Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Miha Bizjak

Generativni globoki modeli slik uhljev

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Peter Peer
Asistent: Žiga Emeršič, mag.

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Generativni globoki modeli slik uhljev

Tematika naloge:

Pri uporabi globokih nevronskih mrež velikokrat naletimo na problem, da imamo premalo podatkov za učenje le-teh. Eno izmed takšnih področij je tudi biometrična razpoznava na osnovi slik uhljev. Preučite možnosti generiranja slik uhljev z generativnimi mrežami za ta namen. Dobljene rezultate ovrednotite tako kvalitativno kot kvantitativno.

Zahvaljujem se mentorju izr. prof. dr. Petru Peeru in asistentu Žigu Emeršiču za pomoč pri izdelavi diplomske naloge ter družini in prijateljem za vso podporo v času študija.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
AWE	Annotated Web Ears	anotirani uhlji iz spleta
ReLU	rectified linear unit	pragovna linearna funkcija
CNN	convolutional neural network	konvolucijska nevronska mreža
GNN	generative neural network	generativna nevronska mreža
GAN	generative adversarial network	generativna nasprotniška mreža
VAE	variational autoencoder	variacijski avtokodirnik
CVAE	conditional variational autoencoder	pogojni variacijski avtokodirnik
CPU	central processing unit	centralna procesna enota
GPU	graphical processing unit	grafična procesna enota
CMC	cumulative match curve	krivulja kumulativnih zadetkov
AUCMC	area under CMC	območje pod CMC krivuljo
ROC	receiver operating characteristics	karakteristika delovanja sprejemnika
TAR	true acceptance rate	delež pravilnih pozitivnih zaznav
FAR	false acceptance rate	delež napačnih pozitivnih zaznav
AUC	area under curve	območje pod krivuljo

Povzetek

Naslov: Generativni globoki modeli slik uhljev

Avtor: Miha Bizjak

Povzetek: Za dobro delovanje potrebujejo globoke nevronske mreže veliko podatkov. V primeru biometrične modalnosti uhljev - največje anotirane baze slik uhljev v nekontroliranem okolju zajemajo nekaj tisoč slik, kar je premalo za globoko učenje in razpoznavo. Ta problem skušamo rešiti z uporabo generativnih nevronskih mrež za obogatitev baze. Implementiramo dva tipa generativnih nevronskih mrež: generativno mrežo in variacijski avtokodirnik. Obe mreži naučimo s pomočjo slik iz obstoječe baze in z vsako generiramo množico umetnih podatkov (slike uhljev). Z vsako od teh množic nato učimo mreže za razpoznavo in primerjamo rezultate. Kljub uporabi umetno generiranih slik, ne uspemo doseči visoke stopnje razpoznave na bazi AWE-v1, vseeno pa so opazne izboljšave v primerjavi z rezultati učenja razpoznave brez umetno generiranih slik.

Ključne besede: nevronske mreže, globoko učenje, obogatitev podatkov, biometrija.

Abstract

Title: Generative deep models for ear images

Author: Miha Bizjak

Abstract: Deep neural networks require large amounts of data to perform well. In the case of the biometrical modality of the human ear, the largest annotated databases of images of ears in an uncontrolled environment consist of a few thousand images, which is insufficient for recognition using deep learning. We try to solve this problem using generative neural networks for data augmentation. We implement two types of generative neural networks: a generative network and a variational autoencoder. We train both networks on images from the existing database and then use them to generate a new set of artificial data (images of ears) with each. We then use each of these datasets to train neural networks for recognition and compare the results. Even using artificially generated images, we do not manage to achieve a high recognition rate on the AWE-v1 ear database. Despite that, there is a noticeable improvement compared to results of training for recognition without using generated data.

Keywords: neural networks, deep learning, data augmentation, biometrics.

Poglavje 1

Uvod

Na področju razpoznavanja uhljev je bilo v zadnjem času narejenega kar nekaj raziskovalnega dela - tudi na naši fakulteti [1, 2, 3]. Kljub temu pa baze anotiranih slik, ki so trenutno na voljo, vsebujejo premalo slik za efektivno uporabo globokega učenja za razpoznavo. Do neke mere se lahko temu izognemo z obogatitvijo baze s pomočjo uporabe transformacij, mi pa se bomo tega lotili z uporabo generativnih nevronskih mrež. Na podlagi slik iz baze AWE-v1 [4], ki vsebuje 1.000 anotiranih slik uhljev 100 različnih oseb, bomo naučili različne arhitekture mrež za generiranje slik in nato s temi mrežami generirali večjo količino umetnih podatkov. Da ugotovimo, koliko lahko ti podatki pripomorejo pri razpoznavi, bomo z njimi poskusili naučiti globoko nevronsko mrežo za razpoznavo in primerjali natančnosti. Analizirali bomo tudi vektorje značilk, ki jih dobimo s pomočjo globoke nevronske mreže in jih uporabili kot testno množico. Ti novi podatki in ugotovitve bodo lahko koristili pri nadaljnjem delu glede razpoznavanja uhljev z uporabo metod globokega učenja.

S pričetkom v poglavju 2 na kratko predstavimo teoretična ozadja naše naloge: dosedanje delo na področju razpoznavanja uhljev, osnove delovanja nevronskih mrež in različne vrste arhitektur nevronskih mrež, ki smo jih uporabljali. V poglavju 3 predstavimo orodja in knjižnice, ki smo jih uporabljali za pripravo slik za učenje generativnih mrež in za pripravo oziroma uporabo

generativnih in konvolucijskih mrež. Poglavje 4 vsebuje opis uporabljenih metod za pripravo slik na učenje generativnih in konvolucijskih mrež. V poglavju 5 predstavimo rezultate kvalitativne in kvantitativne analize, temu sledi še poglavje 6, v katerem podamo sklepe in smernice za nadaljnje delo.

Poglavje 2

Teoretična ozadja

V tem poglavju najprej naredimo pregled dosedanjega dela na področju razpoznavanja uhljev. Sledi kratek uvod v nevronske mreže. Predstavimo njihovo zgradbo in delovanje ter si podrobneje ogledamo nekaj pogostejših aktivacijskih funkcij. Za tem predstavimo še vrste arhitektur nevronskih mrež, ki so relevantne za našo nalogo.

2.1 Razpoznavanje uhljev

Uhlji, kot biometrična modalnost, imajo zaradi edinstvenosti njihove oblike in strukture dober potencial za uporabo v biometriji [5]. Uhelj ima nekaj prednosti v primerjavi z ostalimi modalnostmi: v primerjavi z obrazom se njegova struktura s staranjem spreminja počasneje, pred prstnimi odtisi pa imajo to prednost, da je slike uhljev praviloma lažje pridobivati, saj za to ne potrebujemo sodelovanja uporabnika.

Kljub temu pa razpoznavanje uhljev še ni tako raziskano kot razpoznavanje obrazov in prstnih odtisov. Med glavnimi problemi sta pomanjkanje efektivnih metod za detekcijo uhljev ter premajhna količina anotiranih slik za efektivno uporabo globokega učenja za razpoznavo [6].

Članek *Ear Recognition: More Than a Survey* [6] vsebuje pregled področja avtomatskega zaznavanja uhljev iz 2D slik. Avtorji v članku opišejo

metode, ki se trenutno uporabljajo. Opišejo tudi obstoječe podatkovne zbirke namenjene učenju razpoznavanja uhljev in predstavijo novo podatkovno zbirko *Annotated Web Ears* [4] ter zbirko metod za delo na razpoznavanju uhljev.

V članku [2] se analizira obstoječe metode za razpoznavo uhljev skozi identifikacijske eksperimente na AWE bazi. Rezultati kažejo, da so metode globokega učenja z manjšimi arhitekturami po hitrosti razpoznave primerljive s klasičnimi metodami na podlagi deskriptorjev, a za učenje potrebujejo precej več časa in zmogljivo strojno opremo. Izkaže se, da na uspešnost razpoznave najbolj vplivata prisotnost dodatkov, npr. uhanov in slike zajete pod velikimi koti. V okviru članka predstavijo tudi novo razširjeno verzijo baze AWE, imenovano AWEx (angl. AWE Extended). Ta baza vsebuje 4.104 slik 346 različnih oseb pridobljenih z interneta.

S problemom zaznavanja uhlja na sliki so se ukvarjali avtorji članka *Pixel-wise Ear Detection with Convolutional Encoder-Decoder Networks* [7]. Uporabili so konvolucijsko nevronsko mrežo, osnovano na arhitekturi *SegNet* [8]. Za razliko od obstoječih rešitev - te označijo samo pravokotno regijo, ki vsebuje uhelj - metoda deluje na nivoju pikslov, kar omogoča odstranjevanje ozadja pred uporabo slik za razpoznavo.

2.2 Nevronske mreže

Nevronske mreže so veja strojnega učenja, ki se zgleduje po delovanju možganov. Omogočajo ustvarjanje sistemov, ki se naučijo opravljati določeno nalogo, na primer prepoznavanje objektov na sliki, zgolj na podlagi primerov.

Osnovni gradniki nevronskih mrež so nevroni, ki si prek povezav med seboj pošiljajo signale. Nevron prek povezav prejme več vhodnih vrednosti, ki se pomnožijo z utežmi njihovih povezav in se seštejejo. Tej vsoti se doda še pristranskost (angl. bias) nevrona, končna vrednost pa gre v aktivacijsko funkcijo, ki določi izhodno vrednost nevrona. Obstaja več različnih aktivacijskih funkcij, nekaj pogostejših je opisanih v nadaljevanju.

Nevroni so tipično povezani v več zaporednih plasti. Podatke o problemu se dobi na vhodni plasti, sledi ena ali več skritih plasti, prek katerih se posredujejo signali, na koncu pa je izhodna plast, v kateri se nahaja rešitev problema. Take mreže imenujemo mreže z razširjanjem naprej (angl. feed-forward), saj se informacije širijo samo v eno smer, obstajajo pa tudi mreže s cikličnimi povezavami, a jih v tej diplomski nalogi nismo uporabljali. Primer preproste mreže z razširjanjem naprej in eno skrito plastjo je prikazan na sliki 2.1.

V procesu učenja nevronske mreže se uporablja postopek vzvratnega razširjanja napake (angl. backpropagation), kjer z gradientnim spustom ali drugim optimizacijskim algoritmom, kot je na primer algoritem Adam [9], poskušamo z iterativnim posodabljanjem uteži minimizirati funkcijo napake in s tem izboljšati točnost napovedi, ki jih daje nevronska mreža.

Pri problemih klasifikacije se za funkcijo izgube pogosto uporablja kategorična križna entropija (angl. categorical crossentropy) [10]. Funkcija, ki jo želimo minimizirati je zapisana v enačbi

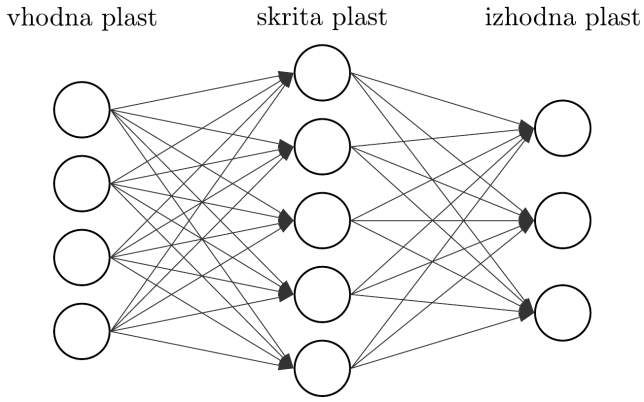
$$H(p, q) = \sum_x p(x) \log(q(x)), \tag{2.1}$$

kjer je $p(x)$ ciljna distribucija razredov, $q(x)$ pa distribucija razredov, ki jo predvidi izhodna plast mreže.

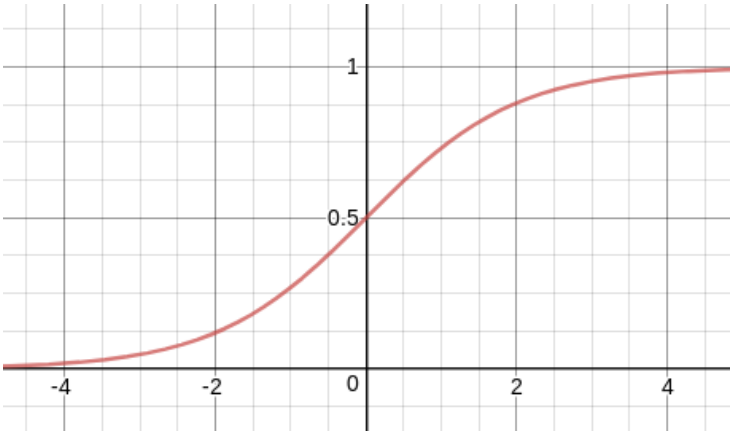
Sigmoidna funkcija kot aktivacijska funkcija izhod nevrona normalizira v interval (0, 1). Zapišemo jo z enačbo

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{2.2}$$

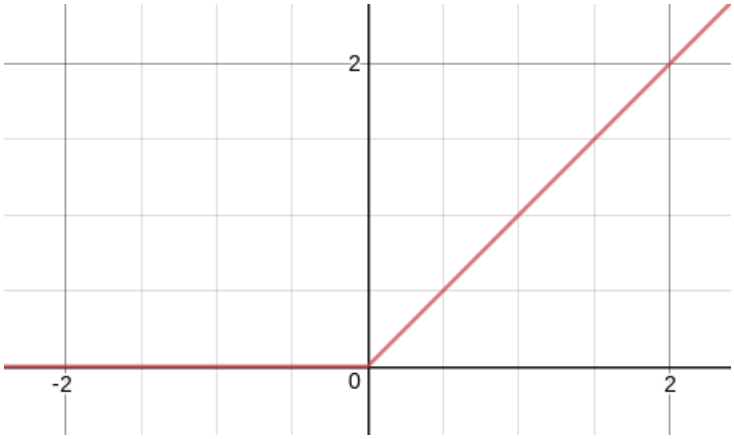
grafično pa je prikazana na sliki 2.2. Največji gradient ima funkcija pri vhodnih vrednostih blizu 0, kar pomeni, da se v tem območju najhitreje odziva na spremembe vhodne vrednosti. Pri vhodnih vrednostih $x \gg 0$ ali $x \ll 0$ pa je gradient skoraj 0, kar zelo upočasni učenje. Ta pojav se imenuje problem izginjajočih gradientov (angl. vanishing gradient problem) in postane še bolj očiten pri globljih nevronskih mrežah, saj se pri vzvratnem razširjanju gradienti z vsakim dodatnim slojem zmanjšujejo.



Slika 2.1: Primer preproste nevronske mreže.



Slika 2.2: Sigmoidna aktivacijska funkcija.



Slika 2.3: ReLU aktivacijska funkcija.

ReLU funkcija, definirana z enačbo

$$f(x) = \max(0, x) \tag{2.3}$$

in prikazana na sliki 2.3, problema izginjajočih gradientov nima, saj je gradient pri pozitivnih vhodnih vrednostih konstanten. To omogoča hitrejše učenje, poleg tega pa je funkcija tudi računsko manj zahtevna. Ima pa svojo slabost, in sicer možnost nastanka t. i. umirajočih nevronov – nevronov, kjer se uteži posodobijo na tako vrednost, da se nevron pri nobenem vhodu ne aktivira več in je vrednost gradienta konstantno 0.

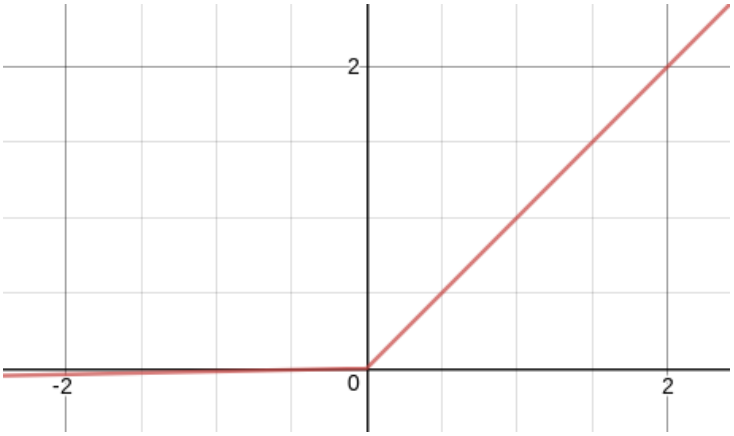
Eden od načinov, da se umirajočim nevronom izognemo, je uporaba puščajoče ReLU (angl. Leaky ReLU) funkcije, zapisane z enačbo

$$f(x) = \max(\alpha x, x). \tag{2.4}$$

To je variacija ReLU funkcije, ki ima pri negativnih x namesto 0 vrednost αx, kjer je α neka majhna konstanta - v članku [11] za α uporabljajo 0,01. Grafično je funkcija prikazana na sliki 2.4.

2.2.1 Optimizacijski algoritem Adam

Adam [9] za izračun uteži uporablja drseči povprečji prvega in drugega momenta gradientov in hitrost učenja sproti prilagaja za vsak parameter v ne-



Slika 2.4: LeakyReLU aktivacijska funkcija.

vronski mreži. Algoritem je preprost za implementacijo, za izvedbo pa potrebuje malo računske moči in spomina. Dobro se obnese pri problemih, kjer delamo z večjim številom parametrov in tipično ne zahteva veliko nastavljanja hiperparametrov.

2.3 Konvolucijske nevronske mreže

Za klasifikacijo slik in detekcijo objektov na slikah se v zadnjih letih večinoma uporabljajo konvolucijske nevronske mreže (angl. convolutional neural networks, CNN) [12, 13]. So vrsta globokih nevronskih mrež, glede na način uporabe konvolucije (angl. convolution). Glavni del konvolucijskih mrež so konvolucijske plasti, ki iz slike izluščijo pomembne značilke. Poleg teh pa tipično vsebujejo še zbirne (angl. pooling) plasti in polno povezane (angl. fully-connected) plasti.

2.4 Generativne nevronske mreže

Arhitektura generativne nevronske mreže (GNN) je na nek način inverzna arhitekturi mreže za razpoznavo, ki je konvolucijska. Kot vhod sprejme parametre slike, ki jo želimo generirati, kot izhod pa dobimo generirano sliko.

V konvolucijskih mrežah (za npr. razpoznavo) je tipično zaporedje nekaj konvolucijskih plasti, ki jim sledi zbirna plast. Pri generativnih mrežah pa te operacije izvajamo ravno v obratnem vrstnem redu: najprej uporabimo višanje vzorčenja in šele potem konvolucijo.

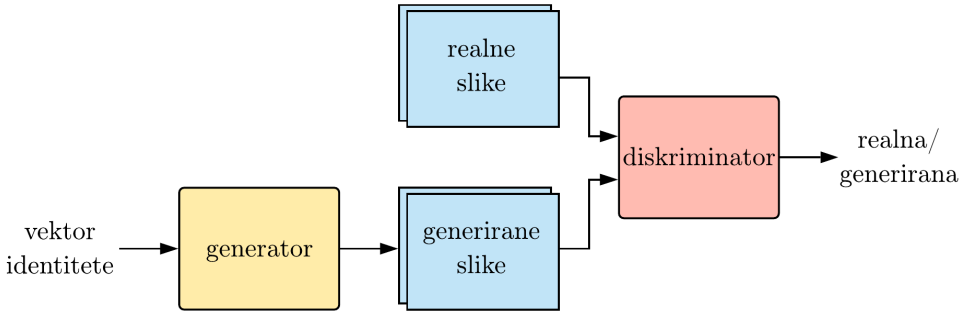
Primer take nevronske mreže je opisan v članku *Learning to Generate Chairs With Convolutional Networks* [14], kjer so avtorji sestavili nevronsko mrežo, ki se na slikah stolov nauči generirati nove slike z določenim stilom, kotom pogleda in barvo.

Na podlagi tega članka obstaja tudi implementacija podobne mreže, ki pa je namenjena generiranju obrazov z *RaFD* bazo obrazov [15]. Ta vsebuje slike 67-ih oseb, ki prikazujejo 8 različnih emocij. Mreža se na slikah iz te baze nauči interpolirati med različnimi identitetami in emocijami, med orientacijami pa ne najbolje. Originalne slike so namreč na voljo le iz petih različnih kotov, za razliko od prej omenjenega članka, v katerem so imeli avtorji na voljo 3D modele objektov in so lahko tako ustvarili slike za učenje iz poljubnih kotov.

2.5 Generativne nasprotniške mreže

Ideja generativnih nasprotniških mrež (GAN, angl. generative adversarial network) [16] je v tem, da imamo dve nevronski mreži – generator in diskriminator, ki tekmujeta med sabo. Generator skuša na podlagi obstoječih podatkov, običajno slik, generirati nove primerke. Diskriminator pa je namenjen temu, da razlikuje med realnimi in generiranimi primerki, ki jih prejema na vhod.

Učenje generatorja in diskriminatorja poteka izmenično. Diskriminator učimo posebej, s slikami, ki so pravilno označene kot realne oz. generirane. Za učenje generatorja pa oba modela združimo. Generator generira slike in z njimi poskuša čim bolj uspešno pretentati diskriminator, tako da jih označi kot realne. Pri tem je zelo pomembno, da je učenje generatorja in diskriminatorja uravnoteženo. Če diskriminator postane preveč natančen,



Slika 2.5: Učenje generatorja in diskriminatorja.

generator ne dobi nazaj dovolj povratnih informacij za nadaljnje učenje. Če pa je diskriminator premalo natančen in tudi slabo generirane slike označi kot realne, se generator ne bo izboljšal. Diagram učenja je prikazan na sliki 2.5.

2.6 Variacijski avtokodirniki

Variacijski avtokodirniki (angl. variational autoencoders, VAE) [17] so generativni modeli, ki se namesto generiranja naključnih novih podatkov, naučijo generiranja variacij obstoječih podatkov. Standardni avtokodirniki so sestavljeni iz dveh mrež, kodirnika in dekodirnika. Naloga kodirnika je, da ustvari manjšo vektorsko reprezentacijo vhodnih podatkov, naloga dekodirnika pa, da iz te reprezentacije čim bolje rekonstruira vhodne podatke. Obe mreži učimo skupaj in skušamo zmanjšati rekonstrukcijsko izgubo (angl. reconstruction loss) – razliko med vhodom in izhodom. Problem standardnih avtokodirnikov pa je v tem, da latentni prostor (angl. latent space) vektorjev, ki jih generira kodirnik, ni nujno zvezen oziroma njihova distribucija ni ugodna za interpolacijo [18].

Variacijski avtokodirniki so zgrajeni tako, da bo njihov latentni prostor zmeraj zvezen. To dosežejo tako, da kodirnik namesto enega vektorja velikosti n generira dva vektorja velikosti n: vektor povprečij μ in vektor standardnih deviacij σ. Na podlagi teh dveh se nato ustvari nov vektor naključnih

vrednosti velikosti n, ki ga prejme dekodirnik. S tem dobimo več različnih variacij istega vhoda, še zmeraj pa ni nujno, da se bodo vektorji različnih primerkov vhodov prekrivali.

Ta problem VAE-ji rešijo z uporabo Kullback-Leibler (KL) divergence v funkciji izgube. KL divergenca izračuna, koliko se dve verjetnostni distribuciji razlikujeta med sabo. KL funkcija izgube, zapisana z enačbo

$$\sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 \log(\sigma_i) - 1, \tag{2.5}$$

je v VAE-jih vsota vseh KL divergenc distribucij, ki jih generira kodirnik. To prepreči, da bi se posamezne skupine vektorjev preveč oddaljile od ostalih. Ker pa bi zgolj uporaba KL funkcije izgube ustvarila povsem naključno distribucijo, skušamo pri učenju VAE-ja hkrati minimizirati obe funkciji izgube, rekonstrukcijsko in KL izgubo. Rezultat tega je, da dobimo zvezen latentni prostor, kljub temu pa so vektorji razporejeni glede na podobnost vhodnih podatkov, kar omogoča interpolacijo med njimi.

Problem klasičnega VAE-ja pa je, da nimamo nadzora nad tem, kakšne podatke generiramo, npr. ne moremo določiti, da hočemo kombinacijo dveh določenih slik.

Ta problem reši pogojni variacijski avtokodirnik (angl. conditional variational autoencoder, CVAE) [19]. CVAE je razširitev VAE, kjer mreži na vhodni plasti ob učenju poleg vsakega podatka (slike) damo še en dodaten parameter, npr. oznako slike. Na naučenem modelu CVAE lahko nato s tem parametrom določamo, kakšne podatke (slike) bomo generirali. Na ta način lahko precej lažje dobimo točno take umetne podatke, kot jih potrebujemo.

Poglavje 3

Uporabljena orodja

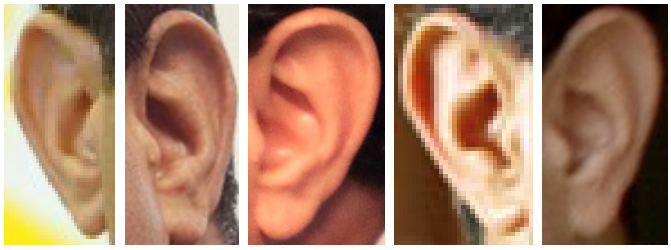
V tem poglavju predstavimo bazo anotiranih slik uhljev AWE [4], ki je služila kot vir slik za učenje generativnih mrež, orodja in knjižnice za pripravo slik za učenje ter za delo z nevronskimi mrežami.

3.1 Podatkovna baza AWE

Annotated Web Ears [4] (AWE) je podatkovna baza iz interneta pridobljenih slik uhljev. Verzija AWE-v1, ki smo jo uporabljali, vsebuje 1.000 slik uhljev, ki pripadajo 100 različnim osebam. Ustvarjena je bila z namenom raziskovalnega dela na prepoznavanju uhljev v nekontroliranem okolju. Vse slike v AWE bazi so označene glede na nagib glave, stopnjo zakritosti uhlja, etničnost, spol in identiteto osebe. Nekaj primerov slik iz baze je vidnih na sliki 3.1. AWE bazo smo za potrebe naših eksperimentov razdelili na učno množico 800 slik, osem na osebo in testno množico 200 slik, dve na osebo.

3.2 Orodje za izrez in anotacijo uhljev

Orodje za izrez in anotacijo uhljev [20] je bilo implementirano za dodajanje novih slik v AWE bazo in njihovo anotiranje, omogoča pa tudi urejanje obstoječih anotacij na slikah iz baze. Uporabniški vmesnik orodja je viden



Slika 3.1: Primeri slik iz baze AWE.

na sliki 3.2. Orodje je na voljo za prenos na spletni strani podatkovne baze AWE [4].

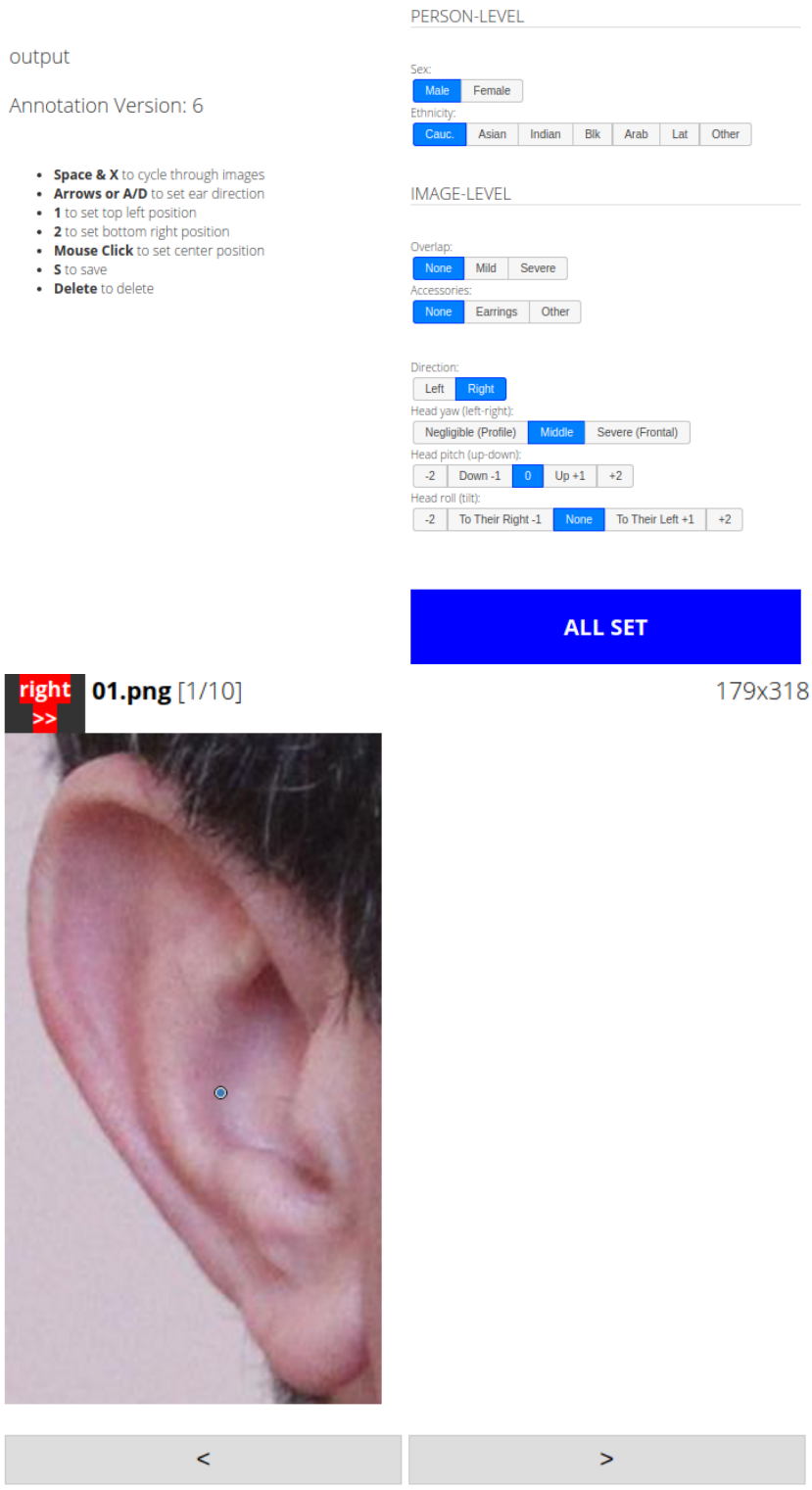
Ker smo za poravnavo slik iz baze AWE potrebovali dodatne anotacije, smo to orodje nadgradili tako, da poleg obstoječih anotacij omogoča shranjevanje štirih dodatnih točk: najvišje, najnižje, najbolj zunanje in najbolj notranje točke uhlja.

3.3 OpenCV

OpenCV (Open Source Computer Vision Library) [21] je odprtokodna knjižnica za računalniški vid, procesiranje slik in strojno učenje. Napisana je v programskih jezikih C in C++, uporabljati pa jo je možno tudi v Pythonu in Javi. Funkcije knjižnice OpenCV smo v našem delu uporabljali za odstranjevanje ozadja iz slik na podlagi mask in za poravnavo slik s homografijo.

3.4 Keras

Keras [22] je knjižnica za globoko učenje v programskem jeziku Python. Služi kot vmesnik za knjižnice Tensorflow, Theano in Microsoft Cognitive Toolkit (za uporabo Kerasa moramo imeti nameščeno eno od teh). Omogoča hitro in preprosto ustvarjanje modelov ter izvajanje učenja na CPU ali na GPU. Keras smo uporabili za implementacijo in učenje generativnih mrež, opisanih v naslednjem poglavju, in pa tudi za učenje mrež za razpoznavo.



Slika 3.2: Uporabniški vmesnik orodja za označevanje.

Poglavje 4

Opis metod

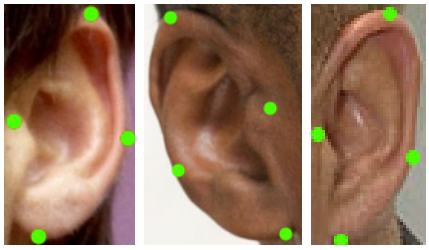
V tem poglavju najprej opišemo, kako smo slike iz baze AWE pripravili za učenje generativnih mrež in kakšne tehnike smo za to uporabili. Nato predstavimo arhitekture in postopke učenja teh mrež ter generiranje umetnih podatkov. Nazadnje opišemo arhitekture mrež, ki smo jih uporabili za učenje in evalvacijo razpoznave.

4.1 Dodatna anotacija in obdelava slik

Slike v AWE bazi so zajete pod različnimi koti, za učenje generativne nevronske mreže pa moramo poskrbeti, da so vse slike enako poravnane. Za potrebe poravnave smo z orodjem, opisanim v podpoglavju 3.2, na vsaki sliki označili najvišjo, najnižjo, najbolj zunanjo in najbolj notranjo točko uhlja (primeri na sliki 4.1). Kot dodatno smo iz slik izločili ozadje na podlagi mask, ustvarjenih v okviru diplomske naloge *Vpliv poravnave na uspešnost razpoznavanja uhljev* [23] in vse desne uhlje zrcalili preko y osi.

4.1.1 Homografija

Homografija [24] je projekcija, ki omogoča preslikavo slike iz ene ravnine v drugo na podlagi štirih poljubnih točk. S to projekcijo smo na podlagi teh štirih dodatnih točk vse uhlje preslikali na isto ravnino in jih na ta način



Slika 4.1: Primeri anotacij skrajnih točk uhlja.

Slika 4.2: Postopek poravnave slik na dveh primerih uhljev iz AWE baze. Od leve proti desni: originalna slika, slika brez ozadja, poravnana slika.

poravnali. Za izvedbo smo uporabili implementacijo homografije v knjižnici OpenCV. Primer odstranjevanja ozadja in poravnave s homografijo je viden na sliki 4.2.

Zaradi različnih oblik uhljev in velikih variacij med koti, pod katerimi so bile slike zajete, ta metoda ne daje vedno idealnih rezultatov – lahko dobimo precej deformirane slike, poleg tega pa postane še bolj izrazit rob mask, s katerimi smo iz slik odstranjevali ozadje. Vseeno takih slik, zaradi že tako majhne osnovne baze, nismo izločali. Nekaj takih primerov je vidnih na sliki 4.3.

Slika 4.3: Primer slabih poravnav uhljev iz AWE baze.

4.2 Učenje generativne nevronske mreže

V naših eksperimentih smo uporabili arhitekturo s štirimi dekonvolucijskimi moduli, sestavljenimi iz plasti višanja vzorčenja, 5×5 konvolucije in nato še 3×3 konvolucije, z vmesnimi normalizacijskimi plastmi. Arhitektura je prikazana v tabeli 4.1.

Na vhod mreže smo dali vektor identitete slike, ki jo generiramo.

Na začetku smo generirali slike velikosti 128×192, a se je izkazalo, da so bile zaradi premajhne povprečne velikosti originalnih slik iz AWE baze (83×160 slikovnih točk) preveč zamegljene. Zato smo najprej poskusili iz baze vzeti le slike z vsaj 200 slikovnih točk višine in učiti mrežo le na podlagi teh slik. Izkazalo se je, da je takih slik zgolj 237, kar pa ni bilo dovolj, da bi se model naučil generirati sprejemljive slike. Odločili smo se, da velikost generiranih slik zmanjšamo na 64×96, kar je prineslo rahlo boljše rezultate. Rezultate smo poskusili izboljšati tudi s povečanjem kontrasta na originalnih slikah in nato z zmanjšanjem na generiranih slikah, a to ni prineslo vidnih izboljšav rezultatov.

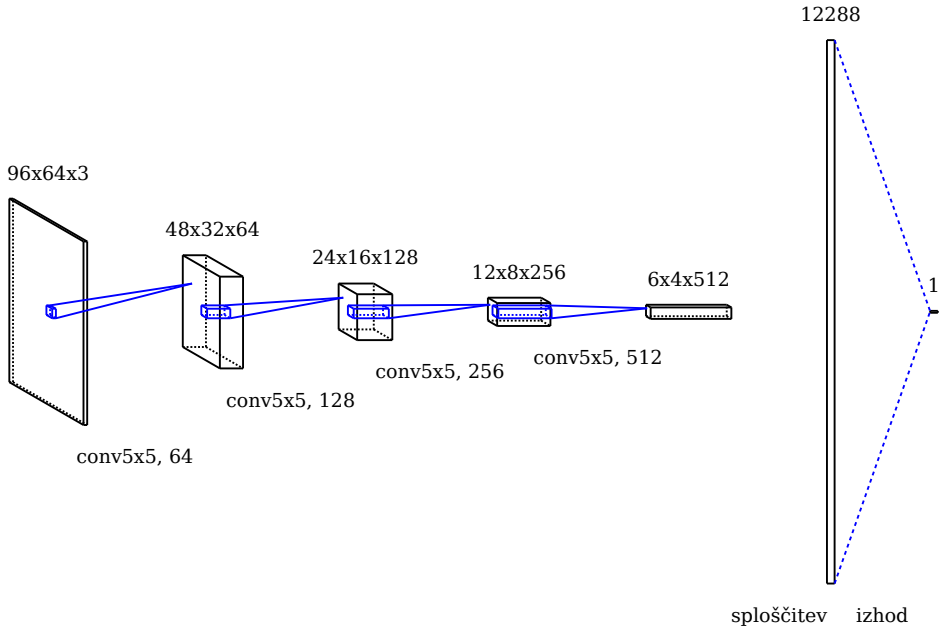
4.3 Uporaba generativne nasprotniške mreže za izboljšavo podatkov

Da bi izboljšali kvaliteto slik, ki jih ustvari generativna mreža, smo jo združili s konvolucijsko mrežo in tako ustvarili GAN.

Za generator smo vzeli generativni model, opisan v prejšnjem podpo-

Tip plasti	oblika izhoda	št. parametrov
vhodna	$100 \times 1 \times 1$	0
polno povezana	$512 \times 1 \times 1$	51.712
polno povezana	$1024 \times 1 \times 1$	525.352
polno povezana	$2560 \times 1 \times 1$	2.624.000
višanje vzorčenja	$10 \times 8 \times 128$	0
konvolucijska	$10 \times 8 \times 128$	409.728
konvolucijska	$10 \times 8 \times 128$	147.584
normalizacijska	$10 \times 8 \times 128$	512
konvolucijska	$20 \times 16 \times 128$	409.728
konvolucijska	$10 \times 8 \times 128$	147.584
normalizacijska	$10 \times 8 \times 128$	512
višanje vzorčenja	$160 \times 128 \times 96$	0
konvolucijska	$160 \times 128 \times 32$	76.832
konvolucijska	$160 \times 128 \times 32$	9.248
normalizacijska	$160 \times 128 \times 32$	128
zbirna	$160 \times 128 \times 32$	0
višanje vzorčenja	$320 \times 256 \times 32$	0
konvolucijska	$320 \times 256 \times 8$	6.408
konvolucijska	$320 \times 256 \times 8$	584
konvolucijska	$320 \times 256 \times 3$	219

Tabela 4.1: Arhitektura generativne nevronske mreže.



Slika 4.4: Diagram arhitekture diskriminatorja.

glavju, za diskriminator pa smo uporabili konvolucijsko mrežo, sestavljeno iz štirih 5×5 konvolucijskih plasti - med katerimi so puščajoče ReLU plasti, na koncu plast sploščitve (angl. flatten), ki obliko izhoda spremeni v vektor, izhod pa je en sigmoidni nevron, ki pove, ali je vhodna slika realna ali umetno generirana. Arhitektura je prikazana na diagramu 4.4.

Za uravnoteženje učenja smo najprej poskusili uporabiti fiksno razmerje med številom epoh učenja generatorja in številom epoh učenja diskriminatorja, a se je ravnovesje precej hitro porušilo. Zato smo algoritem učenja predelali tako, da se vselej uči tista mreža, ki ima trenutno večjo mero izgube, a je diskriminator vseeno čez čas postal preveč natančen. Da bi to preprečili, smo dodali še omejitev, da se diskriminator uči le, dokler je njegova mera izgube večja od določene konstante. V GAN generator tipično ni predhodno učen, a v našem primeru generativni model brez predhodnega učenja ni konvergiral, zato smo uporabili model, ki smo ga predhodno učili 1.000 epoh. To je prineslo dokaj stabilno učenje, a razen ostrejših robov ni bilo bistvenih izboljšav.

4.4 Učenje variacijskega avtokodirnika za generiranje podatkov

Druga metoda, ki smo jo uporabili za generiranje novih podatkov za učenje mreže za razpoznavo, je uporaba VAE za kombiniranje različnih slik iste osebe in s tem generiranje novih podatkov. Bistvo VAE-ja je v tem, da skuša modelirati verjetnostno distribucijo podatkov in nato na podlagi tega generirati nove podatke. Ker smo v našem primeru hoteli zagotoviti čim bolj enakomerno distribucijo generiranih podatkov glede na originalne slike, smo uporabili arhitekturo CVAE. Kot dodatni parameter smo dali vsaki sliki svojo oznako, ki nam bo kasneje omogočala, da pri generiranju slik z naučenim modelom določamo, katere od originalnih slik bomo generirali.

4.5 Generiranje podatkov z generativnim modelom

Naučen generativni model smo uporabili za generiranje nove učne množice, s katero bomo poskusili naučiti mreže za razpoznavo. Za vsako osebo iz originalne baze smo generirali 100 novih slik, in sicer tako, da smo generativnemu modelu na vhod dali oznako tiste osebe, katere sliko hočemo generirati, zraven pa smo vsakič dodali nekaj šuma, da smo dobili več variacij. Tako je nastala baza 10.000 slik, ki je bila uporabljena kot učna množica pri nadaljnjih eksperimentih.

4.6 Generiranje podatkov z variacijskim avtokodirnikom

Ker je bil cilj kombinirati med sabo samo slike iste osebe, smo CVAE model naučili za vsako osebo posebej, nato pa s podajanjem ustreznih oznak na vhod mreže generirali slike tako, da smo za vsak par slik iste osebe generirali

nekaj slik prehoda med njima. Enako kot pri generativnem modelu smo generirali 100 slik vsake osebe, skupaj 10.000.

4.7 Mreže za razpoznavo

Za učenje razpoznave smo uporabili tri različne arhitekture konvolucijskih nevronskih mrež: SqueezeNet [25], VGG16 [26] in Inception-v4 [27]. Njihove lastnosti opišemo v sledečih podpoglavjih.

4.7.1 SqueezeNet

Kot prvo mrežo smo za testiranje izbrali SqueezeNet [25], in sicer zaradi manjšega števila parametrov in velikosti modela, kar pomeni, da za učenje potrebuje manj računske moči in posledično manj časa. Kljub manjši arhitekturi (1,2 milijona parametrov) pa mreža doseže visoko natančnost na bazi slik ImageNet [28].

SqueezeNet v svoji arhitekturi vsebuje module, ki so sestavljeni iz slojev za stiskanje (angl. squeeze) in slojev za razširitev (angl. expand) plasti. Sloj za stiskanje je konvolucijska plast, sestavljena samo iz 1×1 filtrov, sloj za razširitev pa vsebuje 1×1 in 3×3 filtre.

4.7.2 VGG16

VGG16 [26] je eden od globokih modelov, ki jih je razvil Visual Geometry Group iz Oxforda za tekmovanje ImageNet Challenge 2014. Za to arhitekturo je značilna uporaba več skladov - sestavljenih iz dveh do štirih zaporednih konvolucijskih plasti z majhnimi (3×3) filtri namesto posameznih konvolucijskih plasti z večjimi, npr. 7×7 ali 11×11 filtri. Taki skladi konvolucijskih plasti lahko iz istega območja slike izluščijo več informacij kot ena plast z večjim filtrom, kljub temu da vsebujejo manj parametrov. Med temi skladi so zbirne plasti, na koncu pa še tri polno povezane plasti. VGG16 vsebuje skupno 16 plasti z utežmi, kar nanese 138 milijonov parametrov.

4.7.3 Inception-v4

Inception-v4 [27] je ena od različic Inception arhitekture. Njena posebnost so Inception moduli, ki vhod vzporedno pošljejo skozi več konvolucijskih plasti z različnimi velikostmi filtrov in nato njihove izhode spet združijo. To omogoča, da ti moduli iz slike hkrati izluščijo tako lokalne kot večje, bolj kompleksne značilke.

Inception arhitekture potrebujejo manj računske moči od VGG arhitektur, kljub temu pa dosegajo boljše rezultate na ImageNet-u [29]. Inception-v4 vsebuje skupno 42 milijonov parametrov.

Poglavje 5

Rezultati

Rezultate generiranja slik analiziramo kvalitativno in kvantitativno. Kvalitativni eksperimenti nam pomagajo preko slik bolje razumeti, kako se generativne mreže učijo in odkriti morebitne težave. Kvantitativni eksperimenti pa preko metrik pokažejo, kako se generirani podatki dejansko obnesejo pri učenju razpoznave in kakšne izboljšave prinese uporaba teh podatkov.

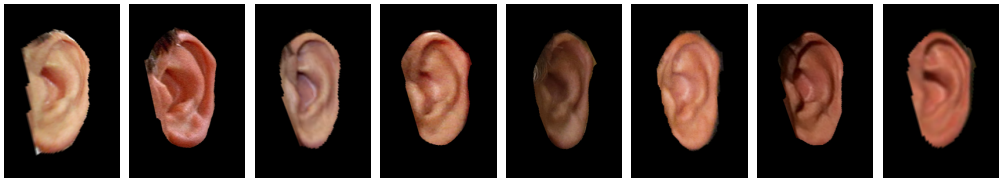
5.1 Kvalitativna analiza

Generirane slike najprej analiziramo kvalitativno. Vizualno ocenimo njihovo uporabnost za učenje razpoznave in ugotavljamo, s čim so imele generativne mreže težave.

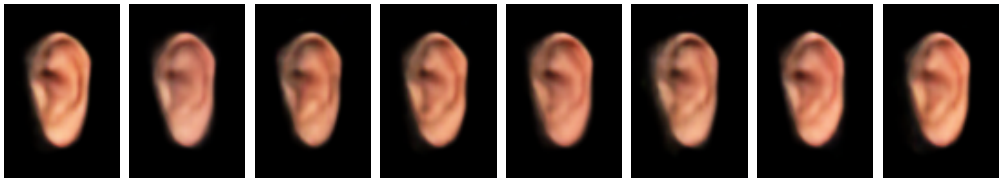
5.1.1 Slike generirane z generativnim modelom

Pri identitetah, kjer so bile originalne slike zajete pod podobnimi koti in jih lahko ustrezno poravnali, se je mreža naučila generirati slike, na katerih se vidi dovolj podrobnosti, da bi bile lahko uporabne za učenje razpoznave. Rezultati generiranja slik za eno od identitet, skupaj z originalnimi in poravnanimi slikami iste identitete, so vidni na sliki 5.1. Opazi se, da slike niso tako jasne kot učne slike. Poleg tega vidimo, da kljub precejšnjim razlikam v barvi pri originalnih slikah zaradi različnih svetlobnih pogojev, pod kate-

Slike iz baze AWE, z odstranjenim ozadjem.



