Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tim Oblak

Detekcija uhljev s pomočjo konteksta

diplomsko delo

visokošolski strokovni študijski program
prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Peter Peer
Somentor: izr. prof. dr. Vitomir Štruc

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Detekcija uhljev s pomočjo konteksta

Tematika naloge:

Pri biometrični razpoznavi je prvi korak povezan z detekcijo. Na osnovi dela, ki predlaga uporabo konvolucijskih nevronskih mrež za detekcijo uhljev na slikah, razširite pristop tako, da uporabite kontekst obraza. Vse spremembe arhitekture opišite in argumentirajte, nato vse zasnovane arhitekture testirajte nad standardizirano podatkovno bazo ter uporabite standardizirane evalvacijske metrike.

Za dano priložnost in podporo se zahvaljujem mentorjema izr. prof. dr. Petru Peeru in izr. prof. dr. Vitomirju Štrucu. Še posebej se zahvaljujem doktorskemu študentu in asistentu Žigu Emeršiču za potrpežljivost ter vodenje skozi raziskovalni proces. Za koristne napotke se zahvaljujem tudi doktorskemu študentu in asistentu Blažu Mednu.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
CNN	convolutional neural network	konvolucijska nevronska mreža
IoU	intersection over union	razmerje med presekom in unijo
MSE	mean square error	srednja kvadratna napaka
ROC	receiver operating characteristic	karakteristika delovanja sprejemnika
AUROC	area under receiver operating characteristic	površina pod karakteristiko delovanja sprejemnika

Povzetek

Naslov: Detekcija uhljev s pomočjo konteksta

Avtor: Tim Oblak

Zaradi svojih biometričnih lastnosti uhlji predstavljajo zanesljiv in edinstven vir informacij, še posebej uporaben na področju identifikacije ljudi. Pogoj za uspešno prepoznavo uhlja je učinkovit način detekcije, ki kljub prekrivanju in različnim pozam objekte detektira z relativno velikim priklicem. V tej diplomski nalogi predstavimo nov način detekcije uhljev, ki z namenom potencialne izboljšave napovedovanja koristi informacijo o kontekstu obraza. Za potrditev domneve smo v začetku izboljšali eno od obstoječih metod detekcije. Rezultate slednje smo utežili z lokalizacijo potencialnih področij uhljev. Na koncu smo zgradili še lasten cevovod, ki informacijo o kontekstu prejme že na začetku. Rezultat končnega cevovoda je izrazito povečan priklic detekcij glede na posamezne slikovne točke, poleg tega pa ohranimo relativno dobro mero preciznosti. Na testni množici 250 slik tako dosežemo izboljšavo dodatnih 28,5 odstotnih točk po meri Jaccardovega koeficienta podobnosti.

Ključne besede: biometrija, kontekst, uhelj, detekcija, globoko učenje.

Abstract

Title: Context-aware ear detection

Author: Tim Oblak

Because of their biometric properties, ears provide a reliable and unique source of information that is useful in the field of human identification. The condition for successful ear recognition is an effective detection method, which, despite various occlusions and poses, detects objects with a relatively large recall rate. In this thesis we present a novel approach to ear detection, which uses additional face context information for potential prediction improvement. In order to confirm the presumption, we first improved one of the existing detection methods. The results of the latter are weighted using localization of potential ear areas. Finally, we designed our own pipeline, which uses face context information from the beginning. The result of the final pipeline is a significant increase in pixel-wise detection recall rate, while preserving a relatively high measure of precision. On our test set of 250 images, we achieve an improvement of 28.5 percentage points according to the Jaccard coefficient of similarity.

Keywords: biometrics, context, ear, detection, deep learning.

Poglavje 1

Uvod

Uhlji so zaradi svojih biometričnih lastnosti dober kandidat za prepoznavo v namene varnostnega nadzorovanja ali identifikacije ljudi. Njihova struktura zadovoljuje biometričnim karakteristikam, kot so univerzalnost, posebnost, trajnost in sposobnost zbiranja podatkov. Razpoznava uhlja nam v kombinaciji z ostalimi biometričnimi značilkami človeka, kot je prstni odtis ali očesna šarenica, daje dovolj natančne rezultate, da lahko osebo identificiramo z zadovoljivim zaupanjem [33].

V praksi največkrat naletimo na nenadzorovano okolje, kjer je podatke treba ustrezno obdelati, preden so pripravljeni za postopek prepoznave. Prvi korak na področju razpoznave uhljev je torej metoda detekcije. Lahko trdimo, da zaradi problema detekcije v praksi razpoznava uhljev ni tako razvita kakor razpoznava drugih biometričnih značilk. Ta je lahko težavna zaradi različnih poz glave, ali pa zaradi prekrivanj, recimo las, pokrival, uhanov in podobno.

Problem detekcije uhljev rešujemo s pomočjo klasičnih pristopov na področju računalniškega vida, ali pa iz podatkovne zbirke pridobimo znanje z metodami strojnega učenja [3]. Oblika uhljev se s staranjem človeka običajno ne spreminja [20]. To nam omogoča, da zgrajen model skozi življenjsko dobo uporabe daje konsistentne rezultate.

Zaradi hitrega razvoja grafičnih procesorskih enot lahko v zadnjem času

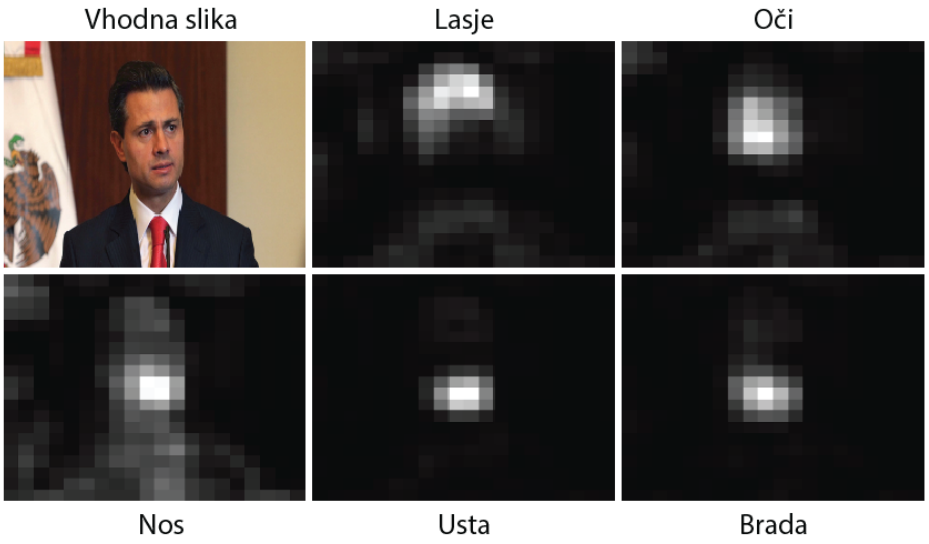
na področju računalniškega vida opazimo močan trend uporabe konvolucijskih nevronskih mrež (angl. *convolutional neural network – CNN*). Z njihovo pomočjo lahko iščemo rešitve za širok spekter problemov, pojavili pa so se tudi sveži pristopi za že obstoječe rešitve [16]. Načrtovana arhitektura naše raziskave v večini temelji na sistemih globokih nevronskih mrež.

Kot rešitev problema predlagamo način detekcije uhljev, ki natančnost napovedi izboljša s pomočjo informacij ostalih delov obraza. V zaključku raziskave [32] avtorji namreč zapišejo, da bi potencialno izboljšanje detekcije dosegli, če bi uhlje iskali le v bližini obrazov ali v relaciji z določenimi deli obraza. To poimenujemo kontekst obraza, saj nam lastnosti ostalih obraznih značilk podajo sveže informacije o morebitni poziciji uhljev. Te naj bi manjkajoče informacije uhljev v primeru prekrivanj ali ekstremnih poz predvidoma dopolnile.

Ključna raziskava, ki nam omogoča realizacijo omenjenega pristopa, je *From Facial Parts Responses to Face Detection: A Deep Learning Approach* [23], v nadaljevanju imenovana *FacenessNet*. Prvi del raziskave obravnava skupek konvolucijskih nevronskih mrež, katerih glavna naloga je, da iz slike obraza izluščijo pomembne obrazne značilke, kot so lasje, oči, nos, usta in brada. Te so prikazane na sliki 1.1. Na zgornjem delu v sredini je prikazana verjetnostna porazdelitev za lase, na desni za oči, spodaj pa v zaporedju od leve proti desni še za nos, usta in brado.

Za vsak del obraza je mrežo treba najprej naučiti na določeni zbirki podatkov, tako da na koncu dobimo pet modelov uteži, ki sovpadajo s posameznim delom obraza. Za vsak del obraza se v osnovi uporabi enaka arhitektura mreže. V nadaljevanju avtorji predlagajo še določene metode, s katerimi lahko na dobljenih verjetnostnih porazdelitvah detektiramo obraze. Nekaj jih bomo omenili v nadaljevanju.

S posameznimi verjetnostnimi porazdelitvami značilk si ne moremo veliko pomagati. Če recimo gledamo le največji odziv znotraj določene porazdelitve, ni nujno, da ta sovpada z ustreznim delom obraza na prvotni sliki. Na napoved mreže *FacenessNet* lahko vpliva šum v sliki ali pa je tekstura na



Slika 1.1: Primer rezultata CNN *FacenessNet*, ki predstavlja verjetnostne porazdelitve za posamezne dele obraza.

tistem območju relativno podobna teksturi dela obraza. Verjetnostne porazdelitve moramo torej gledati globalno in analizirati relacije med vzorci, ki jih vsebujejo.

V poglavju 2 pregledamo sorodna dela, na katerih raziskava temelji. V poglavju 3 sledi opis uporabljenih metod v obliki inkrementalne izboljšave učinkovitosti pristopov. Vmesne rezultate na kratko komentiramo in prikažemo ustrezno slikovno gradivo. V poglavju 4 sledi analiza rezultatov, kjer predstavimo uspešnost opisanih metod in razložimo določene vzorce, ki se pojavijo med končnimi rezultati. Zaključek je podan v poglavju 5, kjer povzamemo rezultate in podamo možnosti za potencialne izboljšave.

Poglavje 2

Sorodna dela

Obstoječe metode za detekcijo uhljev po večini uporabljajo klasične pristope izločanja značilk, strojnega učenja ali kombinacije obeh metod [3]. Klasični pristopi kot referenčno točko največkrat uporabljajo geometrijsko obliko uhlja [9]. Nekateri raziskovalci se detekcije lotijo s pomočjo detektorja robov Canny [13, 22]. Med izluščenimi robovi iščejo določene geometrične lastnosti [10], ali s pomočjo Houghove transformacije računajo prileganje elips [21]. Uporabljene so tudi metode primerjanja predlog, kjer z drsnim oknom primerjamo referenčno sliko z delom prvotne slike [14]. Prav tako izločanje značilk kombinirajo s strojnim učenjem [17].

Klasični pristopi v primeru nekontroliranega okolja običajno dajejo slabše rezultate, saj slike v takšnem okolju vsebujejo različne svetlobne pogoje, poze iskanega objekta in razna prekrivanja na območju interesa [3].

Avtorji članka *Pixel-wise Ear Detection with Convolutional Encoder-Decoder Networks* [32] za namen detekcije uhljev predlagajo uporabo arhitekture *SegNet* [6]. Gre za CNN, ki temelji na konceptu kodirnikov in dekodirnikov. Prvi del mreže iz prvotnih slik izlušči pomembne značilke, drugi del pa jim poveča ločljivost. Na vhodu sprejema slike velikosti 360×480 slikovnih točk, vrne pa nam rezultat v obliki binarne maske, kjer vrednost posamezne slikovne točke določa razred istoležni slikovni točki na prvotni sliki. Glede na rezultate raziskave [32] je ta način v primerjavi s klasičnimi

pristopi bolj odporen na razne moteče elemente in dosega dobre rezultate kljub spremenljivim okoljskim pogojem.

Ta diplomska naloga v večji meri temelji na omenjeni raziskavi detekcije uhljev. V nadaljevanju se na omenjeno raziskavo zato sklicujemo kot na *prvotno raziskavo*. Primer detekcije uhljev je prikazan na sliki 2.1. Na levi strani slike je z zeleno barvo označeno anotirano področje uhlja, na desni strani pa je prikazan rezultat napovedi modela.

Slika 2.1: Prikaz detekcije prvotne raziskave, na kateri diplomsko delo temelji.

Uspešnost obstoječih raziskav težko primerjamo, saj si raziskovalci glede izbire podatkovne baze in metrik uspešnosti na področju detekcije uhljev niso enotni. Zavoljo lažje primerjave te izračunamo na enak način, kot je to storjeno v raziskavi [32].

Poglavje 3

Metode

Za izboljšavo trenutnih rezultatov predlagamo rešitev v treh fazah. V prvih dveh glede na lokacijo utežujemo rezultate obstoječe raziskave detekcije uhljev, poleg tega raziščemo možnosti, ki nam jih verjetnostne porazdelitve delov obraza ponujajo. V zadnji fazi predlagamo nov način detekcije, ki informacijo o kontekstu uporablja že v fazi učenja. Svojo arhitekturo načrtujemo v obliki cevovoda, ki mu na vhod podamo sliko, na izhodu pa nam vrne napovedano detekcijo uhljev. V nadaljevanju na kratko opišimo vsako fazo naloge:

V prvi fazi s pomočjo izluščenih verjetnostnih porazdelitev mreže *FacenessNet* [23] določimo pozicijo dveh koordinat, ki sta uporabljeni kot referenčni točki, v nadaljevanju poimenovani tudi *sidri*. Točki predstavljata predvideno lokacijo uhljev na sliki. S pomočjo teh lokacij nato glede na velikost in oddaljenost utežimo posamezne skupine slikovnih točk, ki so klasificirane kot pozitiven razred uhlja. Pomagamo si z metodami raziskave *FacenessNet*, konkretno z računanjem mere *Faceness* in klasifikacijo obraza. Cilj prve faze je zgolj potrditev domneve, da nam kontekst obraza pri detekciji uhlja lahko koristi.

V drugi fazi načrtujemo svojo metodo iskanja značilk tako, da zmanjšamo časovne stroške prve faze. Ta se lokacij sider nauči iz verjetnostnih porazdelitev obraznih značilk. V ta namen uporabimo regresijsko CNN,

tako da ta napoveduje štiri zvezne vrednosti, ki predstavljajo napovedane lokacije sider. Natančnost ocene merimo s pomočjo evklidske razdalje. Poleg napovedovanja lokacij izboljšamo tudi način uteževanja prve faze.

V tretji fazi predlagamo modifikacijo osnovne mreže *SegNet* [6], tako da ta poleg prvotne slike na vhod prejme tudi verjetnostne porazdelitve posameznih delov obraza. Podatki na vhodu so torej združeni v obliki matrike velikosti $360 \times 480 \times 8$, kjer prvi trije kanali predstavljajo barvne kanale prvotne slike, ostalih pet pa verjetnostne porazdelitve delov obraza.

3.1 Lokalizacija s pomočjo detekcije obrazov

Za lokalizacijo potencialnih področij uhljev smo v začetku implementirali detektor obraza raziskave *FacenessNet* [23].

3.1.1 Pridobitev verjetnostnih porazdelitev

V prvem delu raziskave *FacenessNet* smo s pomočjo CNN generirali verjetnostne porazdelitve za vsak del obraza. Mreža je sestavljena iz sedmih konvolucijskih plasti, kjer prvima dvema sledi še plast združevanja (angl. *pooling*). Arhitektura mreže z veliko zaporedno povezanimi konvolucijskimi plastmi ima namreč sposobnost grobe lokalizacije naučenih objektov na sliki. Uteži že naučene mreže lahko dobimo na spletni strani avtorjev: `http://shuoyang1213.me/projects/Faceness/Faceness.html`. Ko na mrežo naložimo uteži modela in na vhod podamo sliko, nam ta na izhodu vrne verjetnostno porazdelitev preko vseh slikovnih točk na sliki za določen del obraza.

Za vsako podano sliko smo tako dobili pet matrik, kjer višja vrednost elementa matrike pomeni večjo verjetnost, da se na istoležni lokaciji v prvotni sliki nahaja ustrezen del obraza. Na koncu smo matrikam povečali ločljivost, tako da je njihova velikost enaka velikosti slik v zbirki.

3.1.2 Območja interesa

Po uspešnem generiranju verjetnostnih porazdelitev smo se za vsako od slik v naši podatkovni množici lotili iskanja morebitnih območij interesa. Avtorji *FacenessNet* za iskanje predlagajo uporabo enega od algoritmov za detekcijo generičnih objektov, kot so *Selective Search* [19], *Multiscale Combinatorial Grouping* [24] in *EdgeBoxes* [8, 30, 31]. Uporabili smo slednjega, saj je najhitrejši od treh.

Omenjeni detektor na določeni sliki s pomočjo detekcije robov poišče območja interesa, za katera naj bi z veliko verjetnostjo veljajo, da vsebujejo kakršenkoli objekt. Ker detektor predloge vrne z velikim priklicem, smo lahko z veliko verjetnostjo prepričani, da vsaj en predlog vsebuje območje obraza. Podobnih predlogov se znebimo z metodo dušenja ne-maksimumov.

3.1.3 Računanje mere *Faceness*

Kot je bilo omenjeno v poglavju 1, med vsemi verjetnostnimi porazdelitvami iščemo določene relacije. Glede na to, da so deli človeškega obraza s frontalnega pogleda razporejeni v določenih proporcih, lahko to lastnost izkoristimo za računanje korelacije med verjetnostnimi porazdelitvami.

V nadaljevanju raziskava predlaga uporabo t. i. mere *Facensess* . Ta na določenem območju interesa na sliki predstavlja relacijo med posameznimi porazdelitvami obraznih značilk. Pove nam, kolikšna je verjetnost, da je na tem področju slike obraz.

Naj okno W z oglišči $ABCD$ predstavlja predlagano območje, ki ga generiramo z algoritmom za detekcijo generičnih objektov iz poglavja 3.1.2. V predlaganem območju iščemo največji odziv posamezne verjetnostne porazdelitve, območje katerega predstavlja okno w z oglišči $EFGH$. Δ_w predstavlja vrednost mere *Faceness* za okno w. Če z enačbo $\Sigma(X) = \sum_{i=0}^{X_{width}} \sum_{j=0}^{X_{height}} x_{ij}$ izračunamo vsoto vseh vrednosti v oknu X, potem lahko zapišemo enačbo za izračun mere Δ_w (enačba 3.1).

$$\Delta_w = \frac{\Sigma(w)}{\Sigma(W) - \Sigma(w)} \tag{3.1}$$

Oglišča okna w torej predstavljajo točke $EFGH$. Lokacije teh točk se lahko naučimo iz podatkov tako, da izračunamo povprečno območje posameznega dela obraza znotraj anotiranih oken obraza. Koordinate točk $EFGH$ zapišemo s pomočjo linearnih kombinacij (enačba 3.2). Primer oken W in w je prikazan na sliki 3.1.

$$\begin{aligned} E &= (x_A, \lambda_1 y_A + (1-\lambda_1) y_D) \\ F &= (x_B, \lambda_1 y_B + (1-\lambda_1) y_C) \\ G &= (x_A, \lambda_2 y_A + (1-\lambda_2) y_D) \\ H &= (x_B, \lambda_2 y_B + (1-\lambda_2) y_C) \end{aligned} \tag{3.2}$$

Vrednost λ v tem primeru predstavlja koeficient linearne kombinacije y koordinate točk AD ali točk BC. Okno lahko z različnimi vrednostmi λ razdelimo na dva ali tri dele.

Primer 3.1 *Če je običajno največji odziv verjetnostne porazdelitve las v zgornji tretjini prvotnega okna W, potem drži $\lambda_1 = 1$ in $\lambda_2 = \frac{2}{3}$. Tako dobimo oglišča okna w z vrednostmi $E = (x_A, 1y_A + 0y_D) = A$, $F = (x_B, 1y_B + 0y_C) = B$, $G = (x_B, \frac{2}{3}y_B + \frac{1}{3}y_C)$ in $H = (x_A, \frac{2}{3}y_A + \frac{1}{3}y_D)$.*

Mero Δ_w smo na takšen način izračunali za verjetnostno porazdelitev vsakega dela obraza posebej. Vrednosti teh smo nato normalizirali glede na njihovo minimalno in maksimalno vrednost ter jih sešteli. Na koncu smo vsoto modelirali še s sigmoidno funkcijo in kot rezultat dobili verjetnost na območju od 0 do 1.

Okna predlogov so na tej točki omejena le na območja obrazov, saj smo ostala odstranili s filtriranjem mere *Faceness*. Še vedno pa je priklic oken velik, saj je mera *Faceness* lahko velika pri malih oknih, ki vsebujejo le posamezen del obraza, ali pa pri večjih oknih, v katerih je poleg celotnega obraza tudi relativno veliko ozadja.

3.1.4 Klasifikacija obrazov

Dodatno filtriranje oken zahteva uporabo klasifikatorja obrazov. V ta namen smo uporabili konvolucijsko nevronsko mrežo *SqueezeNet* [18]. Ta temelji na mreži *AlexNet* [15], le da je prva bolj kompaktna in vsebuje manj parametrov, hkrati pa dosega primerljivo natančnost. Učenje in napovedovanje sta posledično hitrejša.

