Razvoj Android aplikacije za prepoznavanje lica

Salihbašić, Alen

Master's thesis / Diplomski rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Pula / Sveučilište Jurja Dobrile u Puli**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:137:810003

Rights / Prava: Attribution 4.0 International/Imenovanje 4.0 međunarodna

Download date / Datum preuzimanja: **2024-03-28**

Image not found or type unknown *Repository / Repozitorij:*

Digital Repository Juraj Dobrila University of Pula

zir.nsk.hr Image not found or type unknown

Sveučilište Jurja Dobrile u Puli
Fakultet ekonomije i turizma
«Dr. Mijo Mirković»

ALEN SALIHBAŠIĆ

RAZVOJ ANDROID APLIKACIJE ZA PREPOZNAVANJE LICA

Diplomski rad

Pula, 2018.

Sveučilište Jurja Dobrile u Puli
Fakultet ekonomije i turizma
«Dr. Mijo Mirković»

ALEN SALIHBAŠIĆ

RAZVOJ ANDROID APLIKACIJE ZA PREPOZNAVANJE LICA

Diplomski rad

JMBAG: 0303012108, redoviti student

Studijski smjer: Poslovna informatika

Predmet: Programsko inženjerstvo

Mentor: doc.dr.sc. Tihomir Orehovački

Pula, studeni 2018.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani ALEN SALIHBAŠIĆ, kandidat za magistra ekonomije/poslovne ekonomije ovime izjavljujem da je ovaj Diplomski rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Diplomskog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student



_____

U Puli, 14. prosinca, 2018. godine

IZJAVA

o korištenju autorskog djela

Ja, ALEN SALIHBAŠIĆ dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj diplomski rad pod nazivom RAZVOJ ANDROID APLIKACIJE ZA PREPOZNAVANJE LICA koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, 14. prosinca, 2018. godine

Potpis



_____

Sadržaj

1. Uvod 1
2. Biometrijska identifikacija 3
2.1. Biometrijski sustav 3
2.2. Usporedba različith biometrija 5
2.2.1. DNK 5
2.2.2. Uho 6
2.2.3. Lice 6
2.2.4. Infracrveni termogram lica i tijela 8
2.2.5. Otisak prsta 8
2.2.6. Hod 9
2.2.7. Geometrija ruke i prsta 9
2.2.8. Mrežnica oka 9
2.2.9. Šarenica oka 10
2.2.10. Tipkanje 10
2.2.11. Miris 11
2.2.12. Otisak dlana 11
2.2.13. Potpis 11
2.2.14. Glas 12
3. Sustav prepoznavanja lica 13
3.1. Detekcija lica 13
3.1.1. Metode temeljene na znanju 15
3.1.2. Metode nepromjenjivih značajki 15
3.1.3. Metode podudaranja predloška 16
3.1.4. Metode temeljene na izgledu 16
3.2. Prepoznavanje lica 16
3.2.1. Analiza glavnih komponenti 17
3.2.2. Linearna analiza različitih 18
3.2.3. Lokalni binarni uzorci 19
4. Razvojni alati mobilne aplikacije za prepoznavanje lica 21
4.1. Android platforma 21
4.1.1 Linux jezgra 22

4.1.2. Sloj hardverske apstrakcije 23
4.1.3. Android izvršitelj 23
4.1.4. Nativne C/C++ biblioteke 24
4.1.5. Java API okvir 25
4.1.6. Aplikacije sustava 26
4.2. Struktura Android aplikacije 26
4.2.1. Aktivnosti 27
4.2.2. Usluge 28
4.2.3. Prijemnici emitiranja 29
4.2.4. Pružatelji sadržaja 29
4.3. Android studio 30
4.3.1. Struktura Android projekta 31
4.3.2. Korisničko sučelje 32
4.4. OpenCV 33
5. Programsko rješenje 35
5.1. Dijagram slučajeva korištenja 35
5.2. Dizajn i funkcionalnost 36
5.3. Opis rješenja 37
6. Zaključak 51
Literatura 53
Popis slika 57
Sažetak 58
Abstract 59

1. Uvod

Sustav prepoznavanja lica je jedan od najinteresantnijih i najvažnijih istraživačkih polja u posljednja dva desetljeća. Razlozi dolaze iz potreba za automatskim prepoznavanjem i sustavima nadzora, interesa u ljudskom vizualnom sustavu prepoznavanja lica, kao i dizajna sučelja između čovjeka i računala. Istraživanja uključuju znanja i istraživače iz raznih istraživačkih polja kao što su neuroznanost, psihologija, računalni vid, prepoznavanje uzoraka, obrada slike i strojno učenje. Mnogo istraživačkih radova je usmjereno ka prevladavanju različitih negativnih faktora koji utječu na uspješnu stopu prepoznavanja lica.

Prvi korak ka pouzdanom prepoznavanju lica je detekcija lokacija u slikama gdje se nalaze lica. Detekcija lica se smatra temeljem konačnom prepoznavanju lica. Međutim, to je zahtjevan zadatak za računalni sustav na koji utječu uvjeti poput varijabilnosti u razmjeru, lokacija lica, orijentacija osobe na slici, poza osobe, izrazi lica, prekrivenost osobe ili jednog dijela lica, uvjeti osvjetljenja i sl.

U ovom radu, istražena je mogućnost implementiranja sustava prepoznavanja lica, kao i prepoznavanje spola i dobne skupine, na mobilni uređaj u obliku Android aplikacije. Opisane su metode na kojima se temelji detekcija lica, te najčešće korištene metode za detekciju i prepoznavanje lica u svrhu računalnog vida, odnosno računalnog vida mobilnog uređaja. Implementirano je i programsko rješenje koje pomoću kamere mobilnog uređaja, na temelju slike, omogućuje prepoznavanje spola, dobne skupine i lice osobe.

Drugo poglavlje sadrži uvod u biometrijsku identifikaciju. Opisan je biometrijski sustav i mogućnosti načina rada, kao i uvjeti koje svaka ljudska tjelesna ili ponašajna karakteristika mora zadovoljiti kako bi se smatrala biometrijskom karakteristikom. Nabrojeni su i uspoređeni brojni različiti biometrijski identifikatori. U trećem poglavlju je opisan općeniti sustav prepoznavanja lica i njegov način rada. Zatim slijedi detekcija lica gdje su nabrojeni izazovi s kojima se susreće svaki pokušaj detekcije lica u slici. Nabrojene i opisane su metode detekcija lica. Također, nabrojene su i opisane metode

prepoznavanja lica u slici. Četvrto poglavlje opisuje sve razvojne alate koji su se koristili pri izgradnji Android mobilne aplikacije i Android platformu. U zadnjem poglavlju, pobliže je objašnjeno i prikazano programsko rješenje kroz dijagram slučajeva korištenja. Dizajn mobilne aplikacije i njezine funkcionalnosti detaljno su objašnjeni kroz slike programskog koda i stvarnog izgleda mobilne aplikacije na Android uređaju.

2. Biometrijska identifikacija

Osobna identifikacija je proces povezivanja pojedinca s identitetom. Identifikacija može biti u obliku verifikacije, također poznato kao autentifikacija, koja odgovara na pitanje; „Jesam li ja onaj koji tvrdim da jesam?“ ili identifikacije, također poznato kao prepoznavanje, koja odgovara na pitanje; „Tko sam ja?“ (Jain, et al., 2000.).

Dvije tradicionalne tehnike za osobnu identifikaciju su automatski osobni identifikacijski pristupi temeljeni na znanju i temeljeni na tokenu (Miller, 1994). Pristupi temeljeni na tokenu koriste nešto što pojedincu mora omogućiti osobnu identifikaciju, poput putovnice, vozačke dozvole, osobne iskaznice, kreditne kartice ili ključeva. Pristupi temeljeni na znanju koriste nešto što pojedinac poznaje kako bi omogućio svoju osobnu identifikaciju, kao što je lozinka ili PIN[1] broj. Budući da se ovi tradicionalni pristupi ne temelje na inherentnim atributima pojedinca radi osobne identifikacije, oni pate od očitih nedostataka, tokeni mogu biti izgubljeni, ukradeni, zaboravljeni, a korisnik može zaboraviti PIN broj ili ga varalica može pogoditi.

Pristupi temeljeni na znanju i na tokenu ne mogu razlikovati ovlaštenu osobu i varalicu koja ilegalno stječe token ili znanje ovlaštene osobe, te su oni nezadovoljavajući način postizanja sigurnosnih zahtjeva našeg elektronsko povezanog informacijskog društva. Biometrijska identifikacija odnosi se na prepoznavanje pojedinca na temelju njegovih osebujnih tjelesnih i/ili ponašajnih osobina, biometrijskih identifikatora. Budući da su mnoge tjelesne ili ponašajne osobine karakteristične za svaku osobu, biometrijski identifikatori su inherentno pouzdaniji i sposobniji od tehnika temeljenih na znanju i tokenu u razlikovanju ovlaštene i lažne osobe.

2.1. Biometrijski sustav

Ovisno o kontekstu primjene, biometrijski sustav može raditi u verifikacijskom ili identifikacijskom načinu rada (Jain, et al., 2004.).

[1] Engl. Personal Identification Number – osobni identifikacijski broj

U verifikacijskom načinu rada, sustav provjerava identitet osobe uspoređujući snimljene biometrijske podatke s vlastitim biometrijskim predloškom pohranjenim u bazi podataka sustava. U takvom sustavu, pojedinac koji želi biti prepoznat potvrđuje identitet, obično putem osobnog identifikacijog broja (PIN), korisničkog imena ili pametne kartice, a sustav provodi usporedbu pojedinačnih podataka kako bi se utvrdilo je li zahtjev istinit ili ne. Verifikacija identiteta obično se koristi za pozitivno prepoznavanje, gdje je cilj spriječiti više ljudi da koriste isti identitet.

U identifikacijskom načinu rada, sustav prepoznaje osobu pretraživanjem predložaka svih korisnika u bazi podataka. Stoga, sustav provodi usporedbu jedne ili više osoba kako bi se ustanovio identitet pojedinca bez da pojedinac mora tražiti identitet. Identifikacija je kritična komponenta u primjenama za negativno prepoznavanje gdje sustav utvrđuje da li je osoba ona koja, implicitno ili eksplicitno, negira da jest. Dok tradicionalne metode osobnog prepoznavanja kao što su lozinke, PIN-ovi, ključevi i tokeni mogu poslužiti za pozitivno prepoznavanje, negativno prepoznavanje može se uspostaviti samo putem biometrije.

Biometrijski sustav je u suštini sustav prepoznavanja uzoraka koji čini osobnu identifikaciju utvrđivanjem autentičnosti određene tjelesne ili ponašajne karakteristike koju posjeduje korisnik (Jain, et al., 2000.). Logički se biometrijski sustav može podijeliti na modul za upis podataka i identifikacijski modul. Tijekom upisne faze, biometrijska karakteristika pojedinca prvo se pretražuje pomoću biometrijskog senzora kako bi se dobila digitalna reprezentacija karakteristika. Kako bi se olakšalo podudaranje i smanjili zahtjevi pohrane, digitalni prikaz dodatno se obrađuje pomoću ekstraktora značajki kako bi se stvorio kompaktan, ali izražajan prikaz, nazvan, predložak. Ovisno o primjeni, predložak se može pohraniti u središnju bazu biometrijskog sustava ili može biti zabilježen na magnetnoj kartici ili pametnoj kartici koja se izdaje pojedincu. Tijekom faze prepoznavanja, biometrijski čitač snima karakteristike pojedinca koji se identificira i pretvara ih u digitalni format, koji se dodatno obrađuje pomoću ekstraktora značajki kako bi se dobio isti prikaz kao i predložak. Rezultat prikaza unosi se u značajku koja ga uspoređuje s predloškom kako bi se utvrdio identitet pojedinca.

Svaka ljudska tjelesna ili ponašajna karakteristika može se koristiti kao biometrijska karakteristika ukoliko zadovoljava sljedeće uvjete (Jain, et al., 2004.):

- Univerzalnost; svaka osoba treba imati karakteristiku
- Jedinstvenost; bilo koje dvije osobe trebaju biti dovoljno različite u smislu karakteristike
- Trajnost; karakteristika treba biti dovoljno nepromjenjiva tijekom određenog vremenskog razdoblja
- Prikupljivost; karakteristika se može mjeriti kvantitativno
- Izvedba; odnosi se na dostižnu točnost i brzinu prepoznavanja, resurse potrebne za postizanje željene točnosti i brzine prepoznavanja, kao i operativni i ekološki čimbenici koji utječu na točnost i brzinu
- Prihvatljivost; ukazuje na to koliko su ljudi spremni prihvatiti uporabu određenog biometrijskog identifikatora u svakodnevnom životu
- Obmana; odražava koliko se sustav lako može prevariti pomoću metoda prijevara

2.2. Usporedba različith biometrija

Svaka biometrija ima svoje prednosti i nedostatke, a njen izbor ovisi o primjeni. Ne može se očekivati od jedne biometrije učinkovito zadovoljavanje svih zahtjeva pojedine primjene, tj. niti jedna biometrija nije optimalna. U nastavku će se ukratko objasniti najčešće korištene biometrijske karakteristike (Jain, et al., 2004.).

2.2.1. DNK

Deoksiribonukleinska kiselina (DNK) je jednodimenzionalni jedinstveni kod pojedine individualnosti, dok identični blizanci imaju identične DNK uzorke. Trenutačno se najčešće koristi u kontekstu forenzičkih primjena za prepoznavanje osobe.

Tri pitanja su ograničila korist ove biometrije za druge primjene:

1. Kontaminacija i osjetljivost: lako je ukrasti dio DNK-a nesumnjičavom subjektu koji se može naknadno zloupotrijebiti
2. Problemi automatskog prepoznavanja u stvarnom vremenu: današnja tehnologija za podudaranje DNK-a zahtijeva nezgrapne kemijske metode koje uključuju vještine stručnjaka i nisu namijenjene za online prepoznavanje
3. Pitanje privatnosti: informacije o osjetljivosti osobe na određene bolesti mogu se dobiti iz DNK uzorka, te zbog toga postoji zabrinutost da bi nenamjerno zlostavljanje informacija o genetskom kodu moglo rezultirati diskriminacijom

2.2.2. Uho

Pretpostavlja se da je oblik uha i struktura hrskavog tkiva karakteristično za svaku pojedinu osobu. Pristupi prepoznavanju uha temelje se na poklapanju udaljenosti izbočenih točaka na ušnoj peraji od graničnih mjesta na uhu. Zbog mogućnosti prekrivanja uha kosom, značajke uha nisu prihvaćene kao karakteristične značajke pri utvrđivanju identiteta pojedinca.

2.2.3. Lice

Prepoznavanje lica je nenametljiva metoda, a slike lica su vjerojatno najčešća biometrijska karakteristika koju ljudi koriste za osobno prepoznavanje. Primjena prepoznavanja lica može biti statička, slika lica na kontroliranoj pozadini, i dinamička, pokretna identifikacija lica u nekontroliranim okruženjima.

Najpopularniji pristup prepoznavanja lica temelji se na:

- Lokaciji i obliku značajki lica kao što su oči, obrve, nos, usne i brada i njihovi prostorni odnosi

- Cjelovitoj analizi lica koja predstavlja lice kao ponderiranu kombinaciju brojnih kanonskih oblika lica

Dok izvedba sustava za prepoznavanje lica je zadovoljavajuća, nameću se brojna ograničenja oko prikupljanja slika lica, ponekad zahtijevajući fiksnu i jednostavnu pozadinu ili posebnu rasvjetu. Ti sustavi također imaju poteškoća pri prepoznavanju lica od slika snimljenih iz dva drastično različita kuta gledanja i pod različitim uvjetima osvjetljenja. Još uvijek nije poznato da li je lice, bez ikakvih kontekstualnih informacija, dovoljna osnova za prepoznavanje osobe iz velikog broja identiteta s izuzetno visokom razinom povjerenja. Kako bi sustav prepoznavanja lica dobro funkcionirao u praksi, trebao bi automatski otkriti je li lice prisutno u snimljenoj slici, locirati lice ako postoji, i prepoznati lice s općeg gledišta, tj. iz bilo koje pozicije.

Trodimenzionalni (3D) sustavi prepoznavanja lica su nastali kao pokušaj ispravljanja određenih problema povezanih s dvodimenzionalnim (2D) sustavima. Takvi problemi uključuju degradaciju slike koja dolazi s lošim uvjetima osvjetljenja, kretanjem i lošim poravnavanjem glave. Većina sustava koji koriste 3D tehnologiju upotrebljavaju mjerenja zakrivljenosti lubanje i značajka lica, uz sve površinske značajke. Na primjer, 2D sustav može mjeriti udaljenost između očiju subjekta, dok 3D sustav može mjeriti zakrivljenost očiju u odnosu na most nosa subjekta. 3D sustavi su također sposobniji za hvatanje značajka lica koje se ne mijenjaju tijekom vremena. Takve značajke uključuju most nosa i očne šupljine. Nažalost, ovi moderni sustavi ne mogu ispraviti sve probleme povezane s 2D sustavom kao što je prepoznavanje osobe koja nosi periku, lažni nos ili naočale. Drugi nedostatak je visoka cijena kamera potrebnih za 3D snimanje u odnosu na kamere potrebne za 2D snimanje, a visoka razina točnosti nije dokazana u odnosu na povećanu cijenu. Možda je najveća korist kod sustava prepoznavanja lica činjenica da tehnologija nije nametljiva naspram drugih biometrijskih tehnologija. Međutim, nedostaci mogu biti veliki, lice osobe se može promijeniti, ili može biti promijenjeno, tijekom vremena, obrve se mogu depilirati, nos i jagodice se mogu rekonstruirati, a maske mogu biti nošene preko lica.

2.2.4. Infracrveni termogram lica i tijela

Oblik topline koji zrači ljudskim tijelom je karakterističan za pojedinca i može se snimiti infracrvenom kamerom na nenametljiv način slično kao i obična slika. Tehnologija se može koristiti za skriveno prepoznavanje. Sustav temeljen na termogramu ne zahtijeva kontakt i nije nametljiv, no prikupljanje slika je izazov u nekontroliranim okruženjima gdje su u blizini tijela prisutne površine koje emitiraju toplinu, npr. sobni grijači i ispušne cijevi vozila. Infracrveni senzori su poprilično skupi, što je ujedno i glavni čimbenik koji onemogućava široko rasprostranjeno korištenje termograma.

2.2.5. Otisak prsta

Točnost podudaranja otisaka prstiju pokazala se vrlo visokom. Otisak prsta je uzorak izbočina i udubljenja na površini jagodice prsta, čije se formacije određuju tijekom prvih sedam mjeseci razvoja fetusa. Otisci prstiju identičnih blizanaca su različiti, kao i otisci svakog prsta iste osobe. Metode otiska prstiju su vrlo solidne, pa je naglasak na pronalaženju hardverskog rješenja za što kvalitetnije snimke otiska prsta iz koje se može bolje prilagoditi poboljšana rezolucija, značajke prsta poput oštećenja ili onečišćenja kože. Cijena tehnologije također je znatno smanjena u posljednjih deset godina, što je dovelo do većeg broja implementacija. Također, proizvođači senzora su uspjeli smanjiti veličinu senzora, te troškovi skenera otiska prsta su znatno manji od ostalih biometrijskih tehnologija na tržištu. Skener otiska prsta je pristupačan svima, te se ugrađuje u svakodnevne uređaje koje koristimo poput mobitela i prijenosnih računala.

Točnost trenutačno postojećih sustava prepoznavanja otiska prsta je prikladna za sustave verifikacije, te male i srednje sustave identifikacije koji uključuju nekoliko stotina korisnika. Višestruki otisci prstiju jedne osobe pružaju dodatne informacije kako bi se omogućilo identificiranje osoba u velikim razmjerima. Nedostatak metode može biti zloupotreba pomoću umjetno napravljenih otisaka prstiju, te neprepoznavanje u slučaju deformiranih otisaka prstiju zbog genetskih čimbenika, starenja, okoliša ili profesionalnih razloga kao što su ožiljci i modrice.

2.2.6. Hod

Složena prostorno-vremenska biometrija ponašanja, jedinstven način kretnje jedne osobe. Hod nije vrlo prepoznatljiv, ali je dovoljno profiliran kako bi omogućio verifikaciju u primjenama niske sigurnosti. Hod je ponašajna karakteristika koja je promjenjiva osobito tijekom dugog vremenskog razdoblja, zbog fluktuacija tjelesne težine, većih ozljeda koje uključuju zglobove ili mozak ili zbog opijenosti. Budući da sustavi temeljeni na prepoznavanju hoda koriste videozapis hodajuće osobe za mjerenje nekoliko različitih pokreta svakog artikuliranog zgloba, ulazni podaci su računalno intenzivni i skupi.

2.2.7. Geometrija ruke i prsta

Sustavi prepoznavanja geometrije ruku temelje se na brojnim mjerenjima ljudske ruke, uključujući oblik, veličinu dlana i duljine i širine prstiju. Ekološki čimbenici, kao što je niska ili visoka vlažnost u zraku ili pojedinačne anomalije kao što je suha ili vlažna koža, ne djeluju negativno na točnost verifikacije sustava koji se temelji na geometriji ruku. Izazovi izvlačenja ispravnih informacija geometrije ruku mogu biti nakit pojedinca ili ograničenja u pomicanju ruke poput artritisa. Fizička veličina sustava temeljenog na geometriji ruku je velika i ne može se ugraditi u određene uređaje poput prijenosnih računala. Postoje sustavi za provjeru koji se temelje na mjerenjima samo nekoliko prstiju, najčešće kažiprstu, umjesto cijele ruke. Ti su uređaji manji od onih koji se upotrebljavaju za ručnu geometriju, ali još uvijek mnogo veći od onih koji koriste druge biometrijske podatke, npr. otisak prsta, lice, glas.

2.2.8. Mrežnica oka

Krvne žile mrežnice oka su bogate strukturom i karakteristične su za svakog pojedinca i za svako oko. Jedna je od najsigurnijih biometrijskih karakteristika zbog težine promjene ili repliciranja krvnih žila mrežnice oka. Prikupljanje slika uključuje suradnju

korisnika, uključuje kontakt s okularom i zahtijeva svjestan napor od strane korisnika. Svi ti čimbenici nepovoljno utječu na javnu prihvatljivost ove biometrije. Iako daje najpreciznije rezultate, metoda nije zastupljena zbog svoje visoke cijene i mišljenja javnosti kako je skeniranje mrežnice oka štetno za samo oko.

2.2.9. Šarenica oka

Prstenasta regija oka koju okružuju zjenica i bjeloočnica s obje strane. Složena tekstura šarenice oka nastaje tijekom razvoja fetusa, te sadrži vrlo prepoznatljive informacije korisne za osobno prepoznavanje. Svaka šarenica je jedinstvena i, poput otisaka prstiju, čak i šarenice oka identičnih blizanaca su različite. Tehnologija šarenice oka stječe sve veću popularnost i postala je nasljednik tehnologije mrežnice oka. Jedna od glavnih prednosti ove tehnologije jest činjenica da ne zahtijeva fizički kontakt s uređajem. Sustav pomoću algoritma može lako otkriti umjetne šarenice, npr. kontaktne leće, a otkrivanje staklenog oka moguće je, jer zjenica na umjetnom oku je u nepomičnom stanju dok je kod živog oka podložna konstantnoj kontrakciji i širenju. Cijena tih sustava također je smanjena, što će pomoći u njihovoj povećanoj implementiranosti. Međutim, ti sustavi su i dalje puno skuplji od drugih biometrijskih tehnologija kao što su sustavi prepoznavanja otiska prsta.

2.2.10. Tipkanje

Pretpostavlja se da svaka osoba upotrebljava tipkovnicu na karakterističan način. Ova ponašajna biometrijska karakteristika nije jedinstvena za svakog pojedinca, ali nudi dovoljno svojstvene informacije kako bi se omogućila verifikacija identiteta. Također, tipkanje pojedine osobe koja koristi sustav prepoznavanja temeljen na tipkanju može se nenametljivo pratiti, budući da ta osoba tipkanjem unosi ključne informacije za prepoznavanje.

2.2.11. Miris

Poznato je da svaki objekt izlučuje miris koji je karakterističan za njegov kemijski sastav i to se može upotrijebiti za razlikovanje različitih predmeta. Dašak zraka koji okružuje objekt puše preko niza kemijskih senzora, svaki osjetljiv na određenu skupinu aromatskih spojeva. Komponenta mirisa koju emitira ljudsko, ili bilo koje životinjsko, tijelo razlikuje se od osobe do osobe. Kao nedostatak ove metode može se spomenuti nejasnoća u vezi otkrivanja nepromjenjivosti mirisa tijela usprkos dezodoransnim mirisima i različitim kemijskim sastavima okoline.

2.2.12. Otisak dlana

Dlanovi ljudskih ruku sadrže uzorke specifičnih izbočina i udubljenja poput otisaka prstiju. Budući da skeneri dlana trebaju uhvatiti veliki prostor, oni su kompleksniji i skuplji od senzora otiska prsta. Ljudski dlanovi također sadrže dodatne jedinstvene značajke kao što su glavne linije i bore koje se mogu snimiti čak i sa skenerom niže rezolucije. Konačno, kada se koristi skener s mogućnošću visoke rezolucije otiska dlana, sve značajke dlana, kao što su geometrija ruku, izbočine i udubljenja, glavne linije i bore mogu se kombinirati za izgradnju biometrijskog sustava visoke preciznosti.

2.2.13. Potpis

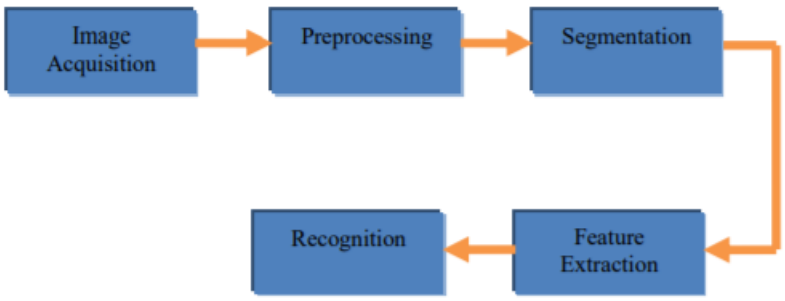
Način na koji osoba potpisuje svoje ime i prezime prihvaća se kao karakteristika tog pojedinca. Iako potpisi zahtijevaju kontakt s instrumentom za pisanje i napor od strane korisnika, prihvaćeni su u vladinim, pravnim i komercijalnim poslovima kao način verifikacije. Potpis je ponašajna biometrija koja se mijenja tijekom određenog vremenskog razdoblja i podliježe tjelesnim i emocionalnim uvjetima potpisnika. Uzastopna pojavljivanja pojedinog potpisa znatno se razlikuju. Također, profesionalni krivotvoritelji potpisa mogu replicirati potpise koji mogu lako prevariti sustav.

2.2.14. Glas

Glas je kombinacija tjelesne i ponašajne biometrijske karakteristike. Značajke pojedinog glasa temelje se na vokalnim traktovima, ustima, nosnim šupljinama i usnicama koji se koriste u proizvodnji zvuka. Tjelesne karakteristike ljudskog govora su nepromjenjive za pojedinca, no ponašajni dio govora osobe se mijenja tijekom vremenskog razdoblja zbog dobi, te medicinskih i emocionalnih stanja. Nedostatak sustava temeljenog na prepoznavanju glasa je osjetljivost značajka govora na niz čimbenika kao što je pozadinska buka.

3. Sustav prepoznavanja lica

Ideja iza sustava prepoznavanja lica je činjenica da svaki pojedinac ima jedinstveno lice. Slično kao i otisak prsta, lice pojedinca ima mnoge strukture i značajke jedinstvene za tu osobu. Automatski sustav prepoznavanja lica temelji se na simetriji lica. Provjera autentičnosti lica i samo prepoznavanje lica izazovni su problemi. Da bi sustavi prepoznavanja lica bili pouzdani, moraju raditi s velikom preciznošću i točnošću. Ako je moguće, nekoliko slika istog pojedinca treba biti uključeno u bazu podataka. Slike snimljene tako da uzimaju u obzir različite izraze lica ili uvjete osvjetljenja omogućuju veću preciznost i točnost sustava u usporedbi sa slučajem u kojem je pohranjena samo jedna slika svakog pojedinca u bazi podataka. Metoda prepoznavanja lica obrađuje snimljenu sliku i uspoređuje ju sa slikama pohranjenim u bazi podataka. Ako se pronađe podudarani predložak, identificiran je pojedinac. U suprotnom, osoba se prijavljuje kao neidentificirana.



Slika 1. Struktura sustava prepoznavanja lica

Izvor: (Bora, et al., 2015., str. 524)

3.1. Detekcija lica

S obzirom na sliku, cilj detekcije lica je utvrditi ima li lica na slici i, ako je prisutno, vratiti mjesto slike i opseg svakog lica.

Postoji mnogo usko povezanih problema otkrivanja lica. Samo neki od problema mogu biti određivanje položaja lica na slici, otkrivanje prisutnosti i lokacije značajki lica, kao što su oči, nos, obrve, nosnice, usta, usne i uši, zatim autentifikacija lica koja potvrđuje identitet pojedinca na slici, kontinuirano praćenje lica i procjenjivanje njegove lokacije i orijentacije u stvarnom vremenu, te prepoznavanje emocionalnog stanja lica. Shodno navedenome, zaključak je da je otkrivanje lica prvi korak u bilo kojem automatiziranom sustavu koji rješava gore navedene probleme (Rizvi, 2011.).

Izazovi povezani s detekcijom lica mogu se pripisati sljedećim čimbenicima (Rizvi, 2011.):

- Poza: slike lica variraju zbog relativne poze prema fotoaparatu, a neke značajke lica poput oka ili nosa mogu postati djelomično ili potpuno prekriveni
- Prisutnost ili odsutnost strukturnih komponenti: značajke lica kao što su brada, brkovi i naočale mogu ili ne moraju biti prisutni i postoji velika varijabilnost među tim komponentama, uključujući oblik, boju i veličinu
- Izraz lica: izraz lica izravno utječe na izgled lica osobe
- Prekrivenost: lica mogu biti djelomično prekrivena drugim objektima. Na slici s grupom ljudi, neka lica mogu djelomično prekriti druga lica
- Uvjeti slike: slike lica izravno se razlikuju zbog različitih rotacija oko optičke osi kamere
- Uvjeti za snimanje: kada se slika formira, čimbenici kao što su osvjetljenje, spektar i intezitet boja, karakteristike fotoaparata, senzor i objektiv, utječu na izgled lica

Metode detekcija lica mogu biti (Rizvi, 2011.):

- Metode temeljene na znanju: metode temeljene na ljudskom znanju o tome što čini tipično lice
- Metode nepromjenjivih značajki: cilj ovih metoda je pronalaženje strukturnih značajki iako se poza, kut gledanja ili uvjeti osvjetljenja razlikuju, a zatim se značajke koriste za pronalaženje lica

- Metode podudaranja predloška: nekoliko standardnih uzoraka lica pohranjuje se za opisivanje lica kao cjeline ili pojedine značajke lica zasebno. Korelacije između ulazne slike i pohranjenih uzoraka izračunavaju se kako bi se pronašlo lice
- Metode temeljene na izgledu: izrađeni model se trenira pomoću skupa slika lica i obuhvaća karakteristične promjenjivosti izgleda lica. Nakon toga, modeli se koriste za pronalaženje lica

3.1.1. Metode temeljene na znanju

U ovom pristupu, metode detekcije lica razvijene su na temelju pravila za opisivanje osobina lica i njihovih odnosa proizašlih iz znanja istraživača ljudskih lica. Na primjer, lice se često pojavljuje na slici s dva oka koji su međusobno simetrični, nosom i ustima. Odnos između značajki može se prikazati njihovom relativnom udaljenošću i položajima. Na ulaznoj slici se prvo izvlače značajke lica, a lica osoba se identificiraju na temelju pravila. Proces verifikacije obično se primjenjuje kako bi se umanjila lažna detekcija. Problem s ovim pristupom je poteškoća u prevođenju ljudskog znanja u dobro definirana pravila. Ako su pravila detaljna, metoda možda neće uspjeti otkriti lica koja ne prolaze sva pravila. Ako su pravila preopćenita, metoda može pronaći mnogo lažno pozitivnih lica. Štoviše, teško je nadograditi ovaj pristup za detekciju lica na detekciju lica u različitim pozama jer je izazov nabrojiti sve moguće slučajeve. S druge strane, metoda dobro funkcionira u otkrivanju frontalnih lica na čistim pozadinama.

3.1.2. Metode nepromjenjivih značajki

Temeljna pretpostavka metode se bazira na shvaćanju da ljudi mogu bez napora detektirati lica i objekte u različitim pozama i uvjetima osvjetljenja, pa stoga moraju postojati svojstva ili značajke koje su nepromjenjive nad tim varijablama. Predložene su brojne metode detekcije značajki lica, a zatim detekcije prisutnosti lica. Značajke lica kao što su obrve, oči, nos, usta i linija kose obično se prepoznaju pomoću detektora ruba. Na temelju prepoznatih značajki izrađuje se statistički model koji opisuje njihove odnose i

provjerava postojanje lica. Problem s metodama temeljenim na nepromjenjivim značajkama je osjetljivost značajka slike na oštećenje koje može uzrokovati osvjetljenje, šum ili djelomična prekrivenost slike, dok sjene mogu uzrokovati brojne lažne rubove koji grupiranje značajki čine beskorisnim.

3.1.3. Metode podudaranja predloška

U metodama podudaranja predloška, standardni uzorak lica, obično frontalni, ručno je unaprijed definiran pomoću funkcije. S obzirom na ulaznu sliku, vrijednosti korelacije standardnih uzoraka se izračunavaju za konture lica, oči, nosa i usta. Postojanje lica određuje se na temelju vrijednosti korelacije. Ova metoda ima prednost u tome što je jednostavna za implementaciju. Međutim, metoda je nedovoljna za detekciju lica jer se ne može učinkovito nositi s razlikama u razmjeru, pozama i obliku.

3.1.4. Metode temeljene na izgledu

Suprotno od metoda podudaranja predloška gdje su predlošci unaprijed definirani od strane stručnjaka, predlošci u metodama temeljenim na izgledu naučeni su iz primjera na slikama. Općenito, metode temeljene na izgledu oslanjaju se na statističku analizu i strojno učenje kako bi se pronašle relevantne osobine lica i ne-lica na slikama. Naučene karakteristike su u obliku distribucijskih modela ili diskriminantnih funkcija koji se koriste za detekciju lica. Također, smanjenje dimenzija slika se obično provodi radi učinkovitosti računanja i učinkovitosti detekcije.

3.2. Prepoznavanje lica

U svom radu, Hubel i Wiesel (2012.) pokazali su da naš mozak ima specijalizirane živčane stanice koje reagiraju na specifične lokalne značajke prizora, kao što su linije, rubovi, kutovi ili kretnje. Budući da ne vidimo svijet oko nas kao rasute dijelove, naš

vizualni korteks mora nekako iskombinirati različite izvore informacija u korisne uzorke. Automatsko prepoznavanje lica se zapravo svodi na izdvajanje značajki iz slika, slažući ih u funkcionalnu reprezentaciju i izvršavanje određene vrste klasifikacije nad njima.

Prepoznavanje lica na temelju geometrijskih značajki lica vjerojatno je najinstinktivniji pristup prepoznavanju lica. Jedan od prvih automatiziranih sustava prepoznavanja lica opisan je u radu Kanade (1973.); točke za označavanje položaja očiju, ušiju i nosa su korišteni za izgradnju vektora značajki, udaljenost između točaka i kut između njih. Prepoznavanje je provedeno računanjem euklidske udaljenosti između vektora značajki testne i referentne slike. Na metodu ne utječu promjene u osvjetljenju, ali ima veliki nedostatak u preciznom obilježavanju točaka značajki koje su čak i sa najnovijim algoritmima kompliciran. U svom radu, Brunelli i Poggio (1992.) upotrijebili su dvadesetdvodimenzionalni (22D) vektor značajki koji je u istraživanjima na velikim skupovima podataka pokazao da same geometrijske značajke ne mogu nositi dovoljno informacija za prepoznavanje lica.

Glavni cilj metoda smanjenja dimenzionalnosti je smanjenje dimenzija uklanjanjem suvišnih i ovisnih značajki pretvarajući značajke iz više-dimenzionalnog prostora u niže-dimenzionalni prostor. Postoje dva glavna pristupa tehnikama smanjenja dimenzionalnosti, nenadzirani pristup (Analiza glavnih komponenti) i nadzirani pristup (Linearna analiza različitih). U nenadziranom pristupu, nema potrebe za označavanjem razreda podataka. Dok u nadziranom pristupu, tehnike smanjenja dimenzionalnosti uzimaju u obzir razredne oznake.

3.2.1. Analiza glavnih komponenti

Metoda svojstvenih lica[2], koju su u svom radu Turk i Pentland (1991.) opisali, pronalazi glavne komponente regija lica, tretirajući sliku kao točku ili vektor u više-dimenzionalnom prostoru. Kako bi klasifikacija bila što jednostavnija uzima se slika iz niže-dimenzionalnog prostora. Prostor se pronalazi pomoću analize glavnih komponenti,

[2] Eigenfaces

koja identificira vektore s maksimalnom varijancom. Pomoću toga se smanjuje broj izračuna, te također čuva memorija potrebna za analizu. Svaka slika u skupu za treniranje je predstavljena kao linearna kombinacija ponderiranih vektora svojstva nazvani eigenface[3]. Vektori svojstva su dobiveni iz kovarijancijske matrice skupa slika za treniranje, a težinka vrijednost je dobivena nakon odabira skupa najrelevantnijih lica svojstva. Slike koje sustav za prepoznavanje lica vidi, spremaju se kao ponderirani skupovi koji opisuju doprinos svakog lica svojstva na pojedinu sliku. Kad se novo lice prezentira sustavu za klasifikaciju, težinske vrijednosti lica se pronalaze projekcijom slike na skup lica svojstva. To pruža skup težinskih vrijednosti koji opisuje prosječno lice. Težinske vrijednosti se zatim klasificiraju prema skupu ponderiranih slika koji se nalazi u sustavu kako bi se pronašlo najbliže podudaranje. Metodom najbližeg susjeda se pronalazi euklidska udaljenost između dva vektora, gdje se najmanja udaljenost može klasificirati kao najbliža (Turk i Pentland, 1991.).

3.2.2. Linearna analiza različitih

Uzimajući u obzir sve podatke, analiza glavnih komponenti će izračunati vektor koji ima najveću varijancu povezanu s njim dok će linearna analiza različitih izračunati vektor koji najbolje razlikuje dva razreda.

Metoda linearna analiza različitih razvijena je kako bi se značajke transformirale u niže-dimenzionalni prostor koji maksimizira omjer razlika među razredima, te ujedno minimizira omjer razlika unutar razreda, čime se jamči maksimalna razdvojenost razreda. Postoje dvije vrste metoda linearne analize različitih za manipuliranje razredima; razredno-ovisna i razredno-neovisna (Tharwat, et al., 2017.). U razredno-ovisnoj metodi, izračunava se jedan zasebni niže-dimenzionalni prostor za projekciju podataka svakog razreda, dok se u razredno-neovisnoj metodi svaki razred smatra različitim u odnosu na druge razrede. Cilj metode je projekcija matrice izvornih podataka na niže-dimenzionalni prostor. Da bi se taj cilj postigao, potrebno je izvršiti tri koraka. Prvi korak je računanje razdvojenosti među razredima, tj. udaljenost između srednjih vrijednosti različitih razreda,

[3] Engl. lice svojstva

što se naziva među-razrednom varijancom ili među-razredna matrica. Drugi korak je računanje udaljenosti između srednje vrijednosti i uzoraka svakog razreda, što se naziva unutar-razredna varijanca ili unutar-razredna matrica. I treći korak je projekcija niže-dimenzionalnog prostora koji maksimizira među-razrednu varijancu i minimizira unutar-razrednu varijancu.

Zatim su se pojavile različite metode za izvlačenje lokalnih značajki. Kako bi se izbjegli visoko-dimenzionalni ulazni podaci, opisane su samo lokalne regije slike što je dovelo do veće otpornosti izvučenih značajki na djelomičnu prekrivenost, osvjetljenje i male veličine uzorka. Metode koji se koriste za izvlačenje lokalnih značajki su gabor valovi, transformacija diskretnih kosinusa i lokalni binarni uzorci.

3.2.3. Lokalni binarni uzorci

Lokalni binarni uzorci[4] (u daljenjem tekstu; LBP) je vrlo učinkovit operator teksture koji označava piksele slike određivanjem granice svakog susjednog piksela i rezultat je promatran kao binarni broj. Zbog svoje diskriminativne snage i računalne jednostavnosti, LBP operator teksture postao je popularan pristup u raznim primjenama. Može se promatrati kao jedinstveni pristup tradicionalnim statističkim i strukturnim modelima analize teksture. Možda najvažnije svojstvo LBP operatora u primjenama u stvarnom svijetu je njegova otpornost prema monotoničnim promjenama sivih tonova uzrokovanih, na primjer, varijacijama u osvjetljenju. Drugo važno svojstvo je njegova računalna jednostavnost, što omogućuje analizu slika u izazovnim okolnostima stvarnog vremena.

Osnovna metodologija za opis lica temeljen na LBP je sljedeća; slika lica je podijeljena na lokalne regije i LBP opisnici teksture su izdvojeni iz svake regije samostalno. Opisnici se zatim spajaju kako bi oblikovali globalni opis lica (Ahonen, et al., 2006.).

[4] Local Binary Patterns

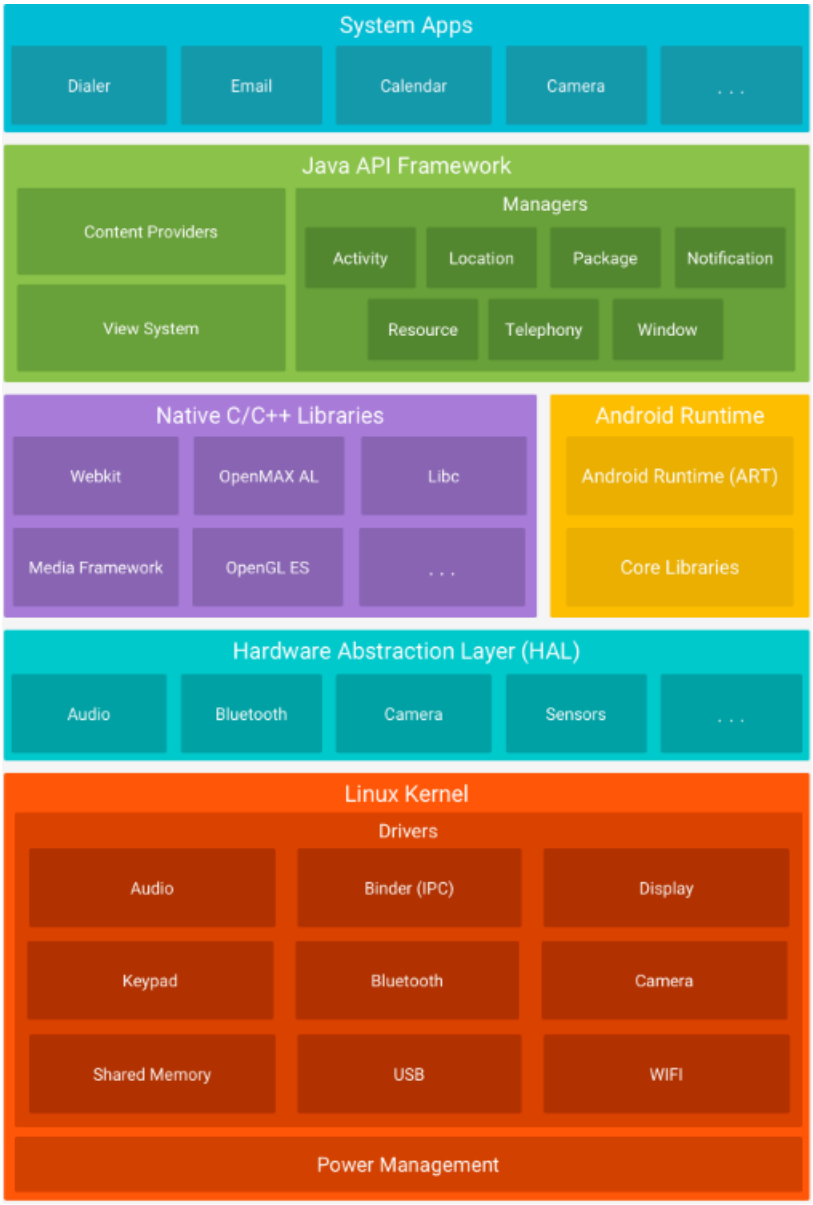
U LBP pristupu za klasifikaciju teksture, pojavljivanje LBP kodova na slici se skuplja u histogram. Klasifikacija se zatim provodi izračunavanjem jednostavnog histograma sličnosti. S obzirom na slične pristupe prikazivanja lica gdje dolazi do gubitka prostornih informacija, stvorila se potreba za zapisivanjem informacija o teksturi uz istodobno zadržavanje njihovih lokacija. Jedan od načina da se to postigne jest korištenje LBP opisnika teksture za izgradnju nekoliko lokalnih opisa lica i njihova kombinacija u globalni opis. Histogram učinkovito opisuje lice na tri različite lokalne razine; LBP oznake za histogram sadrže podatke o uzorcima na razini piksela, oznake se zbrajaju na malom području kako bi se dobile informacije na regionalnoj razini i regionalni histogrami su spojeni kako bi se izgradio globalni opis lica. Metode temeljene na lokalnim značajkama otpornije su na promjene u pozama ili osvjetljenju od metoda temeljnih na izgledu.

4. Razvojni alati mobilne aplikacije za prepoznavanje lica

Cilj rada je pronalazak načina koji omogućava implementaciju aplikacije prepoznavanja lica na Android mobilni uređaj. Stoga, prvi korak je pronalazak rješenja koji omogućuje primjenjivost pristupa prepoznavanja lica na Android platformi. Rješenje je pronađeno u korištenju biblioteke otvorenog koda dizajnirane za aplikacije računalnog vida. Biblioteka se zove Open Source Computer Vision Library (OpenCV).

4.1. Android platforma

Android je softver otvorenog koda temeljen na Linuxu, stvoren za široki niz uređaja. Projektiran je u obliku softvera koji sadrži aplikacije, operativni sustav, izvršno okruženje, middleware, usluge i biblioteke. Svaki stogni sloj i odgovarajući elementi unutar svakog sloja čvrsto su integrirani i pažljivo podešeni kako bi pružili optimalno okruženje za razvoj aplikacija i izvršenje za mobilne uređaje.



Slika 2. Arhitektura Android platforme

Izvor: https://developer.android.com/guide/platform/

4.1.1 Linux jezgra[5]

Postavljen na dnu softverskog stoga Androida, Linux jezgra pruža razinu apstrakcije između hardver uređaja i gornjih slojeva softverskog stoga Androida. Jezgra

[5] Linux Kernel

nudi preventivnu višezadaćnost, usluge niske razine osnovnog sustava kao što su memorija, proces i upravljanje napajanjem uz osiguranje mrežnog stoga i upravljačkih hardverskih programa kao što su zaslon uređaja, Wi-Fi i audio. Korištenje Linux jezgre Androidu omogućuje iskorištavanje ključnih sigurnosnih značajki i omogućava proizvođačima uređaja razvijanje hardverskih upravljačkih programa.

4.1.2. Sloj hardverske apstrakcije[6]

Osigurava standardna sučelja koja otkrivaju mogućnosti hardverskih uređaja višoj razini Java API[7]. Sloj hardverske apstrakcije se sastoji od više modula biblioteka, od kojih svaki implementira sučelje za određenu vrstu hardverske komponente, kao što je kamera ili audio modul. Kada API želi pristupiti hardveru uređaja, Android sustav učitava modul biblioteke za tu hardversku komponentu.

4.1.3. Android izvršitelj[8]

Linux jezgra omogućuje višezadaćnu izvršnu okolinu koja omogućuje istodobno izvršavanje višestrukih procesa. Svaka Android aplikacija se izvršava kao proces izravno na Linux jezgri koji se pokreće u svojem primjeru Android izvršitelja. Učinkovitiji je od standardnog Java virtualnog stroja u smislu korištenja memorije, a posebno je dizajniran kako bi omogućio učinkovito pokretanje više instanci unutar resursnih ograničenja mobilnog uređaja.

Pokretanje aplikacija u virtualnim strojevima pruža niz prednosti. Aplikacije su u zaštićenoj okolini kako bi im se ograničilo štetno djelovanje, namjerno ili nenamjerno, na operativni sustav ili druge aplikacije i zabranjen im je izravan pristup hardveru uređaja. Kako bi se aplikacijski kod izvršio unutar Android izvršitelja, mora se pretvoriti iz

[6] Hardware Abstraction Layer (HAL)
[7] Application Programming Interface – sučelje za programiranje aplikacija
[8] Android Runtime

standardnih Java klasnih datoteka u .dex[9] format. Takav format datoteke ima manji memorijski otisak od standardnog Java bytecode-a[10].

Android osnovne biblioteke sadrže tri glavne kategorije:

- Specifične biblioteke Android izvršitelja: skup biblioteka korištenih pretežno za interakciju izravno s primjerom Android izvršitelja
- Java interoperabilne biblioteke: Android aplikacije uglavnom su razvijene pomoću Java programskog jezika. Standardno okruženje Java razvoja uključuje širok raspon klasa koje su sadržane u osnovnim bibliotekama Java izvršitelja. Biblioteke pružaju podršku za zadatke kao što su rukovanje s nizom, umrežavanje i manipuliranje datotekama
- Android biblioteke: obuhvaćaju biblioteke temeljene na Javi koje su specifične za Android razvoj. Primjeri biblioteka uključuju biblioteke aplikacijskog okvira koje olakšavaju izgradnju korisničkog sučelja, mogućnost grafičkog crtanja i pristup bazi podataka

4.1.4. Nativne C/C++ biblioteke

Osnovne biblioteke Android izvršitelja temelje se na Javi i pružaju primarna aplikacijska sučelja za programere Android aplikacija. Međutim, osnovne biblioteke zapravo ne obavljaju veliki dio stvarnog posla i zapravo su uglavnom Java omoti oko biblioteka temeljenih na C/C++-u. Na primjer, pri pozivanju android.opengl biblioteke za crtanje 3D grafike na zaslonu uređaja, biblioteka zapravo upućuje pozive OpenGL ES C++ biblioteci koja, zauzvrat, radi s osnovnom Linux jezgrom za obavljanje zadataka crtanja. C/C++ biblioteke su uključene kako bi se ispunio širok i raznolik raspon funkcija, uključujući 2D i 3D grafičko crtanje, SSL[11] komunikacija, SQLite[12] upravljanje bazom

[9] Engl. Dalvik Executable - proces virtualnog stroja u Android operativnom sustavu koji izvršava aplikacije napisane za Android
[10] Jezik na koji se Java izvorni kod sastavlja i kojeg Java virtualni stroj razumije
[11] Secure Sockets Layer - standardna sigurnosna tehnologija za uspostavljanje enkriptirane veze između web poslužitelja i preglednika.
[12] Sustav za upravljanje relacijskim bazama podataka sadržane u C programskoj biblioteci

podataka, audio i video reprodukcija, bitmapa[13] i vektor renderiranja fontova, podsustav zaslona i upravljanje grafičkim slojem i implementacija standardne biblioteke sustava C. U praksi, Android razvojni programer aplikacija pristupa tim bibliotekama isključivo putem aplikacijskog sučelja osnovne Android biblioteke temeljene na Javi. U slučaju da je potreban izravni pristup tim bibliotekama, to se može postići pomoću Android nativne razvojne opreme, čiji je cilj pozvati nativne metode programskih jezika, kao što su C i C++, unutar Java koda pomoću Java nativnog sučelja.

4.1.5. Java API okvir

Skup usluga koje zajednički čine okruženje u kojem se pokreću i upravljaju Android aplikacije. Ovaj okvir implementira koncept da su Android aplikacije sastavljene od recikliranih, izmjenjivih i zamjenjivih komponenti.

Ovi API-jevi oblikuju blokove za izgradnju aplikacija na Androidu olakšavanjem ponovne uporabe osnovnih, modularnih komponenti sustava i usluga, a uključuju sljedeće:

- Upravitelj aktivnosti - upravlja svim aspektima životnog ciklusa aplikacije i stanjima aktivnosti
- Davatelji sadržaja - omogućuje aplikacijama objavljivanje i dijeljenje podataka s drugim aplikacijama
- Upravitelj resursa - omogućuje pristup ugrađenim resursima koji nisu kodovi, kao što su nizovi, postavke boja i izgled korisničkog sučelja
- Upravitelj obavijesti - omogućuje aplikacijama prikazivanje upozorenja i obavijesti korisniku
- Prikaz sustava - proširivi skup prikaza koji se upotrebljava za stvaranje korisničkih sučelja aplikacije
- Upravitelj paketa - sustav pomoću kojeg aplikacije mogu saznati informacije o drugim aplikacijama koje su trenutno instalirane na uređaju

[13] Vrsta memorijske organizacije ili format slikovnih datoteka koji se koristi za pohranu digitalnih slika

- Upravitelj telefona - pruža aplikaciji informacije o dostupnim telefononskim uslugama uređaja, kao što su podaci o statusu i pretplati
- Upravitelj lokacije - pruža pristup uslugama lokacije koji aplikaciji omogućuju primanje ažuriranja o promjenama lokacije

4.1.6. Aplikacije sustava

Android dolazi s nizom osnovnih aplikacija za e-poštu, SMS poruke, kalendare, pregledavanje interneta, kontakte i još mnogo toga. Uključene aplikacije nemaju poseban status među aplikacijama koje korisnik odabere za instalaciju. Što znači, bilo koja aplikacija može postati zadani web preglednik, preglednik SMS poruka ili čak zadana tipkovnica. Postoje neke iznimke, kao što je aplikacija Postavke sustava koju nije moguće zamijeniti s nekom drugom aplikacijom.

4.2. Struktura Android aplikacije

Android aplikacije se mogu napisati pomoću Kotlin, Java i C++ jezika. Alati Android SDK-a[14] prevode kod zajedno s podacima i resursnim datotekama u APK[15], Android paket koji je arhivirana datoteka s .apk sufiksom. Jedna APK datoteka sadrži sve sadržaje Android aplikacije te je ujedno datoteka koju Android uređaji upotrebljavaju za instalaciju aplikacije.

Svaka Android aplikacija se pokreće u vlastitoj zaštićenoj okolini, zaštićena sljedećim sigurnosnim značajkama Android sustava:

- Android operativni sustav je višekorisnički Linux sustav u kojem je svaka aplikacija drukčiji korisnik

[14] Engl. Software Development Kit - skup programa koji programeri koriste za pisanje aplikacijskih programa
[15] Android Package Kit

- Prema zadanim postavkama, sustav svakoj aplikaciji dodjeljuje jedinstveni Linux korisnički ID[16], koji nije poznat aplikacijama. Sustav postavlja dopuštenja za sve datoteke u aplikaciji tako da im samo dodijeljeni korisnički ID može pristupiti
- Svaki proces ima svoj virtualni stroj, tako da se kod aplikacije izvodi odvojeno od drugih aplikacija
- Prema zadanim postavkama, svaka se aplikacija pokreće u vlastitom Linux procesu. Android sustav pokreće proces kada treba izvršiti bilo koju komponentu aplikacije, a zatim isključuje proces kada više nije potreban ili kada sustav mora osloboditi memoriju za druge aplikacije

Komponente aplikacije su sastavni dijelovi Android aplikacije. Svaka komponenta je ulazna točka putem koje sustav ili korisnik može ući u aplikaciju. Pojedine komponente su ovisne jedne o drugima.

Postoje četiri različite vrste komponenti aplikacije:

- Aktivnosti
- Usluge
- Prijemnici emitiranja
- Pružatelji sadržaja

Svaki od njih ima različitu svrhu i različiti životni ciklus koji definira kako se komponenta stvara i uništava.

4.2.1. Aktivnosti

Ulazna točka za interakciju s korisnikom. Ona predstavlja zaslon s korisničkim sučeljem. Na primjer, aplikacija e-pošte može imati više aktivnosti, jedna koja prikazuje popis novih poruka, druga koja služi za sastavljanje e-pošte i treća za čitanje e-pošte.

[16] Engl. Identifier - identifikator

Iako su aktivnosti zajednički povezane kako bi stvorile kohezivno iskustvo korisnika u aplikaciji za e-poštu, svaka od njih je neovisna o drugima. Što znači, druga aplikacija može inicirati bilo koju od navedenih aktivnosti ako je aplikacija e-pošte dopusti. Na primjer, aplikacija kamere može pokrenuti aktivnost u aplikaciji e-pošte koja sastavlja novu poštu kako bi korisniku omogućilo dijeljenje slike.

Aktivnost olakšava sljedeće ključne interakcije između sustava i aplikacije:

- Praćenje onoga o čemu korisnik trenutno brine, što je na zaslonu, kako bi se osiguralo da sustav nastavi s procesom koji obavlja aktivnost
- Pretpostavljanje da prethodno korišteni procesi sadrže stvari kojima se korisnik može vratiti, zaustavljene aktivnosti, te posvećuje prioritet pri održavanju tih procesa
- Pomoć aplikaciji pri rukovanju ubijanja procesa kako bi se korisnik mogao vratiti prethodnom stanju obnovljenih aktivnosti
- Pružanje načina aplikacijama da implementiraju međusobne korisničke tokove i da sustav koordinira te tokove, kao što je dijeljenje

4.2.2. Usluge

Općenita ulazna točka za čuvanje aplikacije koja se izvodi u pozadini iz raznih razloga. To je komponenta koja radi u pozadini kako bi obavljala dugotrajne operacije ili obavljala posao za udaljene procese. Usluga ne pruža korisničko sučelje. Na primjer, usluga može reproducirati glazbu u pozadini dok korisnik koristi drugu aplikaciju ili može dohvaćati podatke putem mreže bez blokiranja interakcije korisnika s aktivnošću. Sinkronizacija podataka u pozadini ili reprodukcija glazbe predstavljaju dvije različite vrste pokrenutih usluga koje mijenjaju način na koji ih sustav obrađuje. Reprodukcija glazbe je nešto čega je korisnik izravno svjestan. Aplikacija govori sustavu da želi biti u prvom planu pomoću obavijesti koje prikazuje korisniku. U ovom slučaju sustav zna da bi trebao pokušati zadržati izvođenje te usluge, jer u protivnome korisnik će biti nezadovoljan. Redovita pozadinska usluga nije nešto čime je korisnik izravno svjestan, tako da sustav ima više slobode u upravljanju njegovim procesom. Sustav može proces ukloniti, a zatim

ga ponovo pokrenuti kasnije, ako treba radnu memoriju za stvari koje su od neposredne važnosti za korisnika.

4.2.3. Prijemnici emitiranja

Komponenta koja omogućuje sustavu isporuku događaja aplikaciji izvan redovnog korisničkog toka, omogućujući aplikaciji da odgovori na obavijesti emitiranja preko cijelog sustava. Na primjer, aplikacija može zakazati alarm kako bi prikazao obavijest korisniku o nadolazećem događaju. Isporukom alarma prijemniku emitiranja aplikacije, prestaje potreba prikazivanja aplikacije dok se alarm ne isključi. Mnoga emitiranja potječu iz sustava, primjerice, emitiranje koje prikazuje da je zaslon isključen, baterija slaba ili slika snimljena. Aplikacije također mogu pokrenuti emitiranja, primjerice, kako bi druge aplikacije saznale da su određeni podaci preuzeti na uređaj i dostupni za upotrebu.

4.2.4. Pružatelji sadržaja

Upravlja zajedničkim skupom aplikacijskih podataka koji se mogu pohraniti u datotečnom sustavu, u SQLite bazi podataka, na webu ili na bilo kojoj drugoj lokaciji za pohranu kojoj aplikacija može pristupiti. Putem pružatelja sadržaja ostale aplikacije mogu uputiti upit ili izmijeniti podatke ako im pružatelj sadržaja to dopusti. Rad s pružateljima sadržaja se obično odvija u jednom od dva sljedeća scenarija; mogućnost implementiranja koda za pristup postojećem pružatelju sadržaja u drugoj aplikaciji ili izrada novog pružatelja sadržaja u aplikaciji za dijeljenje podataka s drugim aplikacijama.

4.3. Android studio

Android Studio je službeno integrirano razvojno okruženje temeljeno na IntelliJ IDEA[17] za razvoj Android aplikacija.

Osim alata za uređivanje koda, Android Studio nudi sljedeće značajke koje poboljšavaju produktivnost prilikom izgradnje Android aplikacija:

- Fleksibilni sustav izgradnje temeljen na Gradle-u[18]
- Emulator s bogatim značajkama
- Jedinstveno okruženje u kojem se mogu razvijati aplikacije za sve Android uređaje
- Instant Run značajka koja omogućuje premještanje promjena u aplikaciju koja se izvodi bez izrade novog APK-a
- Predlošci kodova i GitHub integracija koji pomažu u izradi zajedničkih značajki aplikacija i unos predloška koda
- Opsežni alati za testiranje
- Lint[19] alati za postizanje performansi, upotrebljivosti i kompatibilnosti verzije
- C++ i NDK[20] podrška
- Ugrađena podrška za Google Cloud Platform[21], olakšavajući integraciju Google Cloud Messaging-a[22] i App Engine-a[23]

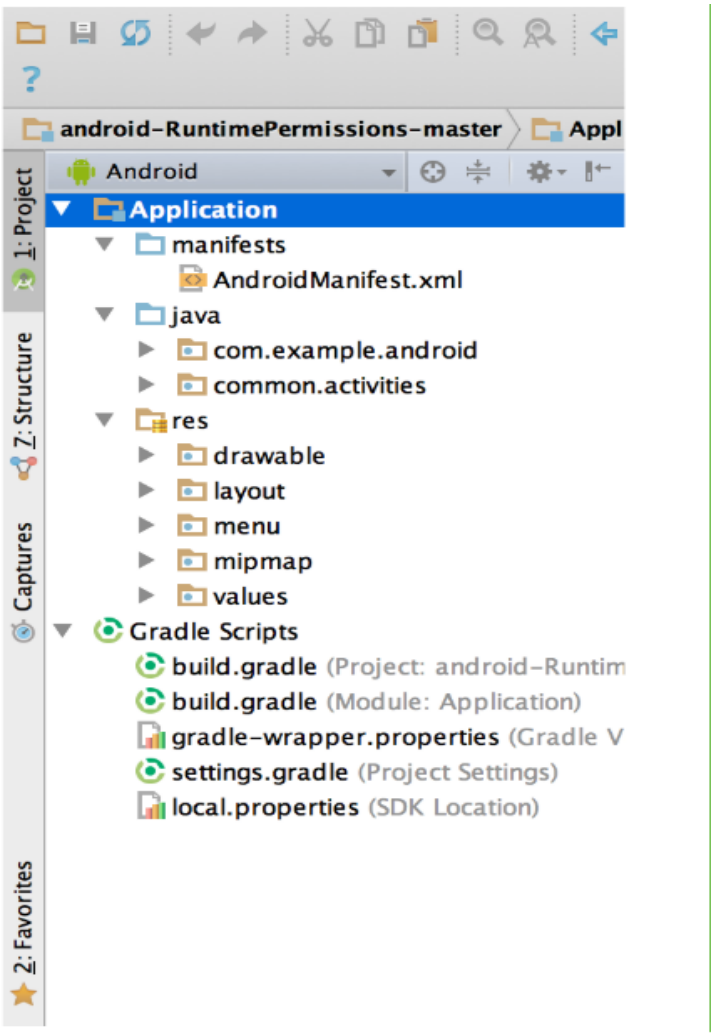
[17] Java integrirano razvojno okruženje za razvoj računalnog softvera
[18] Napredni alat izgradnje za Java virtualni stroj koji upravlja ovisnostima i omogućuje definiranje prilagođene logike izgradnje
[19] Alat za skeniranje Android projekta radi pronalaska potencijalnih grešaka
[20] Engl. Native Development Kit - alat koji omogućuje programiranje u C/C++-u za Android uređaje
[21] Paket usluga računalnog oblaka koji se izvodi na istoj infrastrukturi koju Google koristi interno za svoje proizvode krajnjih korisnika (Google Search, YouTube)
[22] Usluga koja omogućuje slanje podataka s poslužitelja na Android uređaje te primanje poruka s uređaja na istoj vezi
[23] Web okvir i platforma za razvoj u oblaku za razvoj i posluživanje web aplikacija u Google-ovim podatkovnim centrima

4.3.1. Struktura Android projekta

Svaki projekt u Android Studio-u sadrži jedan ili više modula s datotekama izvornog koda i datotekama resursa.

Vrste modula uključuju:

- Module Android aplikacije
- Module biblioteka
- Module Google App Engine-a



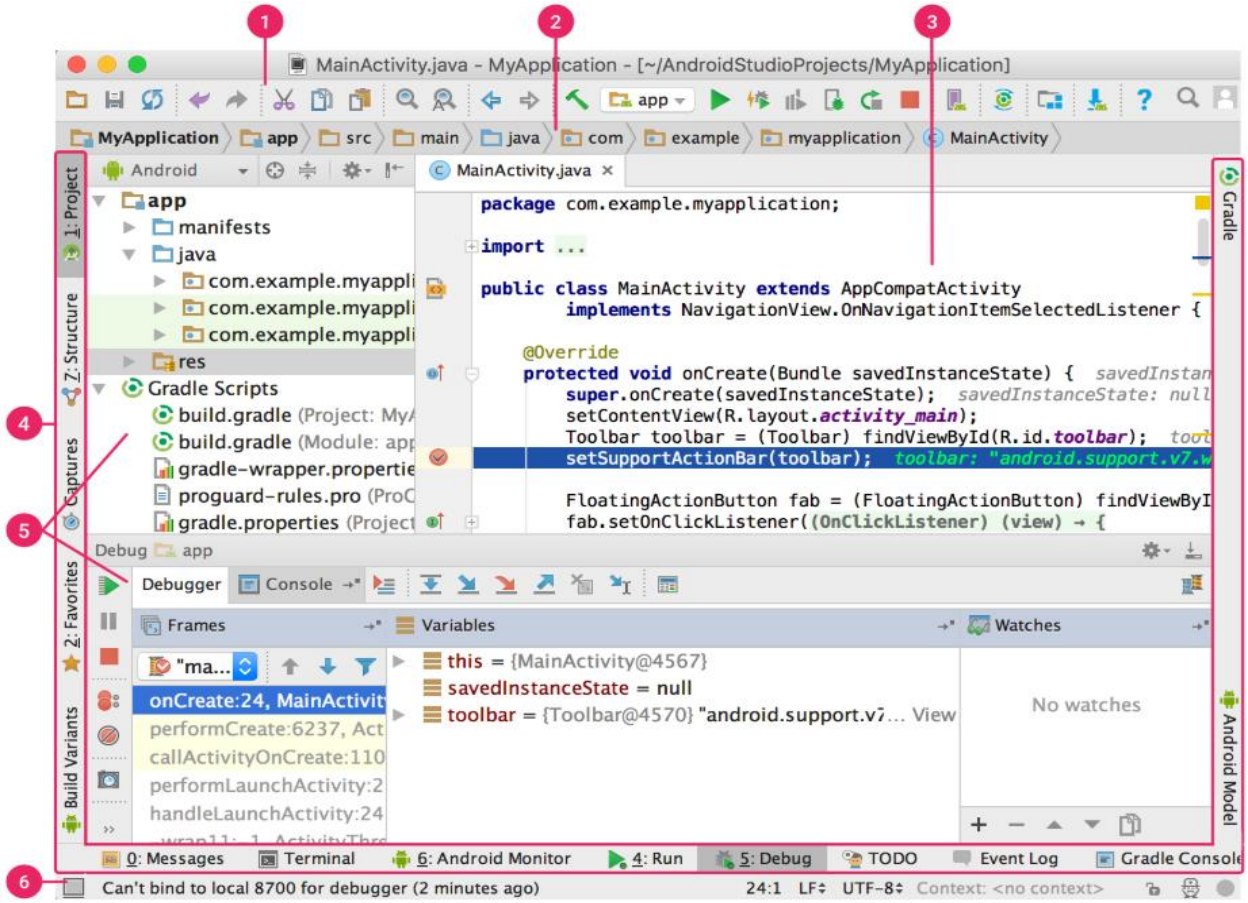
Slika 3. Datoteke projekta u Android pregledu

Izvor: https://developer.android.com/studio/intro/

Sve datoteke za izgradnju aplikacije su vidljive na najvišoj razini pod Gradle Scripts i svaki modul aplikacije sadrži sljedeće mape:

- manifests: Sadrži datoteku AndroidManifest.xml[24]
- java: Sadrži datoteke Java izvornog koda, uključujući i testni kod JUnit[25]
- res: Sadrži sve resurse koji nisu kod, kao što su XML izgledi, nizovi korisničkog sučelja i bitmap slike

4.3.2. Korisničko sučelje



Slika 4. Glavni prozor Android Studio-a

Izvor: https://developer.android.com/studio/intro/

[24] Prikazuje bitne informacije o aplikaciji Android sustavu, informacije koje sustav mora imati prije nego što može pokrenuti bilo koji kod aplikacije
[25] Okvir za testiranje jedinica izvornog koda za Java programski jezik

Objašnjenje Slika 4.:

1. Alatna traka omogućuje širok raspon radnji, uključujući pokretanje aplikacije i pokretanje Android alata.
2. Navigacijska traka pomaže pri kretanju kroz projekt i otvaranje datoteka za uređivanje. Omogućuje kompaktniji prikaz strukture vidljiv u Project pregledu.
3. Prozor urednika je mjesto gdje se piše i mijenja kod. Ovisno o trenutnoj odabranoj vrsti datoteke, uređivač se mijenja. Primjer, prilikom pregledavanja datoteke izgleda, urednik prikazuje uređivač izgleda.
4. Traka prozora alata radi izvan okvira integriranog razvojnog okruženja i sadrži gumbe koji omogućuju proširivanje ili sažimanje pojedinačnih prozora alata.
5. Prozori alata omogućuju pristup specifičnim zadacima kao što su upravljanje projekta, pretraživanje, kontrola verzije.
6. Traka statusa prikazuje status projekta, kao i sva upozorenja ili poruke.

4.4. OpenCV

Open Source Computer Vision Library, OpenCV, je softver biblioteka otvorenog koda za računalnu viziju i strojno učenje. Izgrađen je kako bi osigurao zajedničku infrastrukturu za aplikacije računalnog vida i ubrzao upotrebu strojne percepcije u komercijalnim proizvodima. Budući da sadrži BSD licencu[26], OpenCV olakšava poslovnim subjektima korištenje i izmjenu koda. Izvorno napisan u C++ jeziku, ima C++, Python, Java i MATLAB sučelja i podržava Windows, Linux, Android i Mac OS.

Biblioteka sadrži više od 2.500 optimiziranih algoritama, koji uključuju opsežan skup klasičnih i suvremenih algoritama računalnog vida i strojnog učenja. Algoritmi se mogu koristiti za detekciju i prepoznavanje lica, prepoznavanje objekata, klasificiranje ljudskih akcija u videozapise, praćenje kretnji fotoaparata, praćenje pokretnih objekata, izdvajanje 3D modela objekata, povezivanje slika za dobivanje slike cjelokupnog prizora u visokoj rezoluciji, pronalaženje sličnih slika iz baze podataka, uklanjanje crvenih očiju

[26] Spada u obitelj besplatnih softverskih licenci, sadrži minimalna ograničenja u korištenju i distribuciji pokrivenog softvera

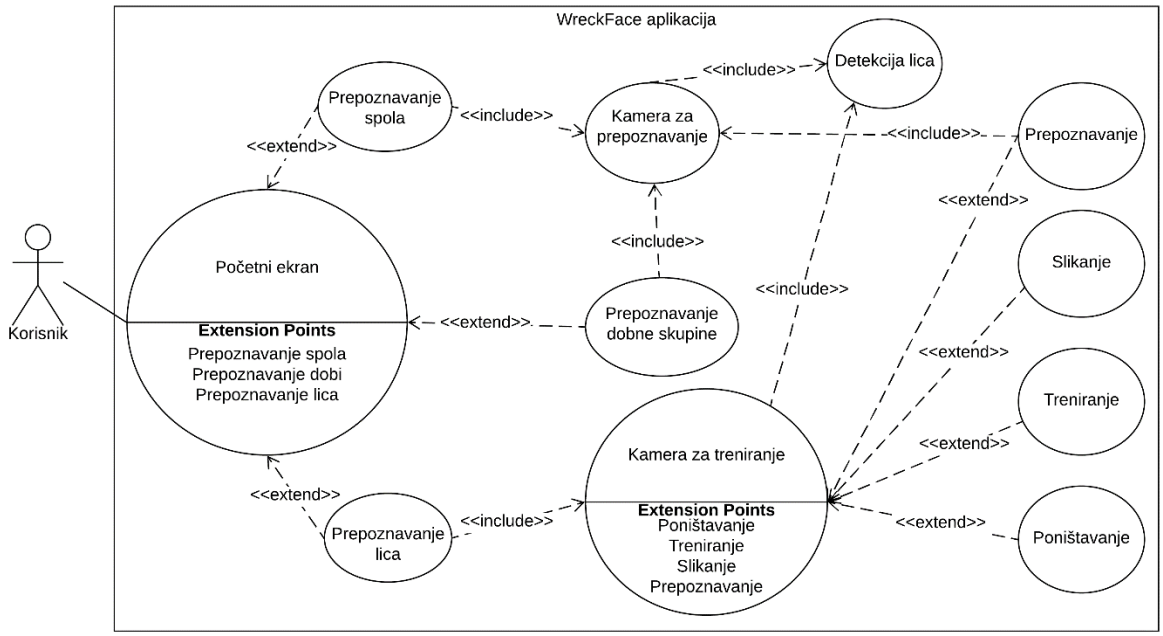
sa slika snimljenih pomoću bljeskalice, praćenje pokreta očiju, prepoznavanje prizora i uspostavljanje markera kako bi ih prekrili proširenom stvarnošću.

5. Programsko rješenje

Prethodna poglavlja su pokazala s kojim problemima se susreće sustav, u ovom slučaju mobilna aplikacija, koji koristi biometrijski identifikator, kao što je lice. Također, navedene su i opisane metode s kojima se omogućuje rješavanje navedenih problema. Cilj programskog rješenja je detekcija svih lica na ulaznoj slici kamere, te na odabranom licu izvršavanje prepoznavanje spola, dobne skupine ili lica u stvarnom vremenu. Za izradu WreckFace-a, mobilne aplikacije za prepoznavanje lica, korišteno je integrirano razvojno okruženje Android Studio, verzija 3.1.4, s pripadajućom OpenCV bibliotekom, verzija 3.4.2, koja nam omogućuje korištenje metoda za detekciju lica, prepoznavanje lica i procesiranje slike kamere. WreckFace mobilna aplikacija je izgrađena i testirana na Samsung S7 mobilnom uređaju.

5.1. Dijagram slučajeva korištenja

Slika 5. prikazuje slučajeve korištenja programskog rješenja u obliku mobilne aplikacije. Korisniku je prikazan početni ekran pri ulasku u mobilnu aplikaciju gdje može odabrati između tri ponuđena izbora, prepoznavanje spola, prepoznavanje dobne skupine i prepoznavanje lica. Odabirom prepoznavanja spola ili dobne skupine, otvara se kamera za prepoznavanje koja ujedno čini i detekciju lica. Nakon što je odabrano jedno lice u kadru kamere, na odabranom licu automatski se vrši prepoznavanje spola ili dobne skupine u stvarnom vremenu. Rezultat prepoznavanja spola ili dobne skupine se ispisuje iznad detektiranog lica na zaslonu uređaja. Odabirom prepoznavanja lica, mobilna aplikacija otvara kameru za treniranje koja sadrži detekciju lica. Na zaslonu se pojavljuju četiri gumba, Poništi, Treniraj, Slikaj i Prepoznaj. Poništavanje omogućuje brisanje svih slika iz memorije uređaja. Treniranje omogućuje treniranje modela prepoznavanja lica nakon odabranog broja slikanih lica. Slikanje služi za slikanje lica, ujedno i za spremanje slikanih lica u memoriju uređaja. Prepoznavanje se izvodi nakon treniranja modela prepoznavanja lica i pokreće novi prozor koji sadrži kameru za prepoznavanje s detekcijom lica gdje se prepoznavanje lica vrši automatski u stvarnom vremenu. Rezultat prepoznavanja lica je, također ispisan iznad detektiranog lica na zaslonu uređaja.



Slika 5. Dijagram slučajeva korištenja WreckFace aplikacije

Izvor: Izrada autora

5.2. Dizajn i funkcionalnost

Mobilna aplikacija sadrži više programskih modula koji su napisani u Java programskom jeziku. Moduli odgovaraju aktivnostima aplikacije. Početni zaslon prikazuje mogućnost izbora tri radnje, prepoznavanje spola, dobne skupine ili lica. Pritiskom na bilo koju željenu radnju, pokreće se odabrana aktivnost. Na početku svake aktivnosti inicijalizira se kamera koja služi za procesiranje slika u stvarnom vremenu. Također, inicijalizacijom aktivnosti koje koriste kameru, učitava se klasifikator zadužen za detekciju lica.

U prepoznavanju spola i dobne skupine, učitavaju se datoteke istreniranog modela koje se koriste za procjenu dobne skupine i spola. Aplikacija podržava istovremenu detekciju svih lica osoba na slici kamere, ali ne podržava istovremeno prepoznavanje svih osoba zbog ograničenih resursa mobilnog uređaja. Odabirom prepoznavanja lica inicijalizira se aktivnost treniranja u kojoj korisnik mora slikati određeni broj slika lica osobe koju želi prepoznati. U aktivnosti treniranja moguće je uzimanje i automatsko

zapisivanje slikanih lica u memoriju uređaja, izrada treniranog modela prepoznavanja lica i automatsko zapisivanje istreniranog klasifikatora značajka lica u memoriju uređaja, brisanje svih slika osobe i istreniranog klasifikatora značajka lica i pokretanje aktivnosti prepoznavanja nakon zadovoljavanja svih prethodno određenih uvjeta. Nakon inicijalizacije aktivnosti prepoznavanja, učitava se prethodno istrenirani klasifikator značajka lica potreban za prepoznavanje lica slikane osobe. U procesu prepoznavanja jedna osoba mora biti u kadru kamere, te se onda automatski inicijalizira proces prepoznavanja spola, dobne skupine ili lica.

Rezultat detekcije lica je vidljiv putem pravokutnog oblika koji je iscrtan oko svakog detektiranog lica na slici kamere, a rezultati prepoznavanja su napisani, sukladno svojoj aktivnosti, iznad pravokutnika koji se pojavljuje oko detektiranog lica. Moguće vrijednosti rezultata prepoznavanja spola su; “MUSKO, ZENSKO”, kod prepoznavanja dobne skupine, moguće vrijednosti rezultata mogu biti; “0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53, 60+”, kod prepoznavanja lica, vrijednost rezultata je zapravo broj najbliže slike koja ima najveću vjerojatnost podudaranja osobi na slici kamere.

5.3. Opis rješenja

OpenCV Manager je Android usluga koja upravlja binarnim OpenCV bibliotekama na uređajima krajnjih korisnika. Prednosti OpenCV Managera su korištenje manje memorije, aplikacije upotrebljavaju iste datoteke i ne čuvaju izvorne biblioteke, optimizacija hardvera za sve podržane platforme, pouzdani izvor OpenCV biblioteke, redovita ažuriranja i popravci programskih pogrešaka. Interakcija između mobilne aplikacije i OpenCV Managera se postiže pomoću apstraktne klase BaseLoaderCallback koja se inicijalizira na samom početku aktivnosti programskog modula.

```
//Veza između aplikacije i OpenCV Manager-a
private BaseLoaderCallback mLoaderCallback = new BaseLoaderCallback( AppContext: this) {
    @Override
    public void onManagerConnected(int status) {
        switch (status) {
            case LoaderCallbackInterface.SUCCESS: {
                Log.i(TAG, msg: "OpenCV loaded successfully");

                (AsyncTask) (voids) → {
                    try {
                        //Učitavanje datoteke klasifikatora iz resursa aplikacije
                        InputStream is = getResources().openRawResource(R.raw.lbpcascade_frontalface);
                        File cascadeDir = getDir( name: "cascade", Context.MODE_PRIVATE);
                        mCascadeFile = new File(cascadeDir, child: "lbpcascade_frontalface.xml");
                        FileOutputStream os = new FileOutputStream(mCascadeFile);

                        byte[] buffer = new byte[4096];
                        int bytesRead;
                        while ((bytesRead = is.read(buffer)) != -1) {
                            os.write(buffer, off: 0, bytesRead);
                        }
                        is.close();
                        os.close();

                        mFaceDetector = new CascadeClassifier(mCascadeFile.getAbsolutePath());
                        if (mFaceDetector.empty()) {
                            Log.e(TAG, msg: "Failed to load cascade classfier");
                            mFaceDetector = null;
                        } else {
                            Log.i(TAG, msg: "Loaded cascade classifier from " + mCascadeFile.getAbsolutePath());
                        }
                        cascadeDir.delete();

                    } catch (IOException e) {
                        e.printStackTrace();
                        Log.e(TAG, msg: "Failed to load cascade. Exception thrown: " + e);
                    }
                    return null;
                }.execute();

                mOpenCvCameraView.enableView();
                break;
            }
            default: {
                super.onManagerConnected(status);
                break;
            }
        }
    }
};
```

Slika 6. Inicijalizacija OpenCV Manager-a u aplikaciji WreckFace

Izvor: Izrada autora

Svaka interakcija s kamerom je obavljena preko OpenCV apstraktne klase CameraBridgeViewBase. Zadaća klase je provođenje interakcije kamere i OpenCV

biblioteke. Glavna odgovornost je kontrola kada kamera može biti omogućena, obrada okvira kamere, poziv eksternog slušatelja radi bilo kakve prilagodbe okvira kamere, te vraćanje rezultata okvira kamere na zaslon mobilnog uređaja. Programski moduli koji koriste kameru implementiraju sučelje CvCameraViewListener i njegove metode onCameraViewStarted, onCameraViewStopped i onCameraFrame, koje omogućuju pokretanje, zaustavljanje i manipuliranje okvirom kamere. Kako bi mogli pistupiti kameri mobilnog uređaja, dodana su prava za pristup u datoteci AndroidManifest.xml.

```
<!--Allow to use a camera-->
<uses-permission android:name="android.permission.CAMERA"/>
<uses-feature android:name="android.hardware.camera" android:required="false"/>
<uses-feature android:name="android.hardware.camera.autofocus" android:required="false"/>
<uses-feature android:name="android.hardware.camera.front" android:required="false"/>
<uses-feature android:name="android.hardware.camera.front.autofocus" android:required="false"/>
```

Slika 7. Dodavanje prava u AndroidManifest.xml datoteci za pristup kameri mobilnog uređaja

Izvor: Izrada autora

Za detekciju lica korištena je OpenCV klasa za detekciju objekata, CascadeClassifier, koja nam omogućuje učitavanje prethodno evaluiranih kaskadnih značajki. Učitavanje istreniranih kaskadnih značajki lica se izvršava u procesu spajanja aplikacije s OpenCV Manager-om, kao što je prikazano na Slika 6. Za detekciju lica upotrebljene su istrenirane LBP značajke lica u XML[27] datoteci.

Kako bi detekcija lica mogla biti uspješna, trenutni okvir kamere u stvarnom vremenu koristi Mat klasu, koja se koristi u svim funkcionalnostima OpenCV 2D računalnog vida. Klasa Mat je zapravo osnovni spremnik slike. Može se prikazati kao klasa s dva dijela podataka. Matrica zaglavlja, koja sadržava informacije kao što je veličina matrice, način korištenja pohrane, adresa pohranjene matrice i pokazivač matrice, koji sadržava vrijednosti piksela.

[27] Engl. Extensible Markup Language - jezik koji definira skup pravila kodiranja dokumenata u obliku koji je čitljiv ljudima i strojevima

Pomoću metode detectMultiScale možemo detektirati lica raznih veličina u datom okviru kamere, te ih spremiti u objekt MatOfRect. Kako bi bili u mogućnosti iscrtati pravokutnik oko svakog pojedinog detektiranog lica na slici, sva detektirana lica se prenose u 2D niz pravokutnika. Iscrtavanje pravokutnika oko detektiranog lica se izvršava pomoću metode rectangle OpenCV modula za obradu slike, Imgproc.

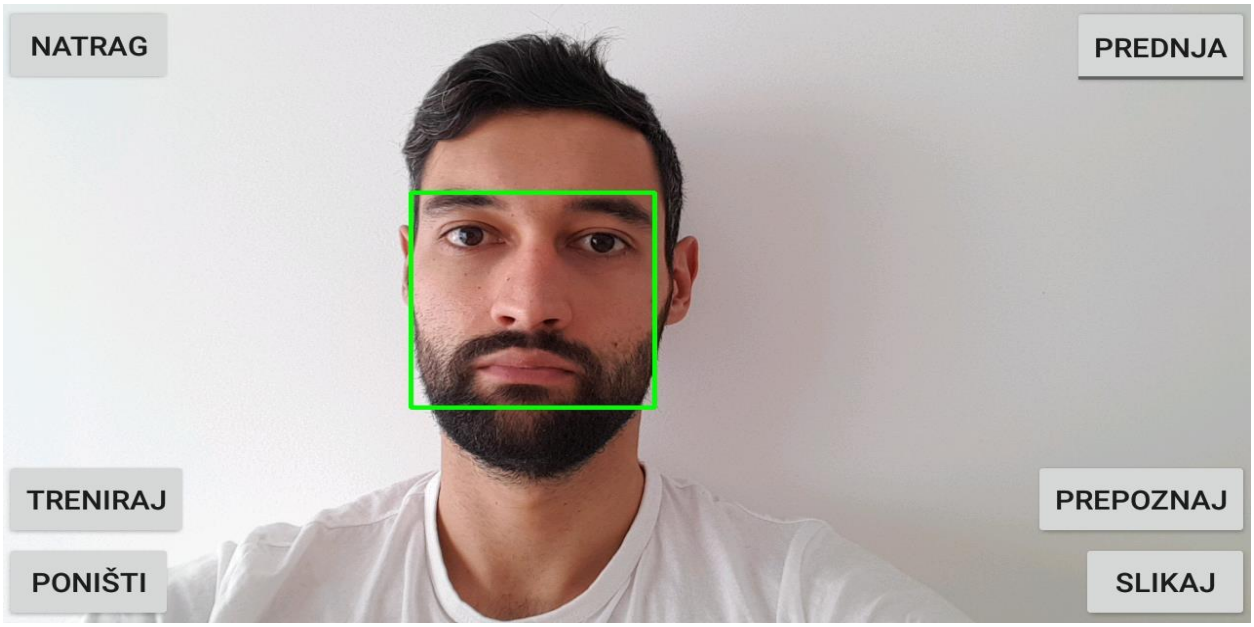
```
MatOfRect faces = new MatOfRect();

//Iskorištavanje klasifikatora za detekciju lica u slici
if (mFaceDetector != null) {
    mFaceDetector.detectMultiScale(mGray, faces, scaleFactor: 1.1, minNeighbors: 5, flags: 2,
            new Size(mAbsoluteFaceSize, mAbsoluteFaceSize), new Size());
} else {
    Log.e(TAG, msg: "Detection is not selected!");
}

//Crtanje pravokutnika oko svakog detektiranog lica u slici
Rect[] facesArray = faces.toArray();
for (int i = 0; i < facesArray.length; i++) {
    Imgproc.rectangle(mRgba, facesArray[i].tl(), facesArray[i].br(), new Scalar(0, 255, 0, 255), thickness: 3);
}
```

Slika 8. Detekcija i iscrtavanje pravokutnika oko lica na slici kamere

Izvor: Izrada autora



Slika 9. Detekcija lica na slici u aplikaciji WreckFace

Izvor: Izrada autora

Za prepoznavanje spola i dobne skupine koristio se istrenirani Caffe[28] model autora Levi i Hassner (Levi i Hassner, 2015.). Model je treniran na Adience kolekciji nefiltriranih lica za spolnu i dobnu klasifikaciju koja sadržava 26.580 slika od ukupno 2.284 subjekata (Eidinger, et al., 2014.).

```
//Učitavanje Caffe modela u duboku neuronsku mrežu
String proto = getPath( file: "deploy_age.prototxt",  context: this);
String weights = getPath( file: "age_net.caffemodel",  context: this);
mAgeNet = Dnn.readNetFromCaffe(proto, weights);

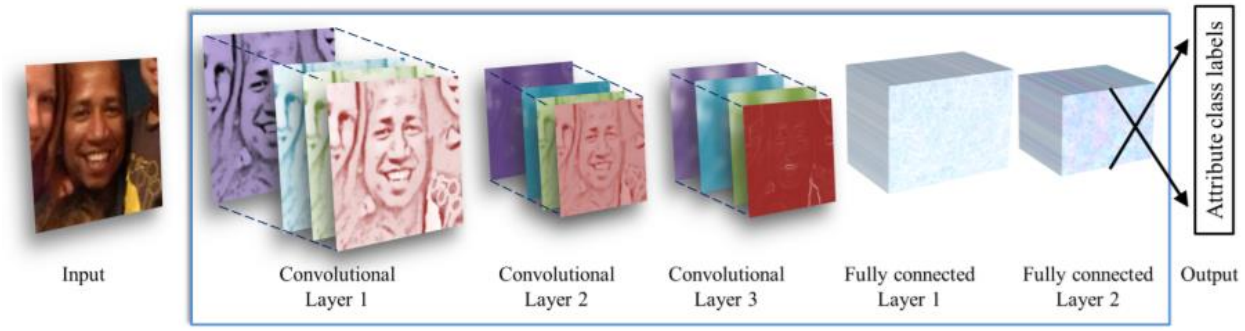
if (mAgeNet.empty()) {
    Log.i(TAG,  msg: "Network loading failed");
} else {
    Log.i(TAG,  msg: "Network loading success");
}
```

Slika 10. Učitavanje Caffe modela za prepoznavanje dobne skupine u duboku neuronsku mrežu

Izvor: Izrada autora

Konvolucijska neuronska mreža se sastoji od ulaznog i izlaznog sloja, kao i višestrukih skrivenih slojeva. Skriveni slojevi konvolucijske neuronske mreže obično se sastoje od konvolucijskih slojeva, slojeva sažimanja, potpuno povezanih slojeva i slojeva normalizacije. Konvolucijski sloj se sastoji od filtera koji se mogu smatrati identifikatorom značajki. Također, filter se sastoji od određenog niza brojeva koji se nazivaju dubine ili parametri. Dubina filtera mora biti iste veličine kao i dubina ulaznog sloja. Konvolucija se događa filtriranjem slike od početka do kraja. Rezultat se dobiva množenjem vrijednosti filtera s originalnom vrijednošću piksela ulaznog sloja. Prije izlaznog sloja, zadnja izlazna vrijednost konvolucijskog sloja se prenosi u potpuno povezani sloj koji izbacuje N dimenzionalni vektor gdje je N broj klasa između koje se odabire konačni rezultat.

[28] Engl. Convolutional Architecture for Fast Feature Embedding - okvir otvorenog koda za duboko učenje, sadrži BSD licencu



Slika 11. Arhitektura konvolucijske neuronske mreže autora Levi i Hassner

Izvor: (Levi i Hassner, 2015., str. 3)

Konvolucijska neuronska mreža autora Levi i Hassner sadrži tri konvolucijska sloja, nakon kojih slijede operacija linearnog ispravljanja i sloj sažimanja. Prvi konvolucijski sloj sadrži 96 filtera od 7x7 piksela, drugi konvolucijski sloj sadrži 256 filtera od 5x5 piksela, treći i konačni konvolucijski sloj sadrži 384 filtera od 3x3 piksela. Na kraju se dodaju dva potpuno povezana sloja, od kojih svaki sadrži 512 neurona. Iz povezanih slojeva se dobiva rezultat oznake klase atributa kojoj ulazna slika pripada (Levi i Hassner, 2015.)

Model se u aplikaciji implementirao pomoću OpenCV modula za duboke neuronske mreže, Dnn. Rezultat prepoznavanja spola opisan je u obliku izlazne vrijednosti 0 ili 1, gdje vrijednost 0 prikazuje mušku osobu, a vrijednost 1 prikazuje žensku osobu. U prepoznavanju dobne skupine, rezultat procjene dobne skupine opisan je u obliku izlaznih vrijednosti od 0 do 7, gdje svaka vrijednost reprezentira određenu dobnu skupinu; 0-2, 4-6, 8-13, 15-20, 25-32, 38-43, 48-53, 60+.

```
//Metoda prepoznavanja dobne skupine
private String predictAge(Mat mRgba, Rect[] facesArray) {
    try {
        for (Rect face : facesArray) {
            Mat capturedFace = new Mat(mRgba, face);
            //Smanjivanje rezolucije slike na potrebnu rezoluciju koju istrenirani Caffe model očekuje
            Imgproc.resize(capturedFace, capturedFace, new Size( width: 227, height: 227));
            //Promjena četverokanalne(RGBA) slike u trokanalnu(BGR)
            Imgproc.cvtColor(capturedFace, capturedFace, Imgproc.COLOR_RGBA2BGR);

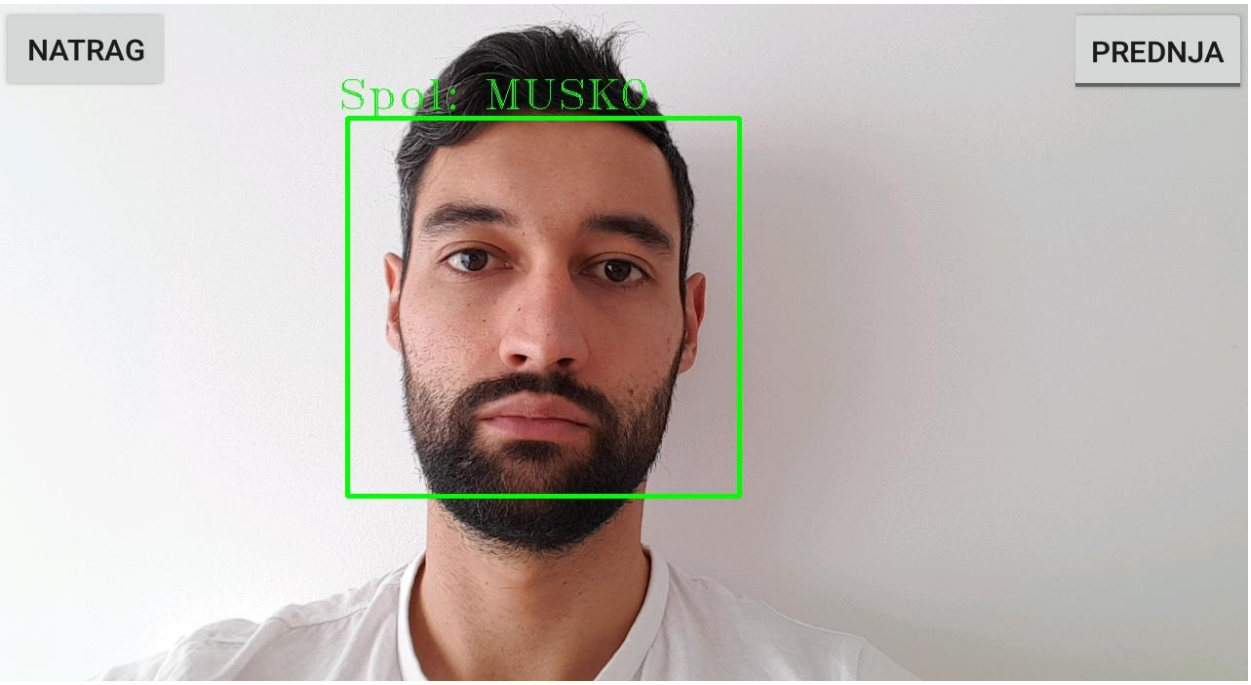
            //Slanje slike lica kroz duboku neuronsku mrežu (Dnn)
            Mat inputBlob = Dnn.blobFromImage(capturedFace, scalefactor: 1.0f, new Size( width: 227, height: 227),
                    new Scalar(78.4263377603, 87.7689143744, 114.895847746), swapRB: false, crop: false);
            mAgeNet.setInput(inputBlob, name: "data");
            Mat probs = mAgeNet.forward( outputName: "prob").reshape( cn: 1, rows: 1);
            Core.MinMaxLocResult mm = Core.minMaxLoc(probs); //Uzimanje najveće vjerojatnosti

            double result = mm.maxLoc.x; //Dobivena dobna skupina
            Log.i(TAG, msg: "Result is: " + result);
            return AGES[(int) result];
        }
    } catch (Exception e) {
        Log.e(TAG, msg: "Error processing age", e);
    }
    return null;
}
```

Slika 12. Metoda prepoznavanja dobne skupine

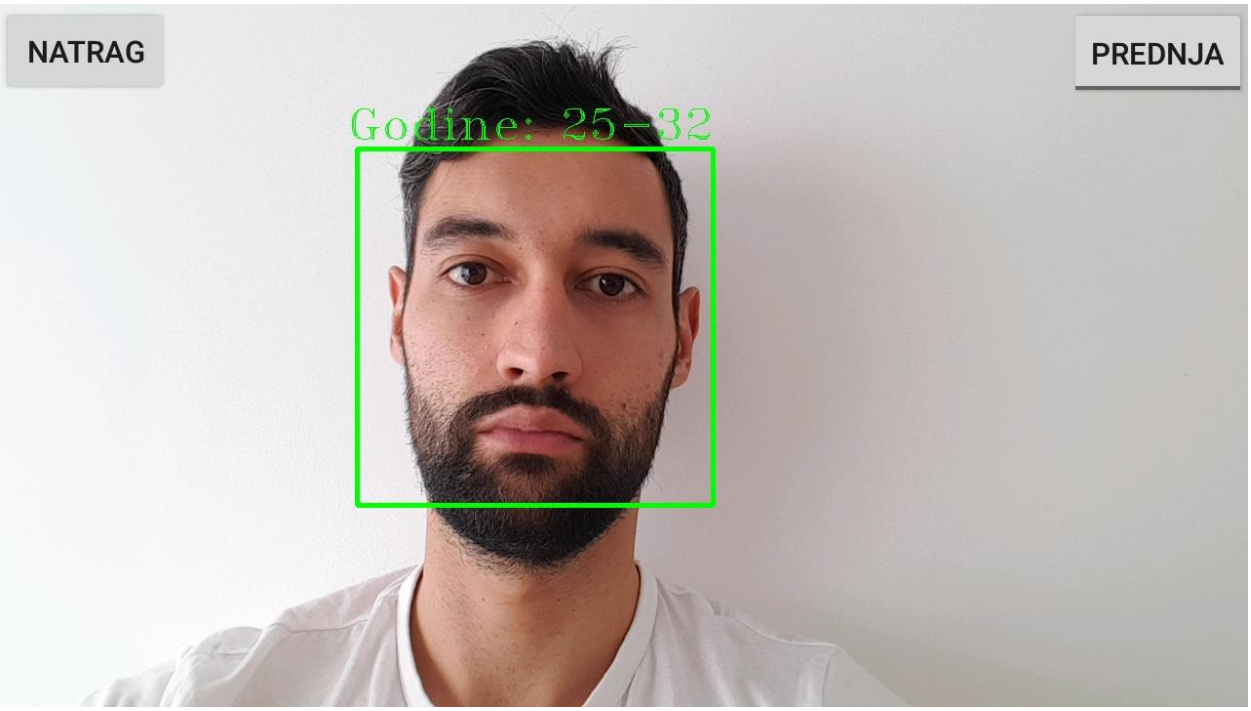
Izvor: Izrada autora

Metoda za prepoznavanje spola i dobne skupine kao parametre uzima četverokanalnu RGBA sliku koju kamera mobilnog uređaja prikazuje i 2D niz pravokutnih detektiranih lica. Unutar metode, stvara se novi Mat objekt koji sadrži samo detektirano lice unutar cjelokupne slike. Zatim se slika lica smanjuje na potrebnu rezoluciju konvolucijske neuronske mreže koju koristimo, 227x227 i prebacuje u trokanalnu sliku formata BGR. Nakon preliminarne obrade slike lica, slika se šalje kroz duboku neuronsku mrežu gdje se rezultat najveće vrijednosti dohvaća putem statičke klase MinMaxLocResult, koja ispisuje broj klase spola ili dobne skupine kojoj detektirano lice pripada te vraća rezultat u obliku String varijable.



Slika 13. Prepoznavanje spola u aplikaciji WreckFace

Izvor: Izrada autora



Slika 14. Prepoznavanje dobne skupine u aplikaciji WreckFace

Izvor: Izrada autora

Prije samog prepoznavanja lica potrebno je istrenirati model prepoznavanja lica putem slika lica određene osobe. Metoda za uzimanje slika lica se izvodi pritiskom na gumb koji se nalazi na zaslonu mobilnog uređaja. Pritiskom se izvodi statička metoda uzimanja slike detektiranog lica na zaslonu mobilnog uređaja.

```
//Metoda uzimanja i spremanja slika
public static void takePhoto(int photoNumber, Mat rgbaMat, CascadeClassifier faceDetector) throws Exception {
    File facePicsPath = new File(String.valueOf(ROOT));

    if (facePicsPath.exists() && !facePicsPath.isDirectory())
        facePicsPath.delete();
    if (!facePicsPath.exists())
        facePicsPath.mkdirs();

    Mat grayMat = new Mat();
    //Pretvaranje četverokanalne slike (RGBA) u jednokanalnu sliku (GRAY)
    Imgproc.cvtColor(rgbaMat, grayMat, Imgproc.COLOR_RGBA2GRAY);

    MatOfRect detectedFaces = new MatOfRect();
    faceDetector.detectMultiScale(grayMat, detectedFaces);
    Rect[] detectedFacesArray = detectedFaces.toArray();

    for (Rect face : detectedFacesArray) {
        Mat capturedFace = new Mat(grayMat, face);
        //Smanjivanje rezolucije na 92x112
        Imgproc.resize(capturedFace, capturedFace, new Size(IMG_WIDTH, IMG_HEIGHT));
        //Ujednačavanje histograma slike
        Imgproc.equalizeHist(capturedFace, capturedFace);

        if (photoNumber <= PHOTOS_TRAIN_QTY) {
            //Format slike u obliku brojSlike.formatSlike (pr. 1.png)
            File savePhoto = new File(facePicsPath, String.format("%d.png", photoNumber));
            savePhoto.createNewFile();
            //Zapisivanje slike lica u direktorij SlikeLica
            Imgcodecs.imwrite(savePhoto.getAbsolutePath(), capturedFace);
            Log.i(TAG,  msg: "PIC PATH: " + savePhoto.getAbsolutePath());
        }
    }
}
```

Slika 15. Metoda uzimanja i spremanja slika lica u memoriju mobilnog uređaja

Izvor: Izrada autora

Parametri metode su broj slike, četverokanalna slika u RGBA formatu i kaskadni klasifikator za detekciju lica na slici. Na samom početku izvršavanja metode, stvara se direktorij nazvan SlikeLica u memoriji mobilnog uređaja gdje će biti pohranjene slike lica, kao i istrenirani model prepoznavanja lica u XML formatu. Zatim slijedi preliminarna obrada slike, pretvaranje slike RGBA formata u jednokanalnu sliku sivih tonova, detekcija

i izvlačenje lica s cjelokupne slike, smanjivanje rezolucije na rezoluciju 92x112 i ujednačavanje histograma slike. Preliminarna obrada slika se obavlja radi optimizacije resursa i performansi potrebnih za računalno izvođenje algoritama treniranja modela prepoznavanja lica. Ako je broj slike manji od maksimalno dopuštenog broja slika, slika se sprema pomoću OpenCV modula za čitanje i zapisivanje slika u memoriju mobilnog uređaja, Imgcodecs, u poddirektorij SlikeLica direktorija Pictures.

Nakon uzimanja slika slijedi priprema slika za treniranje modela prepoznavanja lica, izrada modela prepoznavanja lica, treniranje modela prepoznavanja lica s prikupljenim slikama i odgovarajućim oznakama lica, te spremanje modela prepoznavanja lica u internu memoriju mobilnog uređaja.

```
//Metoda treniranja modela prepoznavanja lica
public static boolean train() throws Exception {
    File facePicsPath = new File(String.valueOf(ROOT));

    if (!facePicsPath.exists()) {
        return false;
    }
    FilenameFilter photoFilter = (dir, name) -> {
            return name.endsWith(".png");
    };
    File[] photosArray = facePicsPath.listFiles(photoFilter);
    opencv_core.MatVector photosMatVector = new opencv_core.MatVector(photosArray.length);
    opencv_core.Mat labels = new opencv_core.Mat(photosArray.length, cols: 1, CV_32SC1);
    IntBuffer intBuffer = labels.createBuffer();
    int counter = 0;

    for (File image : photosArray) {
        //Učitavanje slike u jednokanalnom formatu slike (GRAY)
        opencv_core.Mat photo = imread(image.getAbsolutePath(), opencv_imgcodecs.CV_LOAD_IMAGE_GRAYSCALE);
        Log.d(TAG, msg: "Image: " + image.getName());
        //Odvajanje broja slike
        int intLabel = Integer.parseInt(image.getName().split( regex: "\\.")[0]);
        //Smanjivanje rezolucije na 92x112
        resize(photo, photo, new opencv_core.Size(IMG_WIDTH, IMG_HEIGHT));
        //Ujednačavanje histograma slike
        equalizeHist(photo, photo);
        photosMatVector.put(counter, photo);
        intBuffer.put(counter, intLabel);
        counter++;
    }

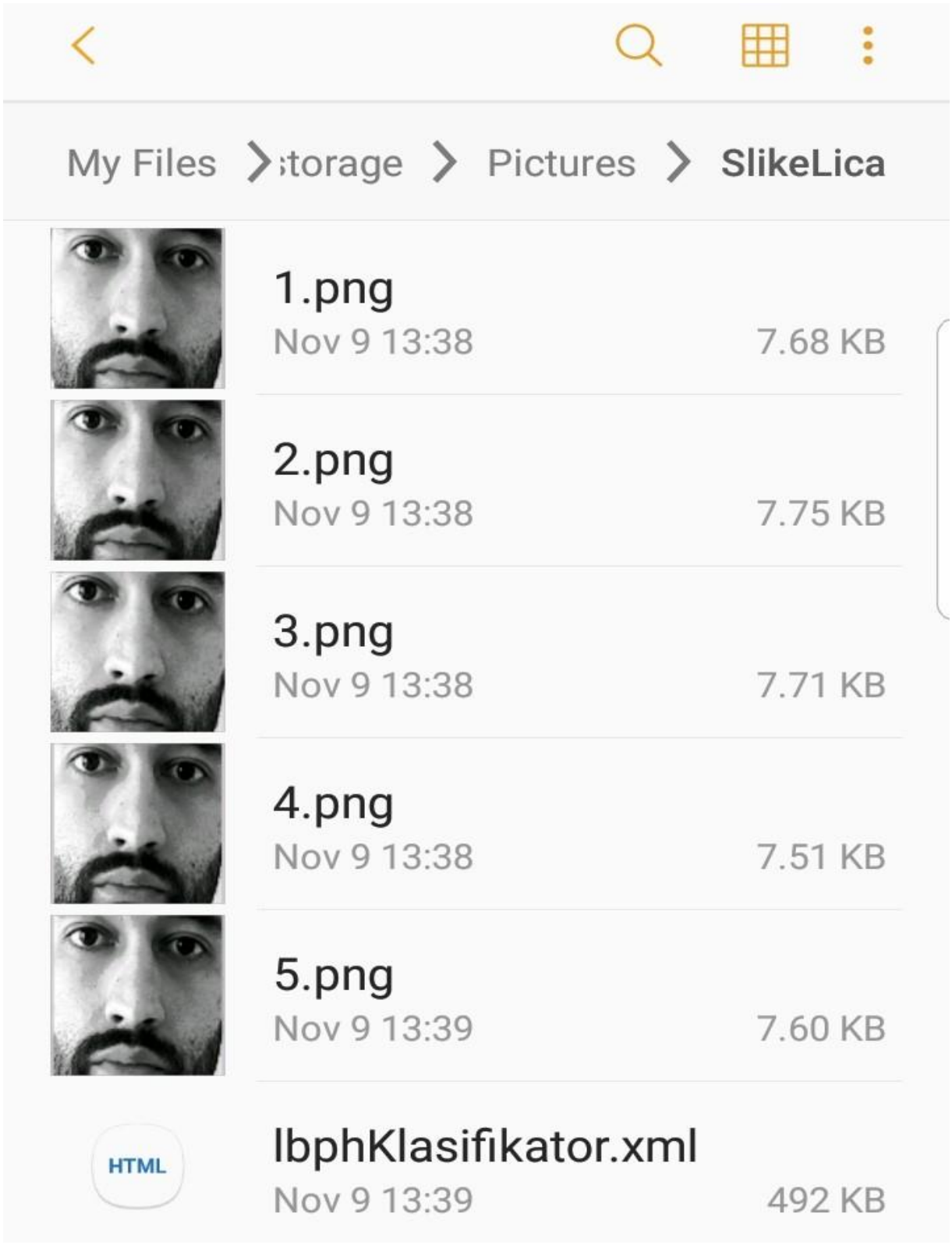
    //Kreiranje, treniranje i zapisivanje LBPH modela prepoznavanja lica
    opencv_face.FaceRecognizer mLBPHFaceRecognizer = opencv_face.LBPHFaceRecognizer.create();
    mLBPHFaceRecognizer.train(photosMatVector, labels);
    File trainedFaceRecognizerModel = new File(facePicsPath, LBPH_CLASSIFIER);
    trainedFaceRecognizerModel.createNewFile();
    mLBPHFaceRecognizer.write(trainedFaceRecognizerModel.getAbsolutePath());
    return true;
}
```

Slika 16. Metoda za pripremu slika, izradu, trening i spremanje modela prepoznavanja lica

Izvor: Izrada autora

Svaka zapisana slika u direktoriju SlikeLica se učitava u jednokanalni format sivih tonova, te se sprema u Mat objekt. Oznaka svake slike je zapravo broj slike koji se sprema u varijablu integer. Zatim slijedi obrada ulazne slike, te naposljetku svaka slika se zapisuje u objekt MatVector s odgovarajućom oznakom koja se zapisuje u objekt Mat. Mobilna aplikacija koristi LBPH[29] model prepoznavanja lica koji se kreira pomoću statičke klase FaceRecognizer koja se nalazi u OpenCV modulu za analizu lica, Face. Nakon kreacije modela prepoznavanja lica, omogućeno je korištenje njegovih metoda. Jedna od metoda koja se koristi je train koja kao parametre zahtijeva set slika koje će se koristiti za treniranje modela i odgovarajuće oznake tih slika. Nakon treniranja modela prepoznavanja lica, model se sprema u internu memoriju mobilnog uređaja putem metode write. Rezultat je datoteka u XML formatu koja sadržava sve informacije o značajkama lica u slikama nad kojima se treniranje izvršilo. Datoteka se sprema u direktorij SlikeLica.

[29] Local Binary Pattern Histogram



Slika 17. Direktorij SlikeLica

Izvor: Izrada autora

Nakon obavljenog treniranja modela prepoznavanja lica, pritiskom na gumb koji se nalazi na zaslonu mobilne aplikacije pokreće se programski modul prepoznavanja lica. Pri samom pokretanju programskog modula prepoznavanja lica u pozadini se učitava istrenirani model prepoznavanja lica u XML formatu. Učitavanje se izvršava pomoću metode read. Istrenirani model služi pri prepoznavanju lica koje je detektirano na zaslonu mobilnog uređaja.

```
//Učitavanje istreniranog klasifikatora iz direktorija SlikeLica u model za prepoznavanje lica
new AsyncTask<Void, Void, Void>() {
    @Override
    protected Void doInBackground(Void... voids) {
        try {
            File folder = new File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES), Methods.FACE_PICS);
            File f = new File(folder, Methods.LBPH_CLASSIFIER);
            Log.i(TAG, msg: "Classifier = " + f);
            mLBPHFaceRecognizer.read(f.getAbsolutePath());
        }catch (Exception e) {
            Log.d(TAG, e.getLocalizedMessage(), e);
        }
        return null;
    }
}.execute();
```

Slika 18. Učitavanje istreniranog modela prepoznavanja lica

Izvor: Izrada autora

Prepoznavanje se izvršava automatski na zaslonu mobilnog uređaja ako je detektirano samo jedno lice. Prije izvršavanja metode prepoznavanja, potrebno je obavljanje preliminarne obrade slike. Osim već spomenutog smanjivanja rezolucije i ujednačavanje histograma, koristila se konverzija slike u JavaCV[30] Mat reprezentaciju slika. Prepoznavanje lica se izvršava metodom predict statičke klase FaceRecognizer. U metodi predict kao parametri se nalaze, slika nad kojom se prepoznavanje izvršava, lista oznaka slika i vjerojatnost poklapanja detektiranog lica s licem iz direktorija SlikeLica. Model prepoznavanja lica koristi metodu najbližeg susjeda za prepoznavanje.

Da bi mogao prepoznati broj najbliže prepoznate slike, detektirano lice mora zadovoljiti određene uvjete istreniranog modela prepoznavanja lica. Postoje dva uvjeta; prvi uvjet je da detektirano lice na zaslonu mobilnog uređaja ima značajke lica koje se uspoređuje prema značajkama lica koje se nalaze u istreniranom modelu prepoznavanja

[30] Java omotač za OpenCV biblioteku

lica, lbphKlasifikator.xml, a drugi uvjet je da vjerojatnost poklapanja detektiranog lica s istreniranim modelom prepoznavanja lica ne prelazi programiranu graničnu vrijednost. Scenarij rezultata prepoznavanja lica može biti negativan, ako jedan ili oba uvjeta nisu zadovoljeni, ili pozitivan, prikaz broja najbliže prepoznate slike lica iz direktorija SlikeLica. Rezultat prepoznavanja lica se ispisuje iznad pravokutnika detektiranog lica putem metode putText OpenCV modula za obradu slike, Imgproc.

```
//Ako je detektirano jedno lice u slici, pokreće se metoda prepoznavanja lica
if (facesArray.length == 1) {
    try {
        //Konverzija OpenCV Mat-a u JavaCV Mat
        opencv_core.Mat javaCvMat = new opencv_core.Mat((Pointer) null) {{address = mGray.getNativeObjAddr();}};
        //Smanjivanje rezolucije na 92x112
        resize(javaCvMat, javaCvMat, new opencv_core.Size(Methods.IMG_WIDTH, Methods.IMG_HEIGHT));
        //Ujednačavanje histograma
        equalizeHist(javaCvMat, javaCvMat);

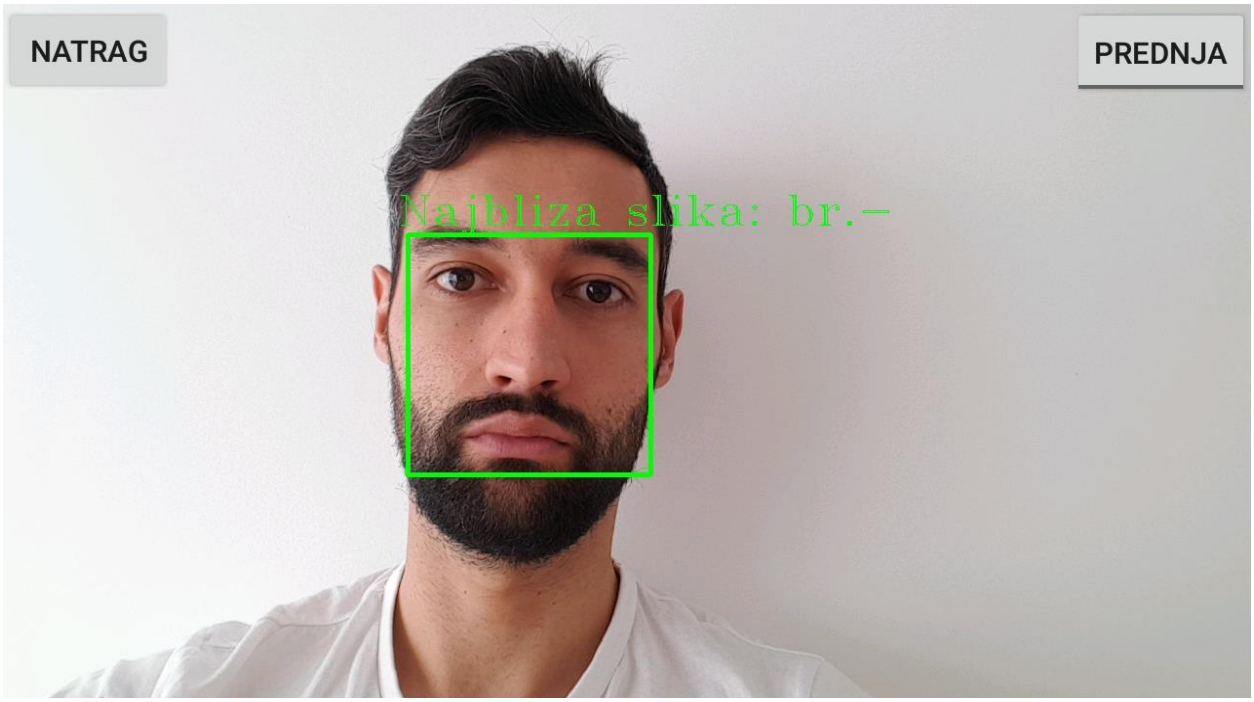
        IntPointer label = new IntPointer( size: 1);
        DoublePointer confidence = new DoublePointer( size: 1);
        //Metoda prepoznavanja lica
        mLBPHFaceRecognizer.predict(javaCvMat, label, confidence);

        //Dohvaćanje najbliže slike iz treniranog modela lica
        int predictedLabel = label.get(0);
        //Dohvaćanje vjerojatnosti. Manji broj = preciznije.
        double acceptanceLevel = confidence.get(0);
        String name;
        Log.d(TAG, msg: "Prediction completed, predictedLabel: " + predictedLabel + ", acceptanceLevel: " + acceptanceLevel);
        if (predictedLabel == -1 || acceptanceLevel >= THRESHOLD) {
            name = "-";
        } else {
            name = Integer.toString(predictedLabel);
        }

        //Ispis broja najbliže prepoznate slike lica iznad pravokutnika
        for (Rect face : facesArray) {
            int posX = (int) Math.max(face.tl().x - 10, 0);
            int posY = (int) Math.max(face.tl().y - 10, 0);
            Imgproc.putText(mRgba, text: "Najbliza slika: br." + name, new Point(posX, posY),
                    Core.FONT_HERSHEY_TRIPLEX, fontScale: 1.5, new Scalar(0, 255, 0, 255));
        }
    }catch (Exception e) {
        Log.d(TAG, e.getLocalizedMessage(), e);
    }
}
```