Poravnane slike, uporabljene za učenje generativne mreže.

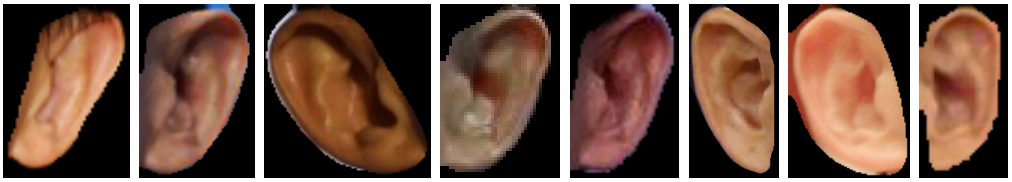


Rezultati generiranja umetnih slik iste identitete.

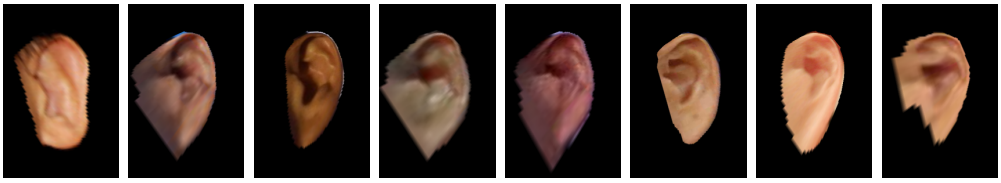
Slika 5.1: Primerjava originalnih, poravnanih in generiranih slik.

rimi so bile posnete, pri generiranih slikah razlik skoraj ni. Predvidevamo, da je razlog v tem, da ob učenju vse slike iste identitete označimo enako, pričakujemo pa vsakič drugačen izhod. Mreža se tako nauči neke povprečne slike za to identiteto, za katero bo funkcija izgube minimalna pri vseh slikah te identitete.

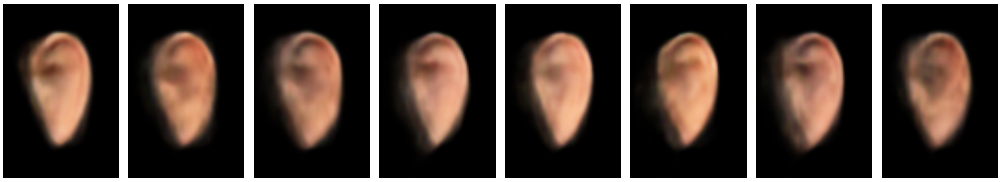
Za nekatere identitete pa se zaradi prevelikih razlik med slikami model ni uspel naučiti generirati jasnih slik. Eden od problemov so različni koti, pod katerimi so bile zajete slike in jih tudi s postopkom homografije nismo uspeli popolnoma poravnati v vseh primerih. Precej se lahko pozna tudi že različna osvetljenost uhlja na sliki. Rezultati generiranja za take identitete so slike, iz katerih razberemo samo približno obliko uhlja, ostalo pa je preveč zamegljeno za kakršnokoli identifikacijo. Primerjava originalnih, poravnanih



Slike iz baze AWE, z odstranjenim ozadjem.



Poravnane slike, uporabljene za učenje generativne mreže.

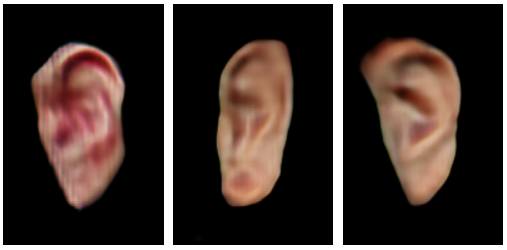


Rezultati generiranja umetnih slik iste identitete.

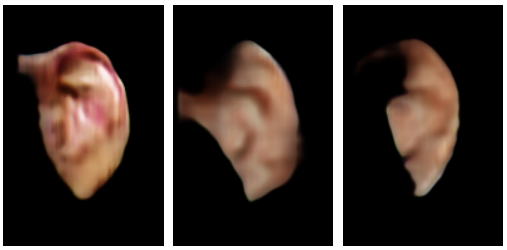
Slika 5.2: Primerjava originalnih, poravnanih in generiranih slik za eno od problematičnih identitet.

in obrezanih ter generiranih slik za eno od problematičnih identitet je viden na sliki 5.2.

Slike, generirane z modelom, ki je bil dodatno učen kot del GAN (nekaj primerov na sliki 5.3), imajo v splošnem bolj ostre robove, jasnost slik pa se v primerjavi s prejšnjim modelom ni vidno izboljšala. Po daljšem učenju GAN so nekatere od generiranih slik začele dobivati tudi razne deformacije, vidne na sliki 5.4, zato smo za učenje razpoznave uporabili slike modela brez dodatnega učenja kot del GAN.



Slika 5.3: Primeri slik generiranih z GAN.

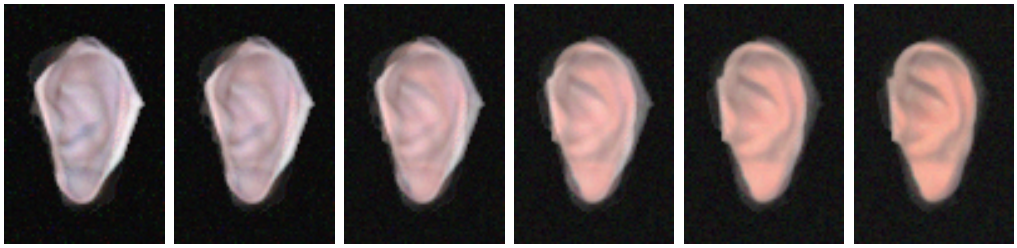


Slika 5.4: Primeri deformacij na slikah generiranih z GAN.

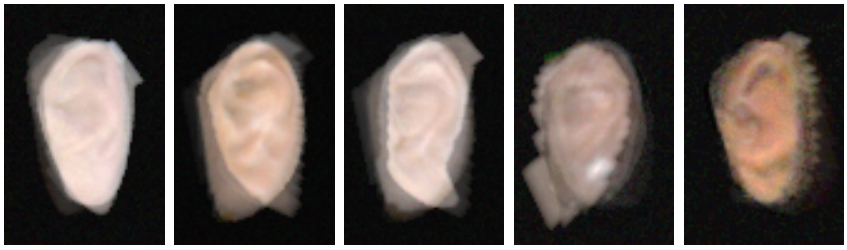
5.1.2 Slike generirane z variacijskim avtokodirnikom

Ker smo model CVAE učili za vsako identiteto posebej, tukaj velike razlike med originalnimi slikami niso tako problematične. Model se ne nauči le nekega povprečja, ampak lahko s podajanjem oznak na vhod mreže izbiramo, katero kombinacijo osnovnih slik želimo generirati. Kot smo omenili v prejšnjem poglavju, smo za vsako sliko generirali več slik prehoda med njima. En primer takega prehoda je viden na sliki 5.5.

Generirane slike so manj zamegljene kot pri generativnem modelu, prav



Slika 5.5: Primer kombinacij dveh slik generiranih z mrežo CVAE. Originalni sliki na skrajni levi in desni, vmes slike prehoda.



Slika 5.6: Nekaj primerov slabih generiranj s CVAE.

tako pa zaradi kombiniranja različnih slik iste identitete lahko dobimo vizualno bolj raznolike slike. Vseeno pa pri identitetah, kjer so bile poravnane slike preveč popačene, nismo dobili dobrih rezultatov, prav tako pa je mreža v nekaterih primerih generirala zelo svetle slike. Nekaj takih primerov je prikazanih na sliki 5.6.

5.2 Kvantitativna analiza

Zanimalo nas je, v kolikšni meri generirani podatki pripomorejo k natančnosti mreže za razpoznavo, ko z njimi povečamo učno množico. Test smo izvedli tako, da smo uporabili tri arhitekture konvolucijskih nevronskih mrež - SqueezeNet [25], VGG16 [26] in Inception-v4 [27].

Učenje modelov smo izvajali na sistemu z grafično kartico Nvidia GeForce GTX 970 s 4 GiB grafičnega pomnilnika, z izjemo učenja modela Inception-v4, kjer smo uporabili grafično kartico Nvidia Geforce GTX 1080 Ti s 11 GiB grafičnega pomnilnika (4 GiB za ta model ni zadoščalo).

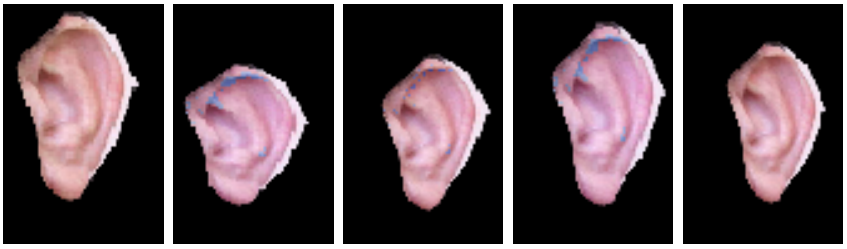
Za funkcijo napake smo izbrali kategorično križno entropijo, kot optimizacijski algoritem pa smo uporabili algoritem Adam.

5.2.1 Učne in testne množice

Za učenje razpoznave smo uporabili 3 različne učne množice, generirane na podlagi 800 slik iz baze AWE, 8 slik na osebo. Množica A vsebuje 10.000 slik generiranih z generativnim modelom, množica B 10.000 slik generiranih

parameter	vrednost
razpon skaliranja po širini	0,1
razpon skaliranja po višini	0,1
razpon povečave	0,2
razpon striženja	0,1

Tabela 5.1: Parametri uporabljeni za augmentacijo slik s pomočjo ogrodja Keras.



Slika 5.7: Primer variacij ene od originalnih slik, ustvarjenih z uporabo transformacij.

s modelom CVAE, množica C pa vsebuje 10.000 slik pripravljenih z uporabo transformacij skaliranja, rotacije in striženja ter spreminjanja barv na osnovnih slikah.

Za pripravo množice C smo uporabili kar v Keras vgrajeno funkcijo *ImageDataGenerator* s parametri navedenimi v tabeli 5.1. Primer variacij, generiranih za eno od slik, je viden na sliki 5.7.

Testno množico pri vseh eksperimentih sestavlja preostalih 200 slik iz AWE baze, ki jih nismo uporabili pri učenju generativnih mrež, in sicer dve sliki na osebo.

5.2.2 Metrike

Pri analizi klasifikatorja za uporabo v biometriji nas tipično zanimata dve meri: natančnost identifikacije in natančnost verifikacije.

Identifikacija

Pri identifikaciji skušamo na podlagi novega podatka (slike uhlja) z uporabo obstoječe baze podatkov in identitet ugotoviti, kateri osebi slika pripada. Če imamo v bazi n identitet in skušamo določiti, kateri od teh identitet pripada nova slika, je to identifikacija zaprtega seta (angl. closed set identification). Če pa testiramo tudi nove identitete, govorimo o identifikaciji odprtega seta (angl. open set identification). Identifikacija je lahko popolnoma avtomatska, lahko pa polavtomatska, kjer klasifikator vrne prvih nekaj rezultatov po stopnji ujemanja, nato pa izmed teh ročno določimo pravo identiteto [5].

Rezultate identifikacije analiziramo s CMC (angl. cumulative match characteristic) krivuljo [5]. Najprej izračunamo χ^2 razdalje med vsakim parom značilk. Te razdalje za vsako sliko uredimo po velikosti od najmanjše do največje in jih rangiramo. Za vsako od teh vrst ugotovimo, na katerem mestu se nahaja korektna identiteta te slike. Na podlagi teh vrednosti lahko potem izračunamo stopnjo razpoznave za rang 1 (angl. Rank 1 recognition rate) in za višje range, recimo za peti ali deseti rang. Stopnja razpoznave za rang n pomeni, kolikšna je verjetnost, da bo za neko sliko pravilna identiteta med prvimi n po izračunani razdalji med značilkami.

Če to izračunamo za vsako vrednost od 1 do n, kjer je n število vseh identitet v bazi, dobljene vrednosti pa povežemo, dobimo CMC krivuljo. Poleg stopnje razpoznave pri CMC krivuljah računamo tudi AUCMC (angl. area under the CMC curve) vrednost – ploščino pod CMC krivuljo.

Verifikacija

Pri verifikaciji ne ugotavljamo identitete, temveč želimo za neko osebo s primerjavo nove slike z obstoječimi - z dovolj visoko gotovostjo pokazati, ali se identiteti res ujemata. Če je stopnja ujemanja opazovane slike z ostalimi nad neko pragovno vrednostjo, imamo pozitivno zaznavo, sicer negativno.

Za ponazoritev uspešnosti klasifikatorja lahko uporabimo ROC (angl. receiver operating characteristic) krivuljo [5]. Z ROC krivuljo prikažemo razmerje med deležem pravilnih pozitivnih zaznav (TAR, angl. true acceptance

rate) in deležem napačnih pozitivnih zaznav (FAR, angl. false acceptance rate) pri verifikaciji, ob spreminjajoči se pragovni vrednosti za pozitivno zaznavo.

Če je L število vseh primerjav, od tega je prvih L_1 primerjav s slikami prave identitete, sledi pa L_0 primerjav s slikami ostalih identitet, ter je s_i rezultat i-te primerjave, lahko FAR in TAR za določeno pragovno vrednost η izračunamo s sledečimi enačbami:

$$FAR(\eta) = \frac{1}{L_0} \sum_{i=L_1+1}^{L} I(s_i \geq \eta), \tag{5.1}$$

$$TAR(\eta) = 1 - \frac{1}{L_1} \sum_{i=1}^{L_1} I(s_i < \eta), \text{kjer} \tag{5.2}$$

$$I(x) = \begin{cases} 1 & \text{, če } x \text{ drži,} \\ 0 & \text{, sicer.} \end{cases} \tag{5.3}$$

Če je pri določeni pragovni vrednosti TAR večji od FAR, to pomeni, da je klasifikacija boljša od naključne, v nasprotnem primeru pa je klasifikacija slabša od naključne. Ta meja je na grafu ROC krivulje diagonala med točkama (0, 0) in (1, 1).

ROC krivulja je uporabna za primerjavo večih klasifikatorjev. Če je TAR klasifikatorja A konsistentno večja od TAR klasifikatorja B pri istem FAR, lahko zaključimo, da je klasifikator A boljši. V primeru, da se krivulji sekata, pa lahko primerjamo deleže ploščine grafa pod krivuljo (angl. area under curve, AUC), ki jo izračunamo z enačbo:

$$AUC = \frac{\sum_{i=1}^{L_1} \sum_{j=L_1+1}^{L} I(s_i > s_j)}{L_0 L_1}. \tag{5.4}$$

Klasifikatorje skušamo pripeljati do tega, da se čim bolj približajo točki (0, 1), ki ponazarja pravilno klasifikacijo vseh primerov.

5.2.3 Klasifikacijski testi

V tem podpoglavju predstavimo rezultate učenja razpoznave na učnih množicah, opisanih v podpoglavju 5.2.1. Natančnost pri teh testih pomeni odstotek

testnih primerov, kjer je predvidena identiteta slike enaka dejanski identiteti (kot zgoraj TAR). Kot predvideno Keras označi tisto identiteto, za katero izhodna plast mreže vrne najvišjo verjetnost. V naši bazi imamo 100 različnih identitet, kar pomeni, da bi naključni klasifikator dosegel natančnost okoli enega odstotka.

Učenje mrež za razpoznavo

Na začetku smo imeli težave z vzpostavljanjem učenja na celotni bazi, zato smo se odločili, da najprej vzamemo bazo 10-ih oseb in najdemo primerne parametre optimizacije, nato pa te parametre uporabimo pri učenju na celotni bazi. Ko smo na bazi 10-ih oseb s SqueezeNet-om dosegli 35 % natančnost na testni množici, smo prešli na učenje mrež na celotnih bazah.

Za testiranje smo uporabili tri že omenjene mreže SqueezeNet, VGG16 in Inception-v4. Na mreži SqueezeNet smo učenja izvajali 2.000 epoh. Pri mrežah VGG16 in Inception-v4 je učenje vzelo precej več časa na epoho, poleg tega pa je natančnost na učni množici že precej prej dosegla skoraj popolno natančnost in se je mreža nehala učiti, zato smo učenje prekinili prej. Na mreži VGG16 smo učenja izvajali 200 epoh, na mreži Inception-V4 pa 150 epoh.

Testiranje natančnosti

Na slikah generiranih z GNN (učna množica A) smo z arhitekturo SqueezeNet dosegli 12 % natančnost, z VGG16 11 % natančnost, z mrežo Inception-v4 pa le 5 % natančnost. Na slikah generiranih s CVAE (učna množica B) smo z SqueezeNet-om dosegli 6 %, z VGG16 4 %, z Inception-v4 pa 3 % natančnost. Z učno množico augmentirano samo s transformacijami (učna množica C) smo s SqueezeNet-om dosegli 8 % natančnost, z VGG16 11 % natančnost, z Inception-v4 pa 17 %.

Zaradi najboljše natančnosti smo, za dodatno analizo na podlagi vektorjev značilk, izbrali model mreže SqueezeNet - naučen na učni množici A (slike generirane z generativnim modelom). Za primerjavo smo v nadalj-

njo analizo vzeli tudi model SqueezeNet-a - naučen na učni množici C (slike augmentirane samo s transformacijami).

5.2.4 Analiza vektorjev značilk

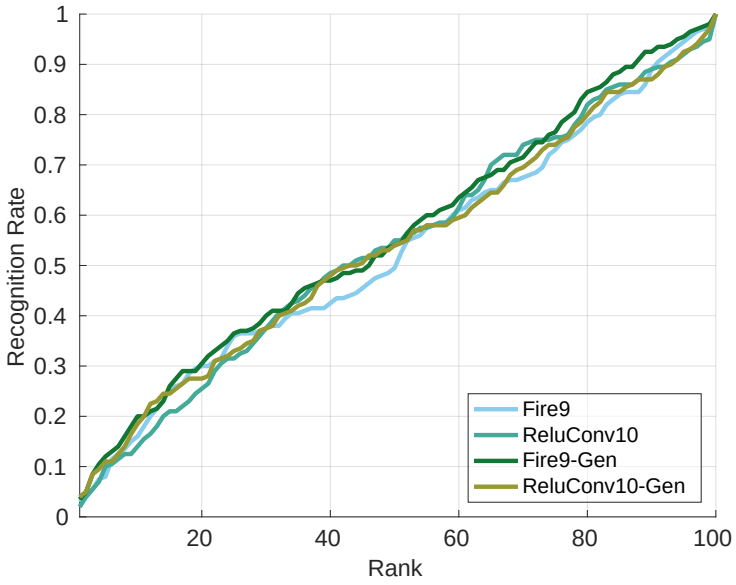
Rezultate učenja mreže SqueezeNet za razpoznavo smo analizirali še z uporabo vektorjev značilk. Te smo dobili tako, da smo shranili izhoda zadnje konvolucijske plasti (Fire9) in zadnje skrite plasti (ReluConv10) v naučenih modelih SqueezeNet-a za vsako od slik v testni množici.

