Usporedba različitih algoritama za detekciju objekata koristeći OpenCV i Dlib

Baudoin, Borna

SVEUČILIŠTE SJEVER
SVEUČILIŠNI CENTAR VARAŽDIN

DIPLOMSKI RAD br. 043/MMD/2021

USPOREDBA RAZLIČITIH ALGORITAMA ZA DETEKCIJU OBJEKATA KORISTEĆI OPENCV I DLIB

Borna Baudoin

Varaždin, rujan 2021. godine

SVEUČILIŠTE SJEVER
SVEUČILIŠNI CENTAR VARAŽDIN

Studij: diplomski sveučilišni studij Multimedija

DIPLOMSKI RAD br. 043/MMD/2021

USPOREDBA RAZLIČITIH ALGORITAMA ZA DETEKCIJU OBJEKATA KORISTEĆI OPENCV I DLIB

Student: Mentor:

Borna Baudoin: 0016115183 izv. prof. dr. sc. Emil Dumić

Varaždin, rujan 2021. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR-42000 Varaždin

Prijava diplomskog rada

Definiranje teme diplomskog rada i povjerenstva

ODJEL Odjel za multimediju

STUDIJ diplomski sveučilišni studij Multimedija

PRISTUPNIK Baudoin Borna | JMBAG 0016115183

DATUM | KOLEGIJ Računalni vid

NASLOV RADA Usporedba različitih algoritama za detekciju objekata koristeći OpenCV i Dlib

NASLOV RADA NA ENGL. JEZIKU Comparison of different object detection algorithms using OpenCV and Dlib

MENTOR Emil Dumić | ZVANJE izv.prof.dr.sc.

ČLANOVI POVJERENSTVA

1. doc. art. dr. sc. Mario Periša - predsjednik
2. doc. art. dr. sc. Robert Geček - član
3. izv. prof. dr. sc. Emil Dumić - mentor
4. doc. dr. sc. Andrija Bernik - zamjenski član
5.

Zadatak diplomskog rada

BROJ 043-MMD-2021

OPIS

U ovom radu će biti uspoređeni različiti algoritmi za detekciju objekata koristeći OpenCV i Dlib programske biblioteke.
U današnje vrijeme, sve se češće koriste različite metode strojnog učenja za računalni vid. Primjerice, kod obrade slike, klasifikacija objekata, detekcija objekata, segmetacija slike su neki od češće korištenih skupina algoritama. Kod detekcije objekata, bitno je odrediti položaj i vrstu objekata na slici. Ako koristimo algoritme strojnog učenja za detekciju objekata, bitno je imati i dovoljnu količinu podataka za trening, kako bi algoritam dobro radio nad testnim podacima.
U praktičnom dijelu zadatka će se koristiti različite metode detekcije objekata te međusobno usporediti koristeći mjere točnosti, F1 mjere, preciznosti i odziva, te AP i mAP mjera (dobivenih na osnovu IoU omjera). Koristit će se neki od javno dostupnih baza s unaprijed označenim položajima objekata, poput "Open Images Dataset" baze, te preuzeti slike s npr. licima, kao objekti za detekciju. Ispitat će se algoritmi poput Viola-Jones, SVM i HOG, te neuronske mreže, koristeći programske bibiloteke OpenCV i Dlib. Dat će se zaključak o točnosti svakog od navedenih algoritama, ali će se raspraviti i o ovisnosti o brzini izvođenja.

ZADATAK URUČEN 8.9.2021. | POTPIS MENTORA Emil Dumić

REPUBLIKA HRVATSKA SVEUČILIŠTE SJEVER KOPRIVNICA

Sažetak

U ovom će radu biti uspoređeni različiti algoritmi za detekciju objekata koristeći OpenCV i Dlib programske biblioteke. U današnje vrijeme, sve se češće koriste različite metode strojnog učenja za računalni vid. Primjerice, kod obrade slike, klasifikacija objekata, detekcija objekata, segmetacija slike su neki od češće korištenih skupina algoritama. Kod detekcije objekata, bitno je odrediti položaj i vrstu objekata na slici. Ako koristimo algoritme strojnog učenja za detekciju objekata, bitno je imati i dovoljnu količinu podataka za trening, kako bi algoritam dobro radio nad testnim podacima.

U praktičnom dijelu zadatka će se koristiti različiti algoritmi za detekciju objekata, poput Viola-Jones algoritma, histograma orijentiranih gradijenata (HOG) te dubokih neuronskih mreža. Ispitivat će se preciznost detekcije lica u različitim položajima na fotografiji te će se na kraju donijeti zaključak koji je od spomenutih algoritama najprecizniji u samoj detekciji.

Ključne riječi: detekcija objekata, detekcija lica, neuronske mreže, Viola-Jones, HOG.

Summary

In this paper, different algorithms for object detection using OpenCV and Dlib program libraries will be compared. Nowdays, various machine learning methods for computer vision are increasingly used. For example, in image processing, object classification, object detection, image segmentation are some of the more commonly used groups of algorithms. When detecting objects it is important to determine the position and type of objects in the image. If we use machine learning algorithms to detect objects, it is important to have enough training data for the algorithm to work well on the test data.

In the practical part of the task, various algorithms will be used to detect objects, such as the Viola-Jones algorithm, histograms of oriented gradients (HOG) and deep neural networks. The accuracy of face detection in different positions in the photograph will be examined and in the end, a conclusion will be made which of the mentioned algorithms is the most accurate in the detection itself.

Keywords: computer vision, object detection, face detection, neural networks, Viola-Jones, HOG, OpenCV, Dlib.

Popis korištenih kratica

ACC Accuracy

AP Average Precision

DNN Deep Neural Networks

CNN Convolutional Neural Networks

FN False Negative

FP False Positive

HOG Histogram of Oriented Gradients

IoU Intersection over Union

mAP Mean Average Precision

PPV Positive Predictive Value

RoI Region of Interest

RPN Region Proposal Network

SSD Single Shot Detector

SVM Support Vector Machines

TP True Positive

TPR True Positive Rate

YOLO You Only Look Once

Sadržaj

1. Uvod

Detekcija objekata je računalna tehnologija usko povezana sa računalnim vidom i obradom slika koja se bavi detekcijom instanci različitih objekata (poput ljudi, životinja, vozila, kuća itd.) na digitalnim fotografijama ili videozapisima. Detekcija objekata danas je široko korištena na području računalnog vida, osobito za prepoznavanje lica i ljudi. U ovom radu glavni fokus će biti upravo na detekciji ljudskog lica na fotografiji. Sustav za detekciju lica jedan je od zanimljivijih stvari, ali i jedan od najvažnijih na području biometrije. Prvotni razlog za potrebom razvoja ovakvih sustava bio je taj da bi se ljudsko lice automatski moglo prepoznati temeljem fizičkih karakteristika putem kamere te tako na brz i efikasan način verificirati samu osobu uglavnom zbog sigurnosnih provjera (npr. identifikacija kriminalaca). Konkretno, detekcija lica podrazumijeva lokalizaciju lica na slici i određivanje njegovih granica. Automatizacija detekcije lica na fotografijama koristi se za formiranje baza podataka lica. Za proces detekcije lica možemo reći da je ono osnovni uvjet i temelj za prepoznavanje samog lica na fotografiji koje se svodi na usporedbu detektiranog lica sa postojećim licima u bazi. Neki od ključnih detalja za prepoznavanje lica općenito su položaj očiju, širina nosa, visina čela te položaj i veličina usnica. U nastavku rada ukratko će se opisati tri najvažnije metode za detekciju objekata općenito, da bi se zatim upoznali i sa programskim bibliotekama pomoću kojih će se implementirati navedene metode kao detektori za detekciju lica. Nadalje, napravit će se usporedba točnosti svih detektora i rezultata detekcije te donijeti zaključak o tome koji je detektor najpreciznji i najtočniji.

2. Metode za detekciju objekata

Kad se govori o detekciji objekata, ona se provodi različitim metodama/algoritmima koji su razvijeni upravo za tu namjenu. U nastavku će se opisati tri najpoznatije takve metode, a to su Viola-Jones algoritam, metoda histograma orijentiranih gradijenata te metoda dubokog učenja (duboke neuronske mreže). Pristup svake od metode se razlikuje međutim zajednička im je primjena za detekciju raznih objekata.

2.1. Viola-Jones algoritam

2.1.1. Karakteristike i evaluacija

Ovaj algoritam predložili su Paul Viola i Michael Jones 2001. godine s ciljem rješavanja problema efikasnosti detekcije lice koja tada još nije bila dovoljno razvijena. Ubrajamo ga u algoritme koji se temelje na pojavljivanju. To znači da su isti naučeni na unaprijed spremljene uzorke slika koja sadržavaju lica. Drugim riječima, Viola-Jones algoritam oslanja se na tehnike statističke analize i strojnog učenja. Iako je primjenjiv i za detekciju nekih drugih objekata, glavna motivacija za razvoj ovog algoritma bila je detekcija ljudskog lica. [4, 5, 11]

Viola-Jones algoritam određen je s tri karakteristike koje ga čine dobrim za prepoznavanje lica: [11]

1. **Robusnost** – vrlo visoka stopa otkrivanja lica

2. **Stvarno vrijeme** – u praktičnoj primjeni moraju se obraditi najmanje dva okvira u sekundi (*frame per second*), iako je danas obrada znatno brža te doseže i do 30 okvira u sekundi

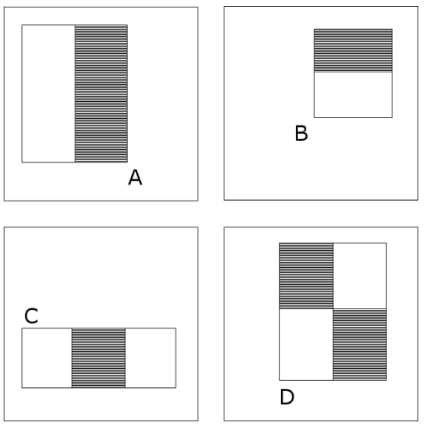
3. **Isključivo otkrivanje lica** – primarni cilj je razlikovati lica od drugih objekata (otkrivanje je prvi korak u procesu prepoznavanja samog lica)

Evaluacija Viola-Jones algoritma sastoji se od četiri faze: [3, 4, 11]

1. **Haarove značajke**
2. **Izrada integralne slike**
3. **AdaBoost algoritam**
4. **Kaskada klasifikatora**

2.1.2. Haarove značajke

Viola-Jones algoritam se bazira na posebnim značajkama pomoću kojih klasificira slike za detekciju. Algoritam u radnom okviru koristi četiri vrste Haarovih značajki prikazanih na Slici 1. Vrijednost svake značajke dobije se tako da se ukupne vrijednosti piksela koji se nalaze pod svakom vrstom pravokutnika u značajki, oduzmu. Na primjer, kod značajke označene slovom „A“ na Slici 2.1. oduzela bi se ukupna vrijednost svih piksela koji se nalaze pod bijelim pravokutnikom od ukupne vrijednosti piksela koji se nalazi pod crnim pravokutnikom. Na Slici 2.2. u nastavku vidimo konkretnu primjenu značajki na licu. Ako značajku označenu slovom „C“ primijenimo u područje nosa na bilo koje lice, dobit ćemo sličan rezultat jer je razlika ukupne vrijednosti piksela pod pravokutnikom koji prekriva sredinu nosa i dva pravokutnika koji prekrivaju područje lijevo i desno slična. Upravo zbog toga što se izdvojene vrijednosti značajki odnose na većinu ljudskih lica koje želimo detektirati, Viola-Jones algoritam donosi vrlo dobrerezultate u samoj detekciji. [4, 11]

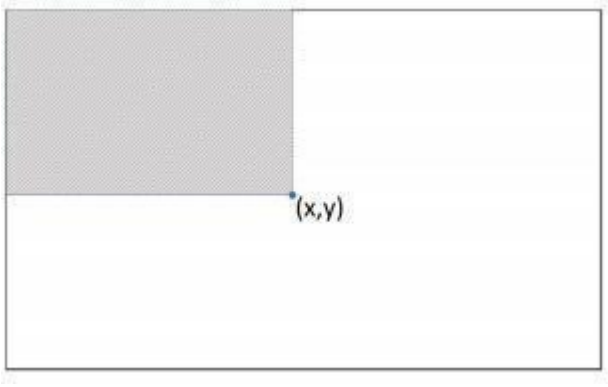


Slika 2.1. Prikaz Haarovih značajki
(izvor: https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework)

Slika 2.2. Prikaz Haarovih značajki na ljudskom licu (izvor: https://zir.nsk.hr/islandora/object/algebra%3A252)

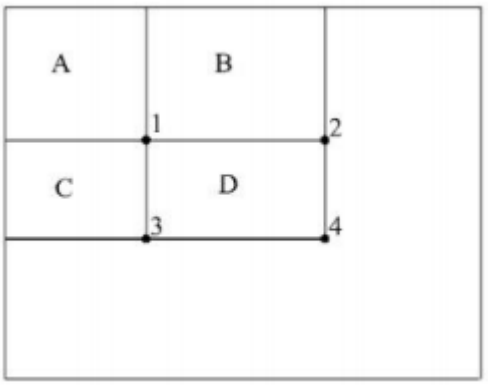
2.1.3. Integralna slika

Drugi korak Viola-Jones algoritma je izrada integralne slike. Ona je jedan od glavnih razloga uspjeha samog algoritma. Omogućuje vrlo brz izračun ukupne vrijednosti piksela u zadanom pravokutniku i to bez obzira gdje se on nalazio na slici. Integralna slika računa se jednom za svaku sliku u kojoj želimo pronaći lica te se potom ona koristi za računanje vrijednosti piksela ispod Haarovih značajki. Svaka se vrijednost piksela pod bilo kojim pravokutnikom na slici može izračunati pomoću vrijednosti četiri točke na integralnoj slici. Na Slici 2.3. možemo vidjeti prikaz vrijednosti integralne slike u točki (x, y). Ta točka jednaka je sumi vrijednosti svih piksela u zatamnjenom području gore lijevo. [4, 9]



Slika 2.3. Vrijednost integralne slike u točki (x,y) (izvor: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=A32ABE3A5AAED0636543B29601502CFF?doi=10.1.1.110.4868&rep=rep1&type=pdf)

Nadalje, na Slici 4 imamo prikaz izračuna ukupne vrijednosti piksela unutar pravokutnika D. Vrijednost integralne slike u točki 1 odgovara ukupnoj vrijednosti piksela u pravokutniku A. Vrijednost u točki 2 odgovara pak ukupnoj vrijednosti piksela A + B, u točki 3 ukupnoj vrijednosti A + C i u točki 4 ukupnoj vrijednosti A + B + C + D. Na kraju dobijemo da je ukupna vrijednost piksela unutar pravokutnika D jednaka 4 + 1 – 2 + 3 kao što vidimo na slici ispod. [4, 9]



Slika 2.4. Izračun ukupne vrijednosti piksela pravokutnika D (izvor: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=A32ABE3A5AAED0636543B29601502CFF?doi=10.1.1.110.4868&rep=rep1&type=pdf)

2.1.4. AdaBoost algoritam

Viola-Jones algoritam u svom radnom okviru koristi potprozor veličine 24 x 24 piksela. Unutar svakog potprozora nalazi se više od 160.000 vrijednosti značajki koje možemo analizirati. Odabir samo ciljanih značajki koje nam trebaju za detekciju lica, omogućuje nam AdaBoost algoritam. U suštini to je algoritam strojnog učenja koji iz 160.000 mogućih značajki izdvaja samo one najbolje za detekciju lica na slici. Nakon što pronađe takve značajke, njihova ponderirana kombinacija koristi se u odlučivanju da li određeni potprozor sadrži lice ili ne. U sam proces odlučivanja se uključuju samo one značajke koje su točne u više od 50% slučajeva. AdaBoost koristi linearnu kombinaciju tzv. slabih klasifikatora da bi stvorio jak klasifikator koji se zatim koristi u odlučivanju da li u potprozoru postoji lice ili ne. U nastavku slijedi prikaz formule 2.1. za dobivavnje jakog klasifikatora. [4]

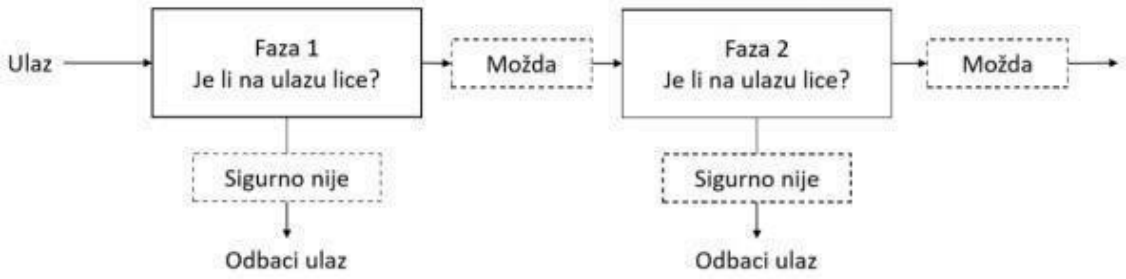
$$F(x) = a_1 f_1(x) + a_2 f_2(x) + a_3 f_3(x) + \cdots + a_n f_n(x)$$

Formula 2.1.

F(x) predstavlja jaki klasifikator, *fn(x)* slabi klsifikator dok *an* predstavlja ponder pridružen slabom klasifikatoru. Ponder će biti veći što je značajka važnija u detekciji lica. Nakon analize pojedinog potprozora, kombinacijom slabih klasifikatora unutar jakog dobivamo jedinstvenu izlaznu vrijednost te ukoliko ta izlazna vrijednost prelazi tzv. vrijednosni prag, u tom slučaju zaključujemo da se u potprozoru nalazi lice. Iako je AdaBoost postupak vrlo efikasan u detekciji lica, nedovoljno je efikasan za primjenu detekcije u realnom vremenu odnosno zahtijeva dulje vrijeme za obradu pojedine slike. Iz tog razloga Viola-Jones koristi kaskade klasifikatora o kojima će biti više riječi u nastavku. [4, 5]

2.1.5. Kaskada klasifikatora

Kaskada klasifikatora uvedena je u Viola-Jones algoritam kako bi se smanjio problem korištenja jednog velikog jakog klasifikatora za svaki potprozor unutar slike jer bi taj proces bio preskup i oduzimao previše vremena za obradu. Kod kasakde klasifikatora svaka faza kaskade izvedena je kao jedan jaki klasifikator. Sve bitne značajke detekcije grupirane su u nekoliko takvih faza u kojima svaka od njih evaluira određeni broj značajki. Zadaća svake od faza je da za određeni potprozor odredi da li se u njemu možda nalazi lice ili se sigurno ne nalazi. Ukoliko se u bilo kojoj fazi kaskade odredi da određeni potprozor ne sadrži lice, isti će biti odbačen. Detekcija lica biti će uspješna samo ako potprozor prođe sve faze kaskade klasifikatora tj. kada ga kaskada označi pozitivnim. Grafički prikaz ovog procesa prikazan je na Slici 2.5. Budući da je za evaluaciju klasifikatora koji sadrže veći broj značajki potrebno više vremena, prilikom optimizacije okvira za detekciju moramonaći balans između tri faktora: broja faza kaskade, broja značajki unutar svake fazei praga vrijednosti svake od faza. [4]



Slika 2.5. Proces kaskade klasifikatora
(izvor: https://zir.nsk.hr/islandora/object/algebra%3A252)

2.2. Histogram orijentiranih gradijenata

Druga važna metoda na području detekcije objekata jest metoda histograma orijentiranih gradijenata (eng. *Histogram of Oriented Gradients*) ili skraćeno HOG. Nakon objave znanstevnog rada Navneet Dalala i Billa Triggsa 2005. godine, HOG metoda postaje jedna od klasičnih metoda računalnog vida u detekciji objekata. [3,7] Temelji se na prebrojavanju pojavljivanja orijentacija gradijenata u dijelovima slike predstavljene pomoću mreže ćelija. Dalal i Triggs su u svom radu opisali postupak detekcije ljudi tj. pješaka na slikama i u videozapisima pomoću HOG detektora. Osnovna ideja HOG metode je da se oblik nekog objekta na slici može opisati uz pomoć raspodjele intenziteta gradijenata ili smjerova rubova. To se radi na način da se slika podijeli na prostorne ćelije gdje pojedina ćelija akumulira smjerove gradijenata. Nadalje, svaka se ćelija sastoji od piksela za koje se računaju gradijenti koji se prethodno grupiraju za svaku ćeliju. Takvo spajanje histograma unutar svake ćelije naziva se HOG deskriptor. [5, 7, 8] Na Slici 2. 6. prikazan je „lanac“ izdvajanja značajki i detekcije objekata tj. proces same HOG detekcije.

Slika 2.6. Proces HOG detekcije objekata
(izvor: http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf)

2.2.1. Gama i boja normalizacija

Ovo je prvi korak HOG procesa. U svom radu, Dalal i Triggs su procijenili nekoliko prikaza ulaznih piksela uključujući nijanse sivih tonova te RGB i Lab prostora boja uz gama normalizaciju. Rezultat je da takva normalizacija ima skroman učinak na performanse. Nadalje, RGB i Lab prostori boja daju usporedive rezultate, ali u usporedbi sa sivim tonovima, performanse se smanjuju za 1,5% na 10^{-4} FPPW (*False Positive per Window*). Korijen gama kompresije svakog kanala boje povećava performansu na niskim FFPW-ovima, međutim budući da je log kompresija prejaka, smanjuje učinkovitost za 2% na 10^{-4} FPPW-a. [7, 8].

2.2.2. Izračun gradijenata

Performase detektora osjetljive su na način na koji se izračunavaju gradijenti. Stoga, za njihov izračun, Dalal i Triggs koriste Gaussovo uglađivanje sa diskretnim derivacijskim maskama. Testirano je nekoliko skala uglađivanja uključujući $\sigma = 0$. Testirane maske uključivale su različite 1-D derivate točaka (necentrirani [-1,1], centrirani [-1,0,1] i kubično korigirani [1,-8,0,8,-1], kao i 3x3 Sobel maske i 2x2 dijagonale kao najkompaktnije centrirane 2-D derivacijske maske.[6, 7] Nadalje, uporabom većih maski performanse i uglađivanje se značajno smanjuju, npr. za Gaussovu derivaciju pomakom vrijednosti sa $\sigma = 0$ na $\sigma = 2$ povratna stopa se smanjuje s 89% na 80% pri 10^{-4} FPPW-a. Korištenjem necentriranih [-1,1] derivacijskih maski, performanse se također smanjuju za 1.5%. Kad se govori o slikama u boji, za njih se izračunavaju zasebni gradijenti za svaki kanal boje gdje se pritom uzima onaj s najvećom normom kao vektor gradijenta piksela. [7, 8]

2.2.3. Prostorno povezivanje

Svaki piksel izračunava težinsku vrijednost za kanal rubno-orijentiranog histograma koji se temelji na orijentaciji elemenata gradijenata usmjerenog na njega. Vrijednosti se zatim akumuliraju u orijentacijske ćelije koje mogu biti pravokutne ili radijalne. Ćelije su ravnomjerno raspoređene na gradijent bez predznaka ($0^{\circ}- 180^{\circ}$) ili gradijent sa predznakom ($0^{\circ}- 360^{\circ}$). Smanjivanje nazubljenosti izvršava se tako što se vrijednosti interpoliraju bilinerano između susjednih ćelija u poziciji i orijentaciji. Vrijednost može biti funkcija veličine gradijenata piksela, bilo sama veličina, korijen, kvadrat ili izrezani oblik veličine koja predstavlja meku prisutnost odnosno odsutnost ruba na pikselu. [7, 8]

2.2.4. Blok normalizacija

Normalizacija vrijednosti potrebna je da bi se dodatno smanjila osjetljivost na promjene u osvjetljenju. Vektor gradijenta je otporan na linearne promjene u osvijetljenju, međutim nije otporan na višestruko povećanje vrijednosti piksela iz razloga što se magnituda vektora također višestruko poveća. Taj je problem riješen uz pomoć euklidske (L2) norme vektora u kojoj se vrijednost svake dimenzije vektora podijeli magnitudom tog istog vektora da bi se dobile vrijednosti vektora neovisne o promjenama sveukupnog osvjetljenja. Nadalje, normaliziranjem blokova ćelija dobiju se bolje performanse umjesto da se svaki histogram normalizira zasebno. Kako bi se osiguralo da se svaka ćelija pojavljuje više puta u finalnom deskriptoru, Dalal i Triggs grupiraju ćelije u blokove veličine 2 x 2 uz pomicanje potprozora izbora grupe ćelija uvijek za jednu ćeliju u određenom smjeru. [4, 8] Primjer pomicanja normalizacijskog bloka ćelija je prikazan na Slici 2.7.

Slika 2.7. Pomicanje normalizacijskog bloka ćelija
(izvor: http://mccormickml.com/2013/05/09/hog-person-detector-tutorial/)

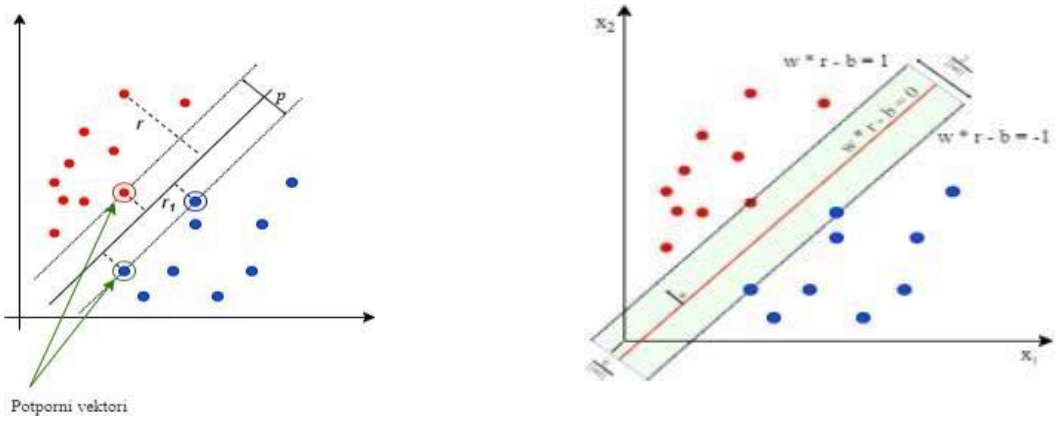
2.2.5. Prozor detekcije

Finalni deskriptor koji je dobiven normalizacijom bilo kojeg prozora detekcije veličine 64 x 128 piksela u sebi sadrži 3780 vrijednosti. Svaki se prozor detekcije dijeli u mrežu blokova veličine 7 x 15 što iznosi 105 različitih blokova ćelija. U svakom se bloku nalazi 36 vrijednosti histograma gradijenata što znači da svaki blok sadrži 4 ćelije od kojih svaka ćelija sadrži histogram od 9 vrijednosti. Konačno, množenjem svih 36 vrijednosti dobivamo ukupni broj vrijednosti sadržanih u svakom finalnom deskriptoru prozora detekcije. [4, 8]

Valja reći da HOG detektori, osim za detekciju ljudi, mogu biti trenirani i za detekciju bilo kojih objekata.

2.2.6. Metoda potpornih vektora

Pristup potpornih vektora (eng. *Support Vector Machines* – SVM) koristi se kao „dodatak“ HOG detektorima. Cilj ove metode jest pronaći optimalnu hiper ravninu između klasa fokusirajući se pritom na vrijednosti koje su prilikom treniranja stavljene na rub deskriptora klase. Takve se vrijednosti zovu potporni vektori po kojima je i sama metoda dobila ime. Sve one vrijednosti koje nisu potporni vektori, se odbacuju. Kako bi klasificirao sve ulaze u višedimenzionalni prostor, SVM gradi hiper ravninu ili skup istih. Svrha potpornih vektora je da povećaju marginu između hiper ravnina i sebe samih. Kvalitetno se odvajanje postiže kada hiper ravnina ima najveću vrijednost do podatkovne točke (koja ustvari predstavlja potporni vektor) bilo koje klase (margine). Najpoznatiji klasifikator koji predviđa svaku ulaznu klasu uzorka između dvije moguće klasifikacije je tzv. linerani SVM. Linearni SVM je ujedno i zadnji korak u procesu HOG detekcije objekata. Prema vrsti odvajanja, linerani SVM razlikuje odvajanje tvrdom odnosno mekom marginom. Na Slici 2.8 grafički su prikazani potporni vektori te hiper ravnine s marginama. [1, 6, 7]



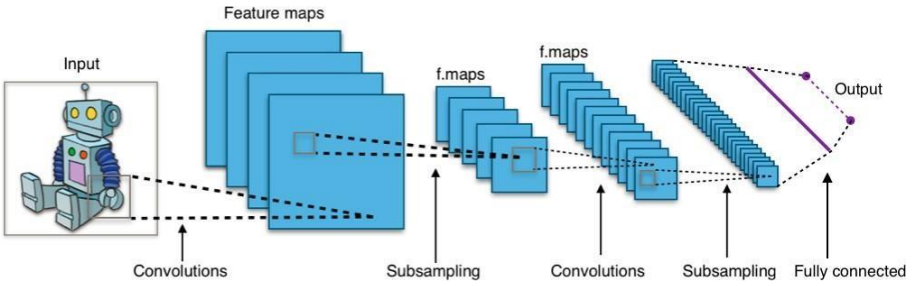
Slika 2.8. Grafički prikaz potpornih vektora (lijevo) i hiper ravnina s marginama (desno) (izvor: https://zir.nsk.hr/islandora/object/unin:3197/datastream/PDF/download)

2.3. Duboke neuronske mreže (DNN)

Duboke neuronske mreže (eng. *Deep Neural Networks*) pripadaju skupini metoda dubokog učenja. Općenito, to su metode strojnog učenja čiji je cilj da uči računalo da radi ono što mogu raditi ljudi. Drugim riječima, računalni model uči izvoditi klasifikacije izravno iz slike. Takvi modeli se danas često koriste u području računalnog vida iz razloga što mogu postići vrhunsku preciznost i točnost. Također, karakteristika tih modela je i ta da se isti treniraju uz pomoć velikog skupa podataka i neuronskih mreža s mnogo slojeva. U nastavku će se detaljnije opisati neke od najkorištenijih neuronskih mreža koje se između ostalog i koriste u svrhu detekcije objekata. [6]

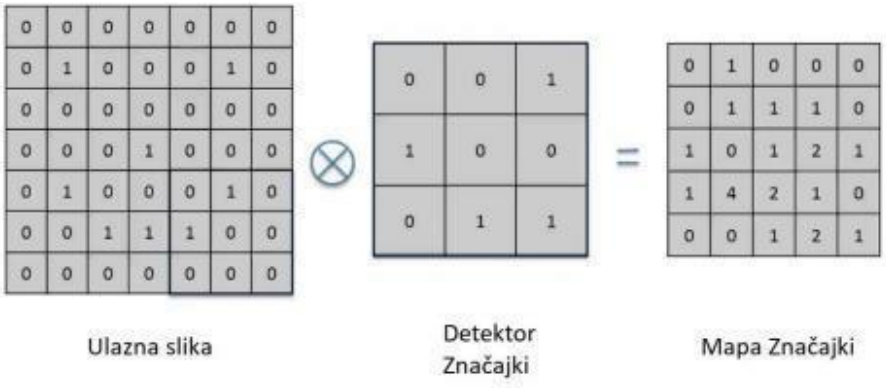
2.3.1. Konvolucijske neuronske mreže (CNN)

Konvolucijske neuronske mreže (CNN) su algoritmi dubokog učenja koji za cilj imaju da na svom izlazu daju klasifikaciju ulazne slike. Mogu se trenirati za klasifikaciju bilo koje vrste objekta, pa se iz toga razloga koriste u aplikacijama za prepoznavanje lica i detekciju objekata, ali i u robotici. Sastoje je se od tri glavna neuronska sloja, o kojima će detaljnije biti riječi u nastavku: **konvolucijski sloj**, **sloj udruživnja** i **potpuno spojeni sloj**. [4, 7]

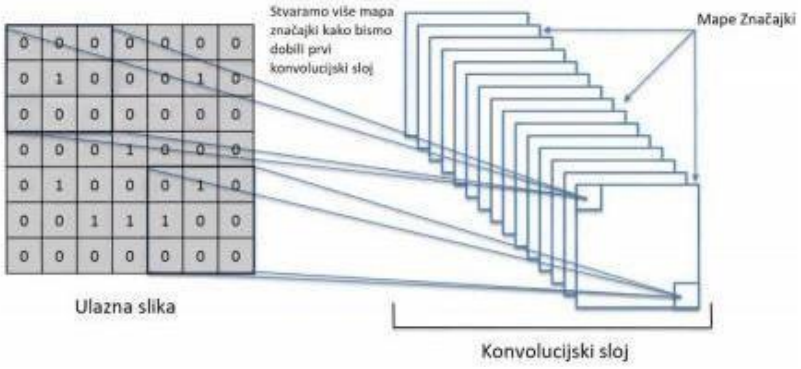


Slika 2.9. Arhitektura CNN-a
(izvor: https://en.wikipedia.org/wiki/Convolutional_neural_network)

Konvolucijski sloj nazvan je prema operaciji konvolucije koja u CNN-u služi za izdvajanje značajki unutar neke slike. Sam proces izdvajanja značajki započinje obradom ulazne slike tako što se detektor značajki povlači preko nje. Zatim se kombiniraju vrijednosti piksela ulazne slike i detektora značajki da bi se na kraju dobio izlazni produkt tj. mapa značajki (Slika 2.10). Valja napomenuti da se u jednom konvolucijskom sloju CNN-a proces izdvajanja značajki odvija više puta pa se iz tog razloga kao izlazni produkt dobije i više mapi značajki, kao što je prikazano na Slici 2.11. [4]

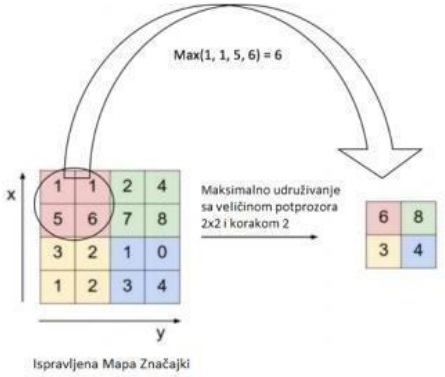


Slika 2.10. Proces izdvajanja značajki
(izvor: https://zir.nsk.hr/islandora/object/algebra%3A252)



Slika 2.11. Stvaranje višestrukih mapa značajki u CNN-u
(izvor: https://zir.nsk.hr/islandora/object/algebra%3A252)

U **sloju udruživanja** odvijaju se procesi prostornog udruživanja koji za cilj imaju smanjiti dimenzije mapa značajki, a da se pritom očuvaju najvažnije informacije koje pojedinu značajku čine značajnom. Iako postoje više vrsta udruživanja, u CNN-u se najčešće koristi tzv. maksimalno udruživanje zato što se značajke očituju kao maksimalne vrijednosti procesa konvolucije. Proces udruživanja započinje tako da se odredi veličina potprozora udruživanja koji se onda povlači preko mape značajki. Nakon svakog pomaka, uzima se maksimalna vrijednost piksela iz tog područja koja se nadalje kopira u novu mapu koja služi kao izlaz sloja udruživanja. Kao rezultat se dobije nova mapa smanjenih dimenzija, ali s očuvanim značajkama ulazne mape. Ovaj se proces ponavlja za svaku ulaznu mapu tako da je i broj izlaznih mapa jednak broju ulaznih. Također, smanjivanjem veličine mapa znatno se povećava i brzina rada same mreže. [4]



Slika 2.12. Maksimalno udruživanje
(izvor: https://zir.nsk.hr/islandora/object/algebra%3A252)

Treći i posljednji sloj u CNN arhitekturi je **potpuno spojeni sloj**. Ovaj sloj je ustvari sloj obične neuronske mreže koji je dodan na prethodna dva sloja (konvolucijski i sloj udruživanja). Svrha potpuno spojenog sloja jest spajanje značajki prethodnih slojeva te stvoriti atribute koji predviđaju klasu ulazne slike s povećanom preciznošću. Konačno, potpuno spojeni sloj pretvara mape dvodimenzionalnih značajki u jednodimenzionalni vektor značajki. [4, 7]

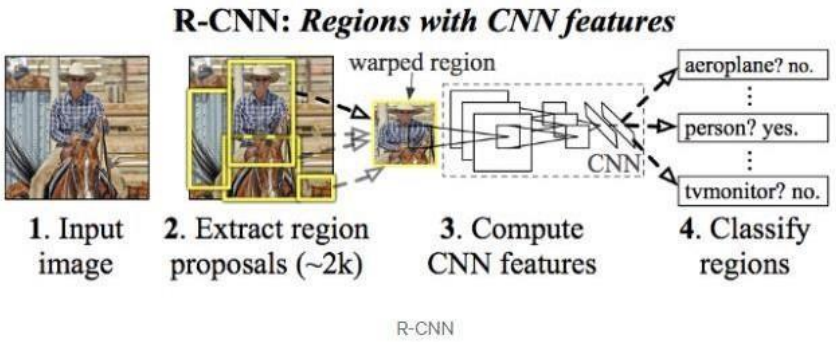
2.3.2. R-CNN

Konvolucijska mreža bazirana na regijama (eng. *Region Based Convolutional Neural Network – R-CNN*) uvedena je kako bi se zaobišao problem selektiranja i izdvajanja velikog broja regija na slici. Ross Girshick i suradnici stoga predlažu metodu tzv. selektivnog pretraživanja pomoću koje se iz slike izdvajaju samo 2000 regija koje se zovu prijedlozi regija. Selektivno pretraživanje sastoji se od 3 koraka: [13, 14]

1. Generiranje početne sub-segmentacije, generiranje velikog broja regija objekta

2. Rekurzivna kombinacija sličnih regija u veće regije pomoću selektivnog algoritma

3. Pomoću generiranih regija dolazi se do konačnih prijedloga regija objekta

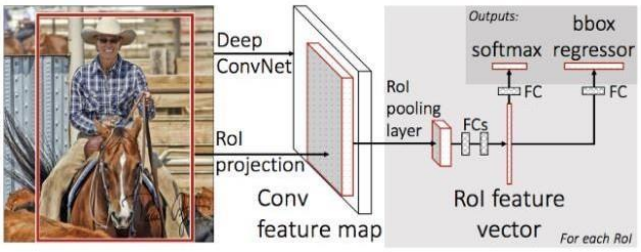


Slika 2.13. R-CNN proces
(izvor: https://sci-hub.se/10.1109/TNNLS.2018.2876865)

Unatoč prednostima u svom pristupu, R-CNN još uvijek ima problema, prije svega, u brzini izvedbe jer je potrebno mnogo vremena kako bi se mreža osposobila budući da se koriste 2000 regija za klasifikaciju po slici. Drugi problem je taj da se R-CNN ne može implementirati u stvarnom vremenu jer je potrebno oko 47 sekundi za svaku testnu sliku, iako vrijeme varira ovisno o korištenom računalu. Konačno, selektivni algoritam je fiksni algoritam što znači da može dovesti i do generiranja loših regija. [14]

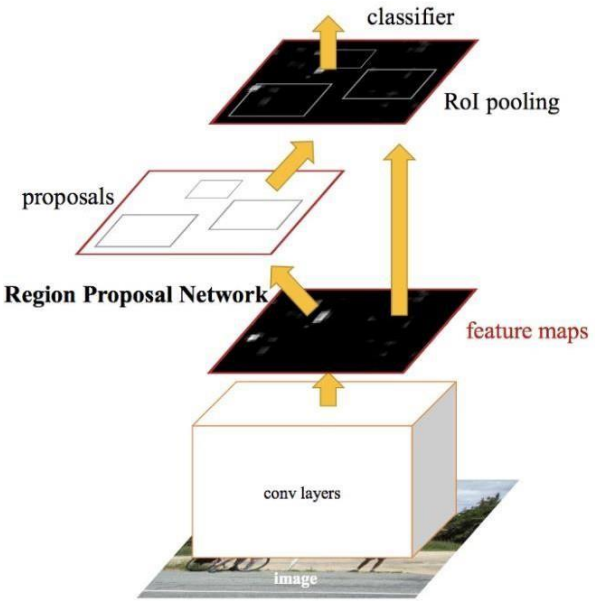
Kako bi se popravili navedeni nedostatci, R-CNN je postupno nadograđivan kroz tri nova algoritma. U nastavku slijedi kratki opis svakog od njih:

a) **Brzi R-CNN** – Ova metoda ispravlja nedostatke „običnog“ R-CNN-a te također poboljšava točnost i brzinu. Nadalje, daje i bolju kvalitetu detekcije. Kao ulaz uzima cijelu sliku i skup prijedloga objekta. Zatim mreža prvo obrađuje cijelu sliku pomoću više konvolucijskih slojeva za izradu konvolucijske mape značajki. Za svaki prijedlog objekta, sloj regija interesa (RoI – *Region of Interest)* izdvaja vektor značajki fiksne duljine iz mape značajki. Svaki se vektor značajki, zatim plasira u niz potpuno spojenih slojeva koji se onda granaju u dva izlazna sloja. Jedan je sloj zadužen za procjenu vjerojatnosti nad klasama objekata dok drugi sloj daje četiri broja za svaku klasu objekta. Konačno, svaki skup od četiri vrijednosti predstavlja granični okvir za jednu od klasa. [6, 14] Na Slici 2.14. prikazana je arhitektura brzog R-CNN-a.



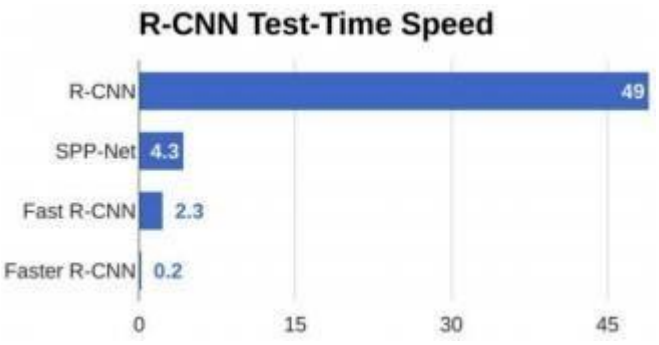
Slika 2.14. Arhitektura brzog R-CNN-a
(izvor: https://sci-hub.se/10.1109/TNNLS.2018.2876865)

b) **<u>Brži R-CNN</u>** – algoritam bržeg R-CNN-a za razliku od prethodna dva, koji koriste selektivno pretraživanje, eliminira isti i omogućava mreži da sama nauči prijedloge regija, s obzirom na to da je selektivna pretraga spor i dugotrajan proces koji utječe na performanse mreže. Brži R-CNN čine dva modula: prvi predstavlja duboku konvolucijsku mrežu koja predlaže regije, dok je drugi modul detektor brzog R-CNN-a koji koristi predložene regije. Kao i kod brzog R-CNN-a, i ovdje slika služi kao ulaz u konvolucijsku mrežu koja daje konvolucijsku mapu značajki. Međutim, umjesto korištenja selektivne pretrage na mapi značajki za identifikaciju prijedloga regija, koristi se zasebna mreža za predviđanje prijedloga regija – RPN (eng. *Region Proposal Network*). Prijedlozi predviđenih regija se onda preoblikuju pomoću RoI „pooling“ sloja (eng. *Region of Interest*) koji se koristi za klasifikaciju slika unutar predloženog područja te za predviđanje vrijednosti pomaka graničnih okvira. [6, 12, 14] Arhitektura bržeg R-CNN-a prikazana je na Slici 2.15.



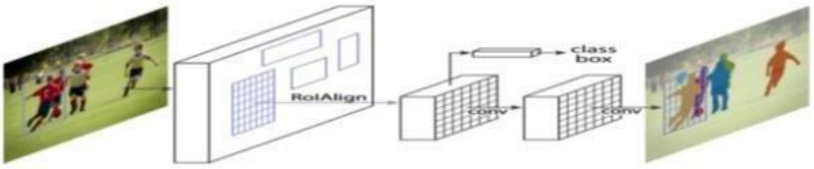
Slika 2.15. Arhitektura bržeg R-CNN-a
(izvor: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e)

Brži R-CNN je značajno brži od prethodna dva algoritma pa se zbog toga može koristiti za detekciju objekata u realnom vremenu. [14] Na Slici 2.16. prikazanaje usporedba brzina R-CNN algoritama.



Slika 2.16. Usporedba brzina R-CNN algoritama
(izvor: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e)

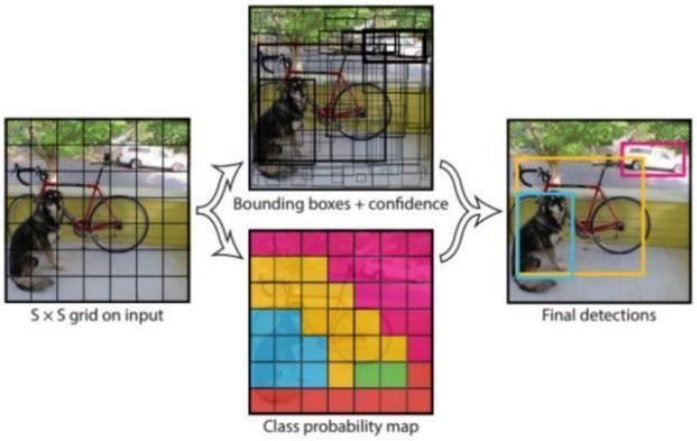
c) **<u>Maskirani R-CNN</u>** – ova metoda R-CNN-a je najefikasnija i najfleksibilnija u odnosu na prethodne. Osim što učinkovito otkriva objekte na slici, maskirani R-CNN istovremeno stvara i visokokvalitetnu segmentacijsku masku za svaku instancu, po čemu je i dobio ime. Koncept maskiranog R-CNN-a dosta je sličan bržem R-CNN-u. Međutim, brži R-CNN ima dva izlaza za svaki objekt, oznaku klase i odstupanje graničnog okvira, dok se maskirani proširuje sa trećim koji predstavlja masku objekta. Model maskiranog R-CNN-a dijeli se u dvije faze: prva faza identična je proceduri bržeg R-CNN-a tj. uključuje RPN mrežu koja predviđa prijedloge regija. U drugoj se fazi generira binarna maska za svaku regiju interesa (RoI) te se uz predviđanje klase i pomak graničnog okvira ujedno primijenjuje klasifikacija i regresija okvira. Također, maskirani R-CNN uvodi i tzv. *pixel-to-pixel* poravnanje što je i glavni nedostatak prethodna dva R-CNN-a.[6, 12, 15] Model makiranog R-CNN-a prikazan je na Slici 2.17.



Slika 2.17. Maskirani R-CNN
(izvor: https://sci-hub.se/10.1109/TNNLS.2018.2876865)

2.3.3. YOLO

YOLO metoda jedna je od najnovijih metoda dubokog učenja razvijena za potrebu detekcije objekata na slikama. Naziv „YOLO“ zapravo je engleska skraćenica za „*You Only Look Once*“ iz razloga što se pomoću YOLO sustava određena slika pogleda samo jednom da bi se predvidjelo koji su objekti na istoj i gdje se nalaze. Glavna značajka YOLO-a je da komponente za detekciju objekata objedinjuje u jednu konvolucijsku mrežu i na taj način samu detekciju svodi na regresijski problem. Za predviđanje graničnog okvira koriste se značajke iz cijele slike što znači da mreža cijelu sliku i sve objekte na slici razmatra globalno.[6, 12, 16]



Slika 2.18. Princip rada YOLO metode
(izvor: https://sci-hub.se/10.1109/TNNLS.2018.2876865)

Princip rada YOLO metode je slijedeći: ulazna slika se dijeli na S x S dijelova te se propušta kroz konvolucijsku neuronsku mrežu za izračun značajki. Zatim se vrši proces linerane regresije pomoću dva potpuno spojena sloja koji za svaku lokaciju generiraju B graničnih okvira. Svaki je granični okvir određen s 5 vrijednosti – x, y, w, h i rezultatom pouzdanosti. X i y koordinate predstavljaju centar okvira u odnosu na ćeliju kojoj pripada, dok se koordinate w i h odnose na širinu i visinu okvira. Pouzdanost pak, odražava da li je objekt prisutan ili ne, a to ovisi o vjerojatnosti da predloženi okvir sadrži objekt i IoU (eng. *Intersection over Union*) mjeri između predviđenog i stvarnog graničnog okvira (Formula 2.2.):

$$\boldsymbol{Pr(Object) * IoU_{pred}^{truth}}$$

Formula 2.2.

Pouzdanost je jednaka nuli ako granični okvir ne sadrži objekt. Pojedina ćelija je zadužena za predviđanje kategorije objekta samo ako se centar objekta nalazi unutar nje, u protivnom to ne vrijedi. Rezultat pouzadnosti za svaku klasu okvira dobiva se množenjem vjerojatnosti uvjetne klase i predviđanja pouzdanosti za pojedinačne okvire, tj. Formulom 2.3.:

$$\boldsymbol{Pr(Class_i|Object) * Pr(Object) * IoU_{pred}^{truth} = Pr(Class_i) * IoU_{pred}^{truth}}$$

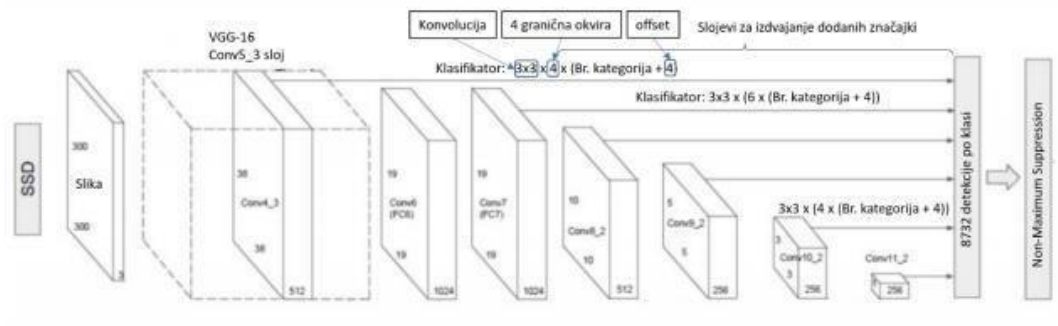
Formula 2.3.

Konačno, YOLO algoritam na svom izlazu daje ukupno S x S x (B x 5 + C) vrijednosti koje su vezane za predikciju graničnog okvira i kategorije objekta. Bez obzira na broj graničnih okvira, za svaku se ćeliju koristi samo jedan vektor C koji se onda množi sa svakim graničnim okvirom unutar ćelije, a rezultat nam govori koliko je objekt dobro obuhvaćen.[6, 12, 16]

Kao što je već rečeno, YOLO objedinjuje sve pojedinačne komponente u jedinstveni i optimizirani model što nudi jednostavnost i bržu detekciju. Međutim, glavni nedostatak YOLO-a je još uvijek neprecizna lokalizacija objekata u odnosu npr. na brzi R-CNN koji je znatno precizniji u tom pogledu.[6] YOLO algoritam, unatoč nedostatcima, obećava na području detekcije objekata budući da se konstantno poboljšava i nadograđuje pa je tako dosad dobio čak 4 novije verzije: **YOLO v2**, **YOLO v3**, **YOLO v4**, te najnoviji **YOLO v5** koji je trenutno jedan od najboljih algoritama za detekciju objekata.

2.3.4. SSD

SSD algoritam (eng. *Single Shot Detector*) je algoritam dubokog učenja koji detekciju obavlja u samo jednom koraku za razliku od metoda baziranih na prijedlozima regija koje su opisane ranije. Princip SSD-a je da ulazna slika prvo prođe kroz CNN mrežu zbog izdvajanja mape značajki. Veličina mape označava se sa *M* x *N* x *P* gdje su *M* i *N* brojevi lokacija, a *P* oznaka za kanal. Nadalje, za svaku se lokaciju dobije *K* graničnih okvira različitih omjera i veličina. Svaki je okvir definiran s *C* kategorija (klasa) i 4 vrijednosti dimenzija okvira. U konačnici, ukupan broj izlaznih vrijednosti dobije se iz izraza – (*C* + 4) x *K* x *M x N*. Zbog prisutnosti višestrukih graničnih okvira, SSD algoritam se još naziva i „*Single Shot MultiBox Detector*“. [12, 17]



Slika 2.19. Arhitektura SSD-a
(izvor: https://sci-hub.se/10.1109/TNNLS.2018.2876865)

Za bolju preciznost detekcije, određene mape značajki se propuštaju kroz 3 x 3 konvolucijski sloj kao što je prikazano na Slici 19. Uzmimo na primjer, prvi sloj Conv4_3 koji je veličine 38 x 38 x 512 i koji se propušta kroz spomenuti 3 x 3 konvolucijski sloj. Ako postoje 4 granična okvira, svaki će okvir imati (*C* + 4) izlaza. Pa je tako izlaz za sloj Conv4_3 jednak 38 x 38 x 4 x (*C* +4). Nadalje, ako pretpostavimo da postoji 20 klasa objekata te dodatna pozadinska klasa, izlaz će onda biti 38 x 38 x 4 x (21 + 4) = 144,400. Što se tiče broja graničnih okvira, on bi iznosio 38 x 38 x 4 = **5776**. Po istom principu broj okvira se izračunava i za ostala 5 sloja s time da se slojeva Conv7, Conv8_2 i Conv9_2 uzima po 6 okvira za svaku lokaciju. Na kraju, ukupan zbroj svih graničnih okvira za sve slojeve je 8732. Ako

taj broj, recimo, usporedimo sa YOLO detektorom koji za svaku lokaciju ima samo po dva granična okvira, a ukupno njih 98, primijećujemo da SSD ima značajno više graničnih okvira, što ga čini preciznijim.[17]

3. Programske biblioteke

U nastavku će se nešto više reći o programskim bibliotekama koje će se koristiti kod implementacije detektora. U ovom radu koristit će se dvije najpoznatije biblioteke otvorenog koda koje se najviše koriste upravo na području računalnog vida, a to su OpenCV i Dlib.

3.1. OpenCV

OpenCV je skup biblioteka otvorenog otvorenog koda koje se bave problemima strojnog učenja i računalnog vida. OpenCV pruža zajedničku infrastrukturu za primjenu računalnog vida te za ubrzanje korištenja strojnog vida u komercijalnim proizvodima. Poseban naglasak stavlja na procesiranje slika u stavrnom vremenu. Sadrži preko 2500 optimiziranih algoritama, uključujući nove i moderne algoritme strojnog učenja i računalnog vida. Algoritmi imaju širok raspon primjena, a to između ostalog uključuje detekciju lica, klasifikaciju ljudskih radnji u videu, identifikaciju objekata, praćenje pokreta kamere te praćenje pokreta očiju. OpenCV je modularno strukturiran što znači da svi paketi posjeduju nekoliko zajedničkih statičkih biblioteka kao što su *core, video* i *objdetect*. OpenCV biblioteke podržavaju sučelja za C++, C, Python, Javu i Mathlab programske jezike kao i operacijske sustave Windows, Linux, Android i MacOS. Mnoge poznate tvrtke poput Google-a, Microsoft-a i Sony-a koriste OpenCV u svrhu razvoja aplikacija koje se u konačnici koriste u robotici, medicini ili autoindustriji.[18]

3.2. Dlib

Dlib predstavlja moderan set alata koji sadrži algoritme strojnog učenja te algoritme za izradu kompleksnih softvera u C++ programskom jeziku radi rješavanja problema u stavrnom svijetu. Dlib se koristi u poslovne i akademske svrhe na širokom rasponu područja uključujući robotiku i ugrađene uređaje. Dlib biblioteka popraćena je kvalitetnom dokumentacijom koja pokriva sve klase i funkcije te također na svojoj službenoj stranici nudi velik broj primjera korištenja istih. Podržan je na svim poznatim operacijskim sustavima (Windows, Linux, MacOS) te ne zahtijeva nikakve druge softverske pakete za rad kao niti dodatnu instalaciju na računalo. Za razliku od OpenCV-a, Dlib nije nužno namijenjen za područje računalnog vida te podržava samo C++ sučelje, međutim može ga se koristiti uz potporu OpenCV biblioteke u programskom jeziku Python.[19]

4. Implementacija detektora za prepoznavanje lica u Pythonu i ispitivanje njihove preciznosti

U praktičnom dijelu rada provjeravat će se preciznost i točnost tri detektora za prepoznavanje lica čiji su općeniti algoritmi opisani prethodno – Haar Cascade, DNN i HOG. Kao što je spomenuto, umjesto detekcije različitih objekata, ovdje će fokus biti isključivo na ljudskom licu. U razmatranje će se uzeti 10 fotografija na kojima će se ispitati sposobnost detekcije lica svakog od navedena tri detektora, kao i njihova preciznost. Fotografije su podijeljene u „kategorije“: lice u frontalnom položaju, više lica u frontalnom položaju, lice iz profila, lice u spuštenom položaju i lice s okluzijom (prekriveno lice). Sve fotografije će biti ispitane u svojim originalnim dimenzijama. Kao mjere preciznosti koristit će se IoU (*Intersection over Union*) gdje je to moguće, te prosječna preciznost. Na kraju će se napraviti usporedba rezultata za sva 3 detektora te procijeniti koji ima najveću preciznost.

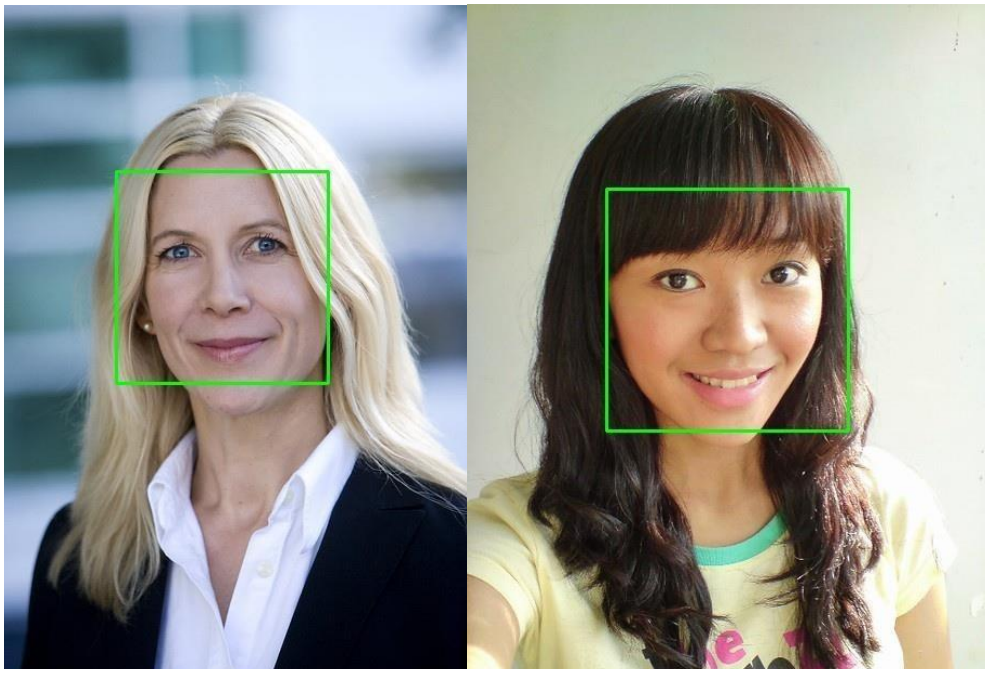
4.1. Haar Cascade detektor

Haar Cascade detektor bazira se na ranije opisanom Viola-Jones algoritmu čiji je razvoj ujedno i bio s naglaskom detekcije ljudskog lica. Detektor će biti implementiran u Python programskom jeziku uz korištenje OpenCV biblioteke. Slijedi programski kod za implementaciju detektora:

```
#import opencv biblioteke
import cv2

#inicijalizacija haar cascade modela
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

#učitavanje slike i definiranje funkcije za crtanje okvira
img = cv2.imread('slika1.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray, 1.1, 4)

for (x, y, w, h) in faces:
    cv2.rectangle(img, (x,y), (x+w, y+h), (0, 255, 0), 2)

#prikaz output slike
cv2.imshow('img', img)
```

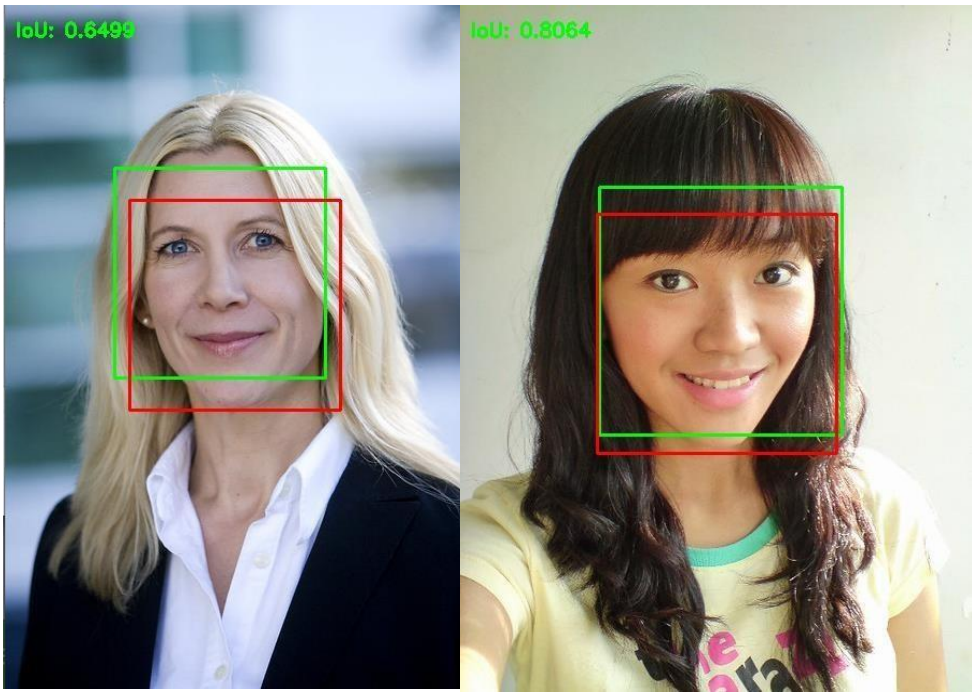
Izvor: https://github.com/opencv/opencv/tree/master/data/haarcascades

Nakon što smo izvršili implementaciju, sada možemo provjeravati svaku sliku pomoću *Haar Cascade* detektora. Za početak, provjeravamo odabrane slike s licem u frontalnom položaju. Rezultat je sljedeći:



Slika 4.1. Lice u frontalnom položaju (Haar Cascade)

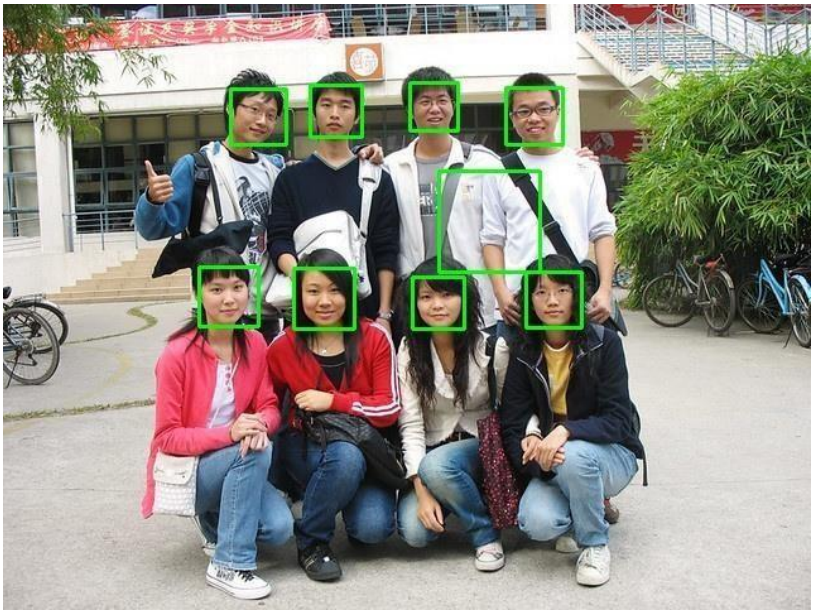
Kao što je već rečeno u uvodu, za procjenu preciznosti koristit će se IoU mjera. U načelu, to je evaluacijska mjera koja nam govori kolika je preciznost detekcije nekog objekta. Računa se kao količina graničnog okvira koji se preklapa s tzv. „*ground-truth*“ graničnim okvirom koji se dijeli sa ukupnom površinom oba okvira. Izračun IoU mjere za obje slike prikazan je na Slici 4.2. Crveni okvir predstavlja predviđeni okvir dok je zeleni stvarni okvir koji je dobiven detekcijom.



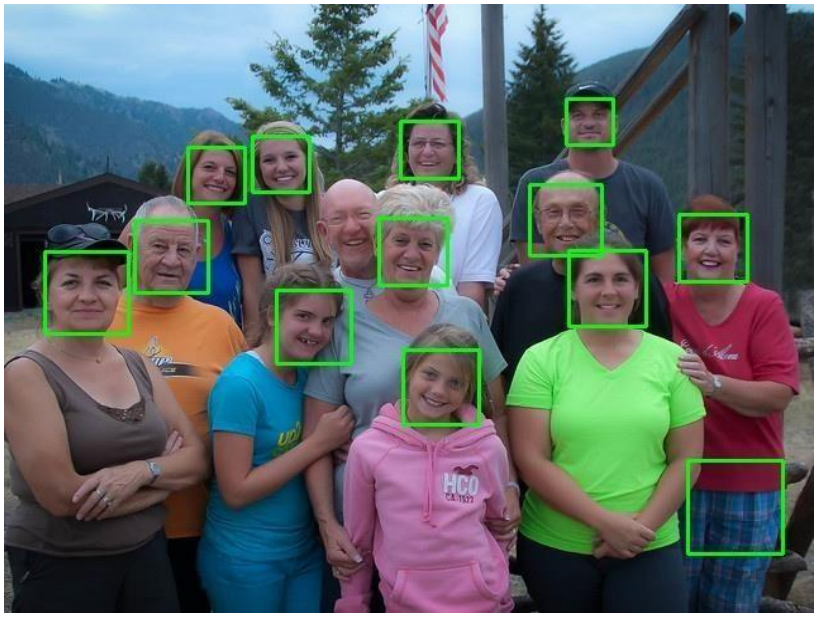
Slika 4.2. Lice u frontalnom položaju (Haar Cascade - IoU)

Iz priloženog se može vidjeti iznos IoU mjere za obje slike. Može se zaključiti da je ovo relativno precizna detekcija s obzirom da je IoU veći od 0.5 pogotovo kod druge slike čiji IoU iznosi čak 0.8 što je prilično dobar rezultat.

Sada ćemo ispitati preciznost za dvije slike koje sadrže više lica u frontalnom položaju. Rezultat je sljedeći:



Slika 4.3. Više lica u frontalnom položaju 1 (Haar Cascade)



Slika 4.4. Više lica u frontalnom položaju 2 (Haar Cascade)

U ovom slučaju kad se na fotografiji nalazi više od jednog lica, odnosno kad lice nije u krupnom planu, možemo vidjeti da *Haar* detektor ima određenih poteškoća u detekciji istih. Na prvoj fotografiji, detektor je uspio prepoznati svih 8 lica, međutim označio je i dio na kojem se ne nalazi lice iako ga je prepoznao kao takvog. Na drugoj smo fotografiji također dobili pogrešnu detekciju lica u donjem desnom kutu, ali za razliku od prve fotografije na kojoj su sva lica prepoznata, na drugoj detektor nije uspio prepoznati jedno lice. U nastavku ćemo izračunati IoU za obje fotografije na isti način kao i kod prethodne dvije uz razliku što ćemo za svako lice izračunati zasebni IoU, te ćemo za krajnji uzeti onaj najveći. Pogrešne detekcije ćemo zanemariti, s obzirom da lica nisu prepoznata pa IoU iznosi 0.

Slika 4.5. Više lica u frontalnom položaju 1 (Haar Cascade - IoU)

Slika 4.6. Više lica u frontalnom položaju 2 (Haar Cascade - IoU)

Najviši IoU iznosi otprilike 0.8 za obje slike. Međutim, detektor ima po jednu pogrešnu detekciju kod svake slike iz čega se može zaključiti da su kod *Haar Cascade* detektora pogreške moguće ako jedno lice nije u krupnom planu te da pozadinski objekti mogu utjecati na ispravnost detekcije.

Za Sliku 4.3. izračunat ćemo i mAP (eng. *Mean Average Precision*) mjeru kako bismo na kraju matematički mogli usporediti sva tri detektora. Da bismo izračunali mAP, potrebno je prije toga izračunati preciznost (PPV), odziv (TPR), F1 mjeru te točnost (ACC). Za prag IoU mjere uzet ćemo 0.5. Preciznost (PPV) računamo prema Formuli 4.1:

$$PPV = \frac{\text{TP } (true\ positive)}{\text{TP } (true\ positive) + \text{FP } (false\ positive)}$$

Formula 4.1.

TP označava broj točnih detekcija objekata (u našem slučaju lica), a FP broj pogrešnih detekcija. Za Sliku 4.3. TP iznosi 8, a FP 1, zbog toga što je detektor prepoznao 8 lica te jedno područje na kojem nema lica pa je stoga zabilježena pogrešna detekcija. Sada te vrijednosti uvrštavamo u gore navedenu formulu kako bismo izračunali PPV:

$$PPV = \frac{8}{8+1} = \frac{8}{9} \approx 0.88888888889$$

Odziv (TPR) računamo prema Formuli 4.2. gdje umjesto FP uvrštavamo FN koji označava negativnu detekciju:

$$TPR = \frac{\text{TP } (true\ positive)}{\text{TP } (true\ positive) + \text{FN } (false\ negative)}$$

Formula 4.2.

Budući da nemamo negativne detekcije za testiranu sliku, uvrštavamo samo vrijednost TP koja iznosi 8:

$$TPR = \frac{8}{8+0} = \frac{8}{8} = 1$$

Nadalje, točnost (ACC) računamo prema Formuli 4.3.:

$$ACC = \frac{\text{TP } (true\ positive)}{\text{TP } (true\ positive) + \text{FP } (false\ positive) + \text{FN } (false\ negative)}$$

Formula 4.3.

Uvrštavanjem dobivamo rezultat za točnost:

$$ACC = \frac{8}{8+1+0} = \frac{8}{9} \approx 0.88888888889$$

Na kraju računamo i F1 mjeru prema Formuli 4.4.:

$$F1 = 2 \times \frac{\mathrm{PPV} \times \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}}$$

Formula 4.4.

Uvrštavanjem dobivamo:

$$F1 = 2 \times \frac{\frac{8}{9} \times 1}{\frac{8}{9} + 1} = 2 \times \frac{\frac{8}{9}}{\frac{17}{9}} = \frac{16}{17} \approx 0.941$$

Prosječnu preciznost (AP) definiramo kao prosječnu površinu ispod *precision-recall* krivulje za određeni skup pragova od IoU. S obzirom da testiramo preciznost za samo jednu sliku sa pragom IoU od 0.5, AP možemo izračunati i kao srednju vrijednost od 11 točaka preciznosti na *precision-recall* krivulji.

$$AP = \frac{1}{11} \sum_{0,0.1,0.2,0.3,..0.9,1} Precision(Recall_i)$$

Formula 4.5.

Uvrštavamo preciznost (*precision*) i odziv (*recall*) mjere koje smo ranije izračunali:

$$AP = mAP = \frac{1}{11} \sum_{0,0.1,0.2,0.3,..0.9,1} (11 \times 0.88888888889) \approx 0.888$$

Kao što možemo primijetiti, prosječna preciznost jednaka je preciznosti (PPV). Isto tako, i mAP je jednaka AP zato što koristimo samo jednu klasu u ispitivanju.

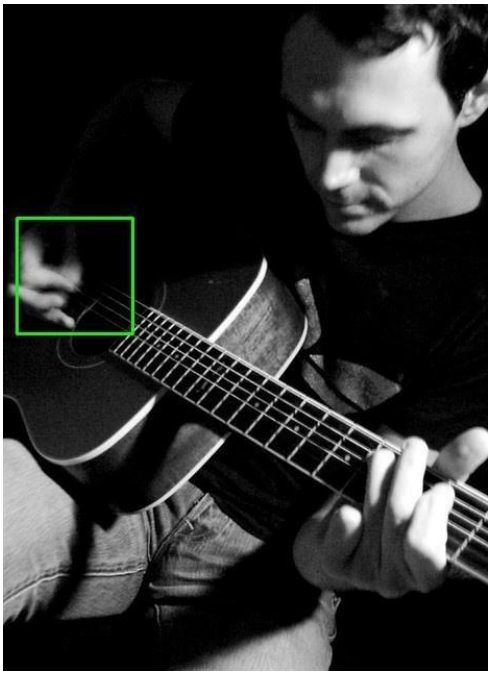
Sada ćemo provjeriti može li *Haar* prepoznati lice ako je ono slikano iz profila. Ispituju se dvije fotografije sa licem iz profila, sa lijeve i desne strane. Rezultati su sljedeći:

Slika 4.7. Lice iz profila (Haar Cascade)

Detektor na lijevoj fotografiji nije uspio prepoznati lice te nema nikakave detekcije, dok je na desnoj detekcija prisutna međutim ista je pogrešna, s obzirom na to da i na njoj lice nije prepoznato. Dakle, zaključujemo da *Haar* detektor ne može prepoznati lice ako ono nije u frontalnom položaju. Budući da detekcije lica nema, za ove fotografije ne moramo računati IoU te on iznosi 0.

Sljedeće dvije fotografije koje ćemo ispitati su one s licem koje nije u frontalnom položaju tj. spušteno prema dolje. Rezultati su sljedeći:

Slika 4.8. Lice u spuštenom položaju 1 (Haar Cascade)



Slika 4.9. Lice u spuštenom položaju 2 (Haar Cascade)

Kao i kod prethodne dvije fotografije kod kojih je lice bilo iz profila, niti ovdje detektor nije uspio prepoznati lice budući da lica nisu u potpunosti vidljiva. Na drugoj fotografiji detektor je ponovno dao pogrešno prepoznavanje za koje je utvrdio da je ljudsko lice iako to naravno nije slučaj. Ovime smo potvrdili tezu da *Haar* ima poteškoća kod raspoznavanja lica ako isto nije u potpunosti vidljivo. IoU i u ovom slučaju iznosi 0.

Na kraju nam za ispitivanje preostaju dvije fotografije na kojima je lice zaklonjeno tj. nije jasno vidljivo ili je vidljiv samo jedan dio lica. Rezultati su sljedeći:

Slika 4.10.. Lice s okluzijom 1 (Haar Cascade)

Slika 4.11. Lice s okluzijom 2 (Haar Cascade)

Na slikama iznad po prvi puta imamo situaciju da ni na jednoj ni na drugoj detektor nije zabilježio nikakvo raspoznavanje, makar i ono pogrešno.

Iz viđenog možemo zaključiti da je *Haar Cascade* detektor najprecizniji ako je na fotografiji lice u prvom i krupnom planu. To smo dokazali i prilično visokim iznosima IoU mjera koje u oba slučaja prelaze prag od 0.5. Na fotografijama s više lica, IoU je također bio visok, međutim dao je i pogrešne detekcije na kojima se lice nije nalazilo. Na ostalim fotografijama, *Haar* je bio neuspješan jer nije uspio uopće prepoznati lica budući da ista nisu bila u potpunosti vidljiva. U nastavku ćermo ispitati i preostala dva detektora te provjeriti njihovu uspješnost.

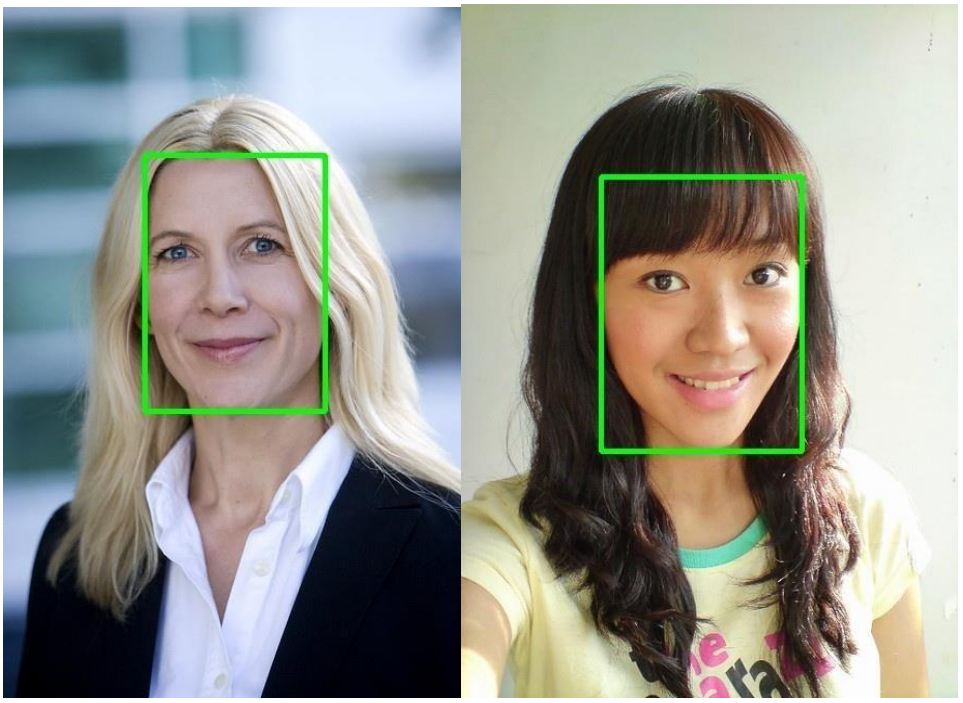
4.2. DNN detektor

Drugi detektor koji će se ispitati je detektor baziran na dubokoj neuronskoj mreži. Za potrebu detekcije koristit će se SSD (*Single Shot Detector*) mreža budući da je ista jedna od najboljih kod detekcije ljudskog lica u odnosu na ostale DNN mreže. Također, u implementaciji će se koristiti OpenCV biblioteka. Slijedi implementacija detektora:

```
#import opencv i numpy biblioteka
import cv2
import numpy as np

#inicijalizacija Caffe modela i konfiguracijske datoteke
modelFile = "res10_300x300_ssd_iter_140000.caffemodel"
configFile = "deploy.prototxt"
net = cv2.dnn.readNetFromCaffe(configFile, modelFile)

#učitavanje slike i definiranje potrebnih funkcija za crtanje okvira
img = cv2.imread('okluzija2.jpg')
h, w = img.shape[:2]
blob = cv2.dnn.blobFromImage(cv2.resize(img, (300,300)), 1.0,
(300, 300), (104.0, 117.0, 123.0))
net.setInput(blob)
faces = net.forward()
for i in range(faces.shape[2]):
        confidence = faces[0, 0, i, 2]
        if confidence > 0.5:
            box = faces[0, 0, i, 3:7] * np.array([w, h, w, h])
            (x, y, x1, y1) = box.astype("int")
            cv2.rectangle(img, (x, y), (x1, y1), (0, 255, 0), 3)

#prikaz output slike
cv2.imshow('img', img)
```

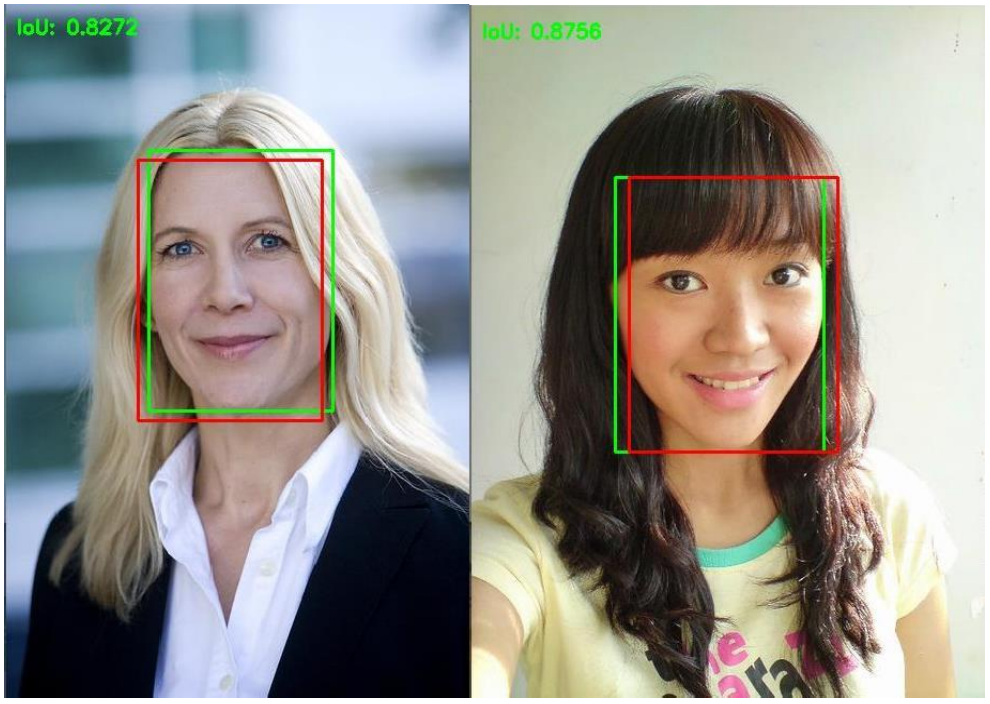
Izvor: https://github.com/opencv/opencv/tree/master/samples/dnn

Ponovno koristimo istih 10 fotografija u ispitivanju detektora. Za početak provjeravamo dvije fotografije sa licem u frontalnom položaju. Rezultat je sljedeći:



Slika 4.12.. Lice u frontalnom položaju (DNN)

Kao što možemo vidjeti, detekcija je uspješna i lica su prepoznata na obje fotografije. Sada ćemo izračunati IoU mjeru za svaku fotografiju.