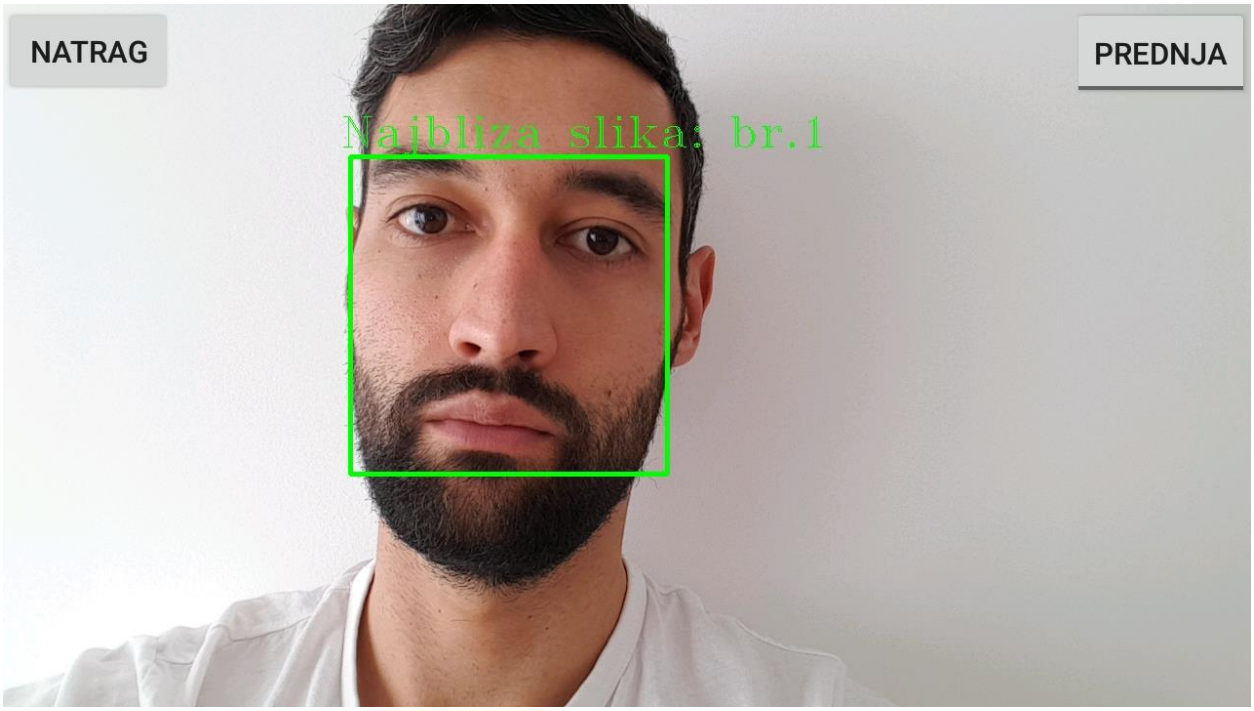
Slika 19. Obrada slike, prepoznavanje lica i ispisivanje rezultata iznad detektiranog lica na zaslonu mobilnog uređaja

Izvor: Izrada autora



Slika 20. Negativan scenarij rezultata aplikacije WreckFace

Izvor: Izrada autora



Slika 21. Pozitivan scenarij rezultata aplikacije WreckFace

Izvor: Izrada autora

6. Zaključak

U ovom radu je detaljno objašnjeno razvijanje sustava prepoznavanja spola, dobne skupine i lica na Android mobilni uređaj. Navedene su metode detekcije lica i najčešće metode prepoznavanja lica, te u programskom rješenju je objašnjeno kako je postignuta detekcija i prepoznavanje lica u Android mobilnoj aplikaciji WreckFace. Cjelokupno programsko rješenje detekcije i prepoznavanja lica u WreckFace aplikaciji je postignuto pomoću OpenCV biblioteke koja omogućuje mnogobrojne funkcionalnosti za razvoj aplikacija računalnog vida. Korišten je LBP klasifikator značajki lica u svrhu detekcije lica i model LBPH u svrhu prepoznavanja lica. Za prepoznavanje spola i dobne skupne koristila se istrenirana duboka neuronska mreža koja omogućuje korištenje funkcije prepoznavanja bez prethodno uzimanja slika i treniranje modela za prepoznavanje spola i dobne skupine. Iako algoritmi prepoznavanja uspješno obavljaju posao prepoznavanja, na njih utječu razni uvjeti poput osvjetljenja, poza osobe, izraza lica, prekrivenosti lica, karakteristike kamere, kao i performanse samog mobilnog uređaja. Prikazano je da unatoč svim navedenim problemima je moguće implementirati sustav prepoznavanja na Android mobilni uređaj.

Budućnost WreckFace aplikacije je u korištenju nativnih metoda, nativne kamere i dubokih neuronskih mreža biblioteke OpenCV. Kako bi se smanjila preopterećenost resursa mobilnog uređaja i povećale performanse, potrebna je izrada velike baze podataka slika pohranjena u oblaku koja bi se koristila za sva prepoznavanja u mobilnoj aplikaciji, tj. za prepoznavanje lica, spola i dobne skupine. Time bi se ostvario ujedinjeni sustav za kompletno prepoznavanje osobe koja koristi usluge WreckFace aplikacije.

Literatura

Ahonen, T., Hadid, A. i Pietikainen, M. (2006.) Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 28(12). str. 2037-2041.

Audet, S. (2014.) *Bytedeco.* [Online] Dostupno na: http://bytedeco.org/. [Pristupljeno 15. ožujak 2018.]

Bora, P. K., Rahman, F. i Hazarika, M. (2015.) A Review on Face Recognition Approaches. *International Journal of Engineering & Technology (IJERT).* 4(05). str. 524-527.

Brunelli, R. i Poggio, T. (1992.) Face Recognition Through Geometrical Features. U: *Proceedings of the Second European Conference on Computer Vision.* Berlin: Springer-Verlag. str. 792-800.

Bytedeco, n.d. *Github.* [Online] Dostupno na: https://github.com/bytedeco. [Pristupljeno 15. ožujak 2018.]

Çarıkçı, M. i Özen, F. (2012.) A Face Recognition System Based on Eigenfaces Method. Istanbul: Elsevier.

Chang-yeon, J. (2008.) Face Detection using LBP features. Stanford: Stanford University.

Deshpande, A. (2016.) *Github.* [Online] Dostupno na: https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks. [Pristupljeno 19. rujan 2018.]

Eidinger, E., Enbar, R. i Hassner, T. (2014.) Age and Gender Estimation of Unfiltered Faces. *Transactions on Information Forensics and Security (IEEE-TIFS).* 9(12). str. 2170-2179.

Emami, S. (2010.) *Introduction to Face Detection and Face Recognition.* [Online] Dostupno na: http://shervinemami.info/faceRecognition.html. [Pristupljeno 28. ožujak 2018].

Google, (2018.) *Android Developers.* [Online] Dostupno na: https://developer.android.com/studio/intro. [Pristupljeno 14. rujan 2018.]

Google, (2018.) *Android Developers.* [Online] Dostupno na: https://developer.android.com/guide/platform. [Pristupljeno 20. travanj 2018]

Google, (2018.) *Android Developers.* [Online] Dostupno na: https://developer.android.com/guide/components/fundamentals. [Pristupljeno 20. travanj 2018.]

Google, (2018.) *Android Developers.* [Online] Dostupno na: https://developer.android.com/guide/topics/providers/content-provider-basics. [Pristupljeno 10. svibanj 2018.]

Gregori, E. (2013.) *Developing OpenCV computer vision apps for the Android platform.* [Online] Dostupno na: https://www.embedded.com/design/programming-languages-and-tools/4406164/Developing-OpenCV-computer-vision-apps-for-the-Android-platform. [Pristupljeno 18. srpanj 2018.]

Grgić, M. i Delač, K. (2017.) *Face Recognition Homepage.* [Online] Dostupno na: http://www.face-rec.org/algorithms. [Pristupljeno 5. ožujak 2018.]

Hubel, D. i Wiesel, T. (2012.) *Cell.* [Online] Dostupno na: https://www.cell.com/neuron/fulltext/S0896-6273(12)00618-6#articleInformation. [Pristupljeno: 9. kolovoza 2018.]

Jain, A., Hong, L. i Sharath, P. (2000.) Biometric Identification. *Communications of the ACM.* 43(2). str. 91-98.

Jain, A. K., Ross, A. i Prabhakar, S. (2004.) An Introduction to Biometric Recognition. *IEEE Transactions on circuits and systems for video technology.* 14(1). str. 4-20.

Kaehler, A. i Bradski, G. (2016.) *Learning OpenCV 3.* 1st edition. Sebastopol: O'Reilly Media, Inc.

Kanade, T. (1973.) *Picture Processing System by Computer Complex and Recognition of Human Faces.* Kyoto: Kyoto University.

Levi, G. (2018.) *Github.* [Online] Dostupno na: https://gist.github.com/GilLevi/c9e99062283c719c03de. [Pristupljeno 2. rujan 2018.]

Levi, G. i Hassner, T. (2015.) *Age and Gender Classification Using Convolutional Neural Networks.* Boston: IEEE Workshop on Analysis and Modeling of Faces and Gestures (AMFG).

Martinez, A. (2011.) *Scholarpedia.* [Online] Dostupno na: http://www.scholarpedia.org/article/Fisherfaces. [Pristupljeno 8. travanj 2018.]

Martinez, A. M. i Kak, A. C. (2001.) PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 23(2). str. 228-233.

Miller, B. (1994.) Vital signs of identity. *IEEE Spectrum.* 31(2). str. 22-30.

Moore, A.-M. (2007.) Biometric technologies – an introduction. *Biometric Technology Today.* str. 6-7.

OpenCV, (2017.) *OpenCV.* [Online] Dostupno na: https://docs.opencv.org/3.4.0/d5/df8/tutorial_dev_with_OCV_on_Android.html. [Pristupljeno 17. srpanj 2018.]

OpenCV, (2018.) *OpenCV.* [Online] Dostupno na: https://opencv.org. [Pristupljeno 3. ožujak 2018.]

OpenCV, (2018.) *OpenCV 2.4.13.7 documentation.* [Online] Dostupno na: https://docs.opencv.org/2.4/doc/tutorials/core/mat_the_basic_image_container/mat_the_basic_image_container.html. [Pristupljeno 23. kolovoz 2018.]

Paul, L. C. i Al Sumam, A. (2012.) Face Recognition Using Principal Component Analysis Method. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET).* 1(9). str. 135-139.

Pietikäinen, M. (2010.) *Scholarpedia.* [Online] Dostupno na: http://www.scholarpedia.org/article/Local_Binary_Patterns#Face_description_using_LBP. [Pristupljeno 8. travanj 2018.]

Radmilović, Ž. (2008.) Biometrijska identifikacija. *Policija i sigurnost.* 3-4. str. 159-180.

Rizvi, D. Q. M. (2011.) A Review on Face Detection Methods. *Journal of Management Development and Information Technology.* 11.

Salton do Prado, K. (2017.) *Towards Data Science.* [Online] Dostupno na: https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b. [Pristupljeno 8. travanj 2018.]

Techotopia, (2018.) *Techotopia.* [Online] Dostupno na: https://www.techotopia.com/index.php/An_Overview_of_the_Android_Architecture. [Pritupljeno 10. svibanj 2018.]

Tharwat, A., Gaber, T., Ibrahim, A. i Hassanien, A. E. (2017.) Linear discriminant analysis: A detailed tutorial. *Ai Communications.* 30(2). str. 169-190.

The-Crankshaft, P. n.d. *what-when-now.* [Online] Dostupno na: http://what-when-how.com/face-recognition/local-representation-of-facial-features-face-image-modeling-and-representation-face-recognition-part-1. [Pristupljeno 15. travanj 2018.]

Turk, M. i Pentland, A. (1991.) Eigenfaces for Recognition. *J. Cognitive Neuroscience.* 3(1). str. 71-86.

Wagner, P. (2011.) *Face Recognition with OpenCV.* [Online] Dostupno na: https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html. [Pristupljeno 3. ožujak 2018.]

Zhang, S. i Turk, M. (2008.) *Scholarpedia.* [Online] Dostupno na: http://www.scholarpedia.org/article/Eigenfaces#Using_Eigenfaces_in_Face_Processing. [Pristupljeno 8. travanj 2018.]

Popis slika

Slika 1. Struktura sustava prepoznavanja lica 13
Slika 2. Arhitektura Android platforme 22
Slika 3. Datoteke projekta u Android pregledu 31
Slika 4. Glavni prozor Android Studio-a 32
Slika 5. Dijagram slučajeva korištenja WreckFace aplikacije 36
Slika 6. Inicijalizacija OpenCV Manager-a u aplikaciji WreckFace 38
Slika 7. Dodavanje prava u AndroidManifest.xml datoteci za pristup kameri mobilnog uređaja 39
Slika 8. Detekcija i iscrtavanje pravokutnika oko lica na slici kamere 40
Slika 9. Detekcija lica na slici u aplikaciji WreckFace 40
Slika 10. Učitavanje Caffe modela za prepoznavanje dobne skupine u duboku neuronsku mrežu 41
Slika 11. Arhitektura konvolucijske neuronske mreže autora Levi i Hassner 42
Slika 12. Metoda prepoznavanja dobne skupine 43
Slika 13. Prepoznavanje spola u aplikaciji WreckFace 44
Slika 14. Prepoznavanje dobne skupine u aplikaciji WreckFace 44
Slika 15. Metoda uzimanja i spremanja slika lica u memoriju mobilnog uređaja 45
Slika 16. Metoda za pripremu slika, izradu, trening i spremanje modela prepoznavanja lica 46
Slika 17. Direktorij SlikeLica 48
Slika 18. Učitavanje istreniranog modela prepoznavanja lica 49
Slika 19. Obrada slike, prepoznavanje lica i ispisivanje rezultata iznad detektiranog lica na zaslonu mobilnog uređaja 50
Slika 20. Negativan scenarij rezultata aplikacije WreckFace 51
Slika 21. Pozitivan scenarij rezultata aplikacije WreckFace 51

Sažetak

Rad detaljno opisuje i objašnjava cjelokupni postupak razvijanja WreckFace Android mobilne aplikacije za prepoznavanje spola, dobne skupine i lica osobe. U teorijskom dijelu su prikazani i objašnjeni različiti biometrijski identifikatori i njihove karakteristike. Opisane su i objašnjene metode detekcije i prepoznavanja lica, kao i razvojni alati koji su korišteni pri izradi mobilne aplikacije. U programskom rješenju su opisani detalji korištenja biblioteke OpenCV i putem slika su prikazani stvarni rezultati mobilne aplikacije.

Ključne riječi: Android, mobilna aplikacija, prepoznavanje lica, dobne skupine, spola, biometrija, detekcija lica, Android Studio, OpenCV, Lokalni binarni uzorci, duboka neuronska mreža, strojno učenje

Abstract

The thesis describes and explains in detail the entire process of developing WreckFace Android mobile application for recognizing person's gender, age group and face. Different biometric identifiers and their characteristics are presented and explained in the theoretical part. Methods of detection and recognition of faces have been described and explained as well as development tools used in the development of mobile application. The program solution describes the details of using the OpenCV library and shows the actual results of the mobile application through the images.

Keywords: Android, mobile application, face recognition, age group, gender, biometrics, face detection, Android Studio, OpenCV, Local binary patterns, deep neural network, machine learning