac{1}{2\hat{\sigma}^2}\|\boldsymbol{r}-\hat{\boldsymbol{\mu}}_k\|_2^2}}, 0) \tag{3.23}$$

gdje je m ponovno margina, $\hat{\boldsymbol{\mu}}_y$ srednji vektor uzorka razreda y, $\hat{\sigma}^2$ varijanca udaljenosti primjera od srednjih vektora njihovih razreda, a $\overline{\mathcal{C}}(y)$ je skup $\{1, \ldots, C\} \backslash \{y\}$.

3.4.5. Korištena arhitektura

Arhitektura mreže koja se koristi za metričko ugrađivanje opisana je u tablici 3.1. Mreža očekuje ulazne slike dimenzija 128x64. Neki od konvolucijskih i rezidualnih [12] slojeva imaju korak 2 što smanjuje dimenzije prostorne ulaznog podatka. Na kraju se potpuno povezanim slojem dobiva vektor značajki dimenzije 128. Na njemu se primjenjuje prethodno spomenuta L2-normalizacija (3.20).

Ime sloja	**Dimenzija jezgre/Korak**	**Dimenzija izlaza**
Conv 1	3 x 3/1	128 x 64 x 32
Conv 2	3 x 3/1	128 x 64 x 32
Conv 3	3 x 3/2	64 x 32 x 32
Residual 4	3 x 3/1	64 x 32 x 32
Residual 5	3 x 3/1	64 x 32 x 32
Residual 6	3 x 3/2	32 x 16 x 64
Residual 7	3 x 3/1	32 x 16 x 64
Residual 8	3 x 3/2	16 x 8 x 128
Residual 9	3 x 3/1	16 x 8 x 128
Dense 10		128
L2 norm		128

Tablica 3.1: Slojevi korištene mreže.

4. Opis sustava

Sada ćemo opisati cijeli sustav za praćenje objekata koji koristimo u ovom radu [2, 34]

Sustav ima 4 glavna dijela koje ćemo opisati zasebno, a čije smo dijelove vezane uz strojno i duboko učenje opisali i u prethodnom poglavlju

Sustav slijedi paradigmu *tracking by detection*, te je jedan od dijelova sustava upravo postupak detekcije objekata. Sustav je namijenjen primijeni u stvarnom vremenu pa za dodjeljivanje identiteta objektima scene u određenom trenutku koristi samo informacije iz trenutne i prijašnjih slika, ne i budućih

Sustav u svakom trenutku t (tj. za svaku sliku iz ulaznog niza slika) provede postupak detekcije objekata na slici. Na samom početku, treba svim detektiranim objektima dodijeliti identitete, tj. početi pratiti objekte. Zatim, na budućim slikama treba nove detekcije objekata iz tih slika točno dodijeliti već praćenim objektima (identitetima), ili pak ako je riječ o novom objektu, početi taj novi objekt pratiti. Uz to, može se dogoditi da je praćeni objekt izašao iz scene, te on nema odgovarajuću detekciju u novoj slici, te ga tada sustav prestaje pratiti

Za točno uparivanje detekcija istog objekta u dvije uzastopne slike sustav koristi dva mehanizma:

1. modeliranje izgleda dijelova slike koje određuju granični okviri detekcija, čime možemo dobiti neku mjeru sličnosti između svih prethodno detektiranih (i praćenih) objekata i novih detekcija iz slike kojoj želimo dodijeliti identitete
2. modeliranje gibanja kojim na temelju prethodnog gibanja određenog objekta previđamo gdje će se on nalaziti u sljedećoj slici

Informacije o međusobnim sličnostima detekcija i očekivanim budućim lokacijama objekata kombiniraju se u jednu funkciju koja predstavlja "cijenu" uparivanja. Što su detekcije sličnije i što su očekivana i detektirana lokacija objekta na novoj slici bliže, to je cijena manja. Za uparivanje koristi se algoritam koji minimizira ukupnu cijenu uparivanja

Iz ovog opisa naslućuju se 4 glavna modula sustava:

1. modul za detekciju objekata
2. modul za predviđanje gibanja objekata u sceni
3. modul za mjerenje međusobne sličnosti izgleda detektiranih objekata
4. modul za optimalno uparivanje do tad praćenih objekata (sa svojim detekcijama i stanjima gibanja) i novih detekcija

U nastavku ćemo svaki od navedenih modula ukratko opisati.

4.1. Modul za detekciju objekata

Ulaz u modul detekcije objekata je jedna slika u trenutku t, a očekivani izlaz su lokacije (granični okviri) svih osoba na toj slici.

Kao prvu inačicu koristimo mrežu FasterRCNN s arhitekturom VGG-16 predtreniranom na skupu ImageNet i fino podešenom na skupovima podataka *ETHZ pedestrian dataset* i *Caltech pedestrian dataset* koje su omogućili autori rada [36]

Kao drugu inačicu koristimo mrežu MaskRCNN s arhitekturom ResNet101 [12] predtreniranu na skupu podataka MS COCO.

Obje navedene mreže mogu detektirati objekte više različitih klasa, a nas zanimaju samo detekcije osoba. Stoga ignoriramo sve ostale izlaze tih mreža osim onog koji odgovara pouzdanosti klase osobe. Za dobivanje konačnih detekcija uzimamo samo one detekcije kod kojih je pouzdanost klase osobe veća od određenog praga, dok ostale detekcije odbacujemo. Optimalni prag pouzdanosti je kod dviju korištenih mreža drukčiji i dobiven je pretragom na intervalu između 0 i 1.

4.2. Modul za procjenu gibanja

Zadatak modula za procjenu gibanja je predvidjeti gdje bi se objekt trebao nalaziti u sljedećem trenutku na temelju informacija o lokaciji objekta u prijašnjim trenucima. Za to se koristi postupak Kalmanov filter s pretpostavkom stalne brzine gibanja [18]

Svaki od praćenih objekata ima svoju točku (svoje stanje) u 8-dimenzionalnom prostoru stanja. Parametri stanja su koordinate središta graničnog okvira objekta *u* i *v*, omjer širine i visine graničnog okvira *gamma*, visina graničnog okvira *h*, te "brzine" (stope promjene) svih četiriju navedenih veličina. Prve četiri veličine opažaju se iz pristiglih detekcija objekata, a preostale četiri veličine inicijalno imaju vrijednost 0 i ažuriraju se nakon svakog dodjeljivanja.

U svakom trenutku, prije dodjeljivanja, napravi se predviđanje sljedeće lokacije svakog objekta na temelju prethodnih lokacija i procijenjenih brzina. Predviđene lokacije koriste se za izračunavanje "cijene" između previđenih lokacija praćenih objekata u sljedećoj slici i lokacija dobivenih detekcijom. Koristi se kvadrirana Mahalanobisova udaljenost između previđenih Kalmanovih stanja i pristiglih mjerenja (detekcija) [34]:

$$d^{(1)}(i,j) = (\mathbf{m}_j - \mathbf{y}_i)^T \mathbf{S}_i^{-1} (\mathbf{m}_j - \mathbf{y}_i) \tag{4.1}$$

Korištenjem Mahalanobisove udaljenosti uzima se u obzir nesigurnost procjene stanja mjerenjem koliko standardnih devijacija je detekcija udaljena od srednje lokacije praćenog objekta. Uz to, uparivanja male vjerojatnosti isključuju se postavljanjem praga $t = 9.4788$ na Mahalanobisovu udaljenost. Taj je broj autori su odabrali jer je to kritična vrijednost točke 95-postotnog intervala pouzdanosti distribucije χ^2 s 4 stupnja slobode (naš prostor mjerenja ima 4 dimenzije). [34] Odluku o zadržavanju ili odbacivanju uparivanja zapisujemo ovako:

$$b_{i,j} = \mathbb{1}\left[d^{(1)}(i,j) \leq t\right] \tag{4.2}$$

Nakon što je dodjeljivanje napravljeno, Kalmanovo stanje svakog od praćenih objekata ažurira se novom lokacijom.

4.3. Modul za izračun sličnosti dijelova slike pomoću metričkog ugrađivanja

Zadatak ovog modula je odrediti sličnost dijelova slike određenih graničnim okvirima. Za to se koristi neuronska mreža prethodno naučena preslikavati ulaznu sliku u vektorski prostor nižih, fiksnih dimenzija, u kojem se tako dobiveni opisnici sličnih slika nalaze blizu jedan drugome, a opisnici različitih su slika daleko jedan od drugog (odjeljak 3.4.5).

Kada se za trenutnu sliku dobiju detekcije osoba, dijelove slika određene graničnim okvirima tih detekcija treba usporediti s dijelovima prijašnjih slika niza koje odgovaraju praćenim osobama, tj. s prijašnjim detekcijama. Kao vizualna reprezentacija prethodno praćene osobe ne koristi se samo njena detekcija iz prethodne slike niza, nego se pamte detekcije te praćene osobe na zadnjih 100 slika. Taj je broj odabran u izvornom radu. [34]

Svaka od pristiglih detekcija za trenutnu sliku uspoređuje se sa svakom od prijašnjih 100 slika svake od praćenih osoba. Kao što smo prije spomenuli, slike se ne uspoređuju izravno, nego ih neuronska mreža naučena provoditi metričko ugrađivanje najprije preslika u vektore opisnike, te se onda računaju međusobne kosinusne, odnosno euklidske (ovisno o tome kako je metrika bila učena) udaljenosti između opisnika pristiglih detekcija i opisnika praćenih osoba. Zatim se kao udaljenost između neke praćene osobe i neke nove detekcije uzima najmanja od svakog od 100 opisnika praćene osobe i nove detekcije:

$$d^{(2)}(i,j) = min\{dist(\mathbf{r}_j^T, \mathbf{r}_k^{(i)}) | \mathbf{r}_k^{(i)} \in R_i\} \tag{4.3}$$

Za kosinusnu udaljenost koristimo:

$$dist(\mathbf{a}, \mathbf{b}) = 1 - \mathbf{a}^T\mathbf{b} \tag{4.4}$$

Za euklidsku udaljenost koristimo:

$$dist(\mathbf{a}, \mathbf{b}) = \sqrt{(\mathbf{a}-\mathbf{b})^T(\mathbf{a}-\mathbf{b})} \tag{4.5}$$

Neuronska mreža korištena za metričko ugrađivanje (3.1) naučena je na skupu podataka za ponovnu identifikaciju MARS. [23] Dvije inačice trenirane su za zadatak

klasifikacije. Jedna kao izlazni sloj ima uobičajeni softmax, dok druga ima modificirani, tzv. kosinusni softmax gubitak [33], opisan u prethodnom poglavlju. Treća je inačica učena magnetnim, a posljednja trojnim gubitkom.

4.4. Modul za optimalno uparivanje praćenih objekata i novih detekcija

Zadatak je ovog modula iskoristiti dvije različite, prethodno opisane, mjere udaljenosti - udaljenost između vizualnih reprezentacija i udaljenost između lokacija - između prethodno praćenih objekata i novih detekcija, te postupkom koji se temelji na mađarskom algoritmu optimalno dodijeliti nove detekcije postojećim identitetima, tako da je ukupna udaljenost ili cijena što manja.

Izraz za ukupnu cijenu između i-tog praćenog objekta i j-te detekcije je sljedeći:

$$c_{i,j} = \lambda d^{(1)}(i,j) + (1-\lambda) d^{(2)}(i,j) \tag{4.6}$$

gdje se parametrom λ kontrolira doprinos svake od dvije komponente cijene. Ovim izrazom dobivamo cijene između svih parova praćenih objekata i novih detekcija, te možemo izgraditi matricu cijene C čiji je element $C_{i,j}$ jednak $c_{i,j}$. U izvornom radu [34] utvrđeno je da u slučajevima kada postoji značajno gibanje kamere, dobra vrijednost za parametar λ je 0.

Pronalazak optimalnog uparivanja se ne provodi globalno u jednom koraku nego postoji kaskada [34] koja rješava niz potproblema. Tom se kaskadom prednost daje češće viđenim objektima. Kaskada uparivanja u jednom koraku poziva funkciju optimalnog uparivanja mađarskim algoritmom. U pseudokodu koji opisuje postupak koristimo matricu praga B. Vrijednosti elemenata te matrice su 0 ili 1, ovisno o tome odbacujemo li ili prihvaćamo uparivanje. Postupak dobivanja vrijednosti te matrice opisan je u odjeljku 4.2. Pseudokod kaskade uparivanja dan je u nastavku:

Algorithm 1 Kaskada uparivanja

Ulaz: Praćeni objekti $T = \{1, ..., N\}$, detekcije $D = \{1, ..., K\}$, maksimalna starost A_{max}

Izračunaj matricu cijene $C = [c_{i,j}]$
Izračunaj matricu praga $B = [b_{i,j}]$
Inicijaliziraj skup parova $M \leftarrow \emptyset$
Inicijaliziraj skup nesparenih detekcija $U \leftarrow D$
Za $n \in \{1, ..., A_{max}\}$ **radi:**
    Izaberi praćene objekte po starosti $T_n \leftarrow \{i \in T | a_i = n\}$
    $[x_{i,j}] \leftarrow$ optimalno_uparivanje(C, T_n, U)
    $M \leftarrow M \cup \{(i, j) | b_{i,j} \cdot x_{i,j} > 0\}$
    $U \leftarrow U \setminus \{j | \sum_i b_{i,j} \cdot x_{i,j} > 0\}$
Vrati M, U

5. Implementacija sustava

Postupak učenja metričkog ugrađivanja na zadatku ponovne identifikacije napisan je u programskom jeziku Python 3.6, uz korištenje knjižnice otvorenog koda TensorFlow [1] za implementaciju dubokih modela. Projekt sadrži više skripti, one za učenje mreže na skupovima podataka Market1501 i MARS, one za učitavanje podataka, one u kojima su sadržane implementacije različitih korištenih funkcija gubitaka i mjera udaljenosti i one koje sadrže definiciju neuronske mreže. Evaluacija naučenih mreža provodi se službenim skriptama u sklopu skupova podataka koje su napisane u programskom jeziku MATLAB. [22, 23]

Projekt koji sadrži implementaciju mreže MaskRCNN čije su detekcije korištene u eksperimentima preuzet je sa sljedeće poveznice.

Postupak praćenja objekata također je napisan u programskom jeziku Python 3.6, uz korištenje raznih knjižnica, između ostalog TensorFlow za učitavanje naučene mreže, numpy, scipy i scikit-learn. Glavni dijelovi implementacije odgovaraju glavnim modulima konceptualnog rješenja - detekcija objekata, procjena gibanja, mjera sličnosti dijelova slika, optimalno dodjeljivanje detekcija praćenim identitetima. Evaluacija postupka, kao i kod ponovne identifikacije, provodi se provodi službenim skriptama u sklopu skupova podataka napisanim u programskom jeziku MATLAB.

6. Skupovi podataka

U ovom poglavlju, za prethodno spomenute zadatke - učenja metričkog ugrađivanja na zadatku ponovne identifikacije i praćenja više objekata u sceni - opisat ćemo korištene skupove podataka.

6.1. Zadatak ponovne identifikacije osoba

6.1.1. Formulacija problema

Zadatak ponovne identifikacije osoba formuliramo kao zadatak pretrage i povrata slika (image retrieval). Imamo bazu slika (gallery) različitih osoba, gdje svakoj osobi pripada više od jedne slike. Za zadanu upitnu (query) sliku iz baze (ili nekog njenog podskupa) želimo vratiti ostale slike koje pripadaju istoj osobi, dok istovremeno ne želimo da se u vraćenim slikama nalaze one koje ne pripadaju toj osobi. [22] Budući da rezultat ove pretrage nije jedna slika nego skup slika, želimo da naš postupak slike tog skupa rangira po pouzdanosti da one pripadaju traženoj osobi, te se onda za evaluaciju sustava mogu koristiti mjere uspješnosti koje uzimaju u obzir da je rezultat uređen po rangu.

Uz pojedinačne slike, skupove podataka za ponovnu identifikaciju mogu činiti i videozapisi (nizovi uzastopnih slika). U tom slučaju, osim klasičnog, prethodno opisanog zadataka pretrage iz slike u sliku (image-to-image), mogu se formulirati i zadaci pretrage iz slike u videozapis (image-to-video), iz videozapisa u sliku (video-to-image) i iz videozapisa u videozapis (video-to-video). [23]

6.1.2. Mjere uspješnosti za ponovnu identifikaciju

Budući da smo problem formulirali kao problem povrata informacija, možemo koristiti klasične mjere uspješnosti koje se obično koriste.

Rezultat jednog upita je rangirani skup slika. Slike u rezultatu mogu biti bitne (one slike koje odgovaraju istoj osobi kao upitna slika) ili nebitne (one slike koje odgovaraju nekoj drugoj osobi). Uzimajući to u obzir, za rezultat svake pretrage može se za svaki pojedinačni rang *(*i) izračunati preciznost (*Precision @ i*) i odziv (*Recall @ i*).

Definicije tih dviju veličina su sljedeće:

$$\text{preciznost(i)} = \frac{\text{broj bitnih slika unutar prvih i slika}}{\text{i}} \tag{6.1}$$

$$\text{odziv(i)} = \frac{\text{broj bitnih slika unutar prvih i slika}}{\text{ukupni broj bitnih slika}} \tag{6.2}$$

Kako rang i raste, preciznost ima tendenciju pada (ali ne monotono), dok odziv monotono raste (za rang jednak ukupnom broju slika, odziv uvijek ima vrijednost 1).

Budući da kao mjeru uspješnosti jedne pretrage želimo samo jednu vrijednost, želimo nekako agregirati preciznost i odziv za različite rangove. Jedan način na koji se to često radi je prosječna preciznost (average precision). U literaturi postoji više pristupa računanju te vrijednosti. Jedan od njih je sljedeći:

$$AveP = \frac{\sum_{k=1}^{n} P(k)rel(k)}{\text{ukupan broj bitnih slika}} \tag{6.3}$$

Ovdje je $P(k)$ preciznost na rangu k , a $rel(k)$ je funkcija čija je vrijednost 1 ako je k-ta slika rezultata bitna, a u protivnom 0. Dakle, u obzir se uzimaju samo rangovi za koje je pretraga vratila bitnu sliku, te se računa zbroj preciznosti na svakom od tih rangova. Taj se zbroj normalizira ukupnim brojem bitnih slika za tu pretragu.

Kao krajnja mjera uspješnosti postupka pretrage, za sve upitne slike u ispitnom skupu podataka, prosječna preciznost može se još jednom uprosječiti po svim slikama, te tada dobivamo srednju prosječnu preciznost (*mean average precision*).

6.1.3. Skup podataka Market1501

Skup podataka Market1501 sadrži 32,668 slika osoba u galeriji i 3,368 upitnih slika, što je u vrijeme nastanka bilo mnogo više od ostalih sličnih skupova podataka. Iako postoje točni, rukom označeni okviri osoba na slikama, oni ne čine ispitni skup. Naime, slike osoba u ispitnom skupu dobivene su postojećim sustavom za detekciju objekata (*deformable parts model object detector* [8]). To čini taj skup podataka realističnijim, budući da je najčešća primjena sustava za ponovnu identifikaciju upravo na dobivenim

rezultatima nekog sustava za detekciju objekata. Takav će skup podataka bolje obuhvatiti nesavršenosti detektora objekata, te će neke slike imati samo dio osobe, ili imati višak prostora oko osobe i slično.

Za razliku od nekih drugih skupova podataka koji za svaku upitnu sliku imati samo jednu odgovarajuću sliku u galeriji, skup podataka Market1501 za svaku upitnu sliku ima više odgovarajućih slika koje su dobivene različitim kamerama.

Uz spomenuti broj slika osoba, ovaj skup podataka ima i oko 500,000 slika koje bi mogle zbuniti neki sustav za ponovnu identifikaciju osobe. Naime, način na koji su prikupljane slike za ovaj skup je sljedeći. Ako je preklapanje u površini ručno označenih okvira i okvira koji su dobiveni detektorom objekata veće od 50%, takvi se detektirani okviri uzimaju kao slike osoba. Pritom preklapanje okvira mjerimo kao omjer površina njihovog presjeka i unije. Ako je spomenuto preklapanje manje od 20%, takav se detektirani granični okvir uzima kao primjer koji bi mogao zbuniti sustav ponovne identifikacije.

Ukupan broj različitih osoba (identiteta) u svim slikama jest 1501. Skup podataka je podijeljen tako da u skupu za učenje ima 750 identiteta, dok u ispitnom skupu ima 751 identiteta.

Skup podataka može se preuzeti na sljedećoj poveznici.

6.1.4. Skup podataka MARS

Skup podataka punog imena *Motion Analysis and Re-Identification Set* (skraćeno MARS) proširenje je skupa Market1501.

Sadrži 1,261 identiteta s preko 1,100,000 slika. Skup podataka napravljen je iz istih slika kao i Market1501, ali je za označavanje korišten postupak praćenja objekata u sceni *GMMCP Tracker*, uz DPM detektor objekata kao i kod skupa Market1501. Tim postupkom dobiveno je oko 20,000 instanci praćenja, koje su zatim ručno svrstane u identitete osoba.

Činjenica da je sakupljen pomoću postupka praćenja osoba čini ovaj skup podataka podatnim za učenje metrike ugrađivanja koja se također koristi u problemu praćenja osoba.

Skup podataka može se preuzeti na sljedećoj poveznici.

6.2. Praćenje više objekata / osoba

Vrednovanje i uspoređivanje metoda praćenja više objekata nije jednostavno. Za razliku od nekih drugih zadataka, teško je jasno definirati rješenje koje se želi postići. U sceni možemo imati samo djelomično vidljive objekte, odraze objekata na reflektirajućim površinama ili objekte koji su jako slični onima koje želimo pratiti, stoga će se i ljudski označivači nekad teško složiti oko točnog rješenja.

Uz to, postoje razne evaluacijske mjere kojima se praćenje može vrednovati i teško je jednom mjerom obuhvatiti sve aspekte problema, što dodatno otežava usporedbu različitih pristupa.

6.2.1. Mjere uspješnosti

Kada evaluiramo praćenje objekata s jedne strane imamo ciljane objekte koje želimo da naš postupak uspješno prati (*target*), a s druge imamo izlaz našeg postupka praćenja (*tracker*), te tu možemo definirati nekoliko veličina. Broj *true positive* (TP) uparivanja je broj praćenih objekata koji odgovaraju nekom označenom objektu. Broj *false positive* (FP) je broj praćenih objekata koji ne odgovaraju nijednom označenom objektu. Na kraju, o broju označenih objekata kojima ne odgovara nijedan praćeni identitet govori broj *false negative* (FN).

Još jedan zanimljiv iznos koji možemo pratiti je broj zamjena identiteta (*identity switch*, IDSW). Zamjena identiteta događa se u trenutku kada je označenom objektu i pridan identitet j, a zadnji identitet dodijeljen tom označenom objektu bio je $k \neq i$.

Pokušamo li objediniti navedene iznose u jednu mjeru, dobivamo *Multiple Object Tracking Accuracy* (MOTA). Riječ je o vrlo često korištenoj evaluacijskoj mjeri koju primarno koristimo i u ovom radu. MOTA je definirana sljedećim izrazom:

$$\mathrm{MOTA} = 1 - \frac{\sum_t (\mathrm{FN}_t + \mathrm{FP}_t + \mathrm{IDSW}_t)}{\sum_t \mathrm{GT}_t} \tag{6.4}$$

gdje je t redni broj slike iz zadanog niza slika, a GT je broj označenih objekata (*target*) u nekoj slici.

6.2.2. Skup podataka MOT 2015

Skup podataka za praćenje više osoba u nizovima slika MOT2015 [21] sadrži 11 sekvenci u skupu za treniranje i 11 sekvenci u ispitnom skupu. Radi se uglavnom o snimkama pješaka na ulici. Za svaku sliku neke sekvence označeni su granični okviri oko svake osobe i točni identiteti. Ukupno ima preko 11,000 slika i preko 1,200 praćenih osoba. Neke od snimki snimane su statičnom kamerom dok je u drugima i kamera u pokretu. Skup podataka može se preuzeti na sljedećoj poveznici.

7. Eksperimenti

Da bismo vidjeli koliki je utjecaj različitih metrika ugrađivanja na performanse postupka praćenja objekata naučili smo 8 varijanti metrika. Za svaki od 4 gubitka spomenuta u sekciji 4.4. naučili smo "ranu" mrežu (oko 10000 iteracija) i "kasnu" mrežu (oko 100000 iteracija), tako da dobijemo dojam u utjecaju drukčijih gubitaka i duljine treniranja.

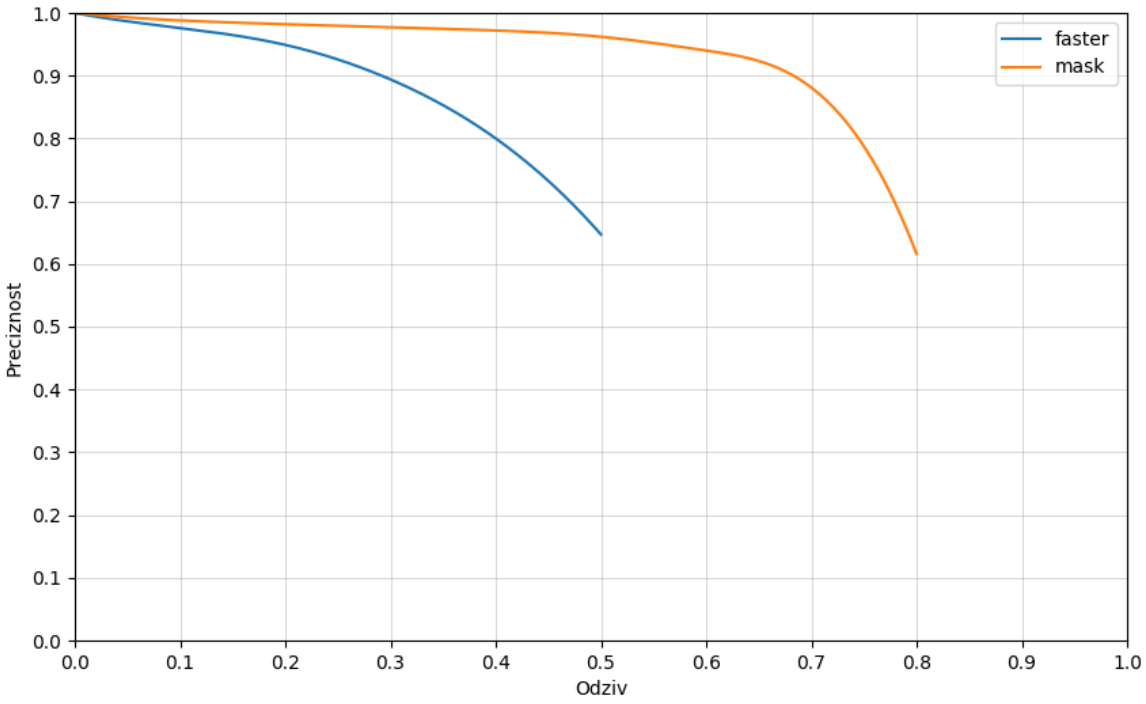
Uz to, zanima nas koliko je utjecaj variranja metrike velik u odnosu na variranje drugog ključnog dijela postupka praćenja - postupka detekcije objekata. Zato smo varirali dva postupka detekcije objekata prethodno opisanih u sekciji 4.3. Za svaki od dva postupka detekcije prethodno smo pronašli optimalan prag pouzdanosti razreda osobe za prihvaćanje određene detekcije kao detekcije osobe. Prilikom traženja optimalnog praga metriku ugrađivanja fiksirali smo na "kasnu" varijantu mreže trenirane gubitkom kosinusnog softmaxa koja je na zadatku ponovne identifikacije imala najbolji rezultat od svih 8 varijanti.

7.1. Detekcija objekata

Arhitekture koje koristimo za detekciju objekata opisane su u odjeljcima 3.3 i 4.1 Evaluirali smo detekciju objekata na skupu podataka MOT 2015 [21]. Evaluiramo preciznost za različite vrijednosti odziva, iscrtavamo krivulju preciznosti u ovisnosti o odzivu [4] i navodimo prosječnu preciznost. Rezultati su prikazani u tablici 7.1 i na slici 7.1. Mreža koja koristi Faster RCNN za detekciju s arhitekturom VGG-16 za ekstrakciju značajki u tablici ima oznaku *faster*. Mreža koja koristi Mask RCNN s arhitekturom ResNet-101 za ekstrakciju značajki označena je s *mask*.

R		0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
P	faster	1.0	0.98	0.95	0.89	0.8	0.65	0.0	0.0	0.0	0.0	0.0
	mask	1.0	0.99	0.98	0.98	0.97	0.96	0.94	0.88	0.62	0.0	0.0
AP	faster	0.479										
	mask	0.756										

Tablica 7.1: Rezultati detekcije



Slika 7.1: Krivulja preciznosti u ovisnosti o odzivu

7.2. Metrička ugrađivanja

Kao što smo rekli, imamo 4 različita gubitka koja isprobavamo: unakrsna entropija na standardnom softmaxu, unakrsna entropija na kosinusnom softmaxu, trojni gubitak i magnetni gubitak (odjeljak 3.4). Dalje u tekstu, tablicama i grafovima na ove ćemo se inačice referirati nazivima *softmax*, *cosine softmax*, *triplet* i *magnet*.

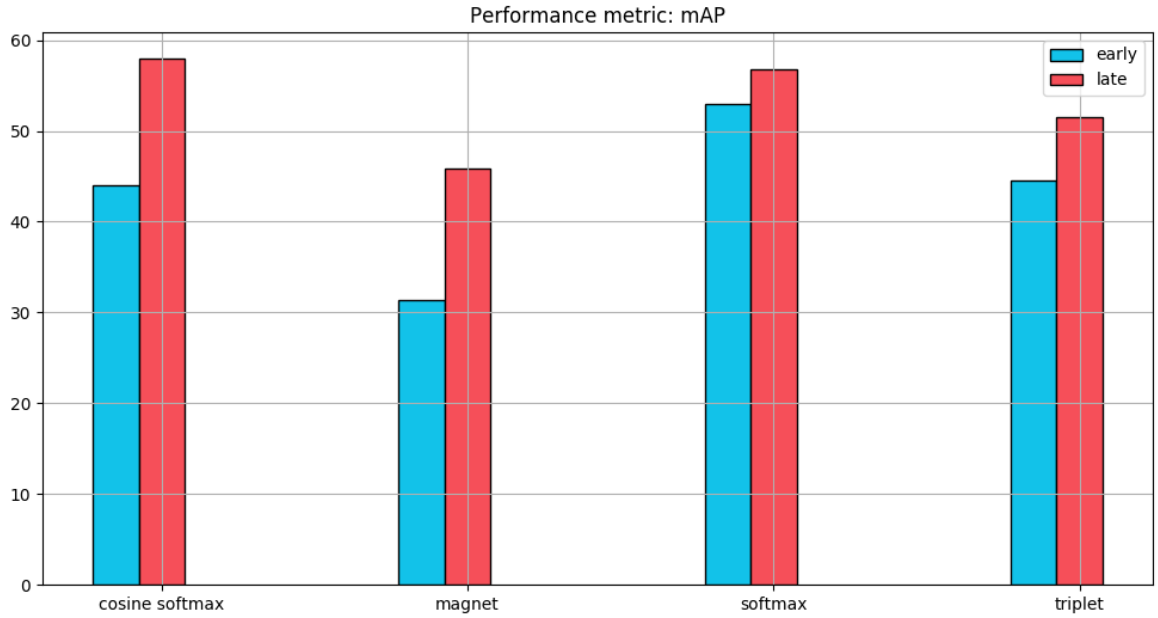
Za svaki od 4 gubitka testiramo mrežu treniranu sa samo 10000 iteracija i mrežu treniranu s 100000 iteracija. Na ove se dvije varijante referiramo nazivima *early* (rana) i *late* (kasna).

Mreže su trenirane na skupu podataka MARS. Koristi se postupak optimizacije Adam sa stopom učenja 1×10^{-3}. Margina magnetnog gubitka m postavljena je na 1. Veličina mini grupe je 128. Kao mjere uspješnosti razmatramo one navedene u sekciji 3.1.: srednju prosječnu preciznost (mAP), te odziv na rangu 1, 5, 10 i 20 (r1, r5, r10, r20).

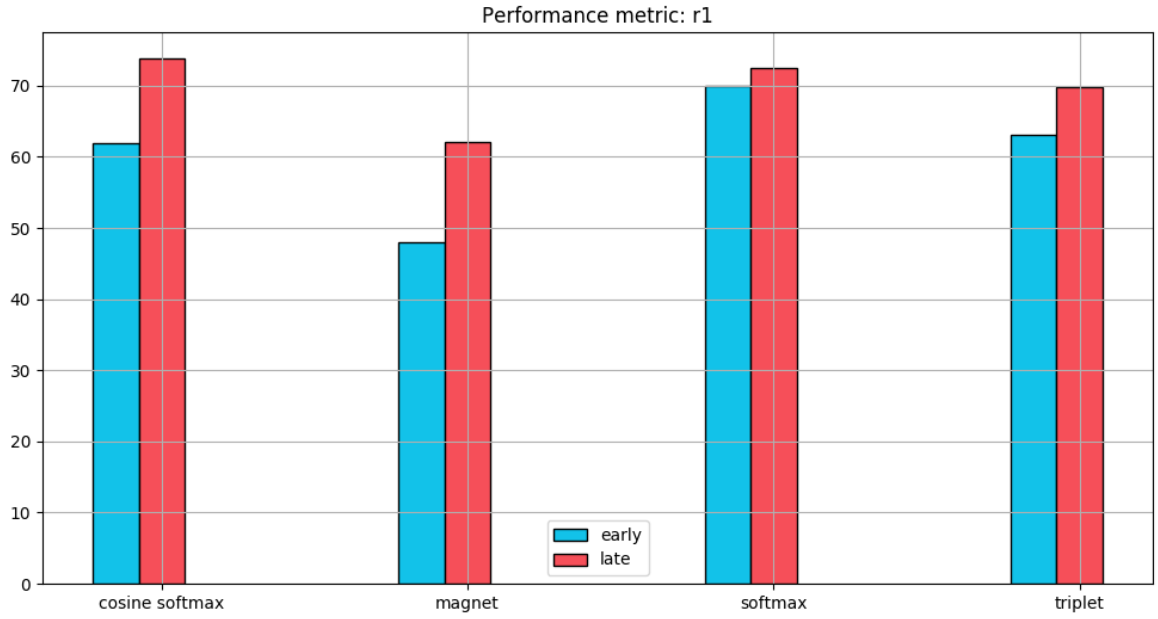
Skripte za evaluaciju na skupu podataka MARS napisane u programskom jeziku MATLAB dostupne su na sljedećoj poveznici. Rezultati su prikazani u tablici i na grafovima.

		Mjera uspješnosti				
Inačica mreže		mAP	r1	r5	r10	r20
softmax	early	52.91	69.95	84.65	88.48	91.41
	late	56.74	72.47	85.35	89.75	92.42
cosine softmax	early	44.01	61.91	78.03	84.24	88.23
	late	**57.97**	**73.83**	**87.17**	**90.40**	**93.38**
triplet	early	44.51	63.08	78.59	83.84	88.03
	late	51.55	69.75	83.64	87.58	90.71
magnet	early	31.41	48.03	67.37	74.19	80.30
	late	45.84	62.02	79.95	85.35	90.00

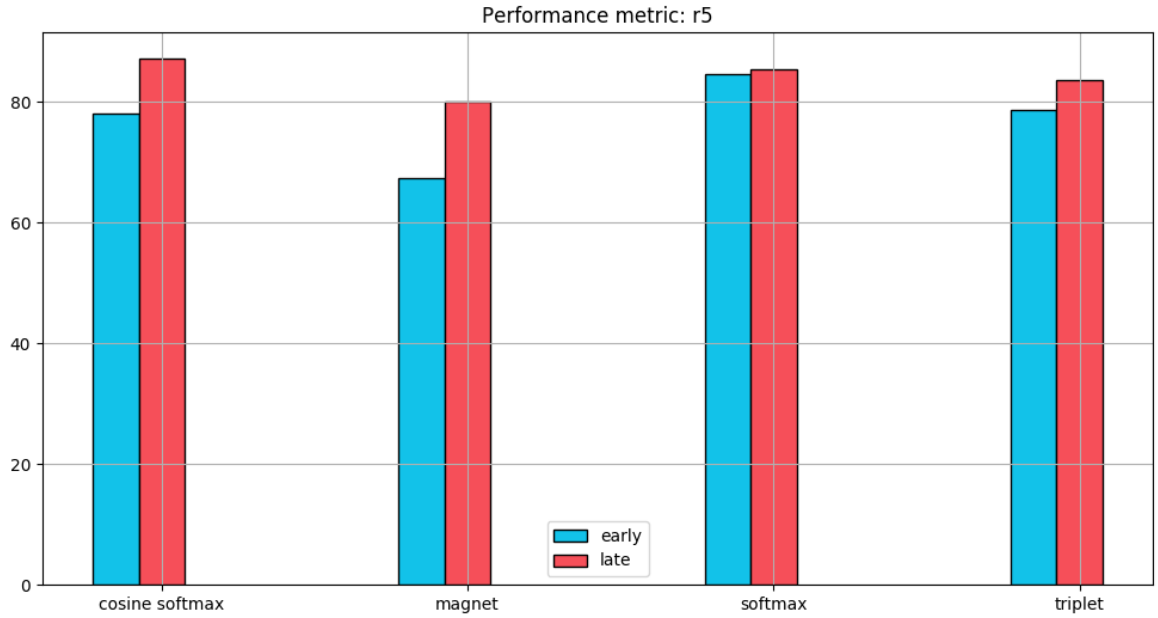
Tablica 7.2: Rezultati ponovne identifikacije



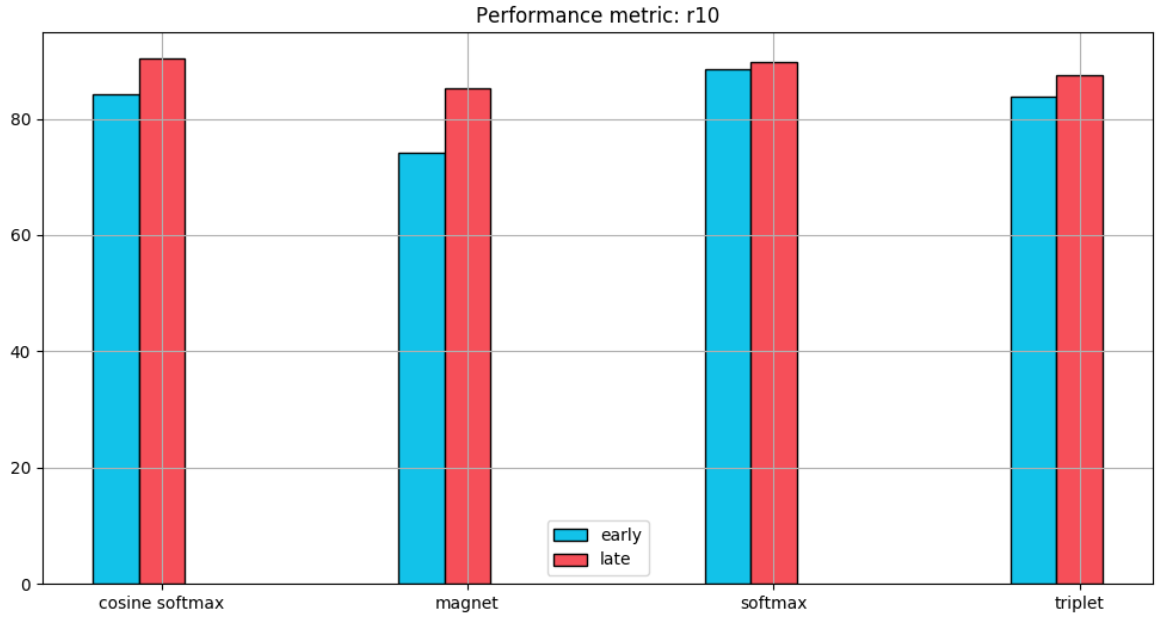
Slika 7.2: Srednja prosječna preciznost



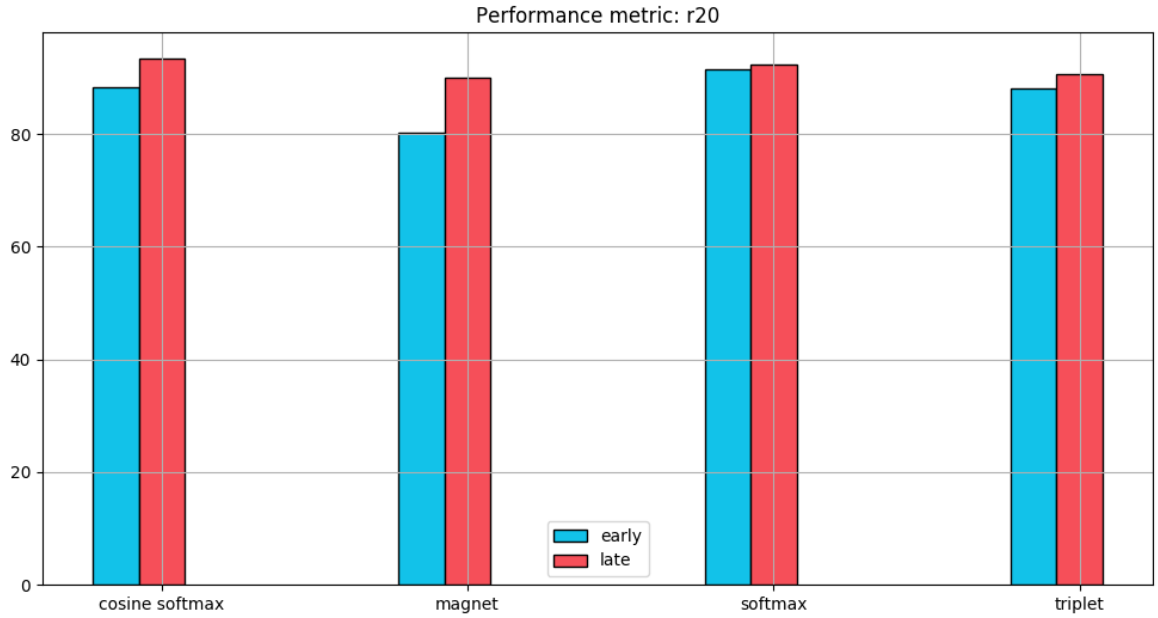
Slika 7.3: Odziv na rangu 1



Slika 7.4: Odziv na rangu 5



Slika 7.5: Odziv na rangu 10

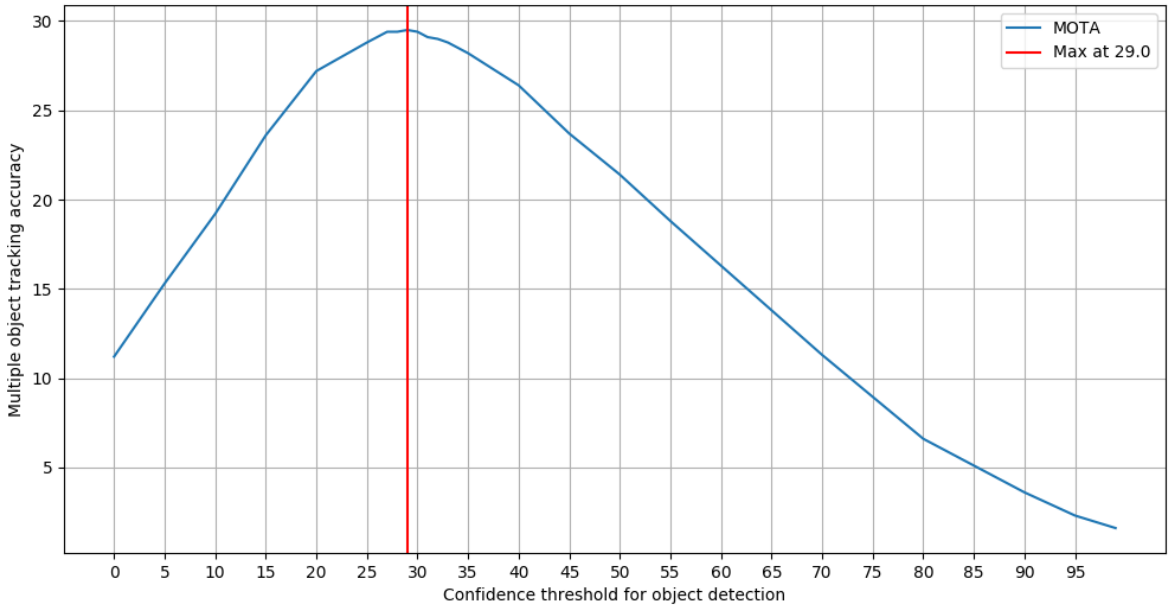


Slika 7.6: Odziv na rangu 20

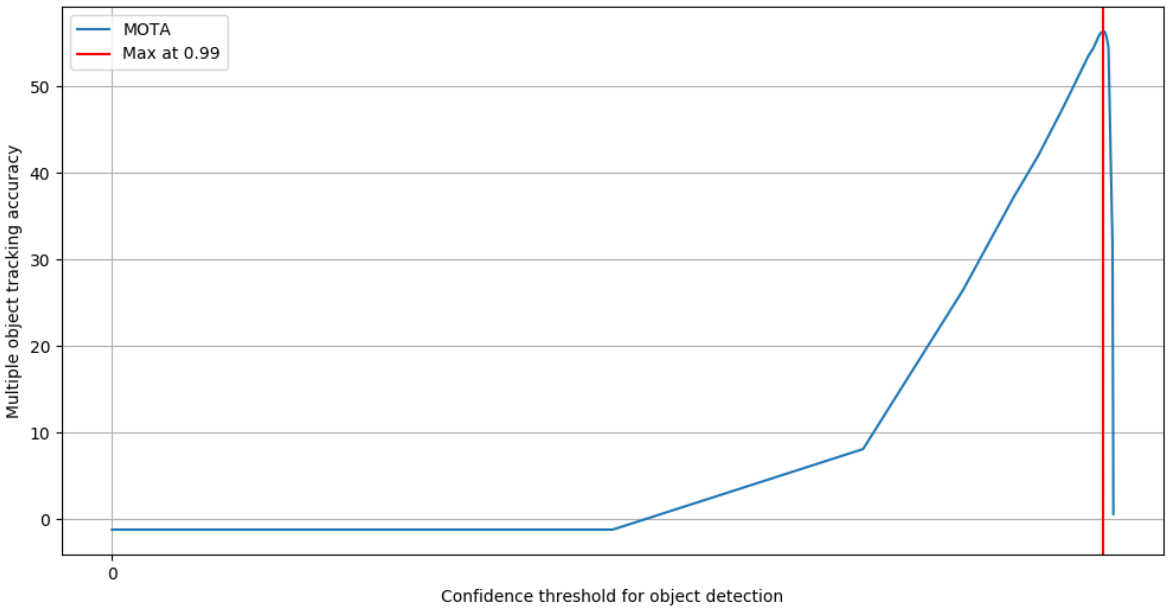
7.3. Prag pouzdanosti detekcije objekata

Prag pouzdanosti za klasu osobe koju na izlazu daje postupak detekcije objekata iznad kojeg se dana detekcija prihvaća kao detekcija osobe važan je parametar i može se znatno razlikovati od detektora do detektora. Zato smo za oba postupka detekcije fiksirali inačicu metrike ugrađivanja (varijanta *cosine softmax*, *late*) i za različite pragove pouzdanosti na intervalu (0, 100) evaluirali postupak praćenja objekata da dobijemo optimalni prag za svaki postupak detekcije.

Kao mjeru uspješnosti razmatrali smo *multiple object tracking accuracy* (MOTA), mjeru opisanu u sekciji 3.3. Rezultati su prikazani grafovima za različite vrijednosti praga. Optimalan prag za prvi postupak detekcije (mreža FasterRCNN, predtrenirana na ImageNetu i fino podešena) je 29%, dok je optimalan prag za drugi postupak (mreža MaskRCNN, predtrenirana na MS COCO) 99%.



Slika 7.7: Performanse za različite pragove pouzdanosti za drugi detektor (FasterRCNN, ImageNet, fino podešavanje)

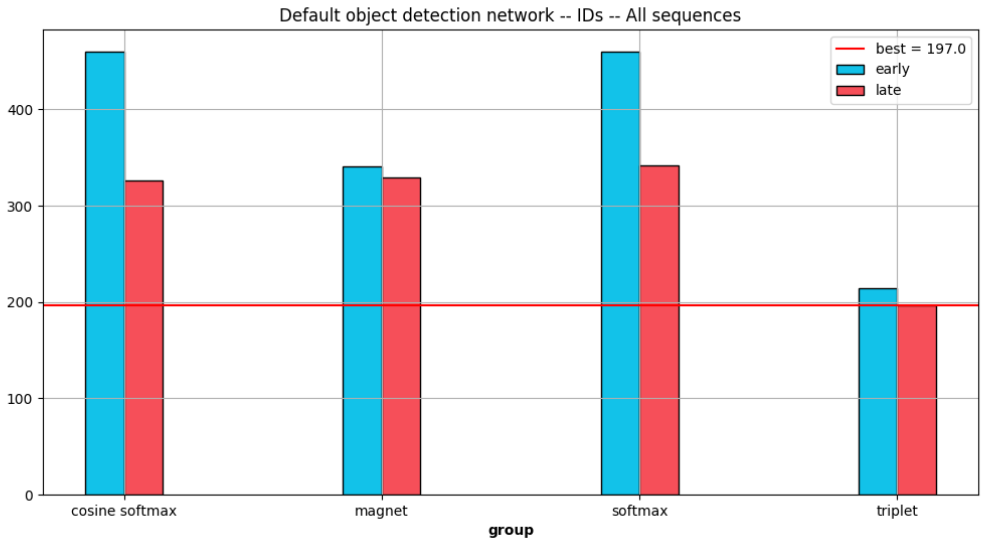


Slika 7.8: Performanse za različite pragove pouzdanosti za drugi detektor (MaskRCNN, MS COCO)

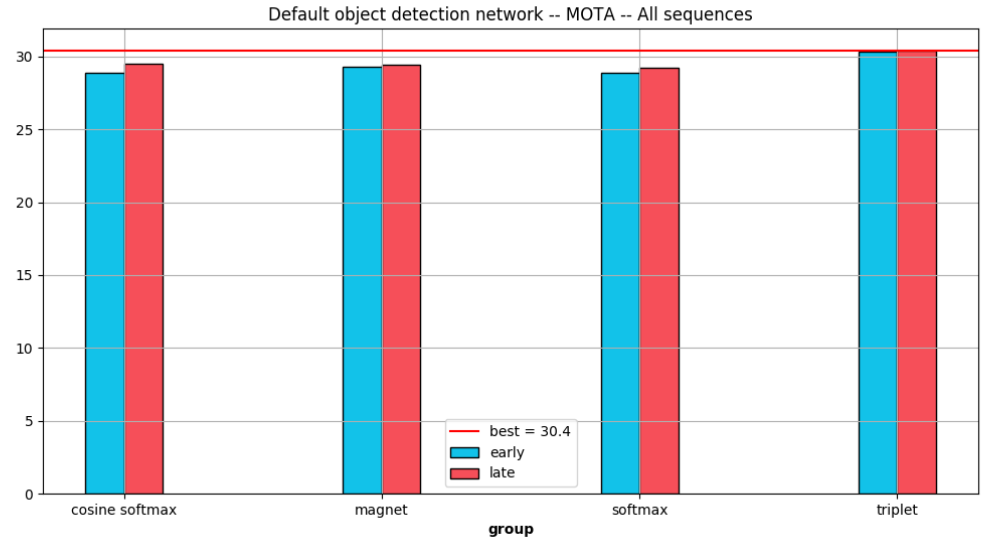
7.4. Praćenje više objekata u sceni

Kod postupka praćenja objekata variramo dva prethodno opisana postupka detekcije objekata (u grafovima i tablicama, *faster* i *mask*) i 8 različitih naučenih metričkih ugrađivanja, za svaku od 4 opisane funkcije gubitka (*softmax*, *cosine softmax*, *triplet*, *magnet*) po dvije mreže, jedna učena kroz 10000 iteracija (*early*) i druga kroz 100000 iteracije (*late*). Koristimo optimalne pragove pouzdanosti detekcije pronađene u odjeljku 7.3. Razmatramo mjere uspješnosti *multiple object tracking accuracy* (MOTA) i broj zamjena identiteta (*identity switch*, IDSW). Skripte za evaluaciju na skupu podataka MOT15 napisane u programskom jeziku MATLAB dostupne su na https://bitbucket.org/amilan/motchallenge-devkit/src. Rezultati su prikazani u tablici i grafovima.

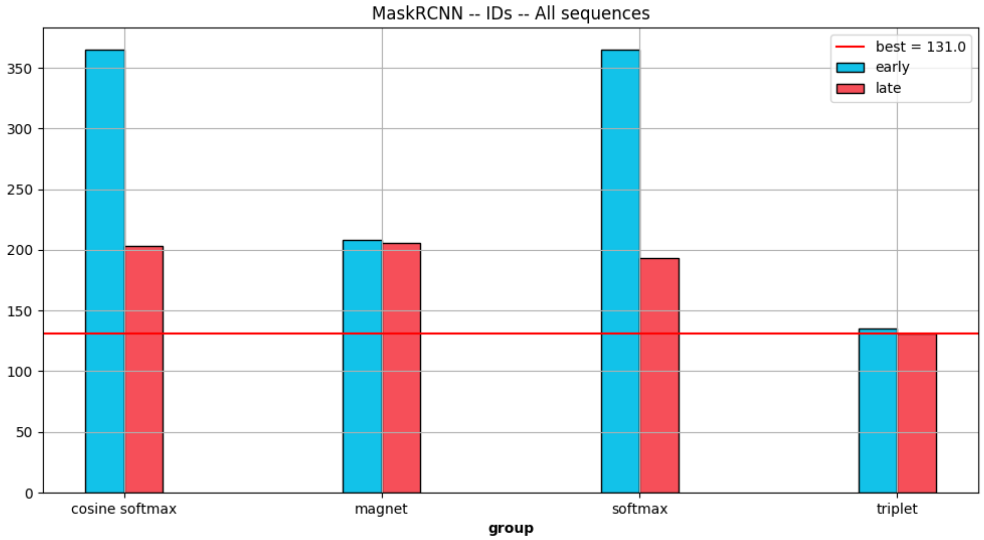
			Mjera uspješnosti	
Inačica postupka			MOTA	IDSW
faster	softmax	early	28.9	460
		late	29.2	342
	cosine softmax	early	28.9	460
		late	29.5	326
	triplet	early	30.3	214
		late	30.4	197
	magnet	early	29.3	341
		late	29.4	329
mask	softmax	early	55.5	365
		late	56.3	193
	cosine softmax	early	55.5	365
		late	56.3	203
	triplet	early	**56.4**	135
		late	**56.4**	**131**
	magnet	early	56.3	208
		late	56.2	206



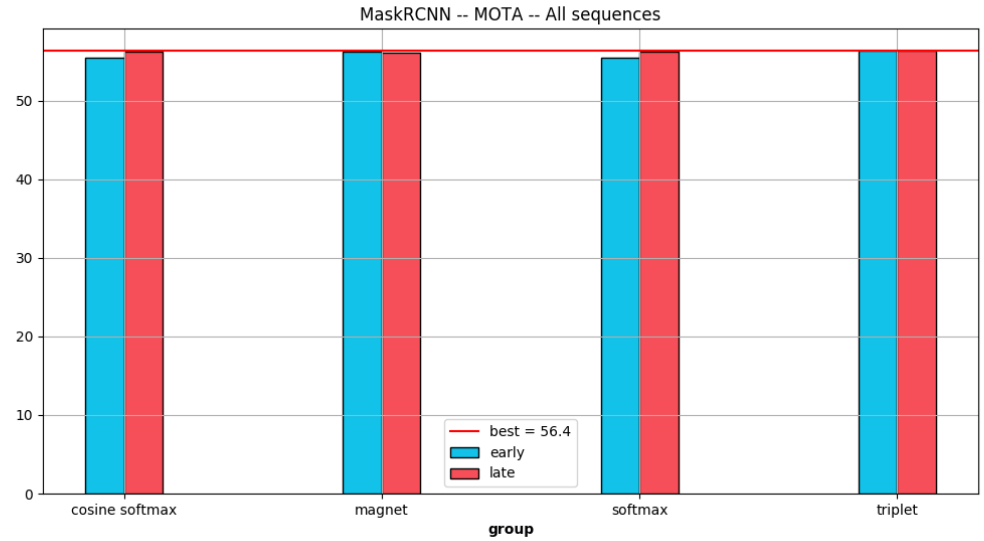
Slika 7.9: Broj zamjena identiteta za postupak detekcije *faster*



Slika 7.10: MOTA za postupak detekcije *faster*



Slika 7.11: Broj zamjena identiteta za postupak detekcije *mask*



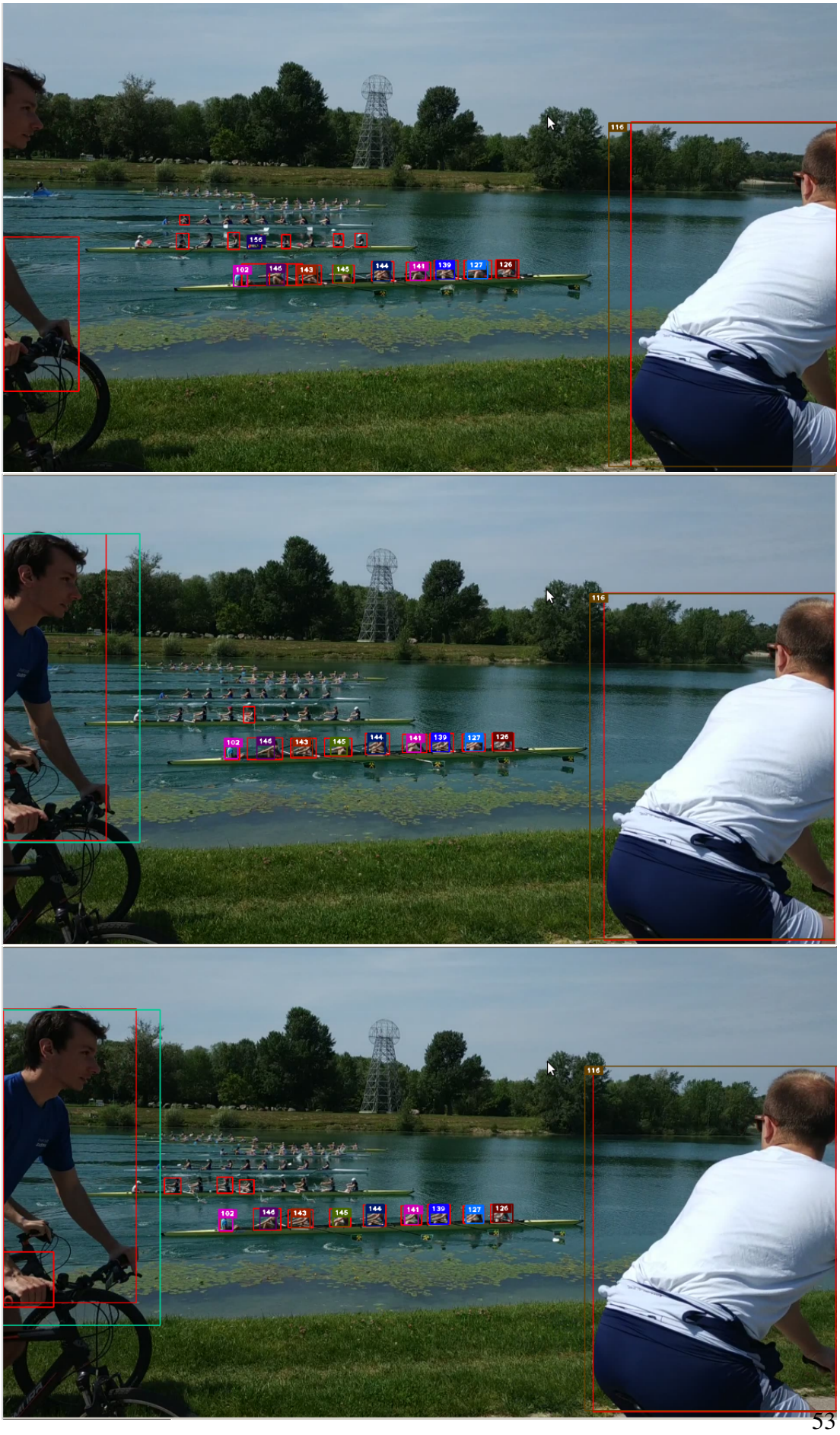
Slika 7.12: MOTA za postupak detekcije *mask*

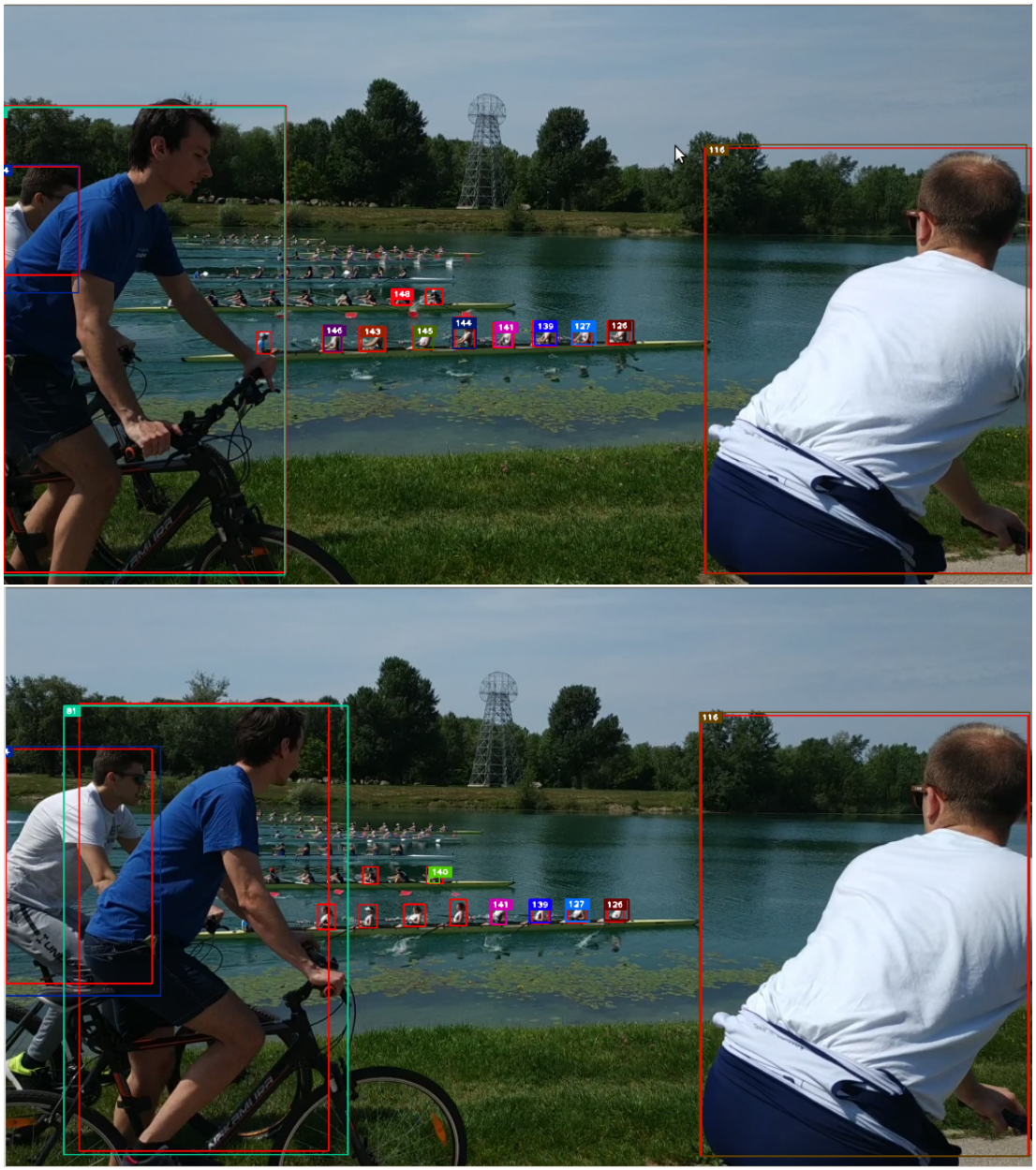
Iz ovih rezultata donosimo nekoliko zaključaka. Bolji postupak detekcije objekata ima puno veći učinak na poboljšanje postupka praćenja nego neka prikladnija funkcija gubitka metričkog ugrađivanja. Razlika u metričkom ugrađivanju malo više dolazi do izražaja kod slabijeg detektora, dok su s boljim detektorom razlike manje. Uz to, iako je na zadataku ponovne identifikacije ugrađivanje s gubitkom *cosine softmax* imalo najbolje rezultate, kada se primijeni na postupak praćenja objekata, ugrađivanje s troj-

nim gubitkom ima bolje rezultate od preostala tri pristupa, i to više dolazi do izražaja kad se gledaju brojevi zamjena identiteta nego MOTA rezultat. Dakle, zaključci izvedeni na temelju usporedbe performansi na zadatku ponovne identifikacije kao nekakve intrinzične mjere performansi metričkog ugrađivanja ne odgovaraju u potpunosti ekstrinzičnoj mjeri performansi tog ugrađivanja kao dijela sustava za praćenje objekata. Također, dulje treniranje mreže nije značajno popravilo MOTA rezultat, ali je značajno smanjilo broj zamjena identiteta. To ukazuje na netrivijalnost evaluacije postupaka praćenja i potrebu za razmatranjem više različitih mjera uspješnosti.

7.5. Demonstracija na primjerima

Na kraju prilažemo niz od 5 slika videozapisa snimljenog mobilnim uređajem u stvarnom svijetu, na veslačkoj regati na jezeru Jarun u Zagrebu. Vidimo da usprkos pomicanju same kamere (videozapis je sniman s bicikla) i promjenama oblika samih objekata koji se prate iz slike u sliku (osobe na slici veslaju) postupak većinom uspijeva pratiti one objekte koje uspije detektirati. U pozadini su osobe koje detektor nije uspio detektirati, te ih se tada niti ne može pratiti.


116
156
102
146
143
145
144
141
139
127
126



Slika 7.13: Pikaz izlaza postupka praćenja

8. Zaključak

U okviru ovog rada proučeni su postupci učenja metričkih ugrađivanja i to u kontekstu primjene na problem praćenja objekata u sceni. Metričko se ugrađivanje uči na odvojenom zadatku (ponovna identifikacija osoba) i skupu podataka, te se razmatra intrinzična uspješnost tog ugrađivanja evaluiranjem na tom zadatku, a zatim se naučeni model ponovno iskorištava za sličnu svrhu (određivanje sličnosti dijelova slika na kojima su osobe) kao dio većeg sustava za praćenje objekata u sceni, te se tada razmatra utjecaj različitih modela metričkog ugrađivanja kao ekstrinzična mjera uspješnosti.

Sustav za praćenje objekata koji koristimo izveden je pomoću više neovisnih modula - detekcija objekata, procjena gibanja, metrika sličnosti dijelova slike i optimalno uparivanje. U tom smo sustavu varirali metrike sličnosti (metrička ugrađivanja), ali i postupke detekcije objekata, da se dobije dojam o relativnom utjecaju različitih metrika. Pokazuje se da bolji postupak detekcije objekata ima znatno veći utjecaj od boljeg metričkog ugrađivanja. Štoviše, što je bolji postupak detekcije objekata, razlike u performansama cijelog sustava dobivene variranjem modela metričkog ugrađivanja sve su manje.

Izlazni sloj s modificiranom funkcijom *softmax*, u radu nazvan *cosine softmax* osmišljen je s ciljem boljeg metričkog ugrađivanja jer prisiljava primjere istog razreda da teže istoj točki u prostoru. Ipak, pokazuje se da iako na zadatku ponovne identifikacije daje najbolje rezultate, to nije slučaj kada se takav model primijeni na praćenje objekata. Naime, trojni gubitak se ističe kao najbolji, gledano po rezultatima na problemu praćenja objekata. To nam govori da podzadatak uspoređivanja dijelova slike u sklopu problema praćenja objekata ne odgovara u potpunosti problemu ponovne identifikacije osoba. Uz to, kad razmatramo problem praćenja objekata, *cosine softmax* nije ništa uspješniji od standardnog *softmaxa*.

U budućem radu zanimljivo bi bilo zamijeniti klasične algoritme koji se koriste za procjenu gibanja i optimalno dodjeljivanje nekim naučenim postupkom strojnog učenja, te istražiti koliki je skok u performansama cijelog sustava kad se neki od tih dijelova poboljša. Također, valjalo bi istražiti kolika je razlika u performansama metričkog ugrađivanja ako se iskoristi neka veća neuronska kao *feature extractor* od one koju smo koristili u ovome radu.

Zbog svoje izazovnosti i teške evaluacije, kao i zbog brojnih i važnih praktičnih primjena, problem praćenja objekata ostaje vrlo zanimljivo područje za daljnja istraživanja.

LITERATURA

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, i Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2016.

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, i Ben Upcroft. Simple online and realtime tracking. 2016. doi: 10.1109/ICIP.2016.7533003.

[3] Taco S. Cohen i Max Welling. Group equivariant convolutional networks. 2016.

[4] Jesse Davis i Mark Goadrich. The relationship between precision-recall and roc curves. U *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, stranice 233–240, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143874. URL http://doi.acm.org/10.1145/1143844.1143874.

[5] Tai Do Nhu, In Na, Guee-Sang Lee, Hyung-Jeong Yang, i S.H. Kim. Tracking by detection of multiple faces using ssd and cnn features, 09 2018.

[6] John Duchi, Elad Hazan, i Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, Srpanj 2011. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1953048.2021068.

[7] Vincent Dumoulin i Francesco Visin. A guide to convolution arithmetic for deep learning, 2016.

[8] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, i Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, Rujan 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.167. URL http://dx.doi.org/10.1109/TPAMI.2009.167.

[9] Costis Georgiou, Hamed Hatami, i Avner Magen. Csc2414 - metric embeddings, lecture 1: A brief introduction to metric embeddings, examples and motivation, 2006.

[10] Ian Goodfellow, Yoshua Bengio, i Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[11] Raia Hadsell, Sumit Chopra, i Yann LeCun. Dimensionality reduction by learning an invariant mapping. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 2:1735–1742, 2006.

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

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, i Ross Girshick. Mask r-cnn. 2017.

[14] Alexander Hermans, Lucas Beyer, i Bastian Leibe. In defense of the triplet loss for person re-identification. 2017.

[15] T. S. Huang. Computer vision: Evolution and promise. *International conference; 5th, High technology: Imaging science and technology*, stranice 13–20, 1996.

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

[17] Katarzyna Janocha i Wojciech Marian Czarnecki. On loss functions for deep neural networks in classification, 2017.

[18] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82:35–45, 1960.

[19] Lawrence K. Saul Kilian Q. Weinberger. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10: 207–244, 2009.

[20] Alex Krizhevsky, Ilya Sutskever, i Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. U F. Pereira, C. J. C. Burges, L. Bottou, i K. Q. Weinberger, urednici, *Advances in Neural Information Processing Systems 25*, stranice 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-netwo pdf.

[21] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, i Konrad Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. 2015.

[22] Lu Tian† Shengjin Wang Jingdong Wang Qi Tian Liang Zheng, Liyue Shen†. Scalable person re-identification: A benchmark. 2015.

[23] Yifan Sun Jingdong Wang Chi Su Shengjin Wang Qi Tian Liang Zheng, Zhi Bie. Mars: A video benchmark for large-scale person re-identification. 2016.

[24] Lu Lu, Yeonjong Shin, Yanhui Su, i George Em Karniadakis. Dying relu and initialization: Theory and numerical examples, 2019.

[25] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, Xiaowei Zhao, i Tae-Kyun Kim. Multiple object tracking: A literature review. 2014.

[26] Yuan Li ; Chang Huang ; Ram Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. 2009.

[27] Chigozie Nwankpa, Winifred Ijomah, Anthony Gachagan, i Stephen Marshall. Activation functions: Comparison of trends in practice and research for deep learning, 2018.

[28] Padmanabhan Soundararajan Vasant Manohar John Garofolo Rachel Bowers Matthew Boonstra Valentina Korzhova Jing Zhang Rangachar Kasturi, Dmitry Goldgof. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:319 – 336, 2009.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, i Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. 2015.

[30] Oren Rippel, Manohar Paluri, Piotr Dollar, i Lubomir Bourdev. Metric learning with adaptive density discrimination. 2015.

[31] Karen Simonyan i Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2014.

[32] Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, i Bastian Leibe. Mots: Multi-object tracking and segmentation. 2019.

[33] Nicolai Wojke i Alex Bewley. Deep cosine metric learning for person re-identification. 2018. doi: 10.1109/WACV.2018.00087.

[34] Nicolai Wojke, Alex Bewley, i Dietrich Paulus. Simple online and realtime tracking with a deep association metric. 2017.

[35] Pin Wu, Yang Yang, i Xiaoqiang Li. Stegnet: Mega image steganography capacity with deep convolutional network. 2018. doi: 10.3390/fi10060054.

[36] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, i Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. 2016.

[37] Matthew D. Zeiler. Adadelta: An adaptive learning rate method, 2012.

[38] Matthew D Zeiler i Rob Fergus. Visualizing and understanding convolutional networks, 2013.

Metrička ugrađivanja za reprezentiranje osoba u slikama

Sažetak

Metrička ugrađivanja vrlo su važan sastojak primjena u kojima ostvarujemo korespondenciju slika iste osobe u različitim trenutcima. Posebno su zanimljive primjene vezane uz praćenje i ponovnu identifikaciju osoba. U posljednje vrijeme, veliki uspjeh u tom području ostvaruju duboki modeli učeni različitim metričkim gubitcima. U okviru rada, proučeni su i ukratko opisani postojeći pristupe za učenje metričkih ugrađivanja. U okviru rada, uhodani su postupci nadziranog učenja prikladnih dubokih modela. Napravljena je usporedba utjecaja različitih metričkih ugrađivanja i različitih postupaka detekcije objekata na performanse postupka praćenja. Prikazani su rezultati eksperimenata i predloženi su pravci budućeg razvoja.

Ključne riječi: računalni vid, metričko ugrađivanje, praćenje, detekcija objekata, duboko učenje, strojno učenje, konvolucijske neuronske mreže, prijenos znanja

Metric embeddings for representing persons in images

Abstract

Metric embeddings are very important when we need to achieve a correspondence between images of the same person at different moments in time. Applications in multiple person tracking and person re-identification are particularly interesting. As of late, deep models trained with different metric losses have been very successful in that area. In this thesis, we use supervised learning to train different deep models for this task. We also compare the effects of different metric embeddings and different object detection procedures on the performance of the tracking procedure. We show the experiment results and suggest promising directions for future work.

Keywords: computer vision, metric embedding, tracking, object detection, deep learning, machine learning, convolutional neural networks, transfer learning