Za analizo teh vektorjev smo uporabili funkcije orodja AWE Toolbox [4]. Rezultati so vidni v tabeli 5.2. Značilki Fire9-Gen in ReluConv10-Gen sta pridobljeni iz modela - naučenega na učni množici A (slike generirane z generativno mrežo), Fire9 in ReluConv10 pa iz modela - naučenega na učni množici C (brez generiranja slik). Rezultate smo vizualizirali tudi s CMC (slika 5.8) in ROC (slika 5.9) krivuljama.

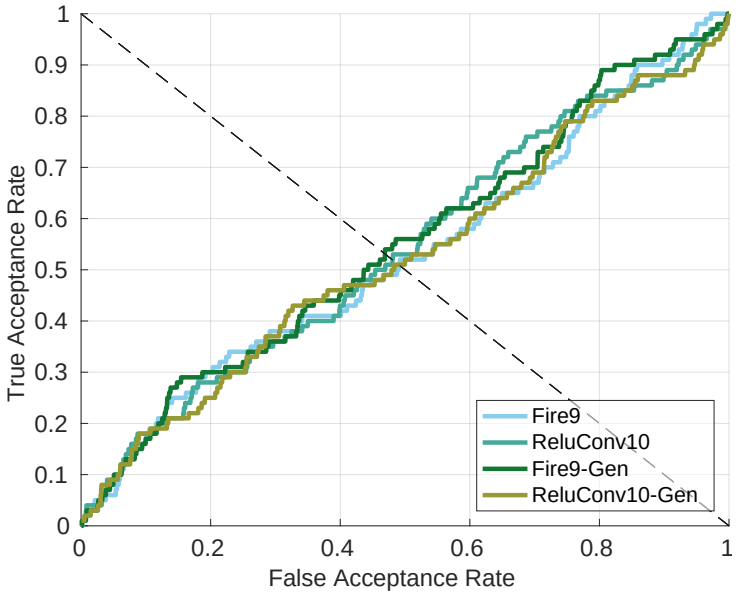
Vir značilk	AUCMC (%)	AUC (%)
Fire9	53,0	52,9
ReluConv10	54,0	54,1
Fire9-Gen	**56,0**	**54,8**
Reluconv10-Gen	53,9	52,6

Tabela 5.2: Rezultati razpoznave na podlagi vektorjev značilk. Fire9 in ReluConv10 sta rezultata mreže naučene na učni množici C (brez generiranja slik), Fire9-Gen in ReluConv10-Gen pa rezultata mreže naučene na učni množici A (generirane z generativno mrežo).

Rezultati so pri značilkah iz obeh modelov precej nizki. Kljub temu pa lahko opazimo nekaj odstotne izboljšave pri merah AUCMC in AUC - pri uporabi generiranih slik v primerjavi z modelom, kjer smo uporabili samo transformacije. To kaže na to, da je v generiranju slik za augmentacijo učne množice potencial, bi pa morali za uporabne rezultate precej izboljšati učenje CNN-jev.



Slika 5.8: CMC krivulje za vektorje značilk obeh mrež.



Slika 5.9: ROC krivulje za vektorje značilk obeh mrež.

Poglavje 6

Zaključek

V diplomski nalogi smo testirali več metod generiranja umetnih slik v namen obogatitve učne množice slik za uporabo v globokem učenju. Implementirali smo generativno nevronsko mrežo za generiranje slik. Naučen model te mreže smo nato povezali z diskriminatorno mrežo in tako ustvarili GAN, s katerim smo poskušali izboljšati kvaliteto generiranih slik. Druga metoda pa je bila implementacija variacijskega avtokodirnika. Pri tej metodi smo se odločili za arhitekturo CVAE, ker lahko s to arhitekturo bolje določamo, kakšne slike želimo generirati. Z obema modeloma smo generirali novo učno množico umetnih slik.

V kvalitativni analizi smo ugotovili, da sta mreži za nekatere identitete generirali uporabne učne podatke, bilo pa je kar precej problematičnih identitet. Problematične so bile predvsem tiste identitete, pri katerih smo imeli težave že s poravnavo originalnih slik oziroma tiste, pri katerih so se precej razlikovali svetlobni pogoji, pod katerimi so bile slike zajete. Pri takih identitetah za generirane slike vizualno težko določimo, kateri identiteti pripadajo.

Za kvantitativno analizo smo z vsako od teh množic učili mreže za razpoznavo in primerjali rezultate. Za primerjavo smo mreže za razpoznavo učili še na tretji učni množici, augmentirani samo z uporabo osnovnih transformacij. Natančneje smo rezultate analizirali še na podlagi vektorjev značilk. Izkazalo se je, da z osnovno bazo 800 slik nobena od teh metod ne izboljša rezulta-

tov do te mere, da bi se lahko zanašali na rezultate razpoznave z nevronsko mrežo.

Ena od metod, ki bi lahko izboljšala rezultate na tej bazi, je projekcija slik na 3D model uhlja iz članka [30] in nato generiranje slik različnih transformacij tega modela, a ima AWE baza trenutno premalo anotiranih točk za izvedbo projekcije. V okviru te diplomske naloge smo za namene poravnave slik dodali 4 točke, baza, ki so jo uporabljali avtorji članka za generiranje modela, pa ima na vsaki sliki označenih 55 točk.

Rezultate bi lahko poskusili izboljšati še s slikami, ki so bile zbrane za tekmovanje UERC [31], na ta način, da bi jim dodali enake anotacije, potrebne za poravnavo.

Literatura

[1] Žiga Emeršič. Razpoznava oseb na podlagi biometričnih podatkov uhljev. Magistrska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2015.

[2] Žiga Emeršič, Blaž Meden, Peter Peer, and Vitomir Štruc. Evaluation and analysis of ear recognition models: performance, complexity and resource requirements. *Neural Computing and Applications*, pages 1–16.

[3] Žiga Emeršič, Dejan Štepec, Vitomir Štruc, and Peter Peer. Training convolutional neural networks with limited training data for ear recognition in the wild. *arXiv preprint arXiv:1711.09952*, 2017.

[4] Žiga Emeršič, Vitomir Štruc, and Peter Peer. Awe toolbox & dataset. Dosegljivo: `http://awe.fri.uni-lj.si/`, 2017. [Dostopano: 10. 9. 2018].

[5] Anil K Jain, Arun A Ross, and Karthik Nandakumar. *Introduction to biometrics*. Springer Science & Business Media, 2011.

[6] Žiga Emeršič, Vitomir Štruc, and Peter Peer. Ear recognition: More than a survey. *Neurocomputing*, 255:26–39, 2017.

[7] Žiga Emeršič, Luka Lan Gabriel, Vitomir Štruc, and Peter Peer. Pixel-wise ear detection with convolutional encoder-decoder networks. *CoRR*, abs/1702.00307, 2017.

[8] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[9] Diederik P. Kingma and Jimmy Ba. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, volume 5, 2015.

[10] Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

[11] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[13] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

[14] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1538–1546, 2015.

[15] Oliver Langner, Ron Dotsch, Gijsbert Bijlstra, Daniel HJ Wigboldus, Skyler T Hawk, and AD Van Knippenberg. Presentation and validation

of the radboud faces database. *Cognition and emotion*, 24(8):1377–1388, 2010.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[17] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.

[18] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[19] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015.

[20] Žiga Emeršič and Peter Peer. Toolbox for ear biometric recognition evaluation. In *EUROCON 2015-International Conference on Computer as a Tool (EUROCON), IEEE*, pages 1–6. IEEE, 2015.

[21] Gary Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[22] François Chollet et al. Keras. `https://keras.io`, 2015. [Dostopano: 10. 9. 2018].

[23] Metod Ribič. Vpliv poravnave na uspešnost razpoznavanja uhljev. Diplomska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2016.

[24] Ondřej Chum, Tomáš Pajdla, and Peter Sturm. The geometric error for homographies. *Computer Vision and Image Understanding*, 97(1):86 – 102, 2005.

[25] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, abs/1602.07360, 2016.

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

[27] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI Conference on Artificial Intelligence*, volume 4. AAAI, 2017.

[28] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[29] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. *arXiv preprint arXiv:1605.07678*, 2016.

[30] Hang Dai, Nicholas E. Pears, and William A. P. Smith. A data-augmented 3d morphable model of the ear. In *2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018)*, pages 404–408, May 2018.

[31] Žiga Emeršič, Dejan Štepec, Vitomir Štruc, Peter Peer, Anjith George, Adii Ahmad, Elshibani Omar, Terranee E Boult, Reza Safdaii, Yuxiang Zhou, et al. The unconstrained ear recognition challenge. In *Biometrics*

(IJCB), 2017 IEEE International Joint Conference on, pages 715–724. IEEE, 2017.