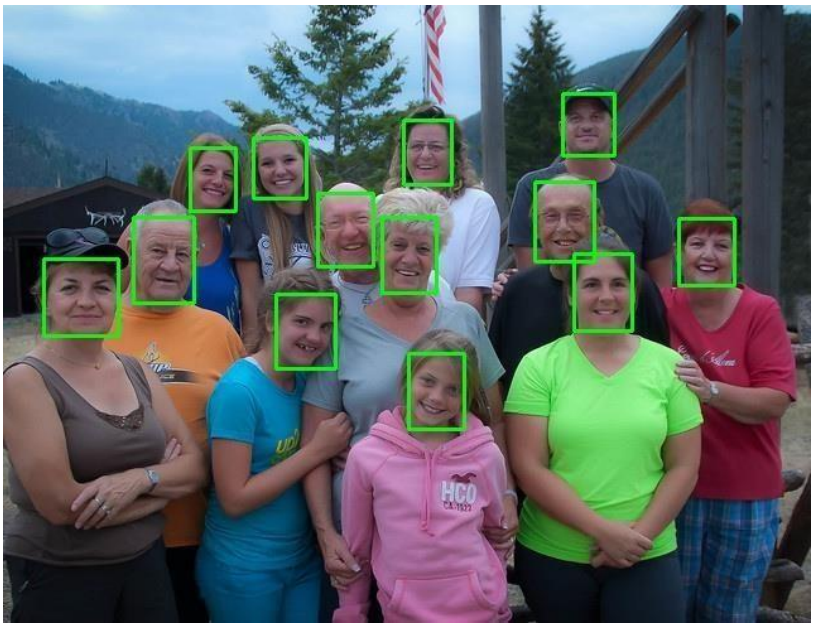


Slika 4.13. Lice u frontalnom položaju (DNN - IoU)

Možemo primijetiti da je IoU dosta visok, čak preko 0.8 za obje fotografije, što nam govori da je preciznost detekcije vrlo dobra. U odnosu na *Haar* detektor, preciznost je još i bolja, poglavito za prvu fotografiju kod koje je IoU veći za 0,2.

Prelazimo na ispitivanje fotografija sa više lica. Rezultat je sljedeći:

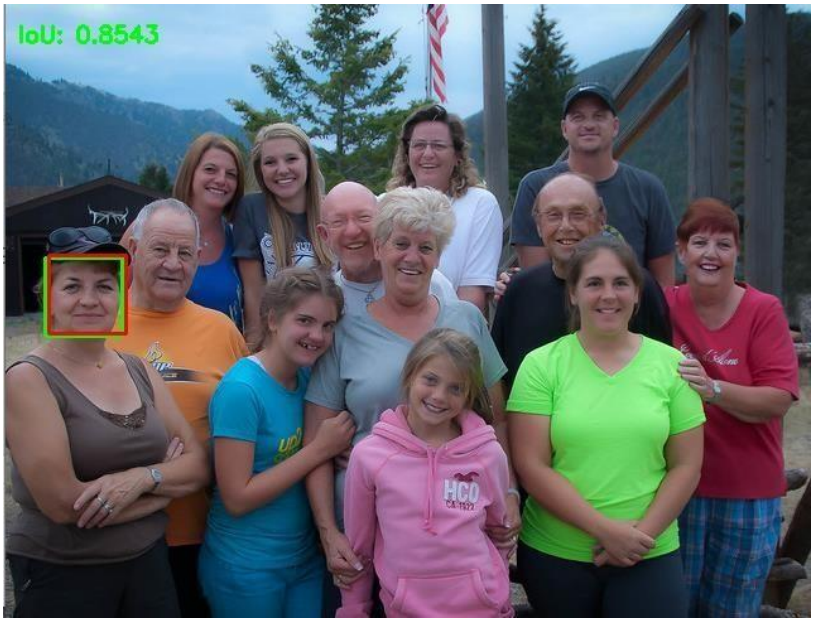
Slika 4.14. Više lica u frontalnom položaju 1 (DNN)



Slika 4.15. Više lica u frontalnom položaju 2 (DNN)

Na prvoj fotografiji, primijećujemo da je DNN prepoznao sva lica osim jednog, dok je na drugoj uspješno prepoznao svako lice. Ono što je također primjetno jest to da nema pogrešnih detekcija, kakvih smo imali u slučaju *Haar-a*. U nasatvku računamo IoU te biramo onaj najveći za svaku fotografiju.

Slika 4.16. Više lica u frontalnom položaju 1 (DNN - IoU)



Slika 4.17. Više lica u frontalnom položaju 2 (DNN - IoU)

Najviši IoU, kao što vidimo, za obje fotografije iznosi preko 0.8 što je ponovno vrlo dobra detekcija. Također, IoU se za ostale detekcije kretao između 0.7 i 0.8 što je isto prilično precizno.

Za Sliku 4.14. ponovno ćemo izračunati prosječnu preciznost. TP vrijednost iznosi 7 zato što imamo 7 od osam prepoznatih lica, FP iznosi 0, a FN iznosi 1 jer za jedno lice nema detekcije. Za početak, računamo preciznost, odziv, točnost i F1 mjeru.

$$PPV = \frac{7}{7+0} = \frac{7}{7} = 1$$

$$TPR = \frac{7}{7+1} = \frac{7}{8} = 0{,}875$$

$$ACC = \frac{7}{7+0+1} = \frac{7}{8} = 0.875$$

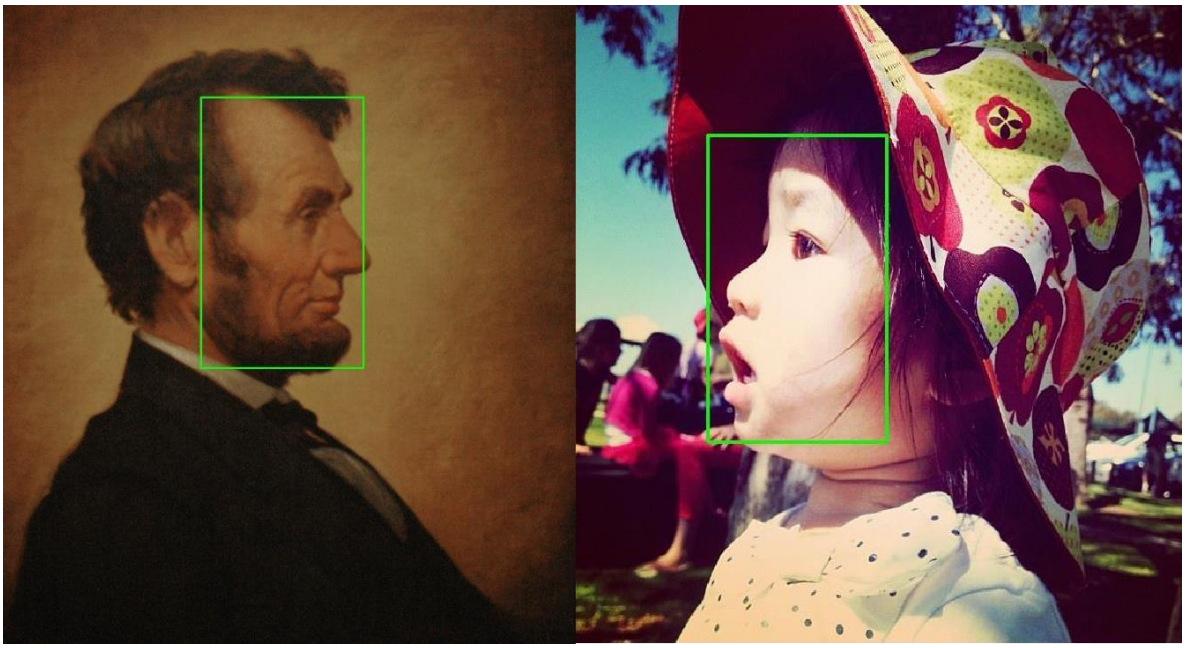
$$F1 = 2 \times \frac{\frac{7}{8} \times 1}{\frac{7}{8} + 1} = 2 \times \frac{\frac{7}{8}}{\frac{15}{8}} = \frac{14}{15} \approx 0.933$$

mAP i AP se ponovno računaju jednako zbog jedne klase. Budući da PPV iznosi 1, a TPR 0,875, uzimamo 10 od 11 *precision* točaka na *precision-recall* krivulji.

$$AP = mAP = \frac{1}{11} \sum_{0,0.1,0.2,0.3,..0.9,1} (10 \times 0.875) \approx 0.795$$

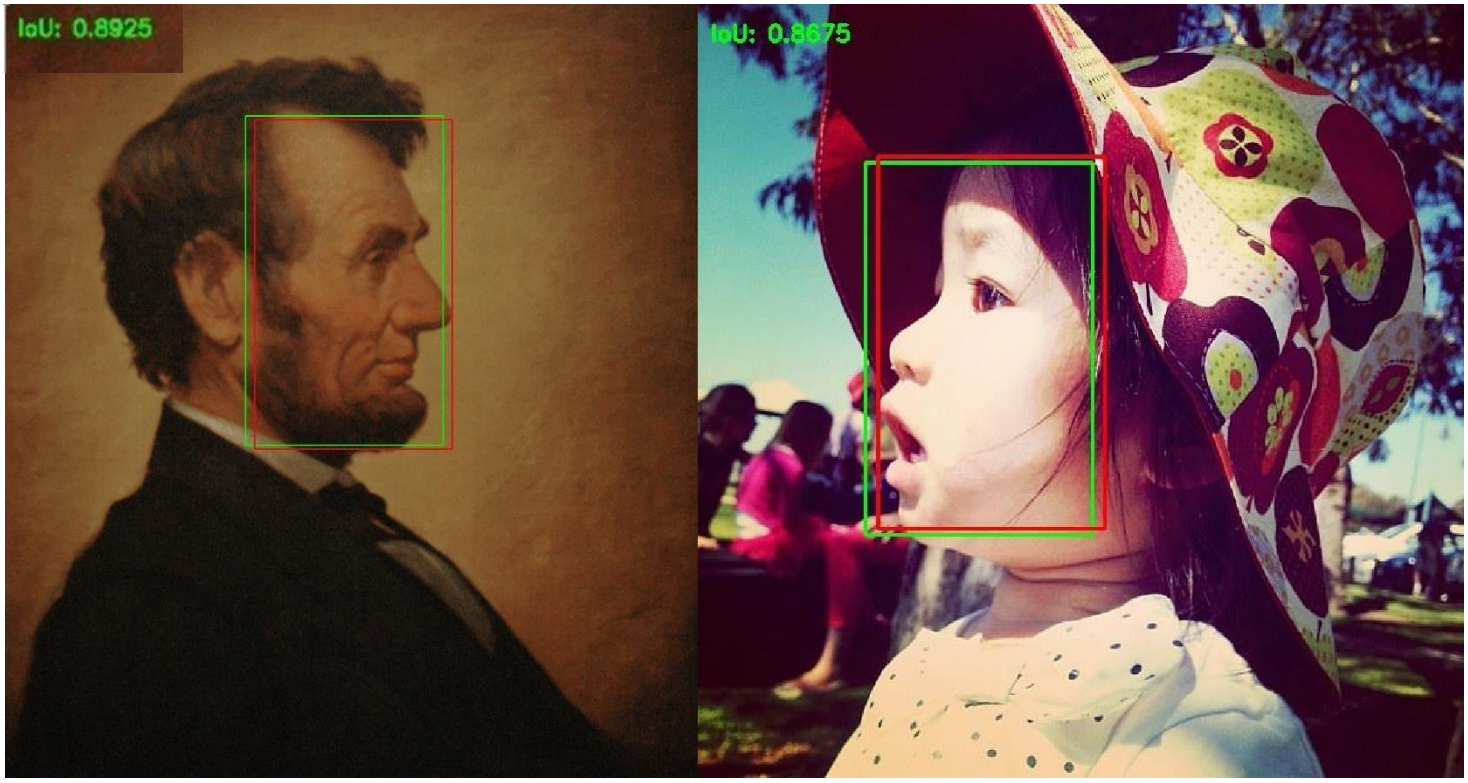
Prosječna preciznost je, kao što vidimo, 0.795 za testiranu sliku što je manje u odnosu na *Haar* detektor, međutim razlog tomu je taj što DNN nije uspio prepoznati sva lica na slici, dok je *Haar-u* to uspjelo unatoč jednoj pogrešnoj detekciji.

U nastavku ispitujemo detekciju lica iz profila. Dobili smo sljedeće rezultate:



Slika 4.18. Lice iz profila (DNN)

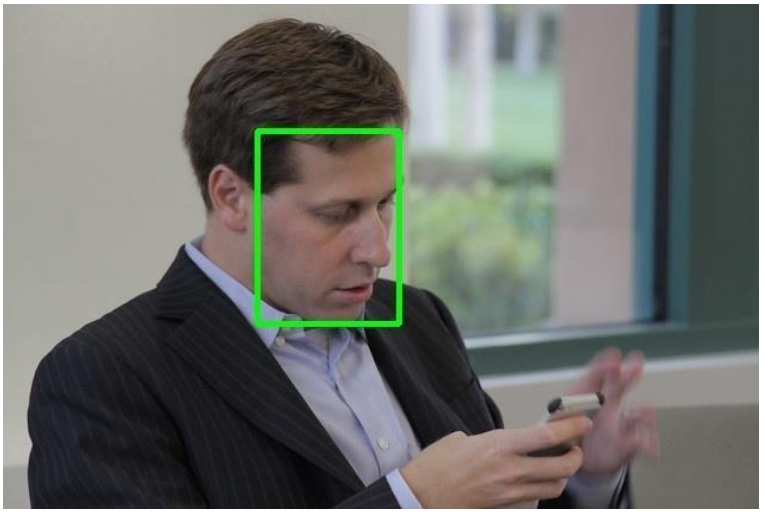
DNN je uspješno prepoznao lica iz profila na obje fotografije. Također, nema niti pogrešnih detekcija, kao što je to bio slučaj kod *Haar* detektora. S obzirom da je detekcija uspješna, možemo izračunati i IoU mjeru.



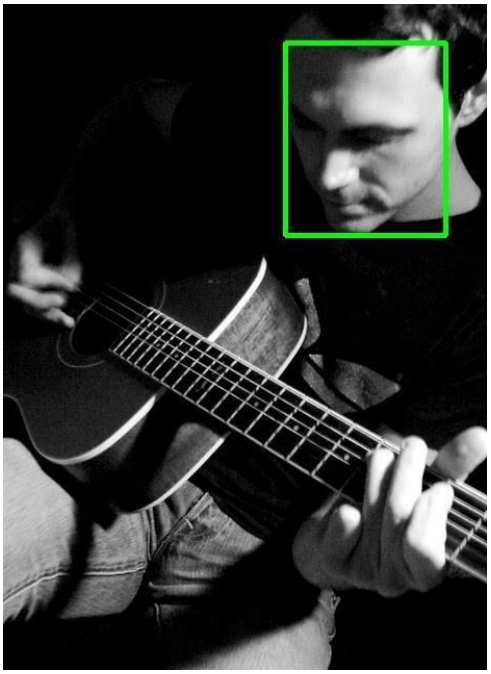
Slika 4.19. Lice iz profila (DNN - IoU)

Iz priloženog vidimo, da IoU iznosi 0.89, odnosno 0.86 što je gotovo pa savršena točnost u detekciji. Pogrešnih prepoznavanja također nema, pa možemo zaključiti da je DNN uspješniji u odnosu na *Haar* što se tiče detekcije lica iz profila.

Sljedeće dvije fotografije koje ćemo ispitati, su one na kojima je lice spušteno prema dolje. Rezultati su sljedeći:

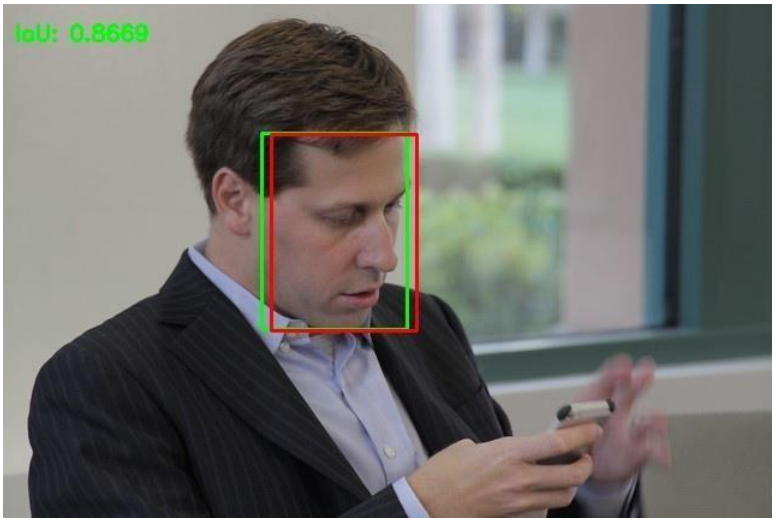


Slika 4.20. Lice u spuštenom položaju 1 (DNN)

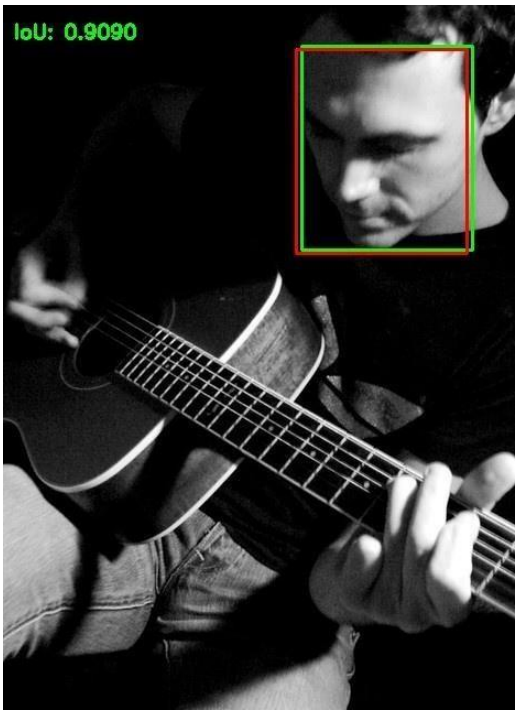


Slika 4.21. Lice u spuštenom položaju 2 (DNN)

Kao i na prethodnim fotografijama, DNN je i u ovom slučaju bio uspješan te je prepoznao lica na obje fotografije. Pogrešnih detekcija nema. Slijedi izračun IoU mjera.



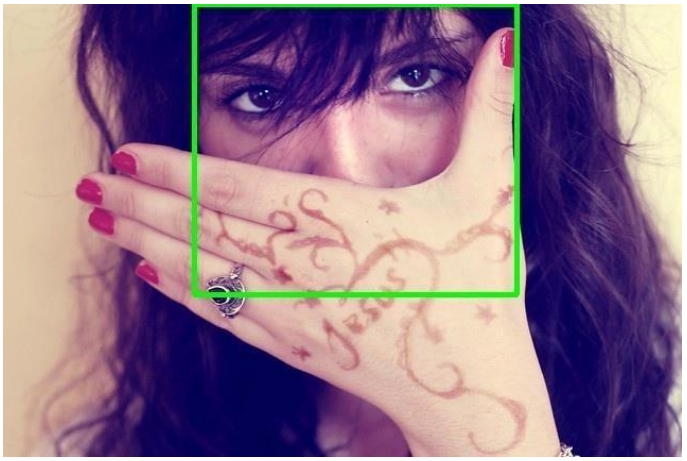
Slika 4.22. Lice u spuštenom položaju 1 (DNN - IoU)



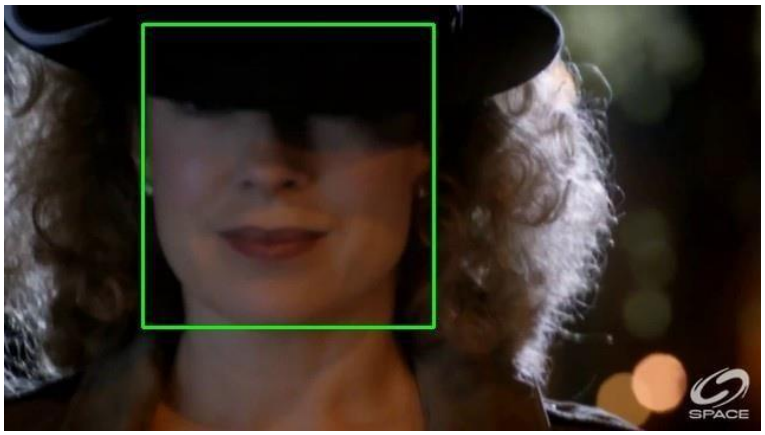
Slika 4.23. Lice u spuštenom položaju 2 (DNN - IoU)

Kao rezultat, ponovno smo dobili visoke IoU iznose, naročito na drugoj fotografiji za koju IoU iznosi čak 0.9 što je ujedno i najviši rezultat do sada.

Na kraju ispitujemo posljednje dvije fotografije, a to su one na kojima je prisutna okluzija lica. Rezultati su prikazani u nastavku:

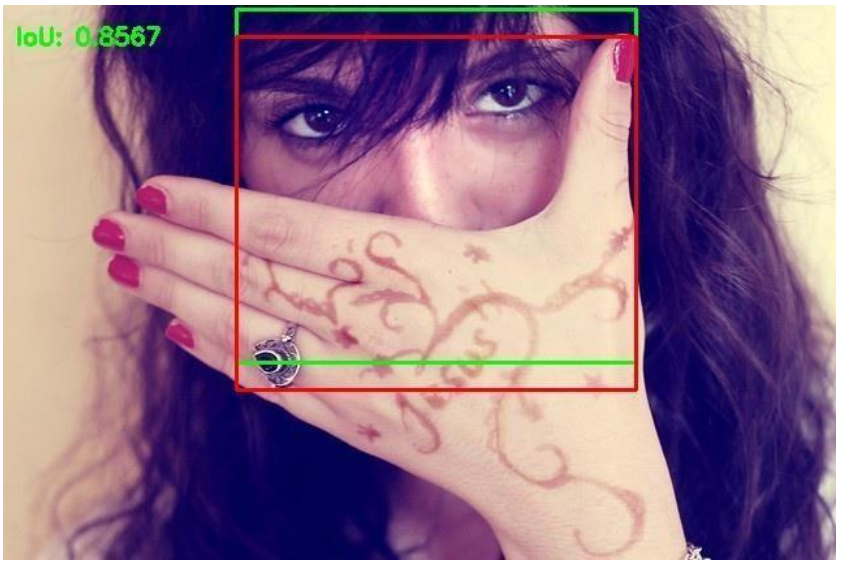


Slika 4.24. Lice s okluzijom 1 (DNN)

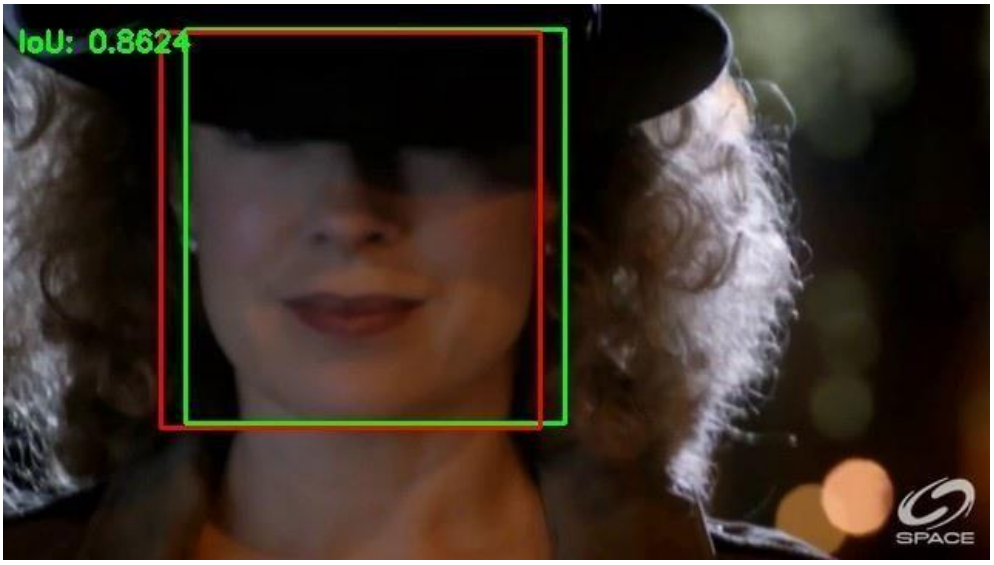


Slika 4.25. Lice s okluzijom 2 (DNN)

Detektor je i ovdje uspješno prepoznao lica, unatoč tome što su bila djelomično vidljiva. Pogrešne detekcije ni u ovom slučaju nisu prisutne. Slijedi izračun IoU mjera.



Slika 4.26. Lice s okluzijom 1 (DNN - IoU)



Slika 4.27. Lice s okluzijom 2 (DNN - IoU)

Za ispitane fotografije, IoU iznosi su malo niži u odnosu na prethodne dvije, no i dalje prelaze prag od 0.5.

Nakon što smo ispitali sve fotografije, zaključujemo da je DNN detektor izuzetno precizan u raspoznavanju ljudskog lica bez obzira iz kojeg kuta je ono fotografirano. Izuzetak je jedino nemogućnost prepoznavanja jednog lica na prvoj grupnoj fotografiji. Nadalje, niti za jednu fotografiju detektor nije dao pogrešnu detekciju, dok za *Haar* detektor to, recimo nije bio sličaj. Već sada iz viđenog možemo utvrditi da je DNN znatno precizniji od *Haar-a*, međutim konačni sud će se moći donijeti nakon što se ispita i posljednji detektor.

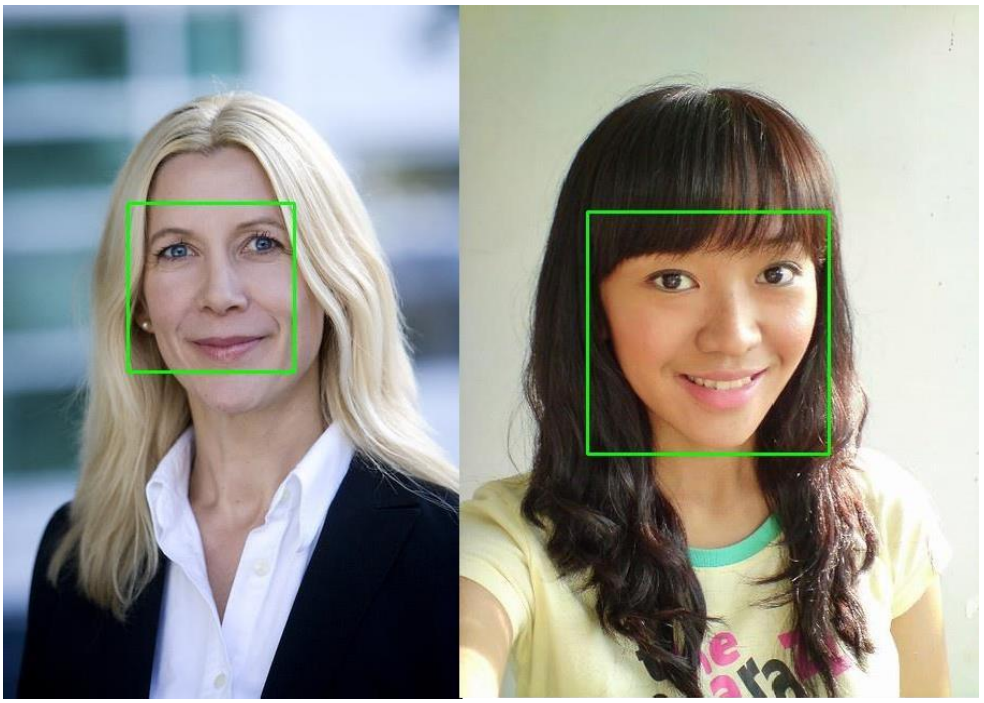
4.3. HOG detektor

Treći i posljednji detektor koji će se ispitati jest HOG detektor. Isti se bazira na već ranije opisanim histogramima orijentiranih gradijenata kao i na potpornim vektorima (HOG + SVM). HOG detektor impementirati ćemo koristeći Dlib biblioteku, međutim iako ona podržava samo C++ programsko sučelje, može se koristiti i u Python-u uz posredovanje OpenCV-a. HOG model koji ćemo koristiti izgrađen je od 5 filtera: prednji pogled, lijevi pogled, desni pogled, prednji pogled rotiran ulijevo i prednji pogled rotiran udesno. Slijedi implementacija HOG detektora:

```
#import dlib i opencv biblioteke
import dlib
import cv2

#inicijalizacija dlib funkcije za prepoznavanje lica i učitavanje slike
detector = dlib.get_frontal_face_detector()
img = cv2.imread('slika1.jpg')

#definiranje funkcije za crtanje graničnih okvira
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
faces = detector(gray, 1)
for result in faces:
    x = result.left()
    y = result.top()
    x1 = result.right()
    y1 = result.bottom()
    cv2.rectangle(img, (x, y), (x1, y1), (0, 255, 0), 2)

#prikaz output slike
cv2.imshow('img', img)
```

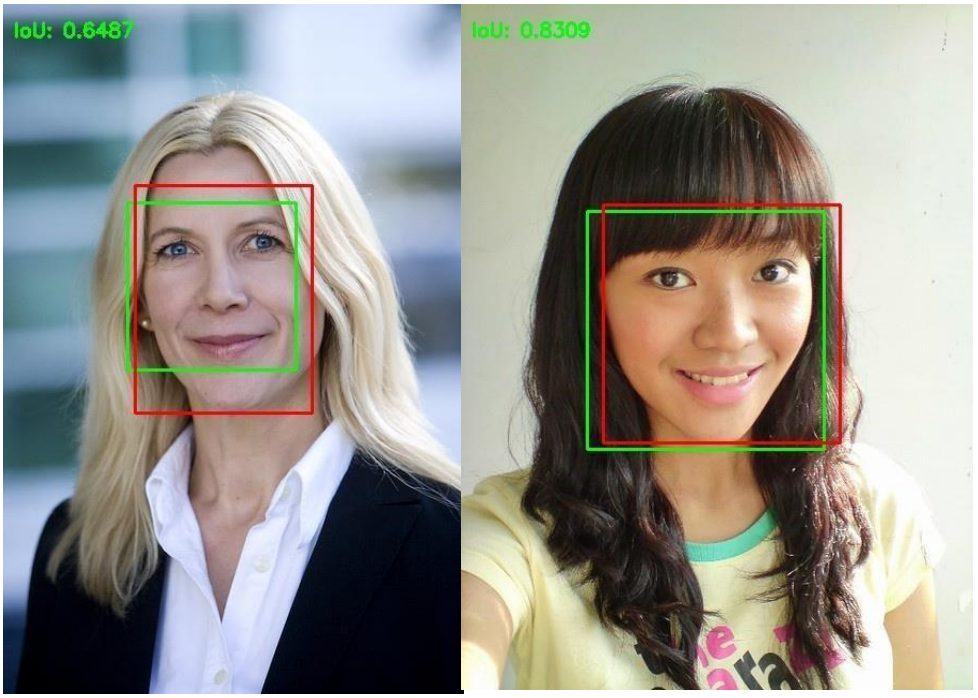
Izvor: https://github.com/davisking/dlib/blob/master/dlib/image_processing/frontal_face_detector.h

Ispitujemo prve dvije fotografije slicem u frontalnom položaju:



Slika 4.28. Lice u frontalnom položaju (HOG)

Očekivano, HOG je uspješno prepoznao oba lica. Sada ćemo provjeriti koliko otprilike iznosi IoU za obje fotografije.

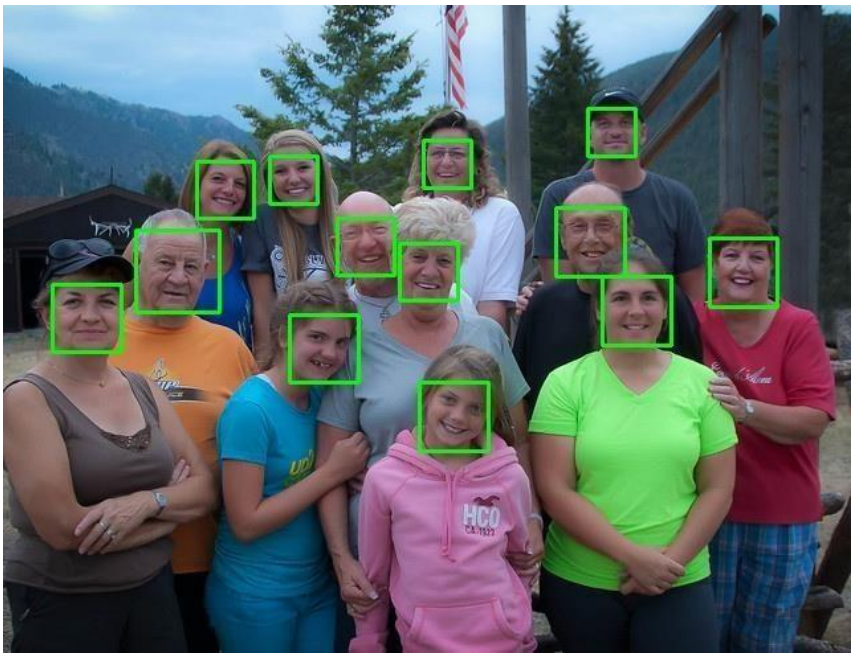


Slika 4.29. Lice u frontalnom položaju (HOG - IoU)

Možemo primijetiti da su dobiveni IoU iznosi usporedivi sa onima kod *Haar* detektora za iste fotografije. Unatoč nešto manjem IoU iznosu za prvu fotografiju, možemo reći da je preciznost detektora zadovoljavajuća za ispitane fotografije.

Slijedi provjera preciznosti za fotografije sa više lica:

Slika 4.30. Više lica u frontalnom položaju 1 (HOG)



Slika 4.31. Više lica u frontalnom položaju 2 (HOG)

Detektor je na obje fotografije prepoznao sva lica, te je tako i jedini kojem je to uspjelo za razliku od prethodna dva. No, moramo provjeriti i očekivanu preciznost pa računamo najviši IoU:

Slika 4.32. Više lica u frontalnom položaju 1 (HOG - IoU)

Slika 4.33. Više lica u frontalnom položaju 2 (HOG - IoU)

Najviši IoU za prvu fotografiju iznosi čak 0.9 što je odlična preciznost, dok je za drugu malo niži, 0.85, međutim i dalje vrlo dobar. HOG detektor, dakle, odlično prepoznaje skupinu lica u frontalnom položaju. Pogrešnih detekcija također nema.

U nastavku računamo prosječnu preciznost za Sliku 4.30. kao i kod prethodna dva detektora. U slučaju HOG detektora, TP iznosi 8, a FP i FN 0 iz razloga što nema pogrešnih detekcija te su sva lica prepoznata. Slijedi izračun preciznosti, odziva, točnosti i F1 mjere:

$$\boldsymbol{PPV = \frac{8}{8+0} = \frac{8}{8} = 1}$$

$$TPR = \frac{8}{8+0} = \frac{8}{8} = 1$$

$$ACC = \frac{8}{8+0+0} = \frac{8}{8} = 1$$

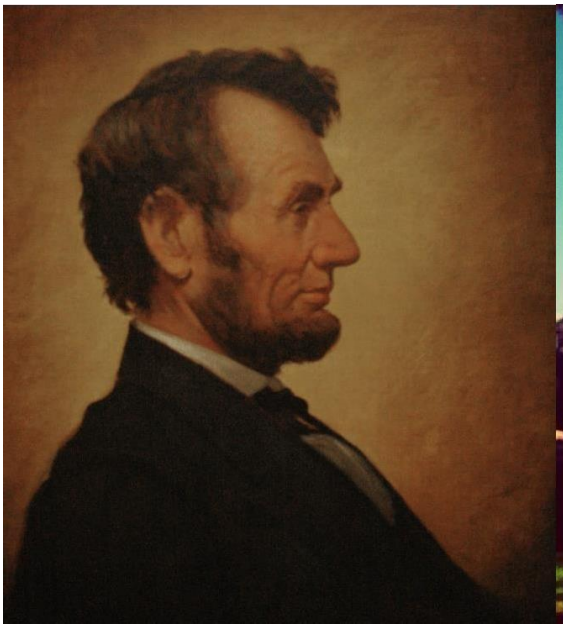
$$F1 = 2 \times \frac{1 \times 1}{1+1} = 2 \times \frac{1}{2} = 1$$

Iz navedenih izračuna, dobili smo da sve mjere iznose 1, pa stoga je i prosječna preciznost jednaka 1.

$$AP = mAP = \frac{1}{11} \sum_{0,0.1,0.2,0.3,..0.9,1} (11 \times 1) = 1$$

Iz priloženog možemo zaključiti da je HOG detektor najprecizniji u odnosu na ostala dva za ispitanu sliku, budući da prosječna preciznost iznosi 1.

Za prve četiri fotografije, HOG detektor je dao odlične rezultate u vidu preciznosti i točnosti detekcije lica. Sada ćemo provjeriti može li prepoznati lica iz profila. Rezultat je sljedeći:

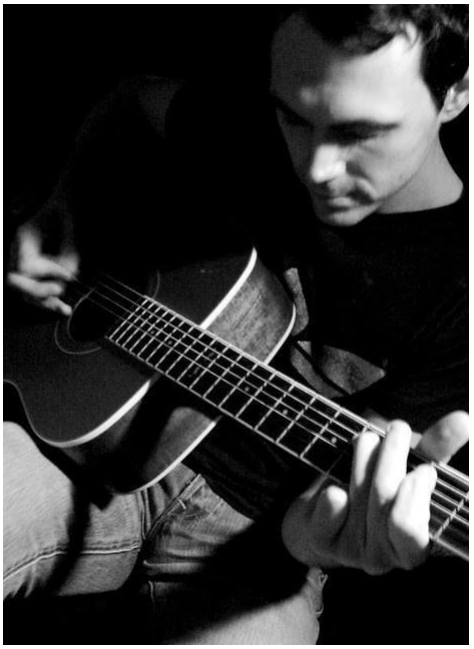


Slika 4.34. Lice iz profila (HOG)

Kao što možemo vidjeti, HOG nije uspio prepoznati lice slikano iz profila, bez obzira na stranu na koju je ono okrenuto. Ono što se još može primijetiti, HOG nije dao pogrešne detekcije, kao što je to bio slučaj kod *Haar-a*. S obzirom da detekcije nema, IoU iznosi 0, pa možemo prijeći na ispitivanje sljedećih fotografija.

Sada ispitujemo preciznost detekcije za fotografije na kojima je lice spušteno prema dolje. Dobili smo sljedeće rezultate:

Slika 4.35. Lice u spuštenom položaju 1 (HOG)



Slika 4.36. Lice u spuštenom položaju 2 (HOG)

Ni u ovom slučaju, HOG nije uspio prepoznati lica, ali isto tako nema niti pogrešnih detekcija. IoU iznosi 0, stoga prelazimo na ispitivanje posljednjih dviju fotografija.

HOG detektor nije uspio prepoznati lica ako ona nisu u frontalnom položaju. Za kraj provjeravamo može li prepoznati lice ako je ono pokriveno odnosno djelomično vidljivo. Rezultati su sljedeći:

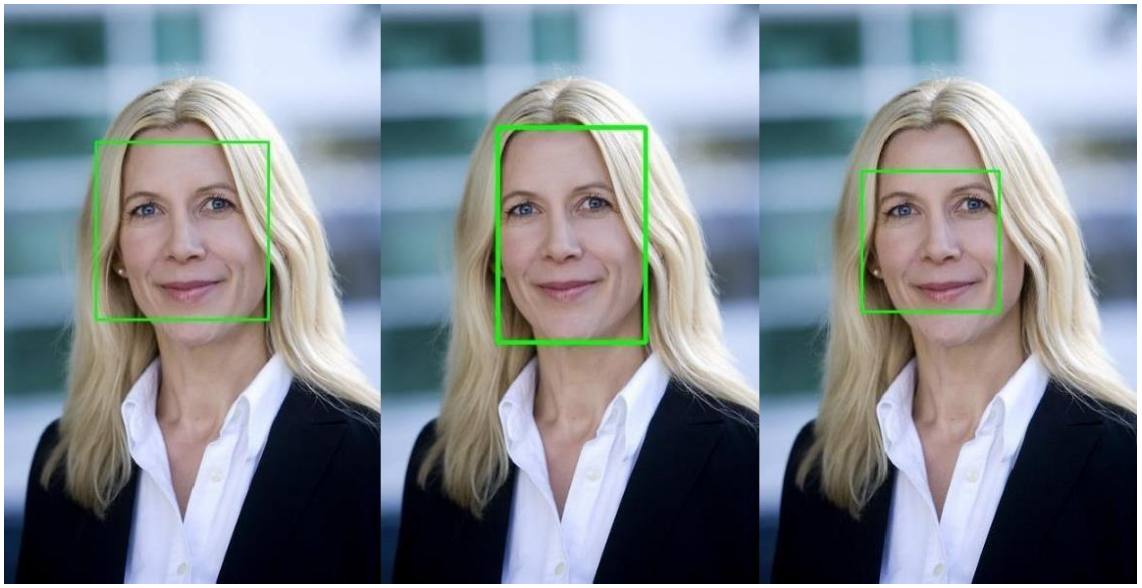
Slika 4.37. Lice s okluzijom 1 (HOG)

Slika 4.38. Lice s okluzijom 2 (HOG)

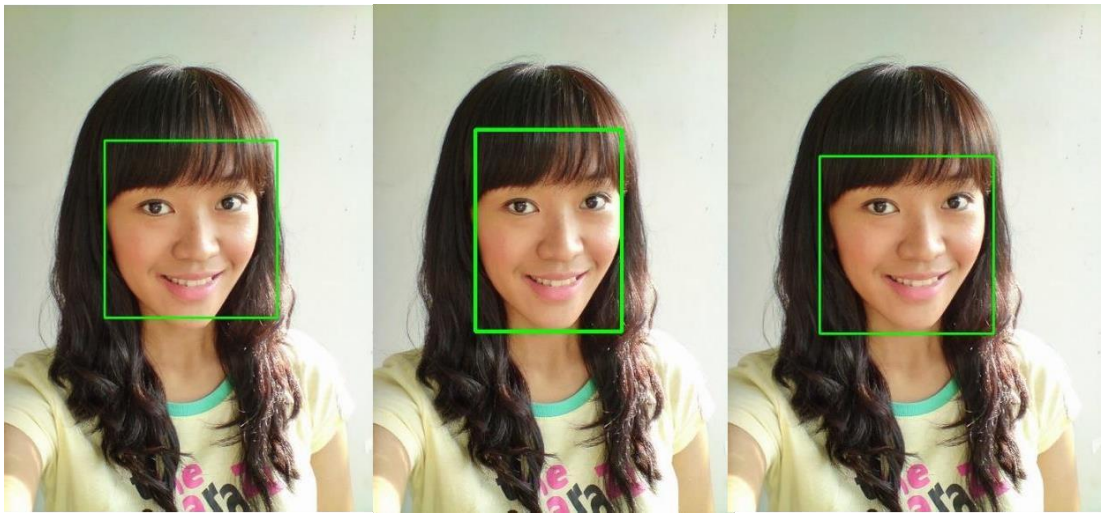
HOG detektor niti na ovim fotografijama nije zabilježio lice, pa možemo zaključiti da HOG ne može prepoznati lice na fotografijama na kojima ono nije jasno i u potpunosti vidljivo, bez obzira na njegovu udaljenost na slici. No, valja naglasiti da ni ovdje nema pogrešnih detekcija, što znači da HOG detektor jasno raspoznaje što je lice, a što nije.

4.4. Analiza i usporedba rezultata

Nakon što smo ispitali preciznost i sposobnost detekcije lica u različitim položajima kod sva tri detektora, sada možemo napraviti usporedbu i presjek dobivenih rezultata.



Slika 4.39.. Usporedba detektora (lice u frontalnom položaju 1)



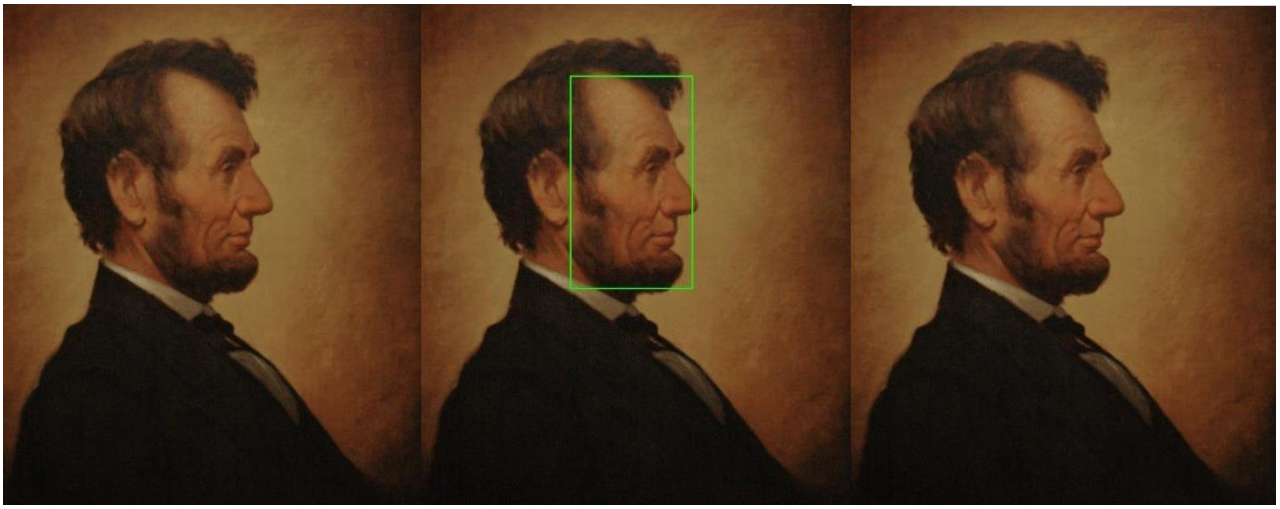
Slika 4.40. Usporedba detektora (lice u frontalnom položaju 2)

Za fotografije s licem u frontalnom položaju, sva tri detektora su uspjela prepoznati lica. Iznosi IoU mjera koje smo izračunali su približno isti, te na svim fotografijama oni prelaze 0.5 što označava vrlo dobru točnost u samoj detekciji. No, DNN detektor jedini je dao malo veće IoU iznose (preko 0.8 za obje fotografije) u odnosu na *Haar* i HOG, pa se stoga može zaključiti da se DNN pokazao najpreciznijim za ovaj slučaj. Niti jedan detektor, također, nije dao pogrešne detekcije koje se ne odnose na lice.

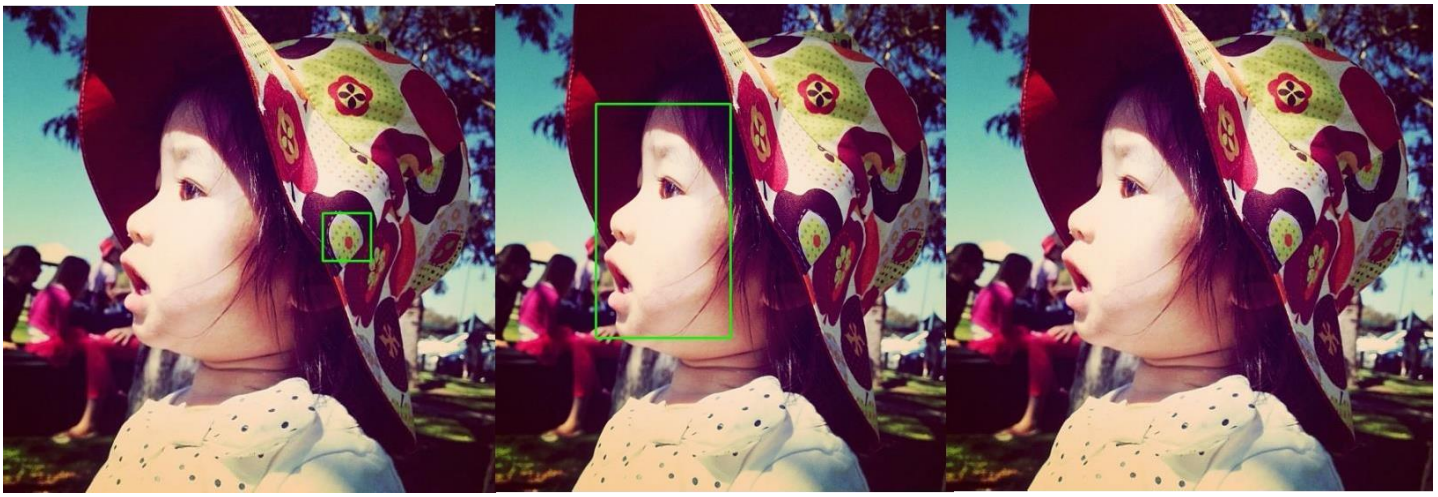
Slika 4.41.. Usporedba detektora (više lica u frontalnom položaju 1)

Slika 4.42. Usporedba detektora (više lica u frontalnom položaju 2)

Kad uspoređujemo rezultate za fotografije sa više lica u frontalnom položaju, možemo primijetiti da se kod nekih detektora javljaju poteškoće u detekciji. To se najbolje vidi u slučaju *Haar* detektora koji daje pogrešne detekcije tj. označuje mjesta na kojima nema lica te ne uspijeva prepoznati sva lica na drugoj fotografiji. Isto tako, DNN detektor također ne uspijeva prepoznati jedno lice iako je ono jasno vidljivo. To nas dovodi do zaključka da DNN može imati poteškoća kod prepoznavanja lica ako ono nije u prvom planu. S druge strane, HOG detektor je u ovom slučaju jedini polučio detekciju lica bez pogrešaka, što je dokazao i najviši IoU iznos koji je za drugu fotografiju iznosio čak 0.9. Za prvu fotografiju računali smo i prosječnu preciznost (mAP) koja je bila najveća za HOG detektor, a najmanja za DNN što je i razumljivo s obzirom da je HOG prepoznao sva lica, a DNN nije.

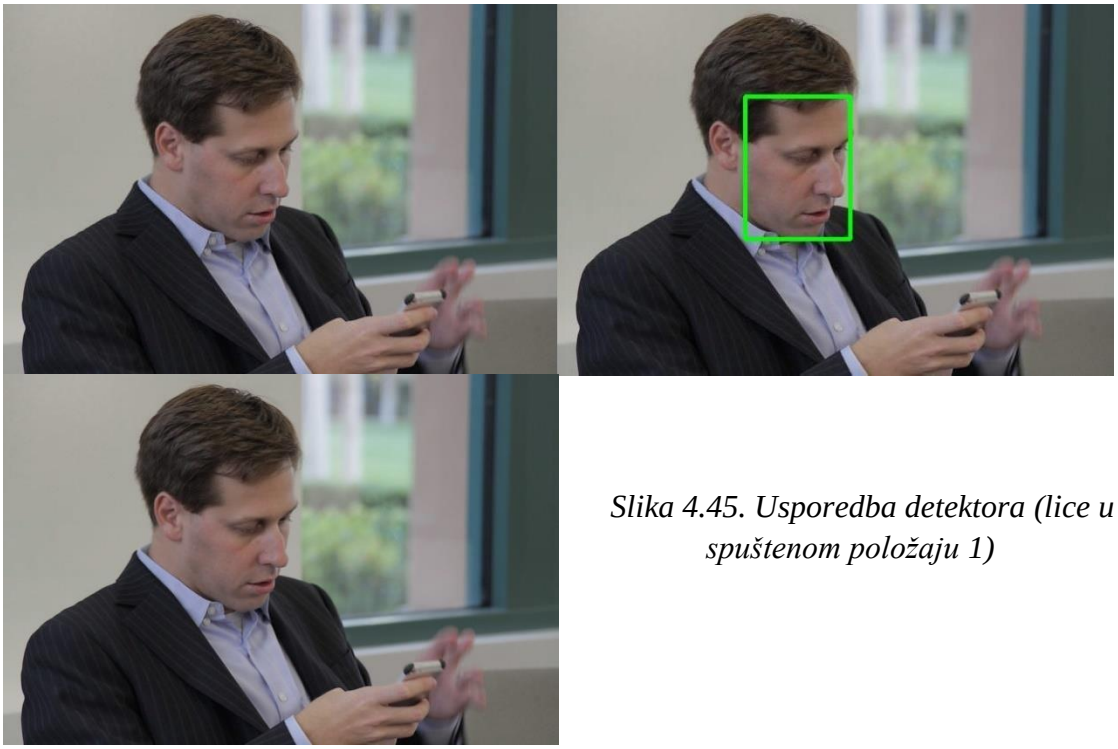


Slika 4.43. Usporedba detektora (lice iz profila 1)

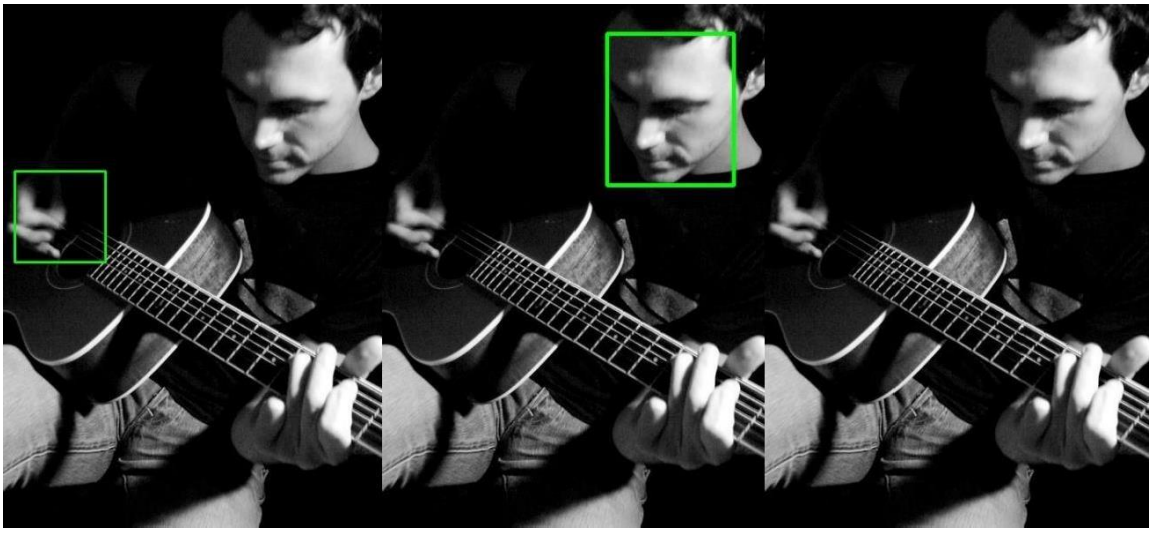


Slika 4.44. Usporedba detektora (lice iz profila 2)

Rezultati detekcije za lice iz profila, dalo je zanimljive rezultate. DNN detektor jedini je uspio prepoznati lice na obje fotografije, i to s priličnom preciznošću. *Haar* detektor za prvu fotografiju nije imao detekciju, ali na drugoj je ponovno imao pogrešno prepoznavanje. HOG detektor jedini nije imao nikakvu detekciju. DNN se, dakle, jedini pokazao uspješnim u prepoznavanju lica koje je fotografirano iz profila.

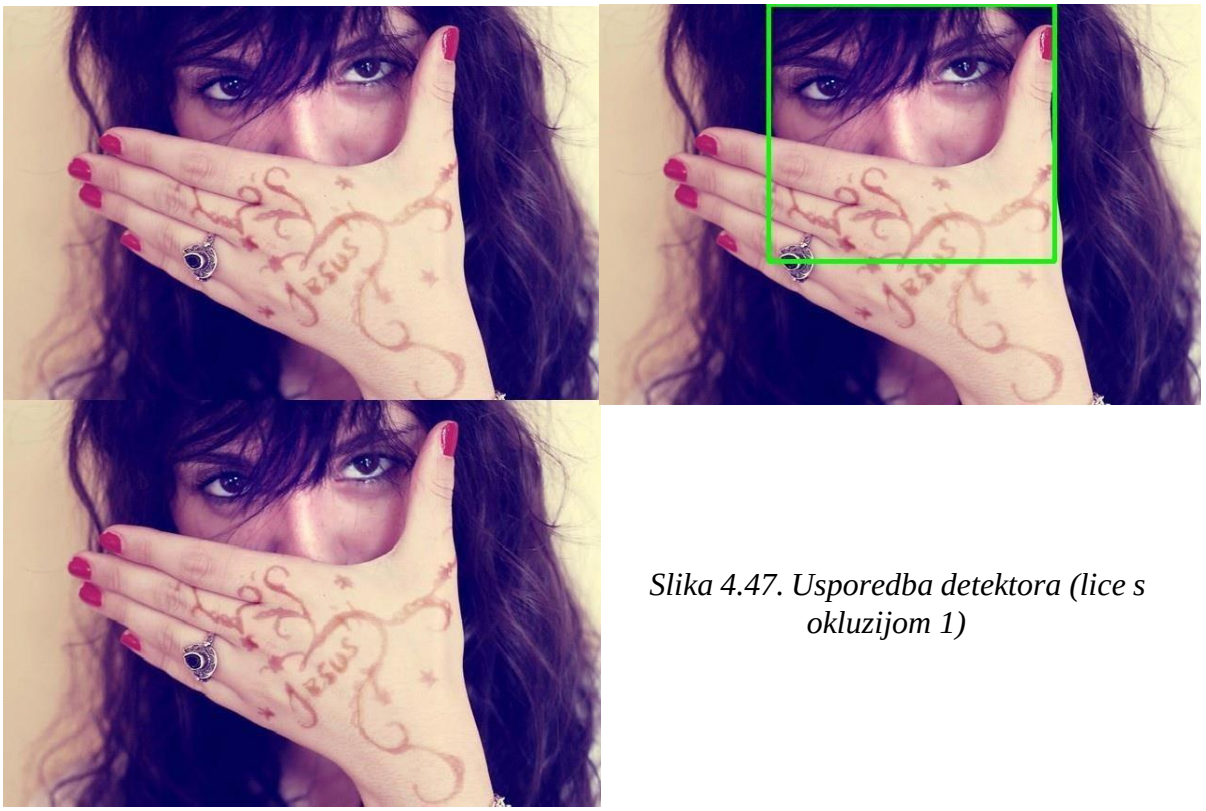


Slika 4.45. Usporedba detektora (lice u spuštenom položaju 1)

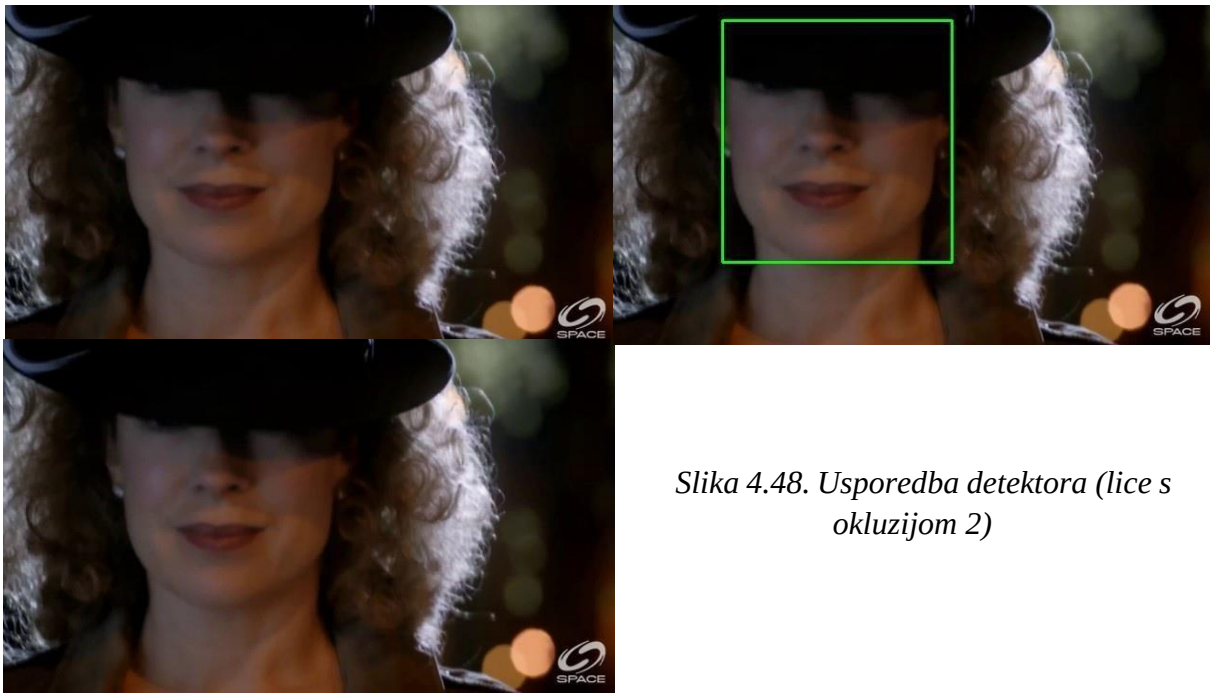


Slika 4.46. Usporedba detektora (lice u spuštenom položaju 2)

Kao i kod fotografija s licem iz profila, i ovdje smo dobili slične rezultate. *Haar* ponovno nije uspio prepoznati lice te na drugoj fotografiji daje pogrešnu detekciju. HOG detektor ne daje nikakvo prepoznavanje, dok se kao jedini uspješan ponovno pokazao DNN koji je sa odličnom preciznošću prepoznao oba lica.



Slika 4.47. Usporedba detektora (lice s okluzijom 1)



Slika 4.48. Usporedba detektora (lice s okluzijom 2)

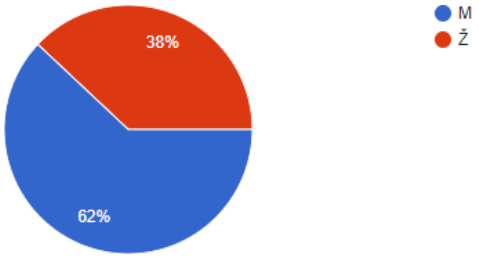
Kod fotografija na kojima lice nije jasno vidljivo, DNN je jedini koji je isto uspio prepoznati. S obzirom na neuspješnost *Haar* i HOG detektora kod prepoznavanja lica koje nije u frontalnom položaju, ovakvi rezultati su bili očekivani.

Što se tiče brzine izvođenja pojedinog detektora, ono ovisi o dimenzijama fotografija koje su ispitane kao i o procesoru računala. S obzirom da smo u ovom istraživanju koristili fotografije u svojim originalnim dimenzijama te one nisu bile istih dimenzija, brzine su varirale. Testirane su prve četiri fotografije budući da su sva tri detektora imali prepoznavanje na istim, a testiranje je provedeno na četverojezgrenom procesoru. Najsporije izvođenje imao je *Haar* detektor, čije je najdulje vrijeme izvršavanja iznosilo 1.20 sekundi. Vrijeme izvršavanja HOG detektora kretalo se između 0.45 i 0.55 sekundi. DNN detektor se pak pokazao kao najbržim, jer je njegovo najbrže vrijeme izvođenja iznosilo 0.07 sekundi, a najsporije 0.14 sekundi.

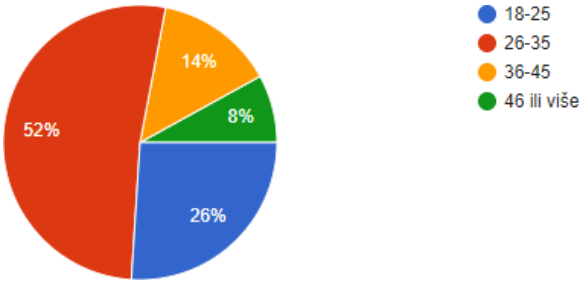
Na kraju ovog ispitivanja, možemo zaključiti da se DNN detektor ipak pokazao najuspješnijim u prepoznavanju lica u svim položajima. DNN nije polučio niti jednu pogrešnu detekciju, međutim jedini nedostatak je da nije prepoznao lice na grupnoj fotografiji. Razlog tomu leži u činjenici promjene originalne dimenzije slike na dimenzije 300x300 prilikom detekcije te iz tog razloga može doći do neuspješnog prepoznavanja lica ako je ono udaljeno na fotografiji. Nadalje, HOG detektor jedini je koji je uspio prepoznati sva lica koja se nalaze u frontalnom položaju na fotografijama te također nije dao pogrešnu detekciju. Glavni nedostatak HOG detektora je nemogućnost detekcije lica u nefrontalnim položajima. Konačno, *Haar* detektor se pokazao najnepreciznijim detektorom. Iako je prilično precizan u detekciji lica u krupnom planu, *Haar* je sklon pogrešnim prepoznavanjima na što utječu pozadinski objekti na fotografijama. Može se zaključiti da svaki od detektora ima svoje prednosti, ali i nedostatke te njihova preciznost uvelike ovisi o dimenziji fotografije, ali i udaljenosti lica na istoj.

5. Anketno istraživanje

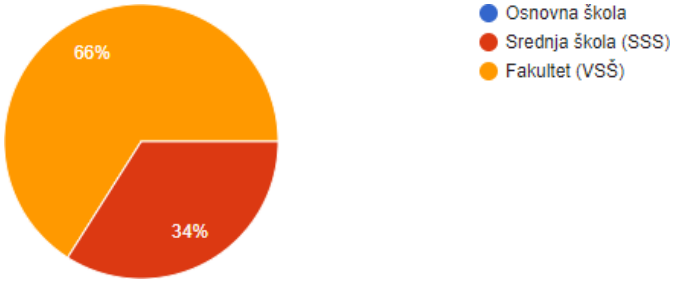
Za kraj je provedeno kratko anketno istraživanje u obliku upitnika. U anketi je sudjelovalo 50 ispitanika sa svrhom provjere jesu li i koliko su općenito upoznati sa tehnologijom detekcije lica kao i sa područjem računalnog vida. U nastavku slijede dobiveni rezulatati u obliku grafikona.



Graf 5.1. Spol ispitanika



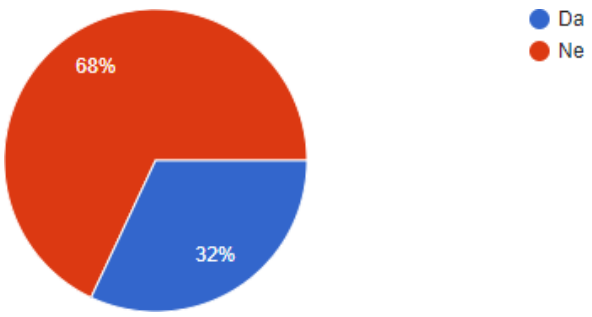
Graf 5.2. Dob ispitanika



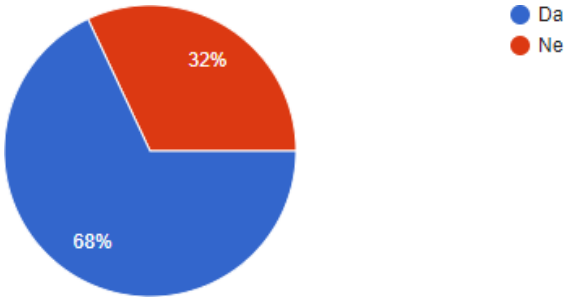
Graf 5.3. Stručna sprema ispitanika

Na pitanje 'Spol?', 62 % ispitanika se izjasnilo kao muško, dok se 38 % njih izjasnilo kao žensko. Što se tiče dobi ispitanika, ona se u većini kretala između 26 i 35 godina (52 %). Ispitanika između 18 i 25 godina bilo je 26 %, između 36 i 45 godina 14 %, dok je najmanji postotak od samo 8 % otpao na ispitanike preko 46 godina. Ovakav rezultat ne čudi s obzirom da su ciljna skupina koja je sudjelovala u ovoj anketi, bili ljudi mlađe populacije s tek završenim fakultetom ili oni koji ga još pohađaju. Na pitanje 'Stručna sprema?' 66 % ispitanika je odgovorilo da imaju završen fakultet odnosno 34 % njih da imaju završenu srednju školu.

Nakon demografskih pitanja, uslijedila su dva pitanja vezana uz područje računalnog vida.



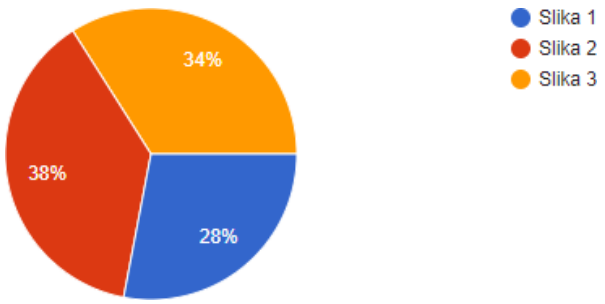
Graf 5.4. Grafički prikaz poznavanja pojma računalnog



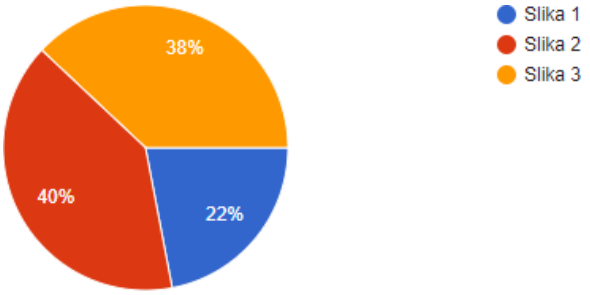
Graf 5.5. Grafički prikaz prema upotrebi aplikacija za detekciju lica

Na pitanje 'Jeste li se ikada susreli s pojmom računalnog vida?', 68 % ispitanika odgovorilo je da nije, dok je 32 % njih odgovorilo da je. Na pitanje 'Jeste li ikada koristili neku od aplikacija za prepoznavanje lica na fotografijama?', 68 % ispitanika odgovorilo je da jesu, a njih 32 % da nije. Ovdje smo dobili zanimljive rezultate, unatoč tome što većina nije upoznata s pojmom računalnog vida, ipak su nekada koristili aplikacije za detekciju lica. Ovakvi odgovori su razumljivi, s obzirom da velik broj ljudi danas posjeduje pametne telefone na kojima se takve aplikacije mogu izvoditi, dok se sa samim područjem računalnog vida susreću ljudi koji pohađaju ili su pohađali škole i fakultete na kojima se takve teme intenzivnije obrađuju.

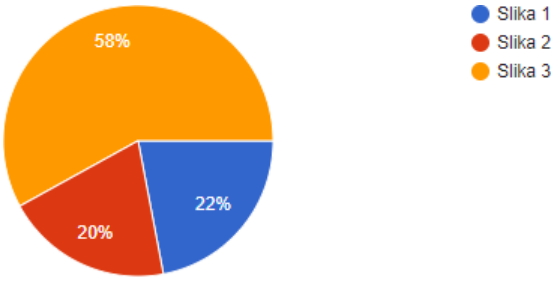
Na sljedećim pitanjima, ispitanicima su dane četiri fotografije koje su ranije ispitivane pomoću detektora: dvije fotografije s jednim licem u frontalnom položaju i dvije s više lica u frontalnom položaju, budući da su sva tri detektora za te fotografije imale uspješnu detekciju lica. Ispitanici su za svaku fotografiju trebali, prema vlastitom vizualnom doživljaju, odrediti koji je detektor postigao najveću preciznost. Fotografije su bile označene nazivima 'Slika 1', 'Slika 2' i 'Slika 3' za koje je 'Slika 1' predstavljala *Haar* detektor, 'Slika 2' DNN detektor, a 'Slika 3' HOG detektor.



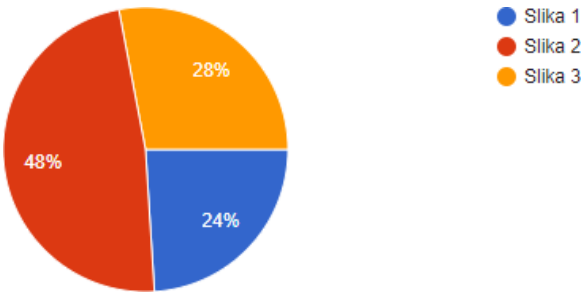
Graf 5.6. Grafički prikaz rezultata za prvu fotografiju s licem u frontalnom položaju



Graf 5.7. Grafički prikaz rezultata za drugu fotografiju s licem u frontalnom položaju



Graf 5.8. Grafički prikaz rezultata za prvu fotografiju s više lica u frontalnom položaju



Graf 5.9. Grafički prikaz rezultata za drugu fotografiju s više lica u frontalnom položaju

Iz navedenih rezultata, za prve dvije fotografije s licem u frontalnom položaju, najviše ispitanika (38 %, odnosno 40 %) se odlučilo za DNN detektor kao najprecizniji. Odmah nakon njega slijedi HOG detektor sa 34 %, odnosno 38 %, te na kraju *Haar* sa 28%, odnosno 22 %. Što se tiče preostalih dviju fotografija, s više lica u frontalnom položaju, rezultati su ovdje odskakali za pojedine detektore. Za prvu grupnu fotografiju, čak 58 % ispitanika izabralo je HOG detektor kao najprecizniji, slijedi *Haar* sa 22 % te, zanimljivo DNN posljednji sa 20 % odabira. Rezultat je vjerojatno takav iz razloga što za tu fotografiju DNN nije uspio prepoznati jedno lice, *Haar* je imao pogrešnu detekciju dok je HOG jedini prepoznao sva lica bez pogrešnih detekcija. Konačno, kod druge grupne fotografije, za najprecizniji detektor 48 % ispitanika je odabralo DNN, 28% HOG te 24 % *Haar* detektor. Može se reći da je većina ispitanika relativno dobro zamijetila da je DNN uistinu najprecizniji za navedenu fotografiju, iako je i HOG detektor dao precizne rezultate u vidu prepoznavnja lica na istoj. *Haar* je, možda i očekivano, dobio najmanji postotak glasova, budući da je ponovno imao pogrešnu detekciju, ali i neuspješnu detekciju jednog lica.

6. Zaključak

Cilj ovoga rada bio je pobliže upoznavanje s principom rada tehnologije detekcije objekata tj. ljudskog lica na fotografijama. Kroz rad upoznali smo se s najpoznatijim tehnologijama u području računalnog vida za tu svrhu, poput Viola - Jones algoritma, dubokih neuronskih mreža te histograma orijentiranih gradijenata. Svaka od navedenih metoda ima svoj jedinstveni pristup u detekciji objekata, što ih također razlikuje jedne od druge. Možda najsloženije od svih su metode dubokih neuronskih mreža koje se sastoje od više zasebnih mreža sa različitim pristupima u detekciji objekata. Jednu takvu mrežu, SSD, smo i korsitili u ovom istraživanju. Nadalje, tehnologija detekcije objekata danas je već vrlo dobro razvijena, te se koristi u svim većim i modernim industrijama. Međutim, naglasak u ovome radu bio je na detekciji ljudskog lica na fotografijama, budući da se ista primijenjuje u različitim oblicima te je dostupna gotovo svima. Za primjer, današnji digitalni fotoaparati i mobilne kamere koriste detekciju lica za automatsko fokusiranje. Isto tako, pojedine društvene mreže poput Facebooka i Instagrama imaju mogućnost obilježavanja lica na fotografijama koje korisnik kasnije može povezati sa drugim korisnicima. U istraživanju su se uspoređivala tri detektora bazirana na spomenutim algoritmima za detekciju objekata, te su se na kraju usporedili rezultati s obzirom na njihovu preciznost, sposobnost same detekcije lica te brzinu izvođenja. Niti jedan od detektora na ispitanim fotografijama nije dao savršene rezultate za svaku, te je zaključak da svaki ima svoje prednosti, ali i nedostatke. Na kraju, kroz anketni upitnik htjelo se provjeriti poznavanje ispitanika sa tematikom računalonog vida odnosno tehnologijom spomenute detekcije lica. Ispitanici su, unatoč većinskom nepoznavanju područja računalnog vida, relativno uspješno prepoznali koji je detektor prema njihovom subjektivnom mišljenju najprecizniji, iako naravno to ne mora biti potpuno točno. Zaključak je da tehnologija detekcije objekata, kao i detekcije lica ima prostora za napredak, a budući da se ista neprekidno razvija, sigurno je da će u bližoj budućnosti dostići najvišu kvalitetu.

Popis literature

[1] Corinna Cortes, Vladimir Vapnik: *Support-Vector Networks*, Machine Learning br. 20, str 273.-297, 1995.

[2] Chris McCormick: *HOG Person Detector Tutorial*, 2013. dostupno na: http://mccormickml.com/2013/05/09/hog-person-detector-tutorial/, 2.8.2021.

[3] Dario Kesić: *Prepoznavanje osoba u stvarnom vremenu*, Osijek, Diplomski rad, 2016.

[4] Javor Marošević: *Detekcija ljudskog lica u videotijeku u realnom vremenu*, Zagreb, Završni rad, 2018.

[5] Krešimir Kukuljan: *Detekcija regija lica na temelju slike kamere mobilnog uređaja*, Osijek, Diplomski rad, 2016.

[6] Marko Rogina: *Detekcija lica korištenjem neuronske mreže YOLO v3 tiny*, Varaždin, Diplomski rad, 2020.

[7] Mario Dubovečak: *Detekcija i prepoznavanje lica koristeći Raspberry Pi računalo*, Varaždin, Diplomski rad, 2020.

[8] Navneet Dalal, Bill Triggs: *Histograms of Oriented Gradients for Human Detection*, Montbonnot-Francuska, str. 2-7

[9] Paul Viola, Michael Jones: *Robust Real-time Object Detection*, Vancouver-Canada, str. 1-25, 2001.

[10] Wikipedia: *Object detection*, dostupno na: https://en.wikipedia.org/wiki/Object_detection, 31.7.2021.

[11] Wikipedia: *Viola-Jones object detection framework*, dostupno na: https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework, 31.7.2021.

[12] Zhang Zhao, Peng Zheng, Shou-Tao Xu, Xindong Wu: *Object Detection With Deep Learning: A Review*, vol. 30, no. 11, 2019.

[13] J.R.R. Uijlings, K.E.A. van de Sande, T. Gevers, A.W.M. Smeulders: *Selective Search for Object Recognition*, 2013.

[14] Ronith Gandhi: *R-CNN, Fast R-CNN, Faster R-CNN, YOLO – Object Detection Algorithms*, 2018., dostupno na: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e, 3.8.2021.

[15] Renu Khandelwal: *ComputerVision: Instance Segmentation with Mask R-CNN*, 2019., dostupno na: https://towardsdatascience.com/computer-vision-instance-segmentation-with-mask-r-cnn-7983502fcad1, 3.8.2021.

[16] Manish Chablani: *YOLO – You only look once, real time object detection explained,* 2017., dostupno na: https://towardsdatascience.com/yolo-you-only-look-once-real-time-object-detection-explained-492dc9230006, 3.8.2021.

[17] Sik-Ho Tsang: *SSD – Single Shot Detector (Object Detection),* 2018., dostupno na: https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11, 3.8.2021.

[18] OpenCV, About OpenCv, dostupno na: https://opencv.org/about.html, 6.8.2021.

[19] Dlib C++ Library, dostupno na: http://dlib.net/, 6.8.2021.

[20] Open Images Dataset V6, dostupno na:

https://storage.googleapis.com/openimages/web/visualizer/index.html?set=valtest&type=detection&c=%2Fm%2F0dzct&id=81bbb29c630c1db8, 25.8.2021.

[21] Adrian Rosebrock: *Intersection over Union (IoU) for object detection*, 2016., dostupno na: https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/, 25.8.2021.

Popis slika i grafova

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, BORNA BAUDOIN *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom USPOREDBA RAZLIČITIH ALGORITAMA ZA DETEKCIJU *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova. OBJEKATA KORISTEĆI OPENCV I DLIB

Student/ica:
(upisati ime i prezime)

Borna Baudoin
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, BORNA BAUDOIN *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom USPOREDBA RAZLIČITIH ALGORITAMA ZA *(upisati naslov)* čiji sam autor/ica. DETEKCIJU OBJEKATA KORISTEĆI OPENCV I DLIB

Student/ica:
(upisati ime i prezime)

Borna Baudoin
(vlastoručni potpis)