Klasifikator smo naučili s pomočjo doučitve (angl. transfer learning) že naučenih uteži podatkovne zbirke *ImageNet* [11]. Potrebna je bila modifikacija zadnje konvolucijske plasti mreže, da namesto 1000 razredov, ta napoveduje le verjetnosti za dva razreda; obraz in vse ostalo. Ostalim plastem smo obstoječe uteži zamrznili, tako da so ohranile naučene značilke z večje baze *ImageNet*. Takšna modifikacija nam omogoča, da se mreža hitreje prilagodi novemu problemu, zato je učenje hitrejše, prav tako je učna množica lahko manjša kot sicer [25].

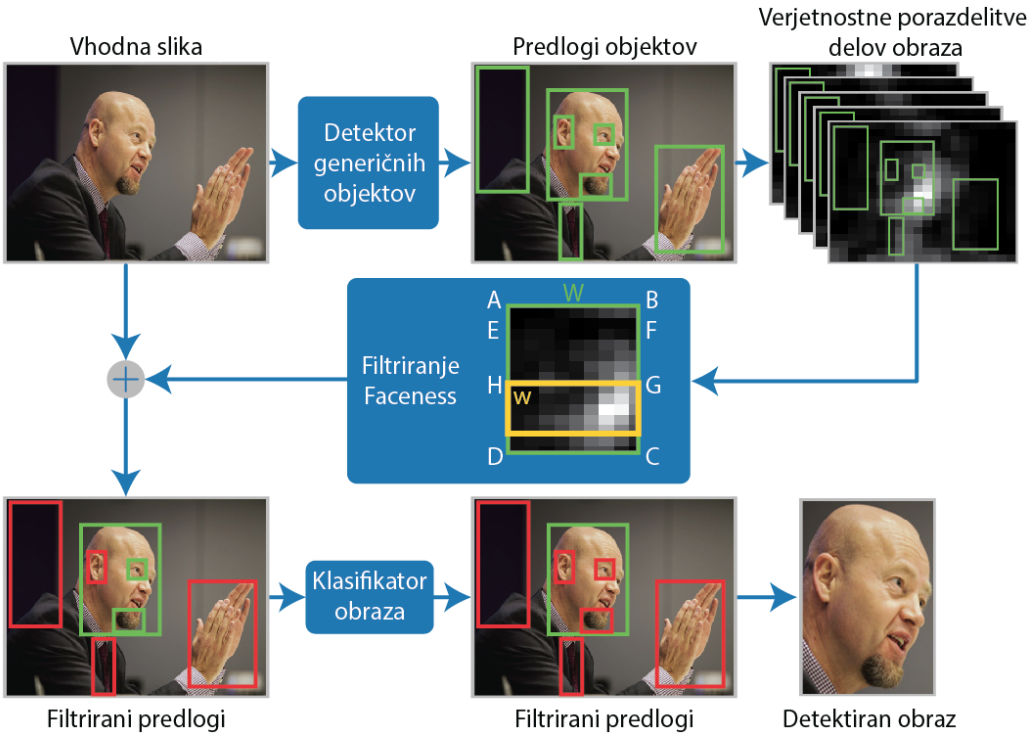
Za učno množico smo uporabili obrezane slike obrazov iz podatkovne množice *AFLW* [2] in slike brez obrazov iz množice *VOC2007* [29]. Slik obrazov je bilo približno 22000, slik brez obrazov pa dodatnih 6000. Testna množica je vsebovala približno 15 % slik.

Problem klasifikacije obraza se je izkazal za dokaj enostavnega, saj je vrednost funkcije napake pri učenju hitro konvergirala, ob hkratni klasifikacijski točnosti modela 99,7 %.

Vsa območja interesa iz prejšnjega koraka smo z opisano metodo klasificirali in odstranili tiste z negativnim razredom ozadja. Ostalo nam je le še nekaj oken z največjo verjetnostjo. Koordinate oken smo na tej točki enostavno povprečili in dobili povprečno velikost ter lokacijo okna za obraz na tistem območju. Celoten diagram detekcije obraza je prikazan na sliki 3.1.

3.1.5 Postavitev sider in izračun verjetnosti

Na oknu, ki predstavlja detektiran obraz, smo zasidrali dve referenčni točki, ki predstavljata morebitni lokaciji uhljev. Ker uhlji s frontalnega pogleda

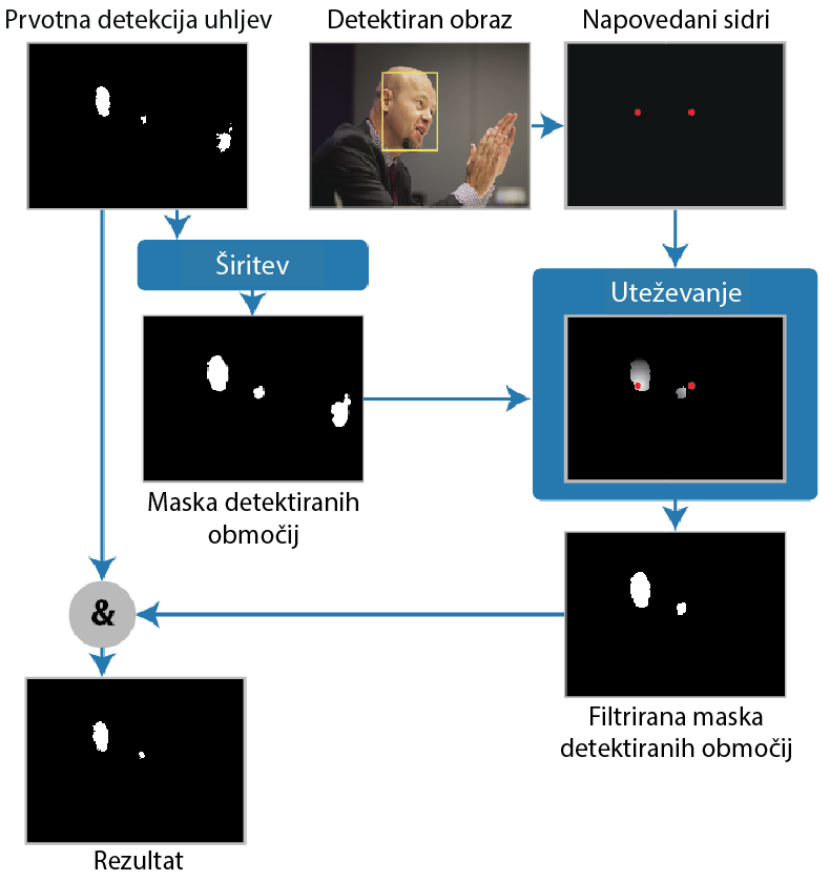


Slika 3.1: Prikaz primera filtriranja predlaganih oken v prvi fazi.

obraza ležijo približno na sredini obraza, lahko točki postavimo na sredino leve in desne stranice najdenega okna. Glede na detektirano okno W z oglišči $ABCD$ izračunamo lokacijo sider $S_{leva} = (x_A, \frac{1}{2}y_A + \frac{1}{2}y_D)$ in $S_{desna} = (x_B, \frac{1}{2}y_B + \frac{1}{2}y_C)$.

Točnost lokaliziranja sider je močno odvisna od učinkovitosti detekcije obraza na sliki. Detektirana okna so poravnana glede na osi slike, kar nas dodatno omejuje. Edine informacije, ki jih z okna lahko pridobimo, so približna višina, širina in lokacija obraza. Posledično dosegamo slabše rezultate na slikah, kjer je obraz v skrajni pozi ali ni poravnan z osmi slike.

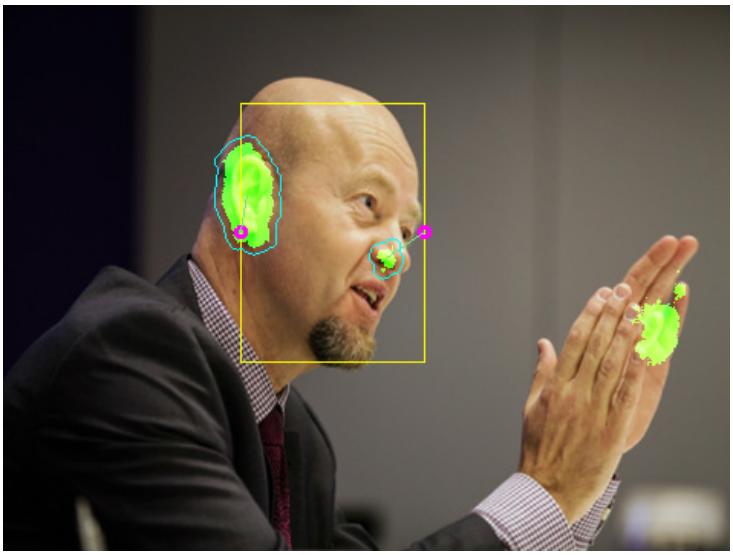
Rezultate prvotne detekcije smo s pomočjo predlogov lokacij ustrezno filtrirali. Vsaki pozitivno klasificirani slikovni točki smo izračunali utež glede na oddaljenost od najbližjega sidra. Iz binarne slike smo prav tako izluščili konture oz. obrobe vseh skupin pozitivno klasificiranih slikovnih točk. Te



Slika 3.2: Diagram uteževanja kontur prve faze.

smo razvrstili na dva dela, da vsaka kontura pripada najbližjemu oz. vodilnemu sidru. Za vsako konturo smo nato izračunali vsoto vseh uteži slikovnih točk, ki jih ta vsebuje. Ker ni nujno, da so v prvotni detekciji slikovne točke na področju uhlja tesno povezane, si lahko pomagamo z morfološkimi operacijami. S pomočjo operacije širitve smo tako generirali masko, ki posamezne detektirane slikovne točke v bližini poveže v skupine. Za vsako skupino pozitivno detektiranih slikovnih točk smo izračunali vsoto njihovih uteži. Na takšen način smo torej ohranili dve skupini slikovnih točk za največjo vsoto uteži, kjer vsaka predstavlja detektirano področje uhlja. Posledica tega je zmanjšano število napačnih pozitivno klasificiranih slikovnih točk izven ob-

sega obraza. Diagram uteževanja je prikazan na sliki 3.2. V zgornjem desnem kotu detektiranega okna pridobimo sidri, s katerima utežimo razširjeno masko prvotne detekcije. Da dobimo rezultat, razširjeno masko z najtežjima konturama množimo s prvotno detekcijo glede na vsako slikovno točko.



Slika 3.3: Prikaz primera filtriranja detekcije prve faze.

Vizualizacija metode je prikazana na sliki 3.3. Rumen pravokotnik na sliki predstavlja okno detektiranega obraza, vijolični točki pa pripadajoči referenčni točki za lokalizacijo uhljev. Modra obroba okoli detektiranih območij predstavlja izbrane skupine slikovnih točk, ki jih obravnavamo kot del uhlja. Na sliki lahko vidimo, da smo uspešno odstranili skupino napačno pozitivnih slikovnih točk na področju rok.

3.2 Lokalizacija z regresijo referenčnih točk

Zaradi odvisnosti od detektorja obrazov so časovni stroški prve faze relativno visoki, zato smo pripravili svojo metodo lokalizacije.

3.2.1 Priprava podatkov

Da bi mrežo naučili lokalizacije potencialnih pozicij uhljev, smo morali svojo podatkovno zbirko najprej ustrezno anotirati. Za vsako sliko v zbirki smo shranili središča kontur na anotiranih maskah uhljev. Tudi če drugi uhelj na sliki ni bil viden, smo manjkajočo točko ročno dodali na drugo stran obraza. V vsakem primeru sta tako anotirani točki postavljeni na nasprotnih straneh obraza.

Ker so nevronske mreže občutljive na obliko vhodnih podatkov, je bilo treba urediti tudi zbirko verjetnostnih porazdelitev. Te smo normalizirali na območje od 0 do 1. Čeprav velikega izboljšanja nismo opazili, smo na koncu normalizirali tudi anotirane koordinate. V tem primeru vrednost 1 za koordinato x predstavlja širino slike, za y pa višino.

3.2.2 Načrtovanje mreže

Za iskanje relacij med verjetnostnimi porazdelitvami ponovno predlagamo uporabo CNN. Ker pa v tem primeru ne napovedujemo razredov, ampak vektor zveznih vrednosti v obliki koordinat, klasifikacijsko nevronsko mrežo zamenjamo za regresijsko. Mreža torej v prvem delu s pomočjo konvolucije izlušči značilke verjetnostnih porazdelitev, v drugem delu pa z uporabo polno povezanih plasti aproksimira funkcijo regresije za ustrezne koordinate.

Konvolucijski del nove mreže smo si sposodili pri arhitekturi *SqueezeNet*, zadnjo plast *softmax* aktivacije pa smo odstranili. Mrežo smo ponovno uporabili zaradi njene velikosti in preprostosti, kar nam je omogočilo hitrejše prototipiranje. Ker značilke iščemo znotraj verjetnostnih porazdelitev, nam predhodno naučene uteži zbirke *ImageNet* v tej situaciji ne pomagajo. Celotno mrežo smo torej učili od začetka.

Zadnji del mreže je sestavljen iz štirih polno povezanih (angl. *dense*) plasti velikosti 4096, 4096, 512 in 4. Med vsako polno povezano plastjo smo vstavili še aktivacijsko funkcijo *ReLu* in funkcijo opustitve (angl. *dropout*) s koeficientom 0,5, tako da smo se izognili prevelikemu prileganju učni množici.

Za zadnjo polno povezano plastjo funkcije opustitve nismo uporabili, dodali pa smo linearno aktivacijsko funkcijo, saj smo pričakovali, da bo rezultat mreže poljubno zvezno število.

Za funkcijo napake smo izbrali mero *MSE* (angl. *mean squared error*), saj ta najbolje ustreza nalogi regresije. Model smo optimizirali s pomočjo optimizacijskega algoritma Adam [1]. Ta med učenjem računa drsečo sredino prvih in drugih momentov, kar mu omogoča učinkovit izračun večjih korakov ob posodobitvi gradientov. Omenjena lastnost zmanjša potrebo po finem uravnavanju parametrov optimizacije.

V fazi učenja smo mreži s pomočjo generatorja podajali svežnje verjetnostnih porazdelitev skupaj z anotiranimi središči uhljev. Mera izgube je začela konvergirati po 2000 epohah, domnevne lokacije uhljev pa je mreža napovedala s povprečno napako 18 slikovnih točk po evklidski razdalji.

Poleg lokalizacije smo izboljšali tudi prostorsko predstavo modela, saj napovedovanje lokacij ni več omejeno na okna, ki so poravnana glede na osi slike. Lokacijo uhljev lahko pridobimo tudi na slikah, ki vsebujejo obraze v skrajnih pozah.

3.2.3 Modeliranje gostote verjetnosti uhlja

Poleg lokaliziranja sider lahko izboljšamo tudi korak uteževanja. Pomagali smo si z modeliranjem gostote verjetnosti.

Da upoštevamo napako naučene mreže, smo iz testne množice vzeli manjši delež podatkov ter z njimi izračunali normalno porazdelitev za povprečno napako napovedi sider po osi x in y. Vprašamo se, kakšna je verjetnost, da naključna spremenljivka pade na določeno območje pod krivuljo porazdelitve. V tem primeru območje predstavlja višino in širino kontur. Izračunali smo torej, kakšna je verjetnost, da določena kontura pripada posameznemu sidru. Tako kot v prvi fazi, smo konture utežili glede na razmerje med njihovo velikostjo in razdaljo do napovedane lokacije najbližjega sidra. Razlika je v tem, da je tokrat funkcija nelinearna in upošteva pričakovano napako naše mreže.

	α	β
x os levega uhlja	2.00	1.87
x os desnega uhlja	1.87	2.00
y os	1.96	2.14

Tabela 3.1: Parametri povprečne porazdelitve beta za oba uhlja.

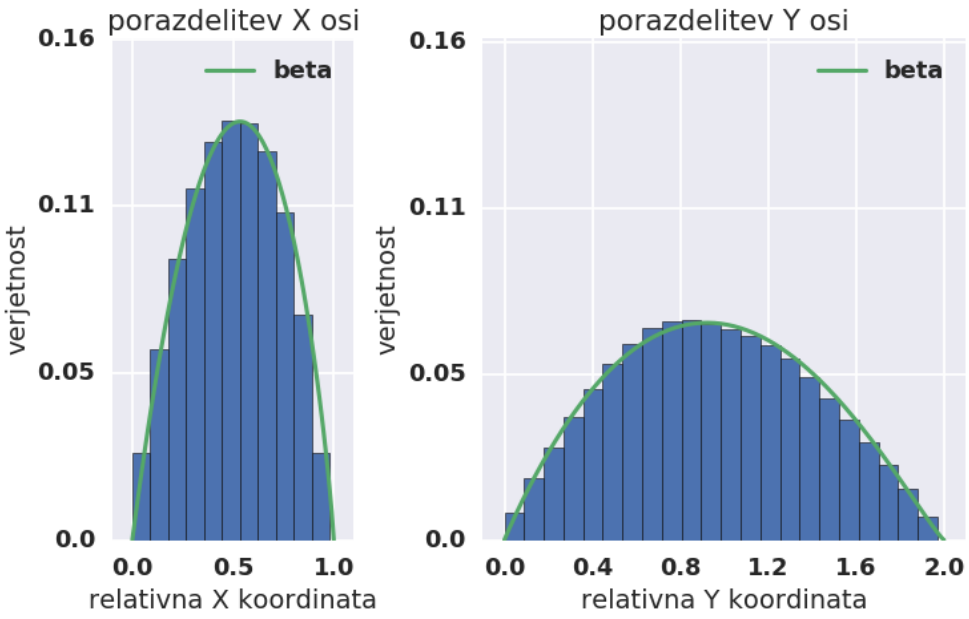
V območju konture z največjo verjetnostjo smo detekcije filtrirali še glede na vsako slikovno točko. V ta namen smo izračunali povprečno porazdelitev pozitivnih slikovnih točk na območju posameznega uhlja. Za referenco smo vzeli območje interesa, kot je to anotirano v učni množici. Območju smo nato prilagodili velikost ene od osi, tako da je med stranicama veljalo razmerje $2 \times \textit{širina} = \textit{višina}$. Povprečno razmerje med stranicama v zbirki je namreč približno $1 : 2$. Vrednost prilagoditve spremenjene osi smo si zapomnili, saj smo jo uporabili v nadaljevanju.

Primer 3.2 *V učni množici obstaja slika, na kateri je prikazan uhelj s prilegajočim oknom višine 40 in širine 18 slikovnih točk. Okno prilagodimo tako, da os x na vsaki strani povečamo za eno slikovno točko. Tako dobimo nove vrednosti višine in širine okna, ki sedaj merita 40 oziroma 20 slikovnih točk, med njima pa velja razmerje* $1 : 2$. *Vrednosti spremembe 2 in 0 za os x oziroma y si zapomnimo.*

Iz vsake maske anotiranih področij uhlja smo torej izluščili okno z razmerjem med stranicama $1 : 2$, kjer se vsaj ena stranica prilega uhlju. Vsakemu oknu smo nato spremenili velikost glede na prej določeno število slikovnih točk, prav tako v razmerju $1 : 2$. Če za vsako os seštejemo število slikovnih točk, ki na njej ležijo, dobimo histogram porazdelitve slikovnih točk uhlja. Histograme smo povprečili, potem pa smo poiskali porazdelitev, ki se povprečnemu histogramu najbolje prilega.

Za obe osi se najbolje prilega porazdelitev *beta*, prikazana na sliki 3.4. Parametra α in β za porazdelitev osi y sta enaka za levi in desni uhelj, za

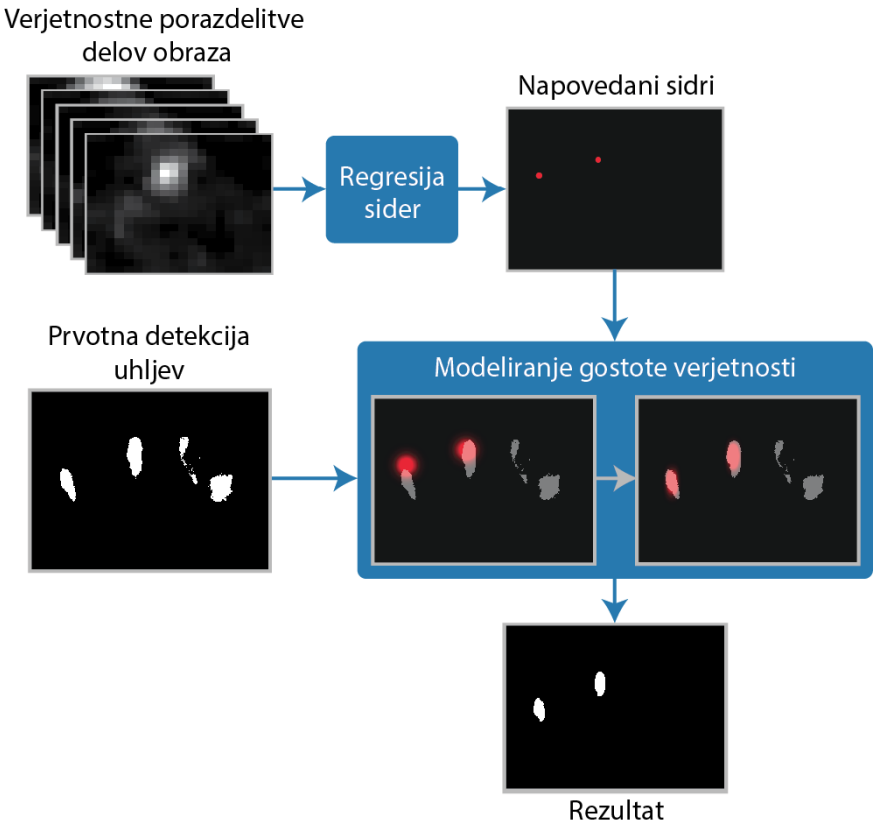
os x pa lahko parametra med sabo zamenjamo, da dobimo zrcalno obliko porazdelitve. Parametri porazdelitve so zapisani v tabeli 3.1.



Slika 3.4: Histogram vsote slikovnih točk in pripadajoča gostota verjetnosti za levi uhelj.

Želeli bi upoštevati tudi določene slikovne točke, ki padejo izven omenjene porazdelitve *beta*. Ta obnašanje naključnih spremenljivk modelira le na intervalih končne dolžine. Porazdelitev smo zato s pomočjo konvolucije zgladili z določeno normalno porazdelitvijo, ki območje razširi na celotno realno os. Gladitveno normalno porazdelitev smo izračunali tako, da smo upoštevali povprečno vrednost prilagoditve stranice. Računanje vrednosti prilagoditve je prikazano v primeru 3.2.

Merilo porazdelitve uhlja smo določili glede na razdaljo med napovedanima točkama. Na sliki 3.5 je prikazan diagram uteževanja druge faze. Z regresijo sider pridobimo referenčni točki, na kateri najprej modeliramo gostoto verjetnosti napake svoje mreže. Na konture z najvišjo verjetnostjo modeliramo še porazdelitev uhlja. To množimo z prvotno detekcijo in ustrezno



Slika 3.5: Diagram postopka regresije sider in modeliranja gostote verjetnosti.

Slika 3.6: Rezultati druge faze.

upragujemo. Z rdečo barvo sta v zgornjem delu slike označeni sidri, v sredini pa modelirane gostote verjetnosti. Primer rezultata druge faze je prikazan na sliki 3.6. Na levi strani slike so z zeleno barvo prikazana prvotno detektirana področja, na desni pa rezultat filtriranja s pomočjo regresije in modeliranja gostote verjetnosti. Verjetnostna porazdelitev uhljev je prikazana na sredini.

3.3 Modifikacija arhitekture *SegNet*

V nadaljevanju smo načrtovali lasten cevovod, ki dodatne informacije verjetnostnih porazdelitev delov obraza uporablja že v fazi učenja.

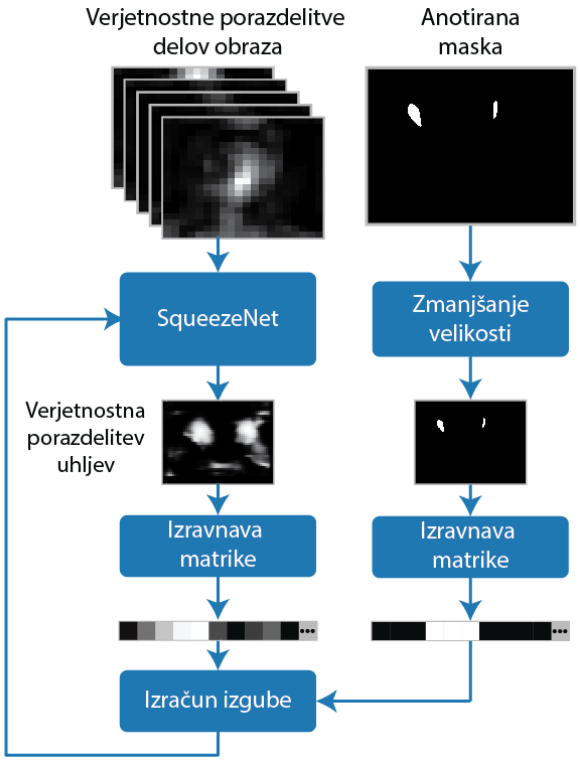
3.3.1 Iskanje značilk

Da bi potrdili smiselnost svoje ideje, smo najprej preverili, kako dobro se CNN potrebnih značilk nauči le iz verjetnostnih porazdelitev delov obraza, brez uporabe dejanske slike. Uporabili smo mrežo, s katero smo v prejšnji fazi napovedovali koordinate referenčnih točk.

Mreži smo polno povezane plasti zamenjali z eno konvolucijsko plastjo s filtrom velikosti 1, saj smo na izhodu potrebovali dvodimenzionalno matriko, ki predstavlja verjetnostno porazdelitev uhljev. Na koncu smo dodali plast, ki matriko porazdelitev oblikuje v enodimenzionalni vektor, in sigmoidno funkcijo aktivacije, ki izhodne vrednosti omeji na interval od 0 do 1.

V fazi učenja smo mreži na vhod podajali verjetnostne porazdelitve drugih delov obraza, izhod mreže pa smo primerjali z anotiranimi maskami uhljev. Maskam smo spremenili ločljivost, tako da se je ta ujemala z ločljivostjo izhodne matrike mreže, nato smo jo prav tako oblikovali v enodimenzionalni vektor. Mero izgube smo izračunali s pomočjo binarne križne entropije. Diagram učenja mreže je prikazan na sliki 3.7. Med izravnanima vektorjema značilk se izračuna mera izgube, ki se nato v sklopu vzvratnega razširjanja prenese nazaj v mrežo.

Slika 3.8 prikazuje verjetnostne porazdelitve uhljev za dva primera iz testne množice. Opazimo lahko, da se je mreža naučila določenih lastnosti



Slika 3.7: Diagram mreže med učenjem verjetnostnih porazdelitev uhljev.

uhljev. Verjetnosti so velike na obeh straneh obraza, tudi v primeru manjkajočega uhlja. Prav tako je na spodnjem delu slike vidno, da model dobro napoveduje tudi, če obraz ni poravnan z osmi slike. V naši zbirki je tudi nekaj sivinskih slik, na katerih prvotna detekcija v večini primerov uhljev ne najde. Iz zgornjega dela slike je razvidno, da mreža tokrat področje uhljev dobro oceni ne glede na barvni spekter prvotne slike. Z analizo omenjenih primerov smo lahko potrdili, da nam verjetnostne porazdelitve ostalih delov obraza nudijo sveže informacije o sliki.

3.3.2 Izpeljava arhitekture

V prejšnjem poglavju smo dokazali, da lahko z uporabo verjetnostnih porazdelitev delov obraza generiramo verjetnostne porazdelitve uhljev. Če na tej

Slika 3.8: Verjetnostne porazdelitve uhljev, ki jih s pomočjo konvolucijske nevronske mreže pridobimo le iz verjetnostnih porazdelitev drugih delov obraza za prikazano sliko.

točki konvolucijskemu delu mreže dodamo njeno zrcalno obliko, ki značilkam poveča ločljivost, se približamo arhitekturi *SegNet*. Ta kot osnovo kodirnika predlaga uporabo konvolucijske mreže *VGG16* [28], ki je od trenutne mreže *SqueezeNet* globlja, vsebuje več parametrov, posledično pa dosega boljše rezultate.

V nadaljevanju smo implementirali različico arhitekture *SegNet*. Ta se od običajnih arhitektur kodirnik-dekodirnik razlikuje po tem, da ob večanju ločljivosti značilk uporablja indekse, ki si jih zapomni ob združevanju značilk. Ker smo se problema lotili postopoma, nas omenjena funkcionalnost pomnenja indeksov še ni zanimala. Za prvi del arhitekture smo torej uporabili mrežo *VGG16*, za drugi del pa smo to obrnili in spremenili število filtrov določenih plasti.

Ker na posameznih slikovnih točkah izvajamo operacijo binarne klasifi-

kacije, na izhodu mreže pričakujemo binarno masko detektiranega področja uhlja. Tako kot v prejšnjem poglavju, smo na izhodu masko sploščili v enodimenzionalni vektor, na koncu pa dodali še plast aktivacije. Ta vrednosti vektorja s sigmoidno funkcijo omeji na vrednosti od 0 do 1.

Za razliko od prvotne arhitekture smo spremenili obliko vhodne plasti tako, da ta prejme sestavljeno matriko, ki vsebuje slike in pripadajoče porazdelitve delov obraza.

Primerjali smo tudi več izvedb arhitekture z manjšimi spremembami. Omenili smo, da je ena od lastnosti arhitekture *SegNet* ta, da si na vsaki od združitvenih plasti zapomni indekse elementov, ki jih znotraj okna prenese na naslednjo plast in posledično zmanjša ločljivost matrike značilk. V fazi dekodiranja značilk mreža vzame omenjene indekse in v plasteh višanja vzorčenja (angl. *upsampling*) z njimi velikost značilk obnovi. Posledično se plastem ni treba učiti, saj potrebne indekse za višanje vzorčenja že poznajo. Končno število parametrov se tako zmanjša v korist manjše velikosti in hitrosti modela.

Druga potencialna izboljšava temelji na raziskavi [5]. Avtorji predlagajo nadgradnjo osnovne arhitekture *SegNet* z uvedbo plasti opustitve. Natančnost modela naj bi se zaradi boljše generalizacije zvišala, poleg tega bi preprečili potencialno preveliko prileganje učni množici. Na izid v veliki večini vplivajo lastnosti učne množice.

3.3.3 Učenje mreže

V fazi učenja smo mreži na vhod podajali sestavljeno matriko velikosti $360 \times 480 \times 8$. Prvi trije kanali v matriki predstavljajo barvne kanale prvotne slike, drugi kanali pa vsebujejo generirane verjetnostne porazdelitve. Vrednosti posameznega kanala so normalizirane na območju od 0 do 1. Na izhodu mreže smo napovedane vrednosti primerjali z anotiranimi maskami uhljev. Razliko med dvema smo ponovno računali s pomočjo binarne križne entropije.

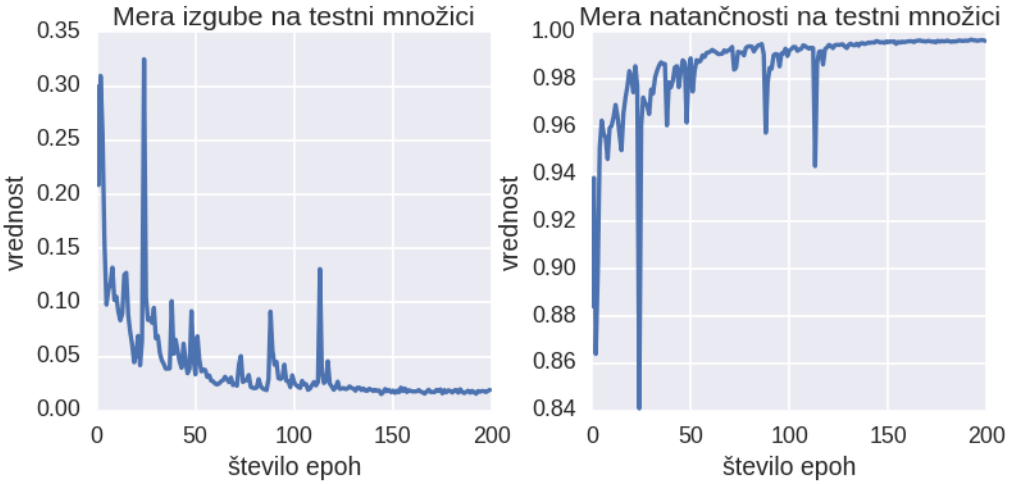
Ob računanju funkcije na izhodu mreže smo tokrat upoštevali tudi velikost večinskega razreda. Ker večina slikovnih točk v podatkovni zbirki

predstavlja ozadje, moramo funkcijo napake ustrezno utežiti. Ta mora biti višja, če eno od relevantnih slikovnih točk klasificiramo napačno. Ob generiranju svežnjev slik smo zato generirali tudi matriko uteži, ki ima velikost enako velikosti anotirane maske. Vsaki slikovni točki, ki ima v anotirani maski vrednost pozitivnega razreda, smo v matriki uteži dodelili višjo vrednost, ostalim pa nižjo.

Vrednost uteži W_c za razred c izračunamo tako, kot je predlagano v raziskavi [12] (enačba 3.3).

$$W_c = \frac{mediana\_frekvenc}{frekvenca(c)}, \tag{3.3}$$

kjer je $frekvenca(c)$ število vseh slikovnih točk razreda c v bazi, deljeno s številom vseh slikovnih točk v bazi; $mediana\_frekvenc$ pa predstavlja mediano vseh izračunanih frekvenc za razred c.

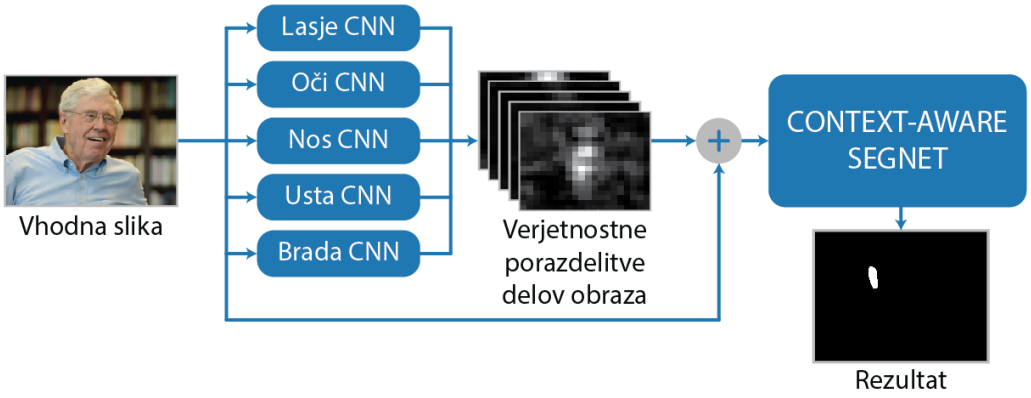


Slika 3.9: Prikaz konvergiranja funkcije izgube in natančnosti modela za eno od različic predlaganih arhitektur.

Za problem optimizacije smo primerjali delovanje algoritmov *Adam* [1] in *Adadelta* [26]. Zaradi velikosti arhitekture mreže, vhodnih podatkov in omejitev spominskega prostora na grafični kartici, je bila velikost svežnjev tokrat omejena na število 4. Potek učenja ene od mrež je prikazan na sliki 3.9.

3.3.4 Načrtovanje cevovoda

Za posamezne elemente svojega sistema smo dokazali, da delujejo v skladu s pričakovanji. Elemente lahko na tej točki povežemo v cevovod detekcije, ki bo kasneje uporabljen kot del večjega cevovoda prepoznave uhljev. Shema cevovoda je prikazana na sliki 3.10. Na vhodu cevovodu podamo sliko. Ta najprej potuje skozi pet konvolucijskih nevronskih mrež, ki izluščijo verjetnostne porazdelitve za lase, oči, nos, usta in brado. Sliko skupaj z verjetnostnimi porazdelitvami pošljemo še skozi novo naučen detektor, na izhodu pa dobimo napovedano masko področij uhljev. Arhitekturi mrež *SegNet* in *FacenessNet* sta prikazani v tabelah 3.3 oziroma 3.2.



Slika 3.10: Prikaz končnega cevovoda.

Številka plasti	Tip plasti	Število filtrov
-	vhodni podatki	
1	konvolucija	96
-	združevanje značilk	
2	konvolucija	256
-	združevanje značilk	
3, 4	konvolucija	512
5	konvolucija	1024
6, 7	konvolucija	512

Tabela 3.2: Prikaz arhitekture *FacenessNet* [23] po plasteh.

Številka plasti	Tip plasti	Število filtrov
-	vhodni podatki	
1, 2	konvolucija	64
-	združevanje značilk	
3, 4	konvolucija	128
-	združevanje značilk	
5, 6, 7	konvolucija	256
-	združevanje značilk	
8, 9, 10	konvolucija	512
-	združevanje značilk	
11, 12, 13	konvolucija	512
-	združevanje značilk	
-	širjenje značilk	
14, 15, 16	konvolucija	512
-	širjenje značilk	
17, 18	konvolucija	512
19	konvolucija	256
-	širjenje značilk	
20, 21	konvolucija	256
22	konvolucija	128
-	širjenje značilk	
23	konvolucija	128
24	konvolucija	64
-	širjenje značilk	
25	konvolucija	64
26	konvolucija	1
-	sigmoidna aktivacija	

Tabela 3.3: Prikaz najboljše različice arhitekture *SegNet* [6] po plasteh.

Poglavje 4

Rezultati

V tem poglavju so predstavljeni rezultati opisanih metod in njihova analiza. Naprej opišemo svoje testno okolje, podamo podatkovno zbirko, s katero smo model učili in vrednotili, potem pa določimo še metrike uspešnosti. Nato predstavimo in komentiramo rezultate uporabljenih metod.

4.1 Strojna oprema in uporabljena orodja

Za učenje in evalvacijo zgrajenih modelov smo uporabljali sistem z grafično kartico *Nvidia GeForce GTX 980 Ti* s 6 GiB pomnilnika, procesorsko enoto *Intel(R) Core(TM) i7-6700K* s taktom 4 GHz in 32 GiB sistemskega pomnilnika.

V veliki večini smo za delo z nevronskimi mrežami uporabljali programsko okolje *Keras* [4], ki na nižjem nivoju koristi funkcionalnosti knjižnice *Tensorflow* [27]. V določenih primerih smo uporabljali tudi ogrodje *Caffe* [7], saj so z njegovo pomočjo zgrajeni modeli za generiranje verjetnostnih porazdelitev delov obraza in model prvotne detekcije uhljev. V prvi fazi smo nekaj funkcij izvedli tudi v *Matlabu*, saj za *Python* implementacij metod *EdgeBoxes* in *Multiscale Combinatorial Grouping* še ni.

4.2 Podatkovna zbirka

Za vrednotenje in učenje svojega modela smo uporabili podatkovno zbirko *Annotated Web Ears – Whole (AWE-W)* [33]. Gre za različico osnovne zbirke *AWE*, ki vsebuje celotne slike in ne le slik uhljev. Ta zajema 100 oseb na 1000 slikah, ki so pridobljene z interneta in prikazujejo področja obrazov različnih oseb. Velikost vsake slike je spremenjena na 360×480 slikovnih točk, zato so nekatere lahko raztegnjene izven prvotnega razmerja stranic.

Vsaki sliki pripada anotirana maska enake velikosti, kjer slikovne točke z vrednostjo 1 predstavljajo področje uhlja, ostali pa predstavljajo ozadje. Shranjene so tudi koordinate oken, ki se uhljem najbolje prilegajo in so poravnane glede na osi. Zbirko smo razdelili na učno množico velikosti 750 in testno množico velikosti 250 slik.

4.3 Vrednotenje modela

Kot je bilo omenjeno v poglavju 2, smo za lažjo primerjavo z referenčnimi raziskavami prevzeli že obstoječe mere uspešnosti. Ker klasifikacijo izvajamo nad vsako slikovno točko na sliki, lahko rezultat svojega modela obravnavamo kot množice binarnih artributov.

Svojo napoved lahko razdelimo na slikovne točke s pozitivnim razredom P in na slikovne točke z negativnim razredom N. Slikovne točke, ki so klasificirane pravilno, prav tako spadajo v množico T, drugače pa v F. Tako dobimo štiri množice klasificiranih slikovnih točk TP, TN, FP in FN.

Ob primerjavi anotirane in napovedane množice lahko slikovne točke razdelimo tudi na relevantne ter izbrane. Relevantne slikovne točke R dejansko pripadajo pozitivno anotiranemu območju uhlja, izbrane slikovne točke S pa so posledica naših pozitivnih napovedi.

Kot glavno metriko modela uporabljamo Jaccardov koeficient podobnosti oziroma razmerje med presekom in unijo dveh množic (angl. *intersection over union – IoU*) (enačba 4.1). Ta nam poda razmerje med velikostjo množice slikovnih točk, ki so hkrati pozitivno anotirane in napovedane, ter

velikostjo unije vseh pozitivno anotiranih in napovedanih slikovnih točk. Za naš tip podatkov je najprimernejša, saj se posveča le razmerju med relevantnimi in izbranimi slikovnimi točkami. Tako naša glavna mera natančnosti ni občutljiva na velikost večinskega razreda negativnih slikovnih točk ozadja. Poleg tega smo sledili meram natančnosti, priklica in preciznosti.

$$IoU = J(R, S) = \frac{R \cap S}{R \cup S} = \frac{TP}{TP + FP + FN} \tag{4.1}$$

Natančnost (enačba 4.2) predstavlja odstotek vseh pravilno klasificiranih slikovnih točk na sliki. Zaradi neuravnotežene porazdelitve ciljnih razredov v naši podatkovni zbirki je ta mera pristranska in deluje v prednost večinskega razreda. Ta predstavlja 98,95 % vseh slikovnih točk v množici.

$$natančnost = \frac{TP + TN}{TP + FP + TN + FN} \tag{4.2}$$

Priklic (enačba 4.3) je mera kvantitete in nam pove, kolikšemu odstotku relevantnih slikovnih točk smo napovedali pozitiven razred. Ker lahko rezultate modela izboljšamo z dodatnimi koraki procesiranja, nam priklic predstavlja zgornjo mejo dodatnih izboljšav modela.

$$priklic = \frac{TP}{TP + FN} \tag{4.3}$$

Preciznost (enačba 4.4) je mera kvalitete in nam pove, kolikšen odstotek pozitivno napovedanih slikovnih točk je relevantnih. Ta je koristna pri ocenjevanju metode za odstranjevanje napačno pozitivnih slikovnih točk v zadnjih korakih cevovoda.

$$preciznost = \frac{TP}{TP + FP} \tag{4.4}$$

Napovedovanje koordinat vrednotimo z uporabo evklidske razdalje (enačba 4.5).

$$d(y, f(x)) = \sqrt{\sum_{i=0}^{n} (y_i - f(x_i))^2} \tag{4.5}$$

4.4 Lokalizacija s pomočjo detekcije obrazov

V prvi fazi smo uporabili določene metode, predlagane v raziskavi *FacenessNet* [23]. Implementirali smo filtiranje predlogov z mero *Faceness*, nato smo naučili klasifikator obraza.

Rešitev predstavlja izboljšavo naknadnega procesiranja prvotne raziskave, kjer sta po opravljeni detekciji ohranjeni le dve področji največjih kontur. Z uporabo detekcije obraza smo tokrat lahko prepričani, da ohranimo največje konture le v bližini uhljev. Rezultate prvotne detekcije smo izboljšali za 1 odstotno točko po meri IoU, točnost na celotni sliki pa smo izboljšali za 0,08 odstotne točke. Časovna zahtevnost metode je relativno visoka. V povprečju obraze detektiramo v dveh sekundah, kar predstavlja veliko oviro za potencialno praktično uporabo v realnem času. Zaradi velike razlike v evklidski razdalji pri napovedovanju sider na določenih slikah uhlji ne padejo v njihovo bližino, posledično pa odstranimo dobre napovedi. Prav tako bi lahko izboljšali metodo uteževanja, da bi ta upoštevala vsako slikovno točko neodvisno od velikosti celotne skupine slikovnih točk, ki ji pripada. Vseeno smo cilj v prvi fazi dosegli. Dokazali smo, da lahko že s preprostimi metodami izboljšamo rezultate segmentacije, če poznamo kontekst obraza.

Rezultati prvih dveh faz iz poglavij 3.1 in 3.2 so skupaj z rezultati prvotne detekcije prikazani v tabeli 4.1. Oznaka v1 predstavlja izboljšavo detekcije prve faze, oznaka v2 pa izboljšavo druge faze. Ker je bila prva faza namenjena bolj za uvod in potrditev domneve, da je kontekst obraza uporaben, se z analizo uteževanja v večji meri osredotočimo na metodo druge faze, s katero dosegamo boljše rezultate.

4.5 Lokalizacija z regresijo referenčnih točk

V drugi fazi smo naučili regresijsko mrežo in za uteževanje uporabili prileganje gostote verjetnosti.

Kot je prikazano v tabeli 4.1, smo prvotno detekcijo uhljev glede na mero IoU izboljšali za približno 5 odstotnih točk, glede na celotno sliko pa smo

	IoU	natančnost	priklic	preciznost
SegNet [32]	$48{,}31 \pm 23{,}01$	$99{,}21 \pm 0{,}58$	$\mathbf{75{,}86 \pm 33{,}11}$	$60{,}83 \pm 25{,}97$
SegNet-v1	$49{,}35 \pm 23{,}29$	$99{,}27 \pm 0{,}51$	$75{,}66 \pm 33{,}75$	$63{,}46 \pm 25{,}43$
SegNet-v2	$\mathbf{53{,}12 \pm 23{,}16}$	$\mathbf{99{,}39 \pm 0{,}49}$	$70{,}68 \pm 31{,}03$	$\mathbf{70{,}28 \pm 25{,}08}$

Tabela 4.1: Primerjava rezultatov med različnimi izvedbami uteževanja prvotne detekcije. Za referenco so prikazani tudi rezultati slednje.

natančnost izboljšali za 0,18 odstotne točke, kar na sliki velikosti 360×480 predstavlja približno 311 slikovnih točk. Prav tako smo zmanjšali časovno zahtevnost metode. V povprečju lokalizacija skupaj z modeliranjem gostote verjetnosti traja približno 20 ms. Čeprav se na prvi pogled modelirana gostota verjetnosti lepo prilega na področja uhlja, je izboljšava modela relativno majhna. Krivdo lahko dodelimo dvema glavnima faktorjema.

Prvi problem predstavlja relativno slab priklic prvotne detekcije. Ta nam postavlja zgornjo mejo dodatne izboljšave modela. Na sivinskih in razmeroma raztegnjenih slikah mreža običajno vrača prazno masko detekcij. Na nekaterih drugih slikah so deli uhlja sicer detektirani, ne povezujejo pa se v večje skupine, ki bi v celoti prekrile področja uhlja. V fazi uteževanja se zato lahko zgodi situacija, kjer gostote verjetnosti ne modeliramo v središče, ampak nekam na rob dejanskega uhlja. Veliko informacij je s tem izgubljenih.

Druga težava so slike z več obrazi. Čeprav se v prvotni raziskavi predpostavlja, da slike vsebujejo en obraz, je v uporabljeni podatkovni bazi tudi nekaj slik z več obrazi. Zaradi oblike izhoda regresijske mreže smo omejeni na napovedovanje dveh koordinat, kar ob prisotnosti več obrazov deluje nepredvidljivo. Včasih mreža napove lokacije enega od obrazov, največkrat pa kar nekje vmes med vsemi prisotnimi obrazi. Omenjeni težavi implementacija iz poglavja 3.3 odpravi.

4.6 Modifikacija arhitekture *SegNet*

Za končno evalvacijo cevovoda primerjamo štiri različice arhitekture. Zanima nas razlika med klasično različico arhitekture kodirnik-dekodirnik in arhitekture s prenašanjem indeksov ob večanju ločljivosti značilk, kot to predlaga raziskava *SegNet*. Funkcionalnost nam v podani situaciji morda ne koristi, zato smo preizkusili obe različici. Prav tako smo preizkusili nadgradnjo osnovne arhitekture *SegNet*, imenovane *Bayesian SegNet* [5]. S kombinacijo različnih nadgradenj smo torej dobili štiri modifikacije predlagane arhitekture.

Pred učenjem mreže smo izbrali še ustrezen optimizator. Kot je omenjeno v poglavju 3.3.2, smo za problem optimizacije med učenjem preizkusili algoritma *Adam* [1] in *Adadelta* [26]. Slednji za inicializacijo ne potrebuje vhodnih parametrov, zato smo začeli z njim. Zaradi adaptivne hitrosti učenja je funkcija napake konvergirala relativno hitro, nato je postala nestabilna. Na tej točki smo bili zadovoljni s potrditvijo, da model med učenjem pridobiva znanje. V prvem poskusu smo dosegli natančnost, ki se primerja z natančnostjo prvotne detekcije. Nad optimizacijo pa smo potrebovali več kontrole, zato smo v nadaljevanju uporabljali algoritem *Adam*.

V povprečju so različice načrtovanih mrež konvergirale po približno 5 urah oziroma 300 epohah. Med učenjem je mreža prejemala svežnje podatkov velikosti 4, tako da je v eni epohi opravila približno 190 posodobitev gradientov.

V fazi testiranja smo opazili izrazito izboljšanje vseh metrik uspešnosti. Za najboljšo različico se je izkazala arhitektura *SegNet* brez prenosa indeksov. Mero *IoU* smo v primerjavi s prvotno detekcijo izboljšali za 28,54 odstotnih točk, natančnost na celotni sliki pa za 0,53 odstotne točke. Rezultati so prikazani v tabeli 4.2. Kratica *CA* v tabeli pomeni *Context-aware* in označuje novo arhitekturo z dodanimi kanali za verjetnostne porazdelitve. Oznaka I predstavlja dodano funkcionalnost prenašanja indeksov, oznaka B pa dodane plasti opustitve (*Bayesian SegNet*). Očitno so nam ob učenju mreže dodatni parametri modela brez prenašanja indeksov koristili. Z zmanjšanjem para-

	IoU	natančnost	priklic	preciznost
SegNet [32]	48,31 ± 23,01	99,21 ± 0,58	75,86 ± 33,11	60,83 ± 25,97
SegNet-v2	53,12 ± 23,16	99,39 ± 0,49	70,68 ± 31,03	70,28 ± 25,08
CA-SegNet	**76,85 ± 20,10**	**99,74 ± 0,35**	**86,40 ± 19,96**	87,15 ± 15,94
CA-SegNet-I	74,77 ± 20,00	99,71 ± 0,36	84,22 ± 21,02	87,25 ± 14,65
CA-SegNet-B	75,02 ± 19,75	99,71 ± 0,40	83,22 ± 20,92	**88,62 ± 13,11**
CA-Segnet-IB	71,05 ± 18,77	99,64 ± 0,41	84,78 ± 21,07	81,54 ± 16,84

Tabela 4.2: Primerjava rezultatov med različnimi izvedbami arhitekture detektorja. Za referenco so prikazani tudi rezultati prvotne detekcije in uteževanja iz druge faze.

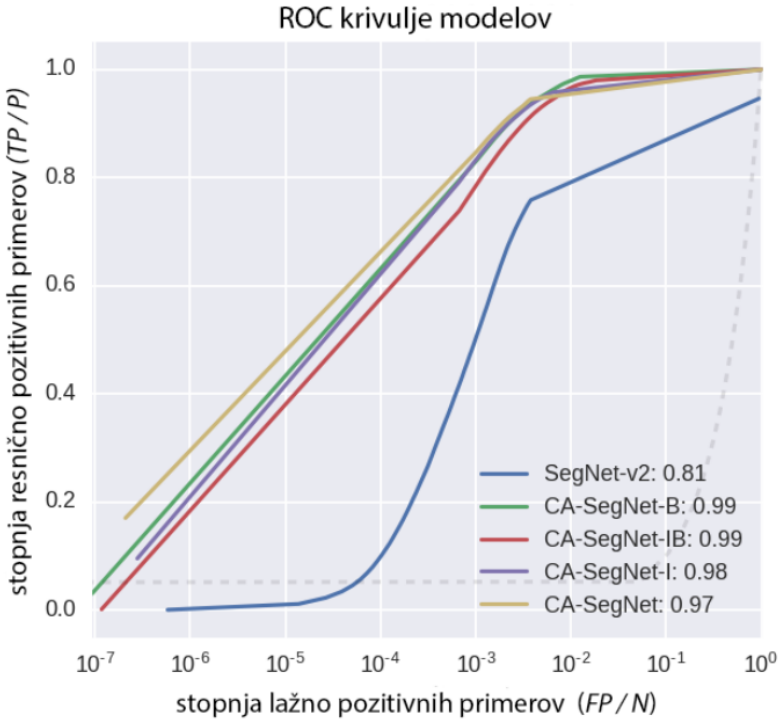
metrov modela se namreč pojavi kompromis, kjer zaradi manjše velikosti modela izgubimo na natančnosti napovedi.

Z dodatkom plasti opustitve mreža ni pridobila dodatnega znanja. Kot je omenjeno v poglavju 4.2, naša učna množica vsebuje slike, ki so raztegnjene izven prvotnega razmerja stranic. Prav tako uhlji na slikah zavzemajo različne poze in velikosti glede na celotno sliko. Učna množica je torej raznolika. V nasprotnem primeru bi verjetno prišlo do prevelikega prileganja učni množici. V tem primeru bi nam plasti opustitve koristile, saj bi mrežo prisilili, da se uči le generalnih značilk, s tem pa preveliko prileganje preprečili.

Ob učenju mreže prevelikega prileganja učni množici torej nismo opazili. Včasih se je zgodilo, da je funkcija izgube močno poskočila, model pa se je prenehal učiti. To smo lahko opazili ob večji vrednosti parametra *learning_rate*, ki predstavlja hitrost učenja mreže. Mreža se je namreč začela učiti v korist večinskega razreda, posledično pa je na večini slikovnih točk določila negativni razred. Točnost modela je bila zaradi manjšinske reprezentacije ciljnega razreda še vedno 99 %, vrednost napake pa je močno poskočila tako na učni kot na testni množici. Problem smo rešili tako, da smo zmanjšali hitrost učenja.

Ker smo morali rezultat mreže zaradi zveznih vrednosti sigmoidne akti-

vacijske funkcije upragovati, analiziramo krivuljo *ROC* (angl. *receiver operating characteristic*) zgrajenih modelov, ki je prikazana na sliki 4.1. Ta nam pove, kolikšna je diskriminacija testnih primerov ob izbiri določenega pragu. Površino pod krivuljo *ROC* označuje mera *AUROC* (angl. *area under receiver operating characteristic*). V idealni situaciji je vrednost *AUROC* enaka 1, kar pomeni, da ob določenem pragu vse testne primere klasificiramo pravilno. Mera *AUROC* je za vse modele zelo visoka, kar pomeni, da prag klasifikacije določimo z minimalno diskriminacijo klasificiranih slikovnih točk. Model *CA-SegNet* dosega najslabšo mero *AUROC*, saj v primerjavi z drugimi slikovne točke na robovih uhlja klasificira z večjo negotovostjo. Sklepamo lahko, da je na izbiro pragu bolj občutljiv, kljub temu pa ob pravi izbiri dosega boljšo klasifikacijsko točnost.



Slika 4.1: Primerjava krivulj ROC za upragovanje rezultatov zgrajenih modelov. Zaradi preglednosti je skala X osi logaritemska. Za vsak model v legendi je podana tudi pripadajoča mera AUROC.

Na podanem sistemu ocenimo še čas procesiranja cevovoda. Prvi del cevovoda je sestavljen iz petih dokaj enostavnih konvolucijskih mrež, ki izluščijo verjetnostne porazdelitve delov obraza. Vsaka mreža za generiranje porazdelitev v povprečju potrebuje 4,7 ms, kar predstavlja približno 23,5 ms za vse dele obraza. Na sliki in porazdelitvah izvedemo še detekcijo uhljev, ki v povprečju traja 69 ms. Skupen čas obdelave ene slike je na koncu v povprečju približno 92,5 ms. Prvotno detekcijo smo torej izrazito izboljšali na račun 5,7 % povečanega časa procesiranja, ki je v prvotni raziskavi znašal 87,5 ms.

Slika 4.2: Pregled rezultatov z različnimi merami IoU.

Za zaključek analizirajmo rezultate še kvalitativno. Na sliki 4.2 predstavimo šest primerov, ki so glede na mero IoU enakomerno razporejeni čez celotno definicijsko območje te metrike. Glede na mero IoU najboljšo detekcijo dosežemo na slikah, na katerih obraz predstavlja večji del slike, poleg uhljev pa so vidni vsi ostali deli obraza. Bolj ko je obraz na sliki oddaljen, slabše je prileganje napovedane maske kljub izpolnjenim ostalim pogojem. Kot je vidno na sliki zgoraj desno, uhljev v ekstremni pozi frontalnega pogleda pogosto ne zaznamo. Še vedno napovemo nekaj napačno pozitivnih slikovnih točk. Prvi primer je predstavljen na sliki spodaj levo, kjer detek-

tiramo logotip, ki vsebuje vzorec podoben področju uhlja. Včasih zaznamo tudi dele rok, ki prav tako vsebujejo omenjene vzorce. Najslabše rezultate dosegamo v primeru, kadar so obrazi na slikah v skrajni pozi pogleda s profila, ali če je na sliki več obrazov.

Poglavje 5

Zaključek

Naša raziskava je pokazala, da lahko z uporabo konteksta obraza pridobimo dodatne informacije o sliki, posledično pa izboljšamo natančnost zgrajenega detektorja uhljev. Povprečna ocena IoU, ki smo jo dosegli na testni množici, ob izbiri najboljše različice arhitekture, znaša 76,85 %, povprečna natančnost na celotni sliki 99,74 %, meri priklica in preciznosti pa 86,40 % oziroma 87,15 %. Glede na prvotno metodo detekcije z arhitekturo *SegNet* smo mero IoU izboljšali za 28,5 odstotnih točk, natančnost na celotni sliki pa za 0,53 odstotne točke.

Na področju detekcije uhljev je v prihodnosti še dovolj prostora za izboljšavo rezultatov. Ker nevronske mreže z večjim naborom razpoložljivih podatkov običajno dosegajo boljše rezultate, bi z večjo učno množico dodatno izboljšali natančnost modela. Prav tako bi podatkovni zbirki koristila boljša reprezentacija uhljev s skrajnimi pozami in različnimi barvami kože. Anotiranje za semantično segmentacijo je sicer draga operacija, daje pa rezultate, iz katerih lahko razberemo tudi obliko detektiranega objekta.

Drugi način izboljšave temelji na združevanju detekcij posameznih atributov želenih objektov. V tem primeru bi lahko uhlje anotirali po posameznih pozah. Uhlju s frontalnega pogleda bi dodelili drugačen ciljni razred kot recimo uhlju s pogleda profila. Vse razrede, ki predstavljajo uhelj, bi na izhodu mreže združili in potencialno izboljšali natančnost detekcije.

Uporaba konteksta pa teoretično ni omejena le na problem detekcije uhljev, ampak predstavlja potencialno izboljšavo številnih področij detekcije. Povsod, kjer detektiramo del večjega sistema, kateremu lahko v prostorski ali projecirani predstavi določimo določene relacije med posameznimi deli, lahko za izboljšavo modela uporabimo dodatne informacije, ki jih pridobimo s konteksta okolja.

Literatura

[1] D. P. Kingma, J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[2] M. Koestinger, P. Wohlhart, P. M. Roth, H. Bischof. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. *International Conference on Computer Vision*, pages 2144–2151, 2011.

[3] A. Pflug, C. Busch. Ear biometrics: a survey of detection, feature extraction and recognition methods. *IET Biometrics*, 1(2):114–129, 2012.

[4] F. Chollet. Keras, 2015. Dostopno na: https://github.com/fchollet/keras (dostopano 13. 9. 2017).

[5] A. Kendall, V. Badrinarayanan, R. Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *CoRR*, abs/1511.02680, 2015.

[6] V. Badrinarayanan, A. Kendall, R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, abs/1511.00561, 2015.

[7] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *International Conference on Multimedia*, pages 675–678, 2014. Dostopno na: http://caffe.berkeleyvision.org/ (dostopano 13. 9. 2017).

[8] C. L. Zitnick, P. Dollár. Edge Boxes: Locating object proposals from edges. *European Conference on Computer Vision*, pages 391–405, 2014.

[9] A. S. Anwar, K. Kamal, A. Ghany, H. Elmahdy. Human ear recognition using geometrical features extraction. *Procedia Computer Science*, 65:529–537, 2015.

[10] N. K. A. Wahab, E. E. Hemayed, M. B. Fayek. Heard: An automatic human ear detection technique. *International Conference on Engineering and Technology*, pages 1–7, 2012.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei. ImageNet: A large-scale hierarchical image database. *Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[12] D. Eigen, R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *CoRR*, abs/1411.4734, 2014.

[13] S. Ansari, P. Gupta. Localization of ear using outer helix curve of the ear. *International Conference on Computing: Theory and Applications*, pages 688–692, 2007.

[14] S. Prakash, U. Jayaraman, P. Gupta. Ear localization from side face images using distance transform and template matching. *First Workshops on Image Processing Theory, Tools and Applications*, pages 1–8, 2008.

[15] A. Krizhevsky, I. Sutskever, G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems 25*, pages 1097–1105, 2012.

[16] Y. LeCun, Y. Bengio, G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[17] P. Viola, M. Jones. Rapid object detection using a boosted cascade of simple features. *Computer Society Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[18] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, K. Keutzer. Squeezenet: Alexnet-level accuracy with 50× fewer parameters and 1 MiB model size. *CoRR*, abs/1602.07360, 2016.

[19] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marqués, J. Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. *CoRR*, abs/1503.00848, 2015.

[20] A. Abaza, A. Ross, C. Hebert, M. A. F. Harrison, M. S. Nixon. A survey on ear biometrics. *ACM Computing Surveys*, 45(2):1–35, 2013.

[21] B. Arbab-Zavar, M. S. Nixon. On shape-mediated enrolment in ear biometrics. *International Symposium on Visual Computing*, pages 549–558, 2007.

[22] S. Attarchi, K. Faez, A. Rafiei. A new segmentation approach for ear recognition. *Advanced Concepts for Intelligent Vision Systems*, pages 1030–1037, 2008.

[23] S. Yang, P. Luo, C. C. Loy, X. Tang. From facial parts responses to face detection: A deep learning approach. volume abs/1509.06451, 2015.

[24] J. R. R Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[25] W. Ge, Y. Yu. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. *CoRR*, abs/1702.08690, 2017.

[26] M. D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[27] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Dostopno na: https://www.tensorflow.org/ (dostopano 13. 9. 2017).

[28] K. Simonyan, A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[29] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. Dostopno na: http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html (dostopano 13. 9. 2017).

[30] P. Dollár, C. L. Zitnick. Structured forests for fast edge detection. *International Conference on Computer Vision*, pages 1841–1848, 2013.

[31] P. Dollár, C. L. Zitnick. Fast edge detection using structured forests. *CoRR*, abs/1406.5549, 2014.

[32] Ž. Emersic, L. L. Gabriel, V. Štruc, P. Peer. Pixel-wise ear detection with convolutional encoder-decoder networks. *CoRR*, abs/1702.00307, 2017.

[33] Ž. Emersic, V. Štruc, P. Peer. Ear Recognition: More Than a Survey. *Neurocomputing*, 255:26–39, 2017.