Универзитет у нишу

Електронски факултет

Катедра за рачунарство

**Праћење регуларности полагања тестова применом техника рачунарског вида**

- дипломски рад -

**Задатак:**

Упознати се са постојећим софтверским решењима која се користе за праћење регуларности онлајн полагања тестова. Идентификовати технике рачунарског вида које се могу искористити за реализацију оваквих система. У практичном делу имплементирати прототип система који употребом камере обезбеђује препознавање особе која полаже тест, врши детекцију нерегуларности у виду окретања и разговора, те генерише извештај са видео доказом спорних ситуација.

**Mентор:** проф. др Александар Милосављевић **Кандидат:** Ивана Миливојевић 16704

Комисија:

1. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Датум пријаве: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Датум предаје: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Датум одбране: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ниш, 2022.

**Садржај**

[**Увод** 5](#_Toc115642908)

[**Софтвери за праћење регуларности полагања тестова** 6](#_Toc115642909)

[Аутоматско надгледање онлајн тестирања 6](#_Toc115642910)

[Примери софтвера за аутоматско надгледање онлајн тестирања 7](#_Toc115642911)

[**Технике рачунарског вида за праћење регуларности полагања тестова** 10](#_Toc115642912)

[Детекција објеката 10](#_Toc115642913)

[**Литература** 14](#_Toc115642924)

# **Увод**

Пандемија настала услед ширења вируса Ковид-19 утицала је на све аспекте живота људи, па тако и на образовање. У условима који су настали као последица ове ситуације било је неопходно прилагодити начин извођења наставе како би се школска година одвијала несметано. Многе образовне институције биле су приморане да пређу на хибридни или у потпуности онлајн модел одржавања наставе, те је један од проблема био како пронаћи најбољи начин за оцењивање ученика/студената.

Полагање онлајн тестова на рачунарима и сличним уређајима је већ било у великој мери заступљено у пракси зато што олакшава и убрзава процес креирања тестова, полагања, као и анализу резултата. Флексибилност и могућност рада од куће разлози су због којих је већина наставника одабрала овакве тестове као меру оцене. Наравно, тиме се отвара простор за различите начине преписивања. Неки од начина да се преписивање спречи су увођење временског ограничења за одговор за свако питање, онемогућавање враћања на претходно одговорено питање, ограничење да сви морају полагати тест истовремено, генерисање различитих питања, приказ питања и понуђених одговора у различитом редоследу... Међутим, ове мере некада нису довољне и ученици/студенти ипак пронађу начин да потраже туђу помоћ. Из тог разлога направљен је софтвер за надгледање полагања онлајн тестирања и генерисање одговарајућег извештаја.

Једано од решења је да дежурна особа надгледа ученике током полагања у реалном времену. Могу се пратити микрофон, камера и екран ученика који полаже тест. Друго решење подразумева снимање ученика и његовог екрана током полагања и касније прегледање тог снимка ради контроле регуларности. Предност оваквог решења је што ученик и дежурни не морају у заказано време бити присутни, већ касније може бити прегледана убрзана верзија комплетног снимка полагања. У овом случају нема могућности да дежурна особа реагује на неправилности у реалном времену, због којих је можда требало онемогућити наставак полагања теста ученику. Треће решење је напредно, аутоматско нагледање полагања, где вештачка интелигенција има кључну улогу у препознавању нерегуларности. Коришћење алгоритама вештачке интелигенције и машинског учења у великој мери мења особу која је била задужена да лично надгледа снимке свих ученика који полажу тестове. Систем генерише упозорења током полагања и бележи све нерегуларне ситуације које су детектоване и које се касније поново прегледају како би се потврдиле неправилности. Још увек не можемо у потпуности да се ослонимо на рачунар и искључимо људски фактор у надгледању регуларности, али овакви системи се могу користити у реалном времену и у будућности ће можда бити довољно поуздани да могу самостално да обављају овај посао. [1]

У овом раду биће обрађене методе рачунарског вида које се могу користити за обраду и валидацију података са камере приликом полагања онлајн тестова. Рад је организован у пет поглавља. Након уводног дела, у другом поглављу је дат преглед постојећих софтверских решења за овај проблем. У трећем поглављу обрађене су технике и алгоритми рачунарског вида који се могу применити за детекцију особа и лица на видео снимку, препознавање лица, праћење положаја главе и погледа и детекцију говора ученика који полаже тест. У наредном поглављу описана је конкретна имплементација прототипа система који се може користити за надгледање регуларности полагања тестова на основу података са камере. Систем детектује невалидне ситуације, приказује упозорења ученику за време теста и све детектоване нерегуларности бележи у видео фајл. Закључно поглавље осврће се на претходна поглавља и сумира теоријске и практичне аспекте описаног проблема.

# **Софтвери за праћење регуларности полагања тестова**

Постоје различити пројекти који су развијени у циљу омогућавања реализације учења и тестирања на даљину. Углавном се ради о веб апликацијама, а има и апликација које је потребно инсталирати на свом рачунару или мобилном телефону. Ови софтвери, поред примене у образовним институцијама, могу се користити и у различитим компанијама и организацијама за потребе било ког типа тестирања попут интервјуа за посао, семинара и добијања сертификата.

Као најважнији захтеви које је потребно да ови софтвери испуне могу се издвојити: једноставност коришћења, превенција варања, сигурност и приватност, скалабилност, прилагодљивост и интеграција са другим сервисима. Једноставност коришћења је веома важна јер софтвер треба да корисницима пружи добро корисничко искуство и не захтева превише њиховог времена. Алати и технике за детекцију нерагуларности су неопходни како би тестирање било еквивалентно тестирању у заједничкој просторији у којој једна или више особа дежура. Потребно је обезбедити сигурност осетљивих података као што су сами тестови и лични подаци корисника. Скалабилност има важну улогу када је у питању употребљивост система зато што може бити потребно да више хиљада корисника истовремено полаже тестове. Како корисници апликације могу бити различите институције и организације, имаће различите потребе, те треба обезбедити могућност избора функционалности које су захтеване за реализацију одређеног тестирања. Такође, систем треба да буде једноставан за интеграцију са постојећим платформама за учење попут LMS-а[[1]](#footnote-1) (енг. Learning Management System).[2]

Као што је напоменуто у уводном поглављу, софтвер за надгледање онлајн тестирања може се поделити у три групе. У прву групу спадају софтвери код којих особа која је задужена за надгледање регуларности уживо посматра ученика који полаже тест и може одмах реаговати уколико примети неке неправилности. Систем захтева од ученика који полаже тест да укључи камеру, микрофон и подели свој екран како би дежурна особа имала увид у све његове активности. Други начин за реализацију система је да се током полагања ученик и његов екран снимају, а касније прегледа убрзана верзија тог снимка ради утврђивања евентуалних нерегуларности. Софтвери из треће групе не захтевају ангажовање дежурне особе током тестирања, засновани су на техникама вештачке интелигенције и рачунарског вида и све више се користе у пракси. У овом поглављу биће дат преглед метода за аутоматско надгледање онлајн тестирања и неколико најпопуларнијих производа који су доступни и користе их многобројне институције и организације.

## Аутоматско надгледање онлајн тестирања

Софтвери за онлајн учење и тестирање углавном имају механизам за регистровање корисника и захтевају да се ученици улогују да би могли да користе апликацију. Ово је први корак у процесу полагања и може бити имплементиран на различите начине, али најчешће се ученик пријављује на систем коришћењем свог и-мејла, а пре самог покретања теста мора да коришћењем камере покаже своју идентификациону картицу или омогући систему да забележи његову слику и упореди са сликом из базе података. У оквиру апликације може постојати више модула који су задужени за обраду различитих података. Модул за праћење аудио података захтева да ученик има повезан микрофон и да обезбеди тишину у просторији у којој ће радити тест. Испитује се да ли ученик разговара са неким, да ли је учеников глас исти током трајања теста и у неким случајевима се може поредити са гласом тог ученика из базе података. Надгледање звука може спречити да неко други ради тест уместо ученика, да му нека особа помаже „добацивајући“ му одговоре или да му одређена апликација саопштава одговоре на питања. Модул који је задужен за праћење екрана кандидата углавном санкционише сликање и снимање прозора у ком је отворен тест, напуштање теста и отварање других страница, апликација или фајлова. Софтвер има контролу над веб претраживачем рачунара у коме ученик полаже тест. Забрањује се отварање апликација које могу олакшати дељење одговора нудећи могућност за дописивање и дељење екрана. Софтвери који захтевају да камера ученика буде укључена током полагања теста могу надгледати његове покрете и пријавити сумњиво понашање. Модул за обраду података са камере може да контролише да ли у кадру има других особа или недозвољених уређаја попут мобилних телефона, да врши препознавање лица ученика, могу се пратити покрети очију ученика и детектовати када он дуже време гледа са стране, или када често на краћи временски период гледа са стране. Софтвер за надгледање може вршити и онлајн претрагу како би проверио да ли је садржај теста „процурео“. Поједини софтвери су у могућности да уз помоћ вештачке интелигенције лоцирају и уклоне садржај теста који је неауторизовано доспео на интернет. [3]

## Примери софтвера за аутоматско надгледање онлајн тестирања

На тржишту су доступни различити софтвери за тестирање. За креирање кратких тестова који не захтевају висок степен сигурности и провере, најједноставније је користити неки од бесплатних веб сајтова за генерисање квизова и тестова. Када су у питању озбиљнији тестови који захтевају висок степен сигурности и надгледања, треба одабрати неки од напреднијих софтвера, у зависности од потреба и буџета организације.

Једно од водећих решења је Honorlock, први сервис за надгледање онлајн тестирањања који комбинује аутоматско надгледање са људским прокторима у циљу очувања интегритета теста. Тестирање надгледа вештачка интелигенција и ако детектује академску нечеститост алармира проктора да се прикључи сесији у реалном времену. Овиме се губи потреба да нека особа све време надгледа више кандидата, што и кандидатима смањује осећај нелагодности због присуства друге особе. Циљ није само детектовати варање, већ учинити сам процес тестирања што лагоднијим како за оног ко полаже тако и за особу која дежура. Апликација нуди интерфејс који је веома једноставан за коришћење, доступна је 24/7, сигурна и скалабилна. Предуслови за израду теста су да особа која полаже тест буде сама у просторији, да буде тишина, да рачунар на коме се ради тест има само један монитор, да корисник поседује 360° камеру како би се скенирала просторија у којој се налази. Honorlock користи софтвер за закључавање веб претраживача који онемогућава приступ другим веб сајтовима, претраживачима и апликацијама, при чему закључава и пречице на тестатури попут штампања, копирања и лепљења (енг. copy/paste), сликања екрана, онемогућава минимизирање претраживача и напуштање теста пре предаје. Наставници имају могућност да по жељи специфицирају којим веб сајтовима желе да омогуће приступ ученицима током тестирања. Од ученика се може затражити да преузме додатну апликацију која је задужена за праћење других апликација које ученик отвара током полагања теста на свом уређају. Могућа је директна интеграција са LMS платформама попут MyMathLab, ALEKS, Pearson, и McGraw Hill. Такође, софтвер идентификује „процурели“ садржај теста на интернету и предузима кораке за његово уклањање. Још једна функционалност која је уведена је да систем детектује ако ученик покуша да приступи материјалу за учење током теста путем неког другог уређаја и бележи снимак екрана током трајања приступа. Honorlock не користи биометријске методе за идентификацију ученика попут препознавања лица, већ пре почетка теста услика ученика који држи своју идентификациону картицу и након 60 секунди омогућава полагање теста. Недостацима се може сматрати ограничење да институција мора имати најмање 250 корисника како би могла да користи апликацију, није доступна бесплатна пробна (енг. trial) верзија и тренутно ради само са Google Chrome претраживачем, односно постоји екстензија за овај претраживач. [4]

ProctorEdu је веб претраживач-решење за онлајн надгледање, снимање и евалуацију корисничног понашања током онлајн тестирања, без потребе да корисник преузима на свом уређају било какав софтвер. Компанија се бави овом темом од 2016. године и фокусирана је на корисничко искуство. Функционалност надгледања је интегрисана са LMS или платформама за тестирање и омогућава надгледање удаљеног тестирања уживо или у аутоматском моду. Око 2 милиона тестова до сада је одрађено преко ове платгорме укључујући до 5 000 особа које инстовремено раде тест. Софтвер нуди поуздане резултате теста, биометријску аутентификацију, подршку за Android и iOS мобилне уређаје, аутоматски опоравак након губљења конекције, могућност повезивања додатне камере са мобилног телефона за преглед простора где се налази ученик у 360° (камера се повезује скенирањем QR кода). Приликом организовања теста може се доделити више проктора који ће надгледати полагање, а систем ће им помоћи тако што ће их обавештавати о нерегуларним активностима у реалном времену. Проктори могу комуницирати са ученицима путем чета, видеа или аудија уколико ученици буду имали неких питања. Особа која полаже тест покреће проверу опреме која траје 30 секунди, доказује свој идентитет показивањем идентификационе картице, гледа видео са правилима и покреће тест. Систем нуди функционалност препознавања лица током трајања теста, детекцију буке, контролише покушај претраге на интернету. Извештај да ли је било невалидности се генерише у видео и пдф формату. Подржава различите претраживаче попут Chrome, Opera, Firefox, Edge, Safari. [5]

Mercer **|** Mettl такође нуди безбедно и фер тестирање. Овај софтвер користи више од 150 универзитета широм света и обављено је више од 12 милиона тестова ове године. Нуди избор између надгледања уживо и аутоматског надгледања, корисничку подршку у свако доба, али не и бесплатну верзију. Може се користити као комплетна платформа за тестирање, али и као сервис за удаљено надгледање тестирања којим се проширује нека друга платформа за тестирање. Систем врши видео и аудио надгледање. Детектује уколико кандидат није присутан или је присутна особа која није она којом се представља, присуство мобилних телефона и других особа и говор других особа. Студент се пре почетка теста пријављује уз помоћ и-мејл аутентификације, мобилне аутентификације коришћењем OTP (енг. One-time password) или своје студентске идентификационе картице, а током трајања теста се у одређеним временским интервалима скенира слика студента и упоређује са његовом сликом из базе података. Тестови се могу полагати и на уређајима попут мобилног телефона и таблета, али за тестове који захтевају већу сигурност препоручљиво је полагати их на лаптоп или десктоп рачунарима. Систем се може скалирати у зависности од потреба. У понуди је и Mettl Secure Browser који искључује све екстерне портове, спречавајући кандидата да повеже секундарни екран. Browser такође искључује све софтвере за дељење садржаја и вебсајтове и не допушта напуштање прозора у коме се ради тест пре предаје теста. AI модул је обучен да детектује до 18 типова нерегуларности. [6]

SpeedExam је још једна платформа за онлајн тестирање која нуди напредне функционалности као што су аутоматско оцењивање и брзи извештаји. Велика предност ове платформе је што нуди бесплатну верзију која није временски ограничена и долази са 25 бесплатних тестова месечно, а цена је генерално најповољнија у односу на конкуренцију. Апликација је сагласна са GDPR (енг. General Data Protection Regulation) и омогућава енкриптовано складиштење података. Овај софтвер користи више од 10 000 користика као што су Microsoft, Amazon, Samsung, University of Toronto, Hewlett Packard Enterprise и многи други. Омогућава 30 000 истовремених тестова, менаџмент полагања и кандидата, као и напредне извештаје и анализу. Када нерегуларне активности пређу одређени ниво, систем онемогућава наставак полагања кандидату. Апликација нуди интерфејс на различитим језицима. [7]

Examus је један од најфлексибилнијих сервиса за онлајн надгледање. Користи веб камеру за препознавање лица и детекцију емоција, праћење погледа, детекцију других особа и гласова. Може се користити на мобилним телефонима, нуди једноставан API, високо је скалабилан, штити корисничке податке. Examus нуди два главна производа: Proctor AI – софтвер за онлајн надгледање коришћењем вештачке интелигенције и Examus EQ – алат који надгледа човекову пажњу и ангажпвање током похађања онлајн курсева. AI proctor комбинује податке из три извора (видео, аудио и радна површина рачунара) и онда их анализира коришћењем алгоритама базираних на неуронским мрежама како би креирао детаљан извештај о корисниковом понашању. Систем детектује. Образовне институције, као клијенти, добијају тај извештај праћен видео доказом. У извештају се налазе линкови који нас одводе директно до релевантног дела видео доказа где можемо лично проверити шта се дешавало у том моменту. Бележе се ситуације попут гледања ван екрана, причања, комуницирања са другим особама, промена прозора на екрану, покушај дељења екрана или звука. Како нечија судбина не би зависила искључиво од алгоритама, постоји могућност укључивања људског надгледача који ће у реалном времену проверити забележене нерегуларности. Компанија нуди обучено особље које може кориговати генерисани извештај и проследити неуронској мрежи одговрајуће параметре како би се повећала поузданост извештаја. Са друге стране, Examus EQ, систем за праћење пажње и ангажовања у реалном времену, процесира видео улаз и шаље резулатате анализе клијенту, што омогућава да предавачи добију у реалном времену информацију о томе како публика реагује на њихова предавања и вебинаре. Ово омогућава увођење промена у начину излагања у ходу. Систем може доставити аналитичке податке за комплетну публику или за сваког појединца. Обрада се врши на клијентској страни, што минимизује трошкове израчунавања на серверској страни и избегава потребу за слањем приватних података. Examus EQ је интегрисан са YouTube, што значи да корисник може делити видео који је уплоадован на YouTube са колегама и ученицима и да добије детаљну анализу утисака. Исто важи и за уживо (енг. live) пренос на YouTube. [8]

# **Технике рачунарског вида за праћење регуларности полагања тестова**

Рачунарски вид је поље рачунарске науке које ради на томе да омогући рачунарима да виде, идентификују и обрађују дигиталне слике на исти начин као што то чини људски вид, а затим дају одговарајући излаз. Тежи се разумевању и аутоматизацији задатака које визуелни систем човека може да уради. Као научна дисциплина, рачунарски вид се бави теоријом вештачких система која издваја информације из слика. Као технолошка дисциплина, настоји да примени теорије и моделе за изградњу система рачунарског вида. [9]

Технике рачунарског вида које се могу применити у апликацијама за праћење регуларности полагања тестова су: детекција објеката, детекција лица и издвајање карактеристичних тачака лица (енг. facial landmarks), препознавање лица, естимација позе главе, праћење покрета очију (трептање и гледање са стране), као и праћење покрета усана особе на камери ради детектовања разговора. У овом поглављу биће обрађена и техника поравнања лица (енг. face align) која се користи како би се побољшали резултати препознавања лица и праћења покрета очију и усана.

## Детекција објеката

Детекција објеката је техника која спада у област рачунарског вида и бави се проналажењем објеката који припадају одређеним класама (нпр. људи, зграде, животиње) на дигиталним сликама и видео снимцима. Свака класа објеката има специфичне особине, односно фичере (енг. features), који одређују припадност објекта тој класи. Детекција објеката има широку примену у свакодневном животу (нпр. за детекцију пешака и аутомобила, мерење растојања између људи у некој просторији, препознавање регистарских таблица), а велику примену има и у спорту и медицини. Не казује нам само који објекат је на слици, већ и где се он налази. Детекција објеката се генерално састоји из два корака: први је пронаћи правоугонике који обухватају по један објекат са слике, а други је класификација слике унутар тог правоугаоника. Jedna od najbolje istraženih područja detekcije objekta su detekcija lica i detekcija pješaka

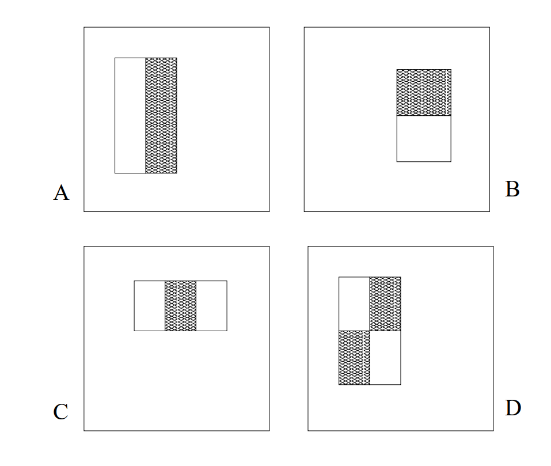
Методе за детекцију објеката се генерално могу поделити на оне које се заснивају на традиционалним техникама обраде слика и оне које су базиране на неуронским мрежама. Код метода које су засноване на традиционалним техникама неопходно је дефинисати фичере објеката и након тога користити неку од техника за класификацију, попут SVM (енг. Support Vector Machine). Примери ових метода су: Viola–Jones детектор, SIFТ детектор (енг. Scale-Invariant Feature Transform) и HOG детектор (енг. Histogram of Оriented Gradients). Технике базиране на неуронским мрежама не захтевају дефинисање карактеристика објеката и најчешће се ослањају на конволуционе неуронске мреже CNN (енг. Convolutional Neural Networks). Примери ових метода су: R-CNN, Fast R-CNN, Faster R-CNN, Single Shot MultiBox Detector (SSD), You Only Look Once (YOLO), Single-Shot Refinement Neural Network for Object Detection (RefineDet), Retina-Net, Deformable convolutional networks. [10]

Данас су рачунари много ефикаснији и имају већу моћ израчунавања него што је то било раније. Тежи се паралелизацији кроз употребу више језгара и AI акцелератора[[2]](#footnote-2). Ово омогућава детекцију објеката у скоро реалном времену. Дубоке конволуционе неуронске мреже у комбинацији са убрзањем које доноси графички процесор значајно су унапредиле процес детекције објеката. Старије методе могу бити ограничене услед постојања комплексне позадине, делимично прекривених објеката, лошег осветљења и шума, док су технике дубоког учења су значајно отпорније на наведене проблеме, али им је недостатак што захтевају велику количину података за тренирање и што процес означавања слика може бити временски захтеван (мада постоје и велики референтни dataset-ови као што су MS COCO, ImageNet и PASCAL VOC који нуде доступне, унапред припремљене податке). Алгоритми дубоког учења појавили су се 2014. године и они се могу поделити на дво-фазне и једно-фазне алгоритме. Ови детектори обављају два задатка: први је проналажење произвољног броја објеката и након тога, други задатак је класификација сваког од њих и одређивање правоугаоника којим се објекат може обухватити. Поступак се може раздвојити на две фазе, а може се и извршити у оквиру једног корака како би се добиле боље перформансе, али по цену поузданости. Детектори који раде у 2 фазе подразумевају проналажење региона објеката, које је праћено класификацијом заснованом на издвојеним фичерима из детектованих региона и bounding-box регресијом[[3]](#footnote-3). Овакви детектори имају велику поузданост, али су спорији. Најпознатији детектори овог типа су: R-CNN, Fast R-CNN, Faster R-CNN и Mask R-CNN. Детектори који раде у једном кораку предвиђају оквирне правоугаонике на слици без претходно издвојених региона од интереса. Бржи су, структурно једноставнији и могу се користити код апликација од којих се очекује да раде у реалном времену. Најпознатији детектори из ове групе су: YOLO који има више верзија и SSD. [12]

У наставку ће бити дат преглед начина функционисања неколико детектора објеката који су били значајни за развој ове области, почевши од Viola-Jones детектора који се први појавио, преко HOG детектора који се појавио пар година касније, до детектора који користе неуронске мреже као што су различите верзије R-CNN-a, YOLO i SSD.

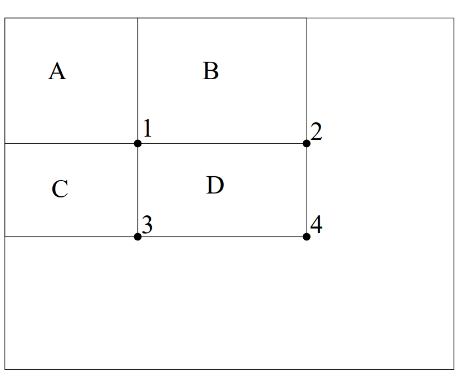
Viola-Jones детектор (Haar каскаде)

Paul Viola и Michael Jones су у свом раду из 2001. године представили приступ за детекцију објеката који примењује машинско учење и обезбеђује веома брзе и поуздане резултате. Iako je prvotno bio namijenjen automatskoj detekciji lica u realnom vremenu, može se koristiti pri detekciji bilo kakvih objekata. Рад обухвата три кључне целине. Прва је представљање новог начина за репрезентацију слике названог „интегрална слика“ који омогућава да се фичери који се користе при детекцији израчунају веома брзо. Коришћен је скуп фичера који подсећају на Haar-ове функције[[4]](#footnote-4). Друга целина је алгоритам за учење заснован на AdaBoost техници који селектује мали број најважнијих фичера из великог скупа фичера и даје веома ефикасан класификатор. Селекција фичера се постиже модификацијом AdaBoost алгоритма тако да сваки генерисан слаби класификатор зависи само од једног фичера. Свака фаза boosting процеса, која селектује нови слаби класификатор може се посматрати као процес селекције фичера. Трећи допринос је метод за комбиновање класификатора у каскаду што омогућава да се позадински региони слике брзо одбаце и обрада врши над регионима који су обећавајући, односно који потенцијално садрже објекат, при чему је сваки класификатор у каскади мало комплекснији од претходног. Слика се дели на прозоре и само прозори који нису одбачени у првом класификатору прослеђују се наредним класификаторима.

Слика се класификује на основу вредности једноставних фичера. Фичери се користе уместо пиксела зато што могу садржати више доменског знања и системи који раде са фичерима су доста бржи од система базираних на пикселима. Користе се три врсте фичера. Вредност фичера који се састоје из два правоуганоника рачуна се као разлика између суме пиксела у оквиру та два правоугаона региона. Правоугаоници су исте величине и хоризонтално или вертикално су поравнати. Вредност фичера од три правоугаоника се рачуна као сума у оквиру два спољашња правоугаоника одузета од суме у средишњем правоугаонику. За фичер од 4 правоугаоника вредност се рачуна као разлика између дијагоналних парова правоугаоника. Основна резолуција детектора је 24x24, а скуп правоугаоних фичера је знатно већи (има их преко 180,000 у сваком потпрозору слике). Примери правоуганих фичера приказани су на слици 1: сума пиксела у белим регионима одузима се од суме пиксела у сивим.

Слика 1: Примери правоугаоних фичера [14]

Интегрална слика на локацији x, y садржи суму пиксела изнад и лево од тачке x, y. На пример, на слици 2, сума пиксела у оквиру правоугаоника D може бити израчуната зависности од вредности у 4 тачке. Вредност интегралне слике на локацији 1 је сума пиксела у правоугаонику А. Вредност у тачки 2 је А+В, у тачки 3 је А+С, а у тачки 4 је А+В+С+D. Сума пиксела у оквиру правоугаоника D би била 4+1 - (2+3). За два правоугаоника сума може израчуната на основу вредности у 8 тачака, међутим пошто су фичери дефинисани са 2 суседна правоугаоника, онда се може израчунати на основу 6 тачака. За случај 3 правоугаоника на основу 8, а за случај 4 правоугаоника 9. Интегрална слика може бити израчуната једном проласком кроз оригиналну слику.



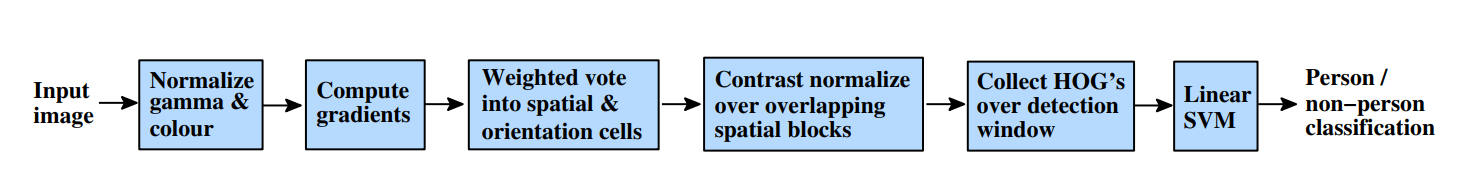
Слика 2: Рачунање суме пиксела у оквиру правоугаоног региона [14]

Иако се појединачни фичери израчунавају прилично једноставно и брзо, рачунање комплетног скупа фичера који има 180,000 елемената је скупо. Да би се издвојио мањи скуп најбитнијих фичера направљен је слабији алгоритам за учење да селектује један правоугаони фичер који најбоље одваја позитивне и негативне примере. За сваки фичер овај алгоритам одређује оптималну threshold класификациону функцију, тако да минимални број примера буде нетачно класификован.

Аутори су се фокусирали на детекцију лица на слици и добили су солидне резултате од 15 фрејмова у секунди на 700 MHz Intel Pentium III за слике димензија 384x288. Систем користи само информације присутне на сивој слици (енг. grey scale image), без узимања у обзир додатних информација попут разлика у сликама у видео секвенци или боје пиксела на обојеним сликама, које могу повећати FPS. Иницијални експерименти су показали да класификатор фронталних лица конструисан са 200 фичера даје detection rate од 95% са false positive rate од 1 у 14084. Са повећањем броја фичера расте и време извршавања. Предложени алгоритам је детектовао објекте на сликама независно од њихове локације и скалирања слике. Једна од највећих предности овог детектора у односу на остале је велика брзина детекције у реалном времену, а недостатак што је склон лажно позитивним детекцијама, захтева подешавање параметара и није толико поуздан као модернији детектори. И даље се користи када се ради на ограниченим уређајима на којима не можемо извршавати детекторе који су рачунски захтевнији. [14]

HOG детектор

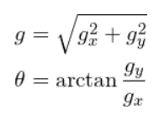
У раду из 2005. године приказано је како се Histogram of Oriented Gradients (HOG) дескриптор слике и Linear Support Vector Machine (SVM) могу користити за детекцију објеката. Аутори су детектовали особе, али генерално могу се детектовати и друге класе објеката. Предложени дескриптори су рачунати над густим гридом униформно распоређених ћелија и користе overlapping local contrast normalizations за побољшање перформанси. In practice this is implemented by dividing the image window into small spatial regions (“cells”), for each cell accumulating a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell. The combined histogram entries form the representation. For better invariance to illumination, shadowing, etc., it is also useful to contrast-normalize the local responses before using them. Tiling the detection window with a dense (in fact, overlapping) grid of HOG descriptors and using the combined feature vector in a conventional SVM based window classifier gives our human detection chain. [15]

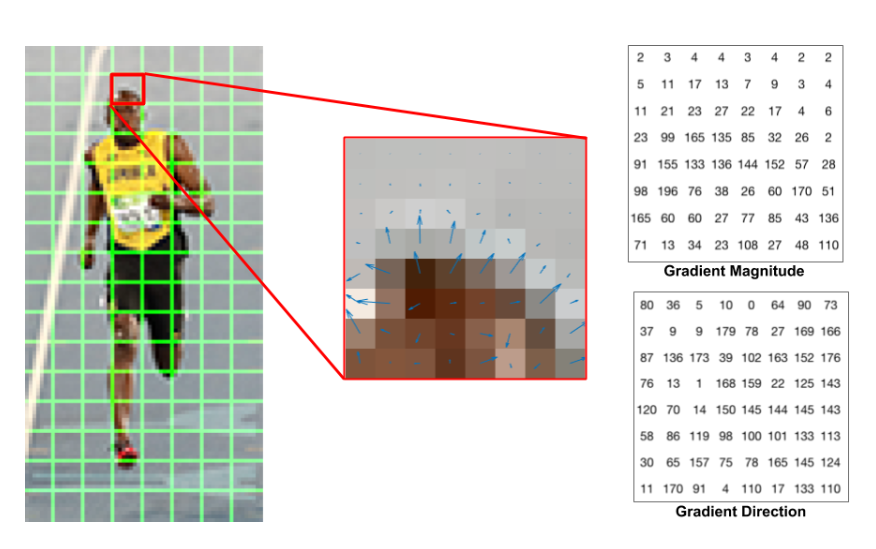


Фичер дескриптор је репрезентација слике или дела слике која поједностављује слику издвајајући корисне информације из ње и одбацивајући ирелевантне информације. Он конвертује слику димензија width x height x 3 (канала ) у вектор (низ) фичера дужине n. Код HOG дескриптора, улазна слика је димензија 64 x 128 x 3, а излазни вектор је дужине 3780. Када се овај вектор фичера проследи алгоритму за класификацију попут Support Vector Machine (SVM) добијају се добри резултати. У HOG фичер дескриптору, дистрибуција (хистограми) смерова оријентисаних градијената се користе као фичери. Градијенти слике ( x и y изводи ) су корисни зато што је магнитуда градијената велика око ивица и ћошкова (региони наглих промена интензитета), а ивице и ћошкови носе много више информација о облику објекта него равни региони.

Први корак у проналажењу хистограма оријентисаних градијената је преприцесирање слике. У оригиналном раду HOG дескриптор фичера је рачунат над деловима слике величине 64×128, али генерално слика може бити било које величине. Посматрају се делови слике са различитим скалирањем и на различитим локацијама. Једино ограничење је да парче које се обрађује има фиксни aspect ratio. У нашем случају, aspect ratio треба да буде 1:2. Дакле, са слике се издвоји део који има овај aspect ratio и димензије му се промене на 64×128. Затим је потребно израчунати хоризонталне и вертикалне градијенте. Ово се може постићи филтрирањем слике Собел филтером.

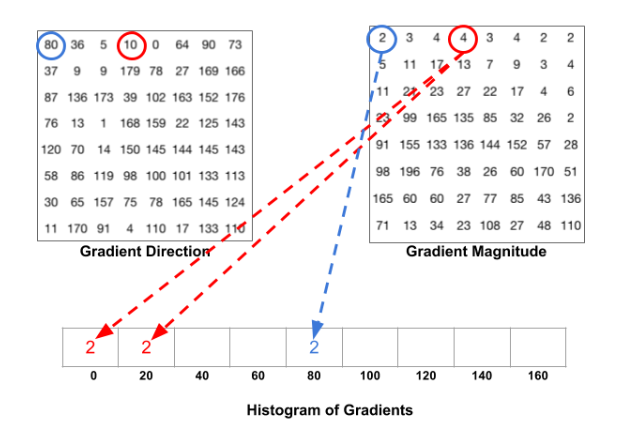
Слика 3Left : Absolute value of x-gradient. Center : Absolute value of y-gradient. Right : Magnitude of gradient.

На слици 3 можемо приметити да x-градијент издваја вертикалне линије, а y-градијент хоризонталне. Магнитуда градијената се јавља где год постоје нагле промене у интензитету. Финална слика садржи истакнуте све ивице, и може се видети да је особа у првом плану, а већина ирелевантних информација је уклоњена. У сваком пикселу градијент има магнитуду и смер. Код слика у боји, рачунају се градијенти за сва три канала, па се за магнитуду у сваком пикселу узима максимална магнитуда та три градијента, а за угао онај угао који одговара максималном градијенту. Магнитуда и угао се рачунају по следећим формулама:

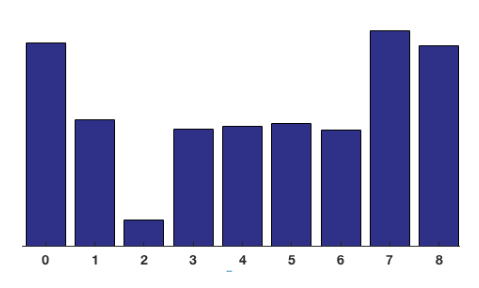
где су gx и gy градијенти по x и y осама. Након овога се слика дели на ћелије 8×8 и хистограм градијената се рачуна за сваку ћелију. Део слике 8×8 садржи 8x8x3 = 192 вредности пиксела. Градијент овог парчета садржи 2 вредности (магнитуду и смер) за сваки пиксел, што даје 8x8x2 = 128 бројева. Индивидуални градијенти могу садржати шум, али хистограм над 8×8 парчетом је много мање осетљив на шум. Ове димензије су одабране зато што је иницијално рађена детекција пешака и ћелије 8×8 над сликом пешака скалираном на 64×128 су биле довољно велике да се издвоје значајни фичери. Хистограм је низ од 9 елемената који одговарају угловима 0, 20, 40, 60 … 160.

Слика 4: Left: 8×8 cells of HOG. Image is scaled by 4x for display.Center : The RGB patch and gradients represented using arrows. Right : The gradients in the same patch represented as numbers

Слика у средини приказује парче слике на коме су стрелицама представљени смерови градијената, а дужина стрелица представља магнитуду. Смерови стрелица показују смер промене интензитета, а њихова дужина колико је велика разлика. Десно можемо видети бројеве који репрезентују градијенте у 8×8 ћелијама, уз разлику што су углови између 0 и 180 степени уместо 0 и 360. Ово су „неозначени“ градијенти јер су градијент и његова негативна варијанта представљени истим бројем. Неке имплементације HOG нам омогућавају да селектујемо да ли желимо да користимо означене градијенте.

Следећи корак је креирање хистограма градијената у оквиру ових 8×8 ћелија. Хистограм садржи 9 елеменета који одговарају угловима 0, 20, 40 … 160. Слика 5 илуструје процес. Посматрамо исто парче као на претходној слици.

Слика 5 Хистограм градијената

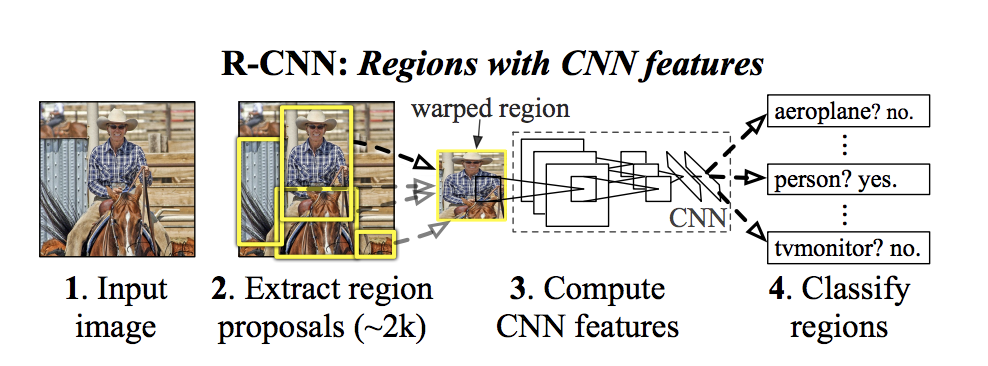
Елемент је селектован на основу смера, а вредност елемента на основу магнитуде. Ако погледамо пиксел означен плавом бојом, приметићемо да он има угао (смер) од 80 степени и магнутуду 2, тако да 2 иде на 5. позицију. Градијент пиксела заокружен црвеном бојом има угао од 10 степени који је на средини између 0 и 20, тако да се његова вредност 4 уписује на оба места (по 2). Ако је угао већи од 160 степени, он је између 160 и 180 степени, а угао 0 и 180 су еквивалентни, тако да би се његова вредност пропорционално уписала у елемент намењен углу од 0 и 160 степени.

Слика 6Хистограм

На слици 6 је приказано како би изгледао хистограм за описани пример. Како у хистограму доминирају вредности близу 0 тј. 180 степени, може се закључити да ова ћелија вероватно садржи ивицу.

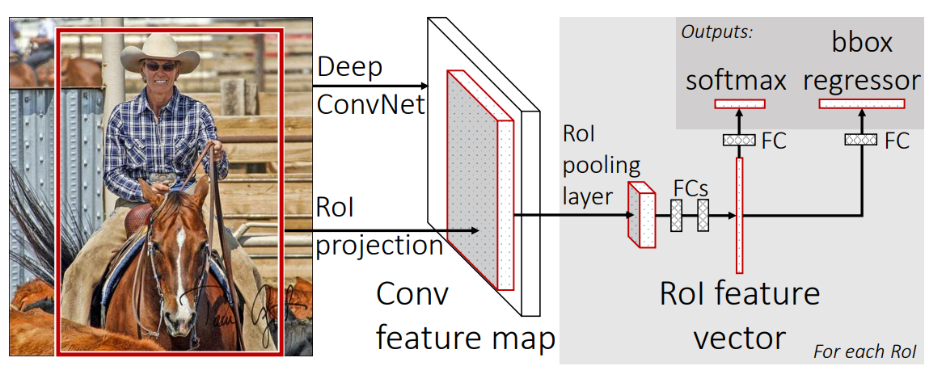
Како би дескриптор био инваријантан на промене у осветљењу, треба да нормализујемо хистограм. Нормализација подразумева дељење сваке вредности вектора са вредношћу вектора (L2 нормом). Два вектора која се разликују само у скалирању свешће се на исти вектор након нормализације. Ако посматрамо блок 16×16, он има 4 хистограма који када се надовежу формирају вектор од 36 елемената који се може нормализовати. Прозор се онда помера за 8 пиксела и процес се понавља. Да би се израчунао коначни вектор фичера за комплетно парче слике, вектори дужине 36 се конкатенирају и креирају један велики вектор од 3780 елемената (36×105, где је 7 x 15 = 105 број позиција на којима може да се нађе прозор). Овај вектор се сада користи за тренирање класификатора као што је SVM како би се обавила детекција објеката. [16]

### R-CNN

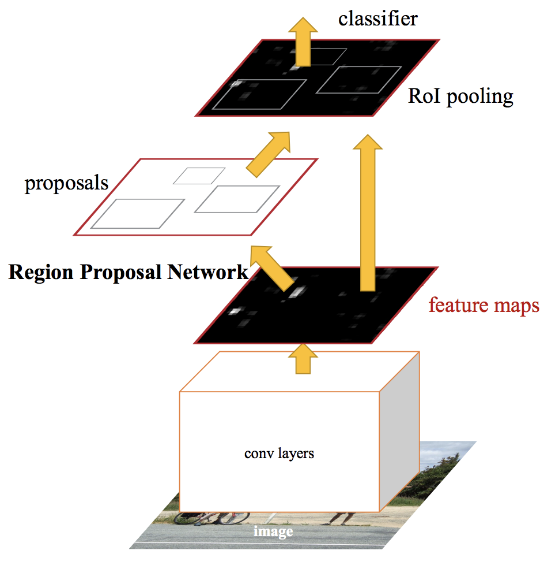
R-CNN (Region-based Convolutional Neural Networks) детектор из 2013. је један од првих који примењује дубоко учење за детекцију објеката. На слици 7 је приказано како модел функционише. Систем за детекцију објеката се састоји из три модула. Први модул применом селективног тражења издваја из улазне слике око 2000 предложених региона који представљају кандидате за детекцију објеката. Ови региони су у суштини правоугаоници за које постоји вероватноћа да садрже објекат (без информације о томе којој класи припада тај објекат). Други модул је конволуциона неуронска мрежа која издваја фичере (векторе фичера фиксне дужине од 4096 елемената) из сваког предложеног региона. Фичери се рачунају пропуштањем mean-subtracted 227 × 227 RGB слике кроз 5 конволуционих слојева и 2 потпуно повезана слоја. Трећи модул је скуп class-specific linear SVMs уз помоћ којих се класификује сваки од региона. За сваку класу се одређује score за сваки издвојени вектор фичера коришћењем SVM-а који је трениран за ту класу. Након класификације, примењује се пост-процесирање да се пречисте оквирни правоугаоници и елиминишу дупликати. R-CNN достиже mAP од 53.7% на PASCAL VOC 2010, али је веома спор. [17]

Слика 7: Начин функционисања R-CNN модела [17]

Fast R-CNN из 2015. године је побољшање R-CNN метода у смислу брзине тренирања и тестирања, као и тачности детекције. На слици 8 приказана је архитектура R-CNN-а. Мрежа која као улаз има комплетну слику и скуп предложених региона процесира слику и генерише конволуциону мапу фичера (conv feature map). Дакле, генерише се једна мапа фичера, а не 2000. Затим, за сваки предложени регион, RoI (region of interest) pooling слој издваја вектор фичера фиксне дужине из мапе фичера за сваки од региона. Сваки фичер вектор се затим користи за класификацију региона у једну од класа и побољшање тачности ориналног обухватајућег правоугаоника коришћењем bounding box regressor-а. [18]

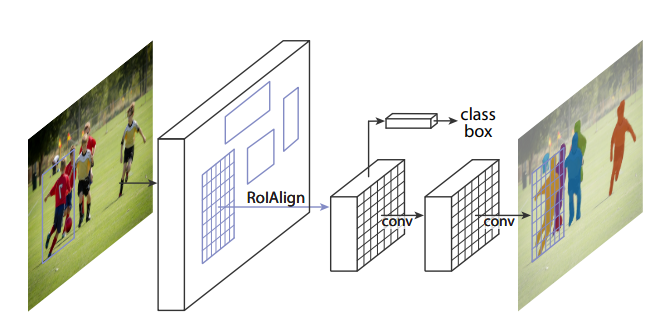


Слика 8: Архитектура Fast RCNN-а [18]

Претходна два алгоритма користила су селективну претрагу за проналажење региона која је веома спора, односно представља уско грло у систему. Уместо овог алгоритма, Faster R-CNN користи дубоку конволуциону мрежу која предлаже регионе. На слици 9 се може видети како изгледа архитектура комплетног система. Један модул је RPN (Region Proposal Network) која на основу конволуционе мапе фичера издваја предложене регионе, а други модул на основу исте мапе фичера и предложених региона ради класификацију. Комплетан систем је јединствена неуронска мрежа за детекцију објеката. [19]

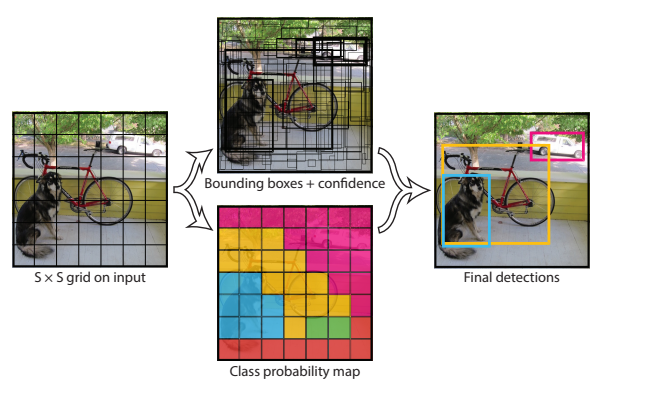
Слика 9: Faster CNN [19]

Mask R-CNN из 2017. године је унапређење Faster R-CNN. Разлика између њих је у томе што Mask R-CNN додаје грану за предикцију маске објекта паралелно са постојећом граном за препознавање оквирног правоугаоника. Маска која се рачуна на нивоу пиксела нам омогућава да раздвојимо објекат од позадине примењујући сегментацију инстанци (енг. instance segmentation). У овом моделу ROI Pooling модул је замењен поузданијим ROI Align модулом и додата је још једна грана из ROI Align модула. Ова додатна грана прихвата излаз ROI Align модула и прослеђује га у два конволуциона слоја, чији излаз је сама маска. [20]



Слика 10. The Mask R-CNN framework for instance segmentation

YOLO детектор

YOLO детектор је презентован у раду који је објављен 2015. године и представљао је нови приступ детекцији објеката. За разлику од детектора који у основи имају класификатор и примењују детекцију над различитим деловима и величинама слике, детекција објеката овде је посматрана као проблем регресије просторно одвојених оквирних правоугаоника и њима придружених вероватноћа. Неуронска мрежа предвиђа оквирне правоугаонике и одговарајуће вероватноће директно из комплетних слика једним погледом на слику. Одатле и потиче назив детектора („погледај само једном“). Модел детектора приказан је на слици 11. Систем дели улазну слику на грид димензија S × S. Ако центар објекта упада у ћелију грида, та ћелија ће бити задужена за његову детекцију. Свака ћелија предвиђа B обухватајућих правоугаоника и поузданости Pr(Object) ∗ IOU за њих. Ове поузданости казују колика је вероватноћа да правоугаоник садржи објекат. Сваки обухватајући правоугаоник има 5 предикција: x, y, w, h, и confidence. Координате (x, y) представљају центар правоугаоника релативно у односу на границе ћелије грида. Ширина и висина (width и height) представњене су релативно у односу на целу слику. Поузданост (confidence) представља IOU[[5]](#footnote-5) (intersection over union) између предвиђеног обухватајућег правоугаоника и било ког ручно означеног правоугаоника у тест скупу слика који специфицира где се објекат налази. Свака ћелија грида такође предвиђа C условних вероватноћа за класе Pr (Classi | Object). Ове вероватноће представљају вероватноћу појављивања објеката сваке од класа у грид ћелији. Предвића се само један скуп вероватноћа по ћелији независно од броја обухватајућих правоугаоника B. Приликом тестирања множе се условне вероватноће припадања класи и индивидуалне вероватноће за сваки обухватајући правоугаоник, што нам даје вероватноћу припадања класи за сваки обухватајући правоугаоник: Pr(Classi|Object) ∗ Pr(Object) ∗ IOU = Pr(Classi) ∗ IOU. Ово нам даје вероватноћу да се та класа појавњује у правоугаику, као и информацију о томе колико добро предвиђени правоугаоник представља објекат. Модел је имплементиран као конволуциона неуронска мрежа и евалуиран је на PASCAL VOC скупу података. Иницијални конволуциони слојеви издвајају фичере из слике, док потпуно повезани слојеви предвиђају излазне вероватноће и координате. Мрежа има 24 конволуционих слојева и 2 потпуно повезана слоја. Бржа варијанта мреже има мање конволуционих слојева (9) и мање филтера у тим слојевима. Како је при тренирању узето S = 7, B = 2 и C = 20 јер коришћени скуп података има 20 класа, излаз мреже је 7 × 7 × 30 тензор. 

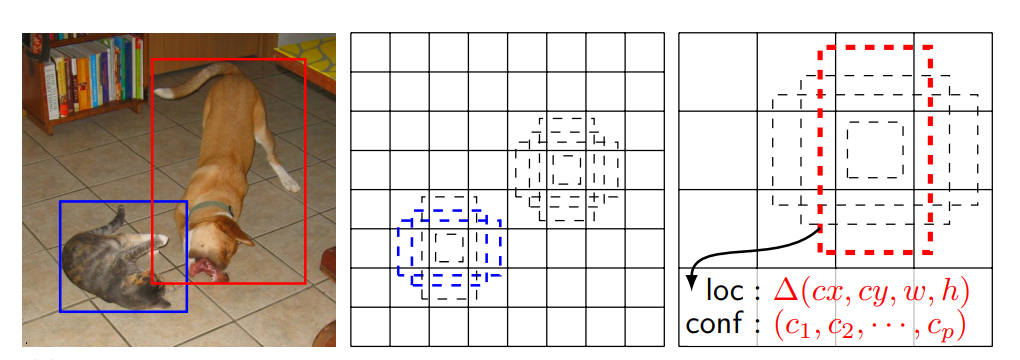
Слика 11Модел YOLO детектора [21]

Архитектура је веома брза, основни модел обрађује у реалном времену 45 фрејмова у секунди на Titan X GPU, док мања верзија мреже Fast YOLO обрађује преко 155 фрејмова у секунди, задржавајући дупло већи mAP од осталих детектора који раде у реалном времену. Мрежа је тренирана отприлике недељу дана на ImageNet 1000-class competition dataset-у и достигнута је тачност од 88% на ImageNet 2012 validation set-у. YOLO прави више грешака у лоцирању (поготово мањих објеката), али знатно мање лажних предикција када нема ничега на слици. Обучаван је на генерализованим репрезентацијама објеката, од природнних слика до уметничких. [21]

SSD детектор

Овај метод из 2016. године детектује објекте на сликама коришћењем дубоке конволуционе неуронске мреже тако што дискретизује излазне оквирне правоугаонике у скуп подразумеваних правоугаоника са различитим aspect ratios и скалирањем по локацијама у фичер мапи. Генерише вероватноће за присуство сваке од класа објеката у сваком од подразумеваних правоугаоника и прилагођава правоугоник да боље одговара облику објекта. Мрежа комбинује предикције из различитих мапа фичера са различитим резолуцијама како би се обухватили објекти различитих величина.

his paper presents the first deep network based object detector that does not resample pixels or features for bounding box hypotheses and and is as accurate as approaches that do. The fundamental improvement in speed comes from eliminating bounding box proposals and the subsequent pixel or feature resampling stage. The core of SSD is predicting category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps. – To achieve high detection accuracy we produce predictions of different scales from feature maps of different scales, and explicitly separate predictions by aspect ratio. – These design features lead to simple end-to-end training and high accuracy, even on low resolution input images, further improving the speed vs accuracy trade-off.



Слика 12: SSD модел

На слици 12 приказан је SSD модел. SSD захтева само улазну слику и ground truth boxes за сваки објекат током тренирања. Евалуира се мали скуп (нпр. 4) default boxes различитих aspect ratios на свакој локацији у неколико мапа фичера са различитим скалирањем (нпр. 8 × 8 и 4 × 4 на друге две слике). За сваки default box, предвиђа се shape offsets and the confidences за све категорије објеката ((c1, c2, · · · , cp)). Током тренирања, прво се ове default boxes поклопе са ground truth boxes. For example, we have matched two default boxes with the cat and one with the dog, which are treated as positives and the rest as negatives. The model loss is a weighted sum between localization loss (e.g. Smooth L1 [6]) and confidence loss (e.g. Softmax).

We associate a set of default bounding boxes with each feature map cell, for multiple feature maps at the top of the network. The default boxes tile the feature map in a convolutional manner, so that the position of each box relative to its corresponding cell is fixed. At each feature map cell, we predict the offsets relative to the default box shapes in the cell, as well as the per-class scores that indicate the presence of a class instance in each of those boxes. Specifically, for each box out of k at a given location, we compute c class scores and the 4 offsets relative to the original default box shape. This results in a total of (c + 4)k filters that are applied around each location in the feature map, yielding (c + 4)kmn outputs for a m × n feature map.

Детектор је једноставан за тренирање и интеграцију са системима који захтевају компоненту за детекцију објеката. За улаз 300×300 достиже 72.1% mAP на VOC2007 и 58 FPS на Nvidia Titan X а за 500×500 улаз, достиже 75.1% mAP, чиме превазилази Faster R-CNN модел. [23]

SSD уводи неколико побољшања која укључују multi-scale features и default boxes. Детекција објеката се састоји из два дела: издвајање мапа фичера и примена конволуционих филтера за детекцију објеката. Слика се дели на 38x38 ћелија и за сваку ћелију (локацију) се генеришу 4 предикције за објекат. Свака предикција садржи оквирни правоугаоник и 21 вредност за сваку класу (једна додатна класа за „no object“) и онда се одабира класа са највећом вредношћу. Генерисање више предикција које садрже оквирне правоугаонике и поузданости се назива multibox. Локација и вредности класа се рачунају коришћењем малих конволуционих филтера.

For each feature map layers, it shares the same set of default boxes centered at the corresponding cell. But different layers use different sets of default boxes to customize object detections at different resolutions. The 4 green boxes below illustrate 4 default boundary boxes. Default boundary boxes are chosen manually. SSD defines a scale value for each feature map layer.

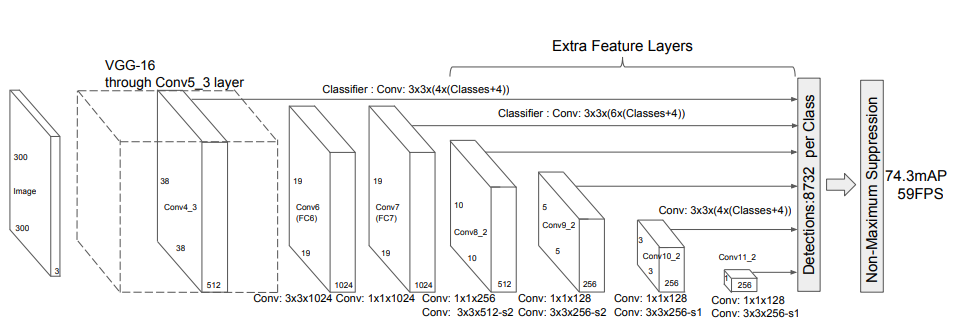
SSD predictions are classified as **positive** matches or negative matches. SSD only uses positive matches in calculating the localizationcost (the mismatch of the boundary box). If the corresponding **default boundary box** (not the predicted boundary box) has an IoU greater than 0.5 with the ground truth, the match is positive. Otherwise, it is negative. (**IoU,**the intersection over the union, is the ratio between the intersected area over the joined area for two regions.)

Here is an example of how SSD combines multi-scale feature maps and default boundary boxes to detect objects at different scales and aspect ratios. The dog below matches one default box (in red) in the 4 × 4 feature map layer, but not any default boxes in the higher resolution 8 × 8 feature map. The cat which is smaller is detected only by the 8 × 8 feature map layer in 2 default boxes (in blue).

Higher-resolution feature maps are responsible for detecting small objects. The first layer for object detection conv4\_3 has a spatial dimension of 38 × 38, a pretty large reduction from the input image. Hence, SSD usually performs badly for small objects comparing with other detection methods. If it is a problem, we can mitigate it by using images with higher resolution.

SSD uses **non-maximum suppression**to remove duplicate predictions pointing to the same object. SSD sorts the predictions by the confidence scores. Start from the top confidence prediction, SSD evaluates whether any previously predicted boundary boxes have an IoU higher than 0.45 with the current prediction for the same class. If found, the current prediction will be ignored. At most, we keep the top 200 predictions per image.

# Single Shot MultiBox

* **Single Shot:**this means that the tasks of object localization and classification are done in a single forward pass of the network
* **MultiBox:**this is the name of a technique for bounding box regression developed by Szegedy et al. (we will briefly cover it shortly)
* **Detector:**The network is an object detector that also classifies those detected objects

SSD’s architecture builds on the VGG-16 as the base network. Instead of the original VGG fully connected layers, a set of auxiliary convolutional layers (from conv6 onwards) were added, thus enabling to extract features at multiple scales and progressively decrease the size of the input to each subsequent layer.

The bounding box regression technique of SSD is a method for fast class-agnostic bounding box coordinate proposals. Interestingly, in the work done on MultiBox an [Inception](https://arxiv.org/abs/1409.4842)-style convolutional network is used. The 1x1 convolutions that you see below help in dimensionality reduction since the number of dimensions will go down (but “width” and “height” will remain the same).

## Детекција лица

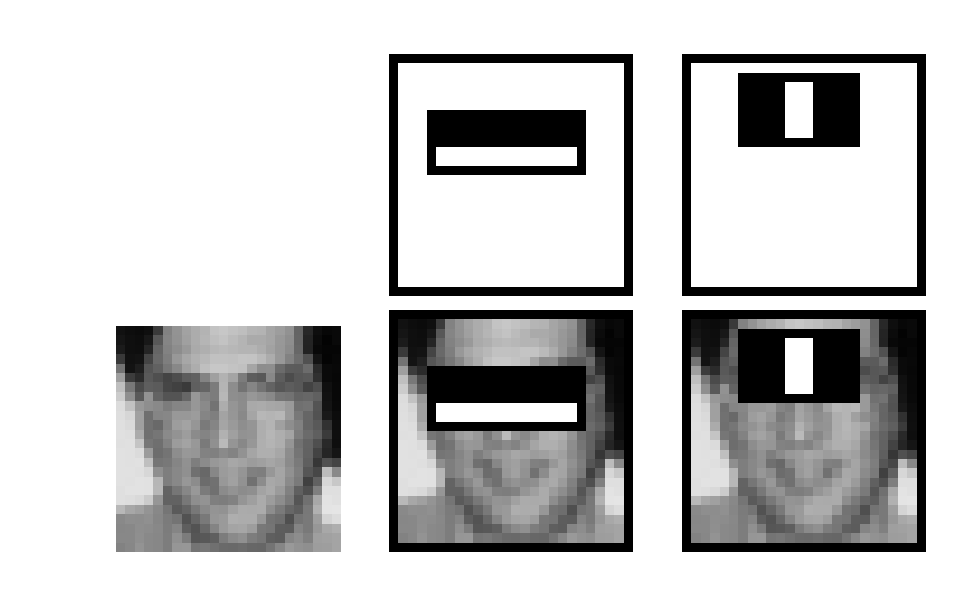
Детекција лица је рачунарска техника која се користи за проналажење људског лица на дигиталним сликама. Може се сматрати спефицичним случајем детекције објеката. Односи се на испитивање да ли се на слици налази људско лице, где је лоцирано и које је величине. Први корак било које обраде лица, попут налажења карактеристичних тачака лица, препознавања лица и детекције покрета лица, је детекција лица. При проналажењу лица на слици неопходно је узети у обзир оријентацију лица, израз лица, осветљење, прекривеност лица (нпр. наочаре, коса, брада), резолуцију слике, комплексност позадине (присуство великог броја објеката), боју коже итд.

Постоји велики број алгоритама за детекцију лица. Yang, Kriegman, and Ahuja представили су у свом раду класификацију алгоритама за детекцију лица у 4 категорије: алгоритми засновани на знању (енг. Knowledge-based methods), алгоритми засновани на непроменљивим особинама (енг. Feature invariant approaches), алгоритми који користе поклапање шаблона (енг. Template matching methods) и алгоритми засновани на изгледу (Appearance-based methods). Алгоритам може припадати већем броју категорија.

1. Алгоритми засновани на непроменљивим особинама лоцирају лице издвајањем структурних карактеристика лица које не зависе од позе и осветљења, као што су нос, уста, очи и обрве. Најпре се ртенира класификатор који се након тога користи за разликовање региона на којима је лице и не којима није лице.
2. Алгоритми засновани на знању ослањају се на скуп правила која се односе на познате карактеристике људског лица. На пример, лице обично садржи два ока, нос и уста који се налазе на одређеном растојању и позицији једни у односу на друге. Могу се формирати и детаљнија правила која се добијају експериментално тражењем просечних вредности за растојања и позиције специфичних црта лица. Проблем код овог модела је креирати одговарајући скуп правила. Ако су правила сувише општа резултоваће добијање великог броја лажно позитивних лица, а ако су сувише детаљна, резултоваће појављиваер великог броја лажно негативних лица. Нису применљиви за све боје коже и зависе од осветљења које може знатно утицати на нијансу коже особе на слици.
3. Алгоритми засновани на поклапању шаблона користе предефинисане шаблоне лица који добро описују лице за детекцију лица рачунајући разлику између шаблона и улазне слике. Проблем са овим методама је што не подржавају веријације у пози, валичини и облику.
4. Алгоритми засновани на изгледу зависе од скупа тренинг слика које се користе за налажење модела лица. Заснивају се на машинском учењу и статистичким методама како би се одредиле релевантне карактеристике лица. Ове методе су показале знатно боље перформансе у порешењу са осталим. [24]

У овом поглављу биће дат преглед неколико најпознатијих детектора који се могу користити за детекцију лица, као што су Haar cascades i DNN детектор из OpenCV библиотеке, HOG + Linear SVM i CNN из dlib библиотеке i MTCNN (Multi-task Cascaded Convolutional Neural Network). [25]

1. Haar cascades

Овај модел је већ описан када је било речи о детекцији објеката. Када се ради о детекцији лица, први фичер наглашава особину да је регион око очију углавном тамнији од региона носа и образа. Други фичер се заснива на особини да је регион очију тамнији од региона носа. Ова два фичера су селектована AdaBoost алгоритмом и на слици 13 се може видети како изгледа када се они поклопе са типичним лицем које се користи за тренирање. Први фичер мери разлику у интензитету између региона очију и региона горњег дела образа. Други филер упоређује интензитете региона очију и моста носа.

Слика 13фдсфсд [14]

Ово је оригинални детектор лица који је испоручен уз OpenCV библиотеку, истовремено и напознатији детектор. Веома је брз и погодан за извршавање у реалном времену, нема велике хардверске захтеве и има мали модел (око 400KB). Са друге стране, подлежан је лажно позитивним детекцијама и захтева ручно подешавање параметара. Треба га користити када је брзина извршавања приоритет и спремни смо да жртвујемо поузданост. Неће радити ако је лице прекривено нечим или није адекватно оријентисано. Prednosti upotrebe ovog algoritma jesu jednostavnost, brzina i visoke performanse

1. Dlib’s HOG + Linear SVM

Овај детектор ради тако што фичере издвојене преко Histogram of Oriented Gradients (HOG) прослеђује SVM. Algoritam se zasniva na piramidama slika[[6]](#footnote-6) i klizećem prozoru[[7]](#footnote-7). Поузданији је од Haar cascades, има стабилнију детекцију и добро је документован. Недостаци су што ради само са фронталним лицима јер HOG дескриптор не толерише промене у ротацији и углу гледања, није толико поуздан као детектори засновани на дубоком учењу и скуп је у виду захтева за рачунањем јер ради конструкцију клизећег прозора и рачунање HOG фичера за сваки прозор.

1. Dlib’s CNN (A Max-Margin (MMOD) CNN face detector [12] [11]

Метода је веома поуздана захваљујући самом дизајну алгоритма и квалитетном тренинг скупу. Модел је мали (мањи од 1MB), веома добро имплементиран и документован. Захтева да dlib буде инсталирана, конвертовање прозора за уоквиравање лица ако се користи OpenCV[[8]](#footnote-8), немогуће га је користити у реалном времену без GPU убрзања. Препорука је да се користи када не треба бринути о перформансама у реалном времену.

1. MTCNN [13]

Представљен је 2016. године и поред детекције лица, детектује и 5 карактеристичних тачака лица. Користи каскадну структуру са три нивоа CNN-a. Прво, користи конволуциону мрежу да прикупи прозоре-кандидате и њихове регресионе векторе који су преклопљени коришћењем on-maximum suppression (NMS). Ови кандидати се прослеђују другој конволуционој мрежи која елиминише велики број лажно позитивних детекција и калибрише обухватајуће правоугаонике. У последњој фази се обавља детекција карактеристичних тачака лица.

1. Детектор лица базиран на дубоком учењу из OpenCV библиотеке

Дубоко учење је једна од напопуларнијих и најбрже растућих области у рачунарском виду. Од верзије 3.1 OpenCV библиотеке псотоји модул за дубоко учење који подржава велики број framework-а за дубоко учење попут. [14]

У питању је Caffe модел заснован на Single Shot Detector (SSD) и ResNet мрежи, брз је и поуздан. Може се извршавати у реалном времену и модел није превелики (око 10MB). Поузданији је од Haar cascades i HOG + Linear SVM, али мање поуздан од CNN MMOD из dlib библиотеке. Један од највећих недостатака је што је мање поуздан за детекцију лица са тамнијом бојом коже, али то се може решити тренирањем модела скупом слика који садржи слике људи различитих раса. Постоји и квантована 8-bit unsigned int Tensorflow верзија. Модел ради добро када је лице делимично прекривено, када постоје брзи покрети главе, може детектовати бочна лица и веома је брз.

## Препознавање лица

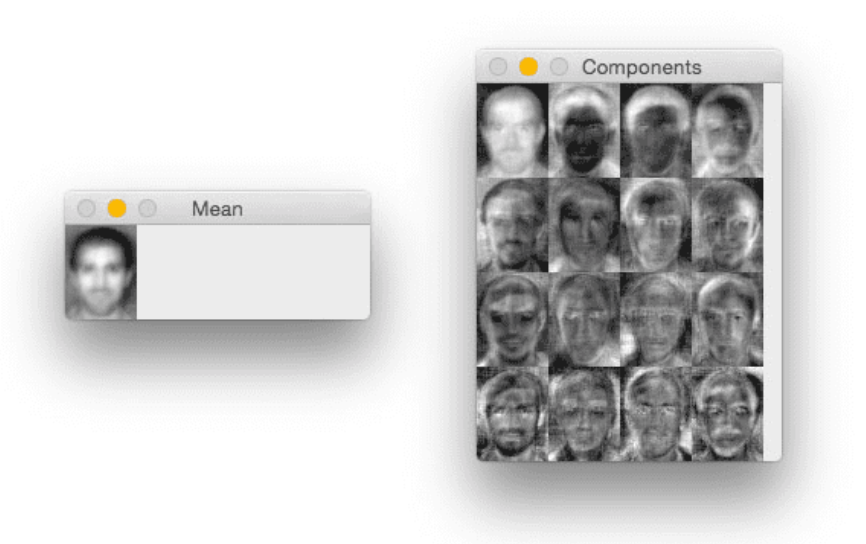
Препознавање лица је процес утврђивања коме одређено лице за слике припада (идентификације особе), односно, упоређивање неког лица са познатим лицима из базе података у циљу проналажења поклапања. Не мора се у свим применама радити упоређивање са лицима из базе података јер је некада потребно само дозволити приступ одређеној особи, а осталим забранити, као на пример код откључавања телефона. Препознавању најчешће претходи детекција лица како би се обрађивао само део слике који представља лице. Постоје и 3D системи за препознавање лица који укључују додатне информације о облику лица у простору и они могу дати боље резултате него 2D системи. Алгоритми се генерално разликују по понашању при екстремним условима као што су лоше осветљење, различити положаји лица, експресија лица, лица мале резолуције, прекривеност лица (наочаре за сунце, капа, брада) итд.

Прво су били развијени системи који користе геометрију лица за идентификацију (позиција и величина очију, носа, образа и браде), затим системи који користе алгоритме машинског учења (издвајање фичера и тренирање класификатора), а у скорије време све више се користе алгоритми дубоког учења. Један од најзначајнијих алгоритама за развој препознавања лица је Eigenfaces, где је коришћена техника линеарне алгебре за редукцију димензија која се зове Principal Component Analysis (PCA). Затим је објављена метода Fisherfaces која користи Linear Discriminant Analysis (LDA). Појавила се и метода базирана на фичерима Local Binary Patterns (LBP) која се и данас користи у многим апликацијама. Када је у питању дубоко учење, постоје специјалне архитектуре које се зову siamese networks. FaceNet и OpenFace су једни од најпопуларнијих модела дубоког учења који се користе за препознавање лица. [26]

Eigenfaces

Први корак у Eigenfaces алгоритму је обезбедити по неколико слика за сваку особу коју желимо да прeпознамо. Слике су представљене у grayscale формату као K×K битмапа пиксела (није нужно да буду квадратног облика). За сваку слику треба креирати један вектор дужине К2 добијен линеризацијом битмапе слике. Затим се креира матрица М димензија ZxK2 где је Z укупан број слика у тренинг скупу, добијена смештањем вектора сваке слике у једну врсту. Над оваквом матрицом примењује се PCA (principal component alalysis – анализа принципалних компоненти) на следећи начин:

1. Рачуна се просечна за сваку колону у матрици, што нам даје просечан интензитет пиксела за сваку (x, y) координату у скупу слика.
2. Израчуната просечна вредност се одузме од сваког елемента одговарајуће колоне чиме се постиже усредњавање података
3. Израчуна се covariance matrix
4. Примењује се eigenvalue decomposition над covariance matrix како би се добиле eigenvalues и eigenvectors
5. Eigenvectors се сортирају по eigenvalues, од највећег до најмањег.
6. Узме се првих N eigenvectors са највећом eigenvalue magnitude
7. Улазни подаци се трансформишу пројектовањем (i.e., taking the dot product) на простор који креира првих N eigenvectors-ови eigenvectors се називају eigenfaces (својствена лица)

Сада имамо матрицу V димензија NxK2. Сваки од eigenfaces је вектор димензија K2 и може се посматрати као битмапа димензија K2. На слици лево представљено је просечно лице лица из скупа лица, а на слици десно приказано је првих 16 eigenface репрезентација које представљају визуелизацију димензија у којима лица из скупа највише варирају. Светлији региони одговарају већим варијацијама, а тамнији мањим. Here, we can see that our eigenface representation captures considerable variance in the eyes, hair, nose, lips, and cheek structure.

Ilustracija 1Linear Combination of EigenFaces [28]

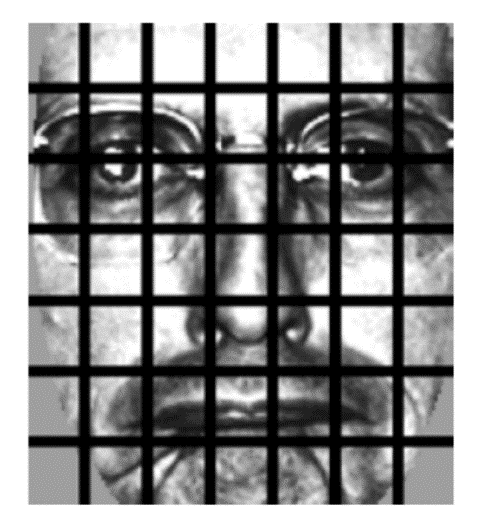
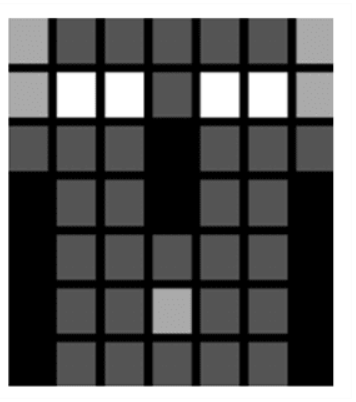
Ново лице које треба препознати се модификује тако што се од њега одузима средње лице и представља се као линерана комбинација principal components. Затим се рачуна Еуклидско растојање између projected eigenface representations тренинг слика и тест слике (k-NN classifier). Prepoznavanje lica je zasnovano na upoređivanju koeficijenata linearne reprezentacije. The smaller the distance, the more similar the faces are considered to be. Лице се идентификује као лице са којим има најмање растојање. Уместо k-NN алгоритма могу се користити напреднији алгоритми попут SVM. Један од највећих недостатака овог алгоритма је што захтева поравнање лица приликом тренирања и препознавања зато што се ради на нивоу пиксела, тако да је потребно да се карактеристике лице скоро савршено поклапају на сликама. [27]

Fisherfaces

Овај алгоритам ради нешто боље и стабилније се понаша при различитим интензитетима осветљења, комплекснији је. Fisherfaces je uz Eigenfaces jedan od najpopularnijih algoritama prepoznavanja lica. Ovaj algoritam uvjetuje poznavanje Eigenfaces algoritma iz razloga što se nadovezuju jedan na drugi. Dok se Eigenfaces algoritam koristi analizom glavnih komponenti pri prepoznavanju lica, Fisherface algoritam koristi Fisherfaces linearnu diskriminantnu analizu. Fisherfaces analiza temelji se na svođenju d-dimenzionalnog vektora značajki na jednu dimenziju, što je potrebno kako bi se izvršila klasifikacija.

LBP

Local Binary Patterns are a texture descriptor. LBPs compute a local representation of texture. This local representation is constructed by comparing each pixel with its surrounding neighborhood of pixels. The first step in constructing the LBP texture descriptor is to convert the image to grayscale. For each pixel in the grayscale image, we select a neighborhood of size r surrounding the center pixel. A LBP value is then calculated for this center pixel and stored in the output 2D array with the same width and height as the input image. Први корак у алгоритму је подела лица на матрицу од 7×7 једнаких ћелија. Затим се за сваку од ћелија израчуна Local Binary Pattern histogram. Хистограм одбацује просторне информације у зависности од тога како су патерни оријентисани једни у односу на друге. Ми рачунањем хистограма за сваку ћелију заправо енкодирамо просторне информације као што су очи, нос, уста... Просторно кодирање нам такође омогућава да придружимо хистограмима ћелија различите тежине, чиме се повећава моћ разликовања фичера лица. На слици 1 је приказано лице подељено на ћелије, а на слици 2 weighting scheme за сваку ћелију. LBP за беле ћелије (очи) су weighed 4 пута више него остале ћелије (њихови хостиграми су помножени са 4). Светло сиве ћелије су 2 пута теже (уста и уши), тамно сиве 1 пута (унутрашњи образи и чело), док црне ћелије имају тежину 0 (нос и спољашњи образи). Вредности тежина су експериментално утврђене. Тежински 7×7 LBP хистограми се надовезују и формирају коначни фичер вектор. Препознавање лица се врши коришћењем растојања и nearest neighbor класификатора. Улазно лице се обрађује као и тренинг лица (LBPs се издвајају, додају им се тежине, конкатенирају се) и затим се примени k-NN (with k=1) са хи квадрат растојањима како би се нашло најближе лице из тренинг података. Овај алгоритам може да се ажурира приликом додавања новог лица у систем, док неки други алгоритми захтевају да сва лица која ће се идентификовати буду присутна у време тренирања. LBP алгоритам је отпотнији на шумове јер не ради директно са интензитетима пиксела и углавном даје боље реултате него Eigenfaces алгоритам. [28]



Дубоко учење

Specialized neural network architectures and training techniques, including **siamese networks**, **image triplets**, and **triplet loss,** enabled researchers to obtain face recognition accuracy that was once thought impossible. These methods are far more accurate and robust than previous techniques. And despite the stigma of neural networks being data hungry beasts, siamese networks allow us to train these state-of-the-art models with very little data. The secret is a technique called *deep metric learning.* If you have any prior experience with deep learning you know that we typically train a network to: Accept a single input image And output a classification/label for that image. However, deep metric learning is different. Instead, of trying to output a single label (or even the coordinates/bounding box of objects in an image), we are instead outputting a real-valued feature vector. For the dlib facial recognition network, the output feature vector is 128-d (i.e., a list of 128 real-valued numbers) that is used to *quantify the face*. Training the network is done using triplets. triplet consists of 3 unique face images — 2 of the 3 are the same person. The NN generates a 128-d vector for each of the 3 face images. For the 2 face images of the same person, we tweak the neural network weights to make the vector closer via distance metric. Optionally we can compute facial landmarks, enabling us to preprocess and align the face.

## Детекција карактеристичних тачака лица

Свако људско лице је јединствено и одређено је специфичним распоредом и величином његових саставних делова. Карактеристичне тачке лица се користе за означавање истакнутих делова лица попут очију, обрва, носа, уста и образа. У рачунарском виду ове тачке се успешно примењују нпр. за поравнање лица, замену лица, одређивање положаја главе, детекцију трептаја и праћење погледа. Детекција истакнутих тачака лица је подскуп проблема предикције облика. На основу улазне слике и региона од интереса (лице), предиктор облика покушава да лоцира значајне тачке које одређују тај облик. Дакле, детекција се одвија у 2 корака: проналажење лица на слици и издвајање значајних тачака из региона лица. Детектор лица даје информацију о правоугаонику у оквиру кога се на улазној слици налази лице. Постоје различити детектори истакнутих тачака лица, а један од најпознатијих и најкоришћенијих је детектор који је укључен у dlib библиотеку. OpenCV библиотека такође нуди уграђени детектор значајних тачака лица, док MediaPipe библиотека нуди детектор који може детектује 3D мрежу лица. Постоји и модел који је трениран на HELEN скупу података и који детектује 194 тачке. [16]

## Поравнање лица

Постоји више начина поравнања лица. Неке методе захтевају 3D модел лица и онда примењују трансформације над улазном сликом тако да се карактеристичне тачке са слике поклопе са тачкама у 3D моделу. Једноставније методе се ослањају само на карактеристичне тачке (конкретно, на тачке које одређују очи) да би извршиле трансформације попут ротације, транслације и скалирања над сликом. Поравнање лица се ради како би се побољшали алгоритми који ће се након ње примењивати над лицем. У нашем случају то су препознавање лица, праћење погледа и детекција говора. Ова техника се може упоредити са нормализацијом података која се иначе примењује над подацима пре тренирања модела машинског учења. Желимо да лице буде центрирано на слици, заротирано тако да очи буду на истој хоризонталној линији и да буде скалирано тако да величина свих лица буде приближно идентична. У ту сврху коришћена је афина трансформација. Афине трансформације се користе за ротацију, скалирање, трасналцију итд. и можемо све три трансформације објединити у један позив cv2.warpAffine функције којој се као један од параметара прослеђује ротациона матрица, М. На почетку се задаје жељена (x, y) позиција левог ока. Координате су углавном изражене у процентима из опсега of 20-40%. (у овом пројекту узето је 0.43). Ови проценти контролишу колико ће лице бити видљиво након поравнања и варирају од апликације до апликације. Што је мањи проценат лице ће више бити зумирано, а што је већи одзумирано. На основу карактеристичних тачака очију израчунавају се центар једног и другог ока, а након тога и угао који линија која их спаја заклапа са хоризонталом. Овај угао је кључан за поравнање слике. Након тога израчунава се жељена x позиција десног ока (од 1 се одузме жељена x позиција левог ока). Сада можемо израчунати жељену дистанцу између два ока по хоризонтали и помножити је са ширином слике јер је добијена вредност из опсега 0-1. Дељењем жељене дистанце са текућом дистанцом по хоризонтали можемо израчунати фактор скалирања. Потребно је наћи централну тачку која се налази између два ока јер ће се око ње вршити ротација. За рачунање ротационе матрице коришћена је функција cv2.getRotationMatrix2D којој су као параметри прослеђени претходно израчунати центар ротације, угао и фактор скалирања. Добијену матрицу потребно је ажурирати, тачније њену транслациону компоненту како би лице било у оквиру слике и након афине трансформације. Прерачунате су две променљиве: tX у којој је смештена половина ширине слике, и tY у којој се налази висина слике помножена жељеном y координатом левог ока. Транслациона компонента матрице је ажурирана тако што су ове две вредности одузете од одговарајућих координата централне тачке. Након овога се може применити афина трансформација коришћењем функције cv2.warpAffine којој се као параметри прослеђују слика, матрица М, жељена ширина и висина излазне слике и опциони параматар којим се специфицира интерполациони алгоритам, у нашем случају INTER\_CUBIC. [23] Слика са поравнатим лицем се враћа позивајућем програму и даља обрада се врши над овом сликом.

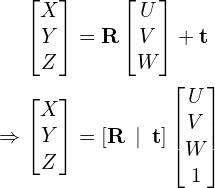
## Детекција окретања главе

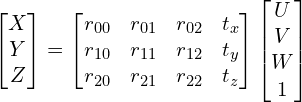
Праћење позе главе је имплементирано коришћењем неколико функција из OpenCV библиотеке и карактеристичних тачака лица. Потребно нам је да знамо да ли је ученикова глава окренута лево, десно, горе или доле и ако јесте да то забележимо у одговарајући бафер, слично као код детектора особа и лица. Детектор позе главе се ресетује уколико у фрејму нема детектованих особа и лица, а карактеристичне тачке које се користе су добијене из детектора карактеристичних тачака лица.

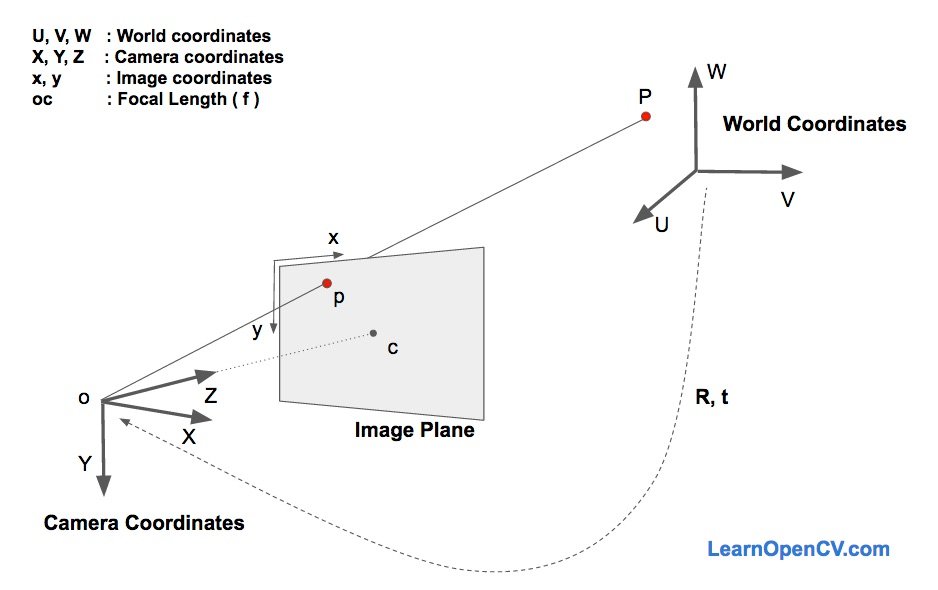
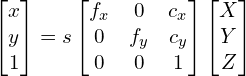
Поза неког објекта се посматра као његова релативна оријентација у односу на камеру. Проблем естимације позе се често назива Perspective-n-Point проблем или PNP. Поза објекта се одређује када имамо калибрисану камеру и знамо локације n 3D тачака које одговарају 2D пројекцијама на слици. Објекат се може кретати у односу на камеру применом транслације или ротације, а то кретање можемо посматрати и као кретање камере око објекта. Транслацијом се назива померање камере са њене тренутне 3D позиције (X, Y, Z) на нову 3D позицију (X’, Y’, Z’) и она се може представити вектором t = (X’ – X, Y’ – Y, Z’ – Z). Ротацијом се назива ротација камере око X, Y и Z осе и може се представити на више начина: коришћењем Ојлерових углова[[9]](#footnote-9), 3x3 ротационе матрице или смером ротације и углом. Оно што ће нам бити потребно за одређивање позе главе је:

1. 2D координате (x,y) неколико тачака на слици: узећемо углове очију, углове усана, врх носа и врх браде.
2. 3D координате истих тачака: У идеалном случају за одређивање ових тачака био би нам потребан 3D модел лица. Међутим, и генерички 3D модел задовољава наше потребе. Користићемо следеће вредности: врх носа (0.0, 0.0, 0.0), брада (0.0, -330.0, -65.0), леви угао левог ока (-225.0, 170.0, -135.0), десни угао десног ока (225.0, 170.0, -135.0), леви угао уста (-150.0, -150.0, -125.0), десни угао уста (150.0, -150.0, -125.0). Ово су тачке у неком произвољном координатном систему (светске координате).
3. Унутрашњи параметри камере: Камера треба да буде калибрисана, тј. потребни су нам: фокална дужина камере, оптички центар слике и параметри радијалне дисторзије. Оптички центар слике се може апроксимирати центром слике, фокална дужина ширином слике и може се претпоставити да радијална дисторзија не постоји.

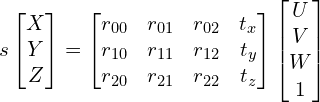
Ми посматрамо 3 координатна система: 3D координате неколико тачака лица представљене су у светским координатама и ако знамо ротацију и транслацију можемо их трансформисати у 3D тачке у координатама камере. Ове тачке могу бити пројектоване на раван слике коришћењем унутрашњих параметара камере чиме се добијају тачке у координатном систему слике. На слици испод, *о* је центар камере, а раван која је приказана је раван слике. Нас интересују једначине које дају пројекцију *p* 3D тачке *P* на раван слике.Нека је (U, V, W) позиција тачке *P* у светским координатама. Ако знамо ротациону матрицу *R* (3x3) и транслациони вектор *t* (3x1), можемо израчунати (X, Y, Z) координате тачке *P* у координатном систему камере на следећи начин:



односно:

Ако имамо довољан број парова (X, Y, Z) и (U, V, W), можемо решити систем линеарних једначина и добити непознате rij и (tx, ty, tz). Нама су познате тачке у 3D моделу (U, V, W), али не знамо (X, Y, Z), већ само 2D тачке (x, y). У одсуству радијалне дисторзије координате (x, y) тачке *p* дате су једначином:

где су *fx* и *fy* фокалне дужине у x и y смеровима, а *(cx, cy)* је оптички центар. Непознати фактор скалирања *s* постоји у једначини услед чињенице да не знамо дубину слике. Ако спојимо било коју тачку P у 3D са центром камере, талка p, у којој зрак пресеца раван слике је пројекција тачке P. Све тачке дуж зрака када се споје са центром камере имаће исту пројекцију на раван слике, тако да коришћењем претходно једначине можемо добити само (X, Y, Z) са скалирањем s.

Сада наша једначина изгледа овако:

и може се решити коришћењем метода који се зове директна линеарна трансформација [(DLT)](https://en.wikipedia.org/wiki/Direct_linear_transformation). Овај метод се може користити кад год имамо проблем где је једначина скоро па линеарна, али је помножена непознатим фактором. Метод није баш поуздан зато што ротациона матрица R има три степена слободе, а матрица која се појављује у овој једначини има 9. DLT решење не минимизује the correct objective function. У идеалном сличају, ми желимо да минимизујемо **reprojection error**. Ако знамо тачну позу (R и t), можемо предвидети 2D локације 3D тачака са слике пројектовањем 3D тачака на 2D слику. Како знамо 2D карактеристичне тачке лица, можемо посматрати растојање између пројектованих 3D тачака и 2D карактеристичних тачака лица. Када процењена поза буде савршена, 3D тачке пројектоване на слику ће се поклопити са 2D карактеристичним тачкама. Када је процењена поза нетачна, можемо израчунати **reprojection error**measure — суму квадрираних даљина између пројектованих 3D тачака и 2D карактеристичних тачака лица. Овај метод се може побољшати итеративном променом вредности R и t тако да се reprojection error смањује: [Levenberg-Marquardt](https://en.wikipedia.org/wiki/Levenberg%E2%80%93Marquardt_algorithm) оптимизација. [24]

## Провера да ли је лице реално

Код система за препознавања лица може бити велики проблем уколико неко злонамерно покуша да приступи систему представљајући се као неко други. Како се препознавање врши на основу података са камере, може се десити да неко подметне слику или видео друге особе. Уколико нема провере да ли је лице на камери реално, систем подметнута, тј. лажирана лица може препознати као валидна. Постоје различити приступи овом проблему, а неки од њих су:

* **Texture analysis,** including computing [**Local Binary Patterns**](https://pyimagesearch.com/2015/12/07/local-binary-patterns-with-python-opencv/) (LBPs) over face regions and using an SVM to classify the faces as real or spoofed.
* Frequency analysis, such as examining the Fourier domain of the face
* Variable focusing analysis, such as examining the variation of pixel values between two consecutive frames.
* Heuristic-based algorithms, including eye movement, lip movement, and blink detection. These set of algorithms attempt to track eye movement and blinks to ensure the user is not holding up a photo of another person (since a photo will not blink or move its lips).
* Optical Flow algorithms, namely examining the differences and properties of optical flow generated from 3D objects and 2D planes.
* 3D face shape, similar to what is used on Apple’s iPhone face recognition system, enabling the face recognition system to distinguish between real faces and printouts/photos/images of another person.
* Combinations of the above, enabling a face recognition system engineer to pick and choose the liveness detections models appropriate for their particular application.

# **Опис практичног дела**

За реализацију прототипа система који врши надгледање полагања онлајн тестова коришћењем камере искоришћен је програмски језик Python (Prvenstveno zbog jednostavne i čiste sintakse a potom i zbog bogate kolekcije podržanih biblioteka.), а пројекат је имплементиран у развојном окружењу PyCharm. Систем не представља комплетну платформу за полагање тестова, већ симулира полагање и надгледање теста укључивањем тајмера и прихватањем фрејмова са камере корисника. У реалној апликацији би се у бази података чували подаци о ученицима и њихове слике (и остали подаци неопходни да функционисање апликације) и пре покретања теста би се захтевало да ученик буде пријављен на систем. У овом пројекту коришћене су Firebase[[10]](#footnote-10) услуге за складиштење основних података о ученицима (Firestore Database) i њихових слика (Storage). Кориснички интерфејс апликације је једноставан: омогућава додавање ученика у систем и позивање функције која покреће тест за одговарајућег ученика. Приликом стартовања функције која симулира покретање теста учитава се слика ученика на основу његовог идентификационог броја (за студенте би то био број индекса) и она се користи за касније упоређивање са сликом са камере. За обраду слика, односно фрејмова са камере, коришћене су библиотеке OpenCV и Dlib. OpenCV библиотека може се инсталирати командом: pip install opencv-python, а Dlib: pip install dlib. Ове библиотеке су отвореног кода (енг. open source), веома су популарне и користе се за машинско учење, а OpenCV (Open Source Computer Vision Library) се првенствено бави рачунарским видом. Као помоћна библиотека за нумеричка израчунавања коришћена је библиотека NumPy.

Након отварања камере ученика, систем контролише да ли се у кадру налази тачно једна особа, да ли је детектовано једно лице, да ли се детектовано лице поклапа са сликом ученика који је улогован, да ли су ученикова глава и поглед усмерени ка екрану, да ли разговара са неким и да ли је његово лице реално, односно проверева се да ли је камери подметнута слика. Сваки од детектора се посматра као одвојени модул, а из главног фајла који је улазна тачка апликације и који садржи функцију за симулацију теста, позивају се функције за детекцију и валидацију које су имплементиране у сваком од детектора. Оваква организација кода омогућава измене у обради фрејмова које су локализоване у оквиру сваког детектора и једноставну замену модела за било који тип детекције. Обрада фрејмова се прекида када прође онолико времена на колико је подешено трајање теста и тада се у видео фајл бележи извештај. Заправо, генеришу се два фајла: један који садржи комплетан снимак полагања, при чему су означене нерегуларне ситуације, и други, који садржи само временске интервале полагања где је било нерегуларности. Видео фајл са нерегуларним ситуацијама се може прегледати како би се потврдиле нерегуларности које је систем генерисао, а може се погледати и комплетан снимак. На слици је приказан архитектурни дијаграм система који представља опис компоненти система и њихове међусобне интеракције.

## Комуникација са базом података

За потребе реализације овог прототипа система било је потребно чувати само податке о ученицима. Подаци који су чувани су: идентификациони број (број индекса за студенте), име, презиме и е-маил адреса ученика. Слике ученика чувају се у Firebase Storage, а линкови до слика заједно са осталим подацима о ученику. Претпоставка је да сваки ученик који је убачен у систем има слику попут слика које се користе за личне документе (особа се види отприлике до рамена, лице јој се јасно види, гледа у камеру и не смеје се). За конекцију са Firebase-ом из Python-а коришћена је библиотека firebase\_admin. У модулу за комуникацију са базом података имплементиране су функције за додавање ученика и учитавање информација о ученику. Firestore Database је document-oriented (оријентисана ка документима) база података, где су подаци организовани у документе, а документи у колекције докумената, тако да је направљена колекција students (ученици) у којој је сваки ученик представљен једним документом. Сваки документ при креирању добија аутоматски генерисан јединствени id, који је искоришћен да се именује фолдер у Firebase Storage где ће бити чувани фајлови везани за тог ученика, као што су његова слика и видео извештаји са тестова које је полагао. Са базом података се интерагује само на почетку полагања приликом учитавања слике ученика, имена ученика како би му се приказала поздравна порука и на крају када треба сачувати генерисани извештај. У реалној апликацији база података би била сложенија и имала би више колекција тако да би и организација модела базе података била другачија, али за потребе овог система модел је поједностављен и чуване су само неопходне информације за функционисање система.

## Детекција објеката на камери

Метрика која се обично користи за упоређивање модела је постигнута Mean Average Precision (MAP). При одабиру модела треба имати на уму конкретан случај коришћења и потребе саме апликације. На пример, YOLO i MobileSSD дају веома добре резултате, али MobileSSD је оптимизован за рад на CPU, a YOLO за рад на GPU.

За детекцију особа у кадру коришћен је модул за дубоко учење (DNN) из OpenCV библиотеке. Овај модул је високо оптимизован за извршавање на CPU, што је један од главних разлога зашто је у коришћен у овом пројекту. Не омогућава тренирање модела, али омогућава коришћење истренираних мрежа из framеwork-a за дубоко учење попут TensorFlow, Torch, Caffe, Darknet и ONNX. Метод који је коришћен комбинује MobileNet архитектуру и Single Shot Detector (SSD) и веома је брз и ефикасан. Коришћена је TensorFlow имплементација. [29] MobileNet SSD је трениран на COCO (Common Objects in Context) скупу података који садржи 80 класа објеката који се могу срести у свакодневном животу (људи, аутомобили, животиње, предмети у кући, храна). Овај dataset садржи 330 000 слика са 1.5 милиона инстанци објеката. [31] Коришћен је тесктуални фајл који садржи лабеле за сваку класу објеката, који је учитан у оквиру класе намењене овом детектору и који је парсиран тако да се од њега направи низ који садржи називе класа које се могу детектовати.

SSD модели су мање захтевни за рачунање у односу на друге моделе за детекцију објеката, па су и генерално бржи. Међутим, због тога су и склонији грешкама. Ове апликација се извршава на CPU, али могуће је извршавање и на GPU, за шта је потребно компајлирати OpenCV DNN модул from the source са GPU. У коду је потребно додати следеће две линије након учитавања модела:

net.setPreferableBackend(cv2.dnn.DNN\_BACKEND\_CUDA)

net.setPreferableTarget(cv2.dnn.DNN\_TARGET\_CUDA)

Прва линија обезбеђује да неуронска мрежа користи CUDA backend ако DNN модул подржава CUDA GPU модел. Друга линија означава да ће се сва израчунавања неуронске мреже извршавати на GPU уместо на CPU. Коришћење GPU доноси већи FPS у односу на CPU. [32]

У овом пројекту у оквиру детектора објеката биле су од значаја класе: person, laptop и cellphone зато што је циљ био детектовати да ли постоји више особа у кадру или се користе недозвољени електронски уређаји попут мобилних телефона и лаптопова. Мрежа је учитана позивом функције cv2.dnn.readNetFromCaffe. Додати параметре Величина улазне слике (фрејма) је промењена на величину 300x300 и прослеђена мрежи. Само детекције које имају поузданост већу од 0,5 су разматране и међу њима су издвајане оне које припадају класи „person“. У оквиру програмске класе која обухвата логику детектора особа постоји неколико бафера. Главна класа у којој се учитавају фрејмови са камере прослеђује овом детектору један по један фрејм и сваки фрејм се одмах обрађује. Како нерегуларна ситуација (у кадру нема ниједне особе или има више од једне особе) неће трајати само један фрејм, већ више узастопних фрејмова, посматрају се „прозори“ фрејмова. У window баферу се чувају сви фрејмови који припадају једном прозору. Тестирањем различитих величина прозора установљено је да је величина прозора од 30 фрејмова најповољнија. На нивоу сваког прозора се броји колико је било нерегуларних фрејмова и ако тај број прелази једну трећину укупног броја фрејмова у прозору, прозор се сматра нерегуларним и „пресипа“ у главни бафер који садржи нерегуларне ситуације из детектора особа на нивоу комплетног теста и који се налази у главној класи апликације. На крају сваког прозора window бафер се ресетује. Постоји још један бафер: people\_cons\_buffer, који чува узастопне нерегуларне фрејмове у једном прозору. Овај бафер има улогу да контролише прелаз између фрејмова. Може се десити да се нерегуларна секвенца налази на прелазу између два прозора, тј. да се један њен део налази у текућем, а други у следећем фрејму, и да због ресетовања window бафера она не буде урачуната у потпуности (један део упада у прозор који у целини нема довољно нерегуларних фрејмова да би се сматрао проблематичним) или не буде урачуната уопште (ниједан од два прозора нема довољан број нерегуларних фрејмова да се сматра пронлематичним). Из тог разлога се на прелазу из једног у други прозор испитује да ли баферу који садржи узастопне нерегуларне фрејмове није празан. Ако није празан, текући прозор се „скраћује“ за број елемената people\_cons\_buffer-а и не отвара се нови прозор док год има сукцесивних нерегуларних фрејмова. Када се наиђе на први валидни фрејм, испитује се да ли у people\_cons\_buffer-у има више од 15 елемената и ако има ти фрејмови се пребацују у главни бафер. Након тога се прелази на следећи прозор. Ово је интерна логика детектора особа и промене су видљиве кроз ажурирање бафера из главне класе који је додељен овом детектору. Резултат обраде сваког појединачног фрејма може утицате на преостале детекторе у систему зато што ако нема само једне особе у фрејму, нема потребе прослеђивати тај фрејм даље, он ће представљати прекид прозора других детектора и главна класа ће позвати одговарајуће функције за ресетовање осталих детектора.

## Детекција лица и карактеристичних тачака лица

За детекцију лица коришћен је OpenCV DNN модул. На почетку било је потребно учитати мрежу и проследити слојеве модедла и тежине.Тежине се могу преузети са линка: https://github.com/opencv/opencv\_3rdparty/raw/dnn\_samples\_face\_detector\_20170830/res10\_300x300\_ssd\_iter\_140000.caffemodel, а слојеви модела (архитектура мреже) са линка: https://github.com/opencv/opencv\_extra/blob/4.x/testdata/dnn/opencv\_face\_detector.prototxt.

Квантована верзија доступна је на линку:

https://github.com/opencv/opencv/blob/4.x/samples/dnn/face\_detector/opencv\_face\_detector.pbtxt

https://github.com/opencv/opencv\_3rdparty/raw/8033c2bc31b3256f0d461c919ecc01c2428ca03b/opencv\_face\_detector\_uint8.pb

Да би се постигла најбоља поузданост моделу треба проследити BGR слике димензија 300x300 над којима је примењено одузимање средње вредности (104, 177, 123) за сваки од B, G i R канала, респективно. Ово је одређено коришћењем функције blobFromImage. Да бисмо добили конкретне предикције на основу дубоке неуронске мреже, потребно је да препроцесирамо податке. Одузимање средишње вредности се користи како би се умањио ефекат у сликама из скупа података (dataset). Пре него што се крене са тренирањем дубоке неуронске мреже израчунају се пресечни интензитети пиксела над целим тренинг скупом за сваки од канала. Резултат је уређена тројка. Пре пропуштања слике кроз мрежу, одузмемо усредњене вредности за сваки од канала улазне слике. Можемо имати и фактор скалирања који се додаје при нормализацији. Функција cv.dnn.blobFromImage(image[, scalefactor[, size[, mean[, swapRB[, crop[, ddepth]]]]]]) ->retval креира 4-D блоб на основу слике. Опционо мења димензију и исеца слику од центра, одузима средње вредности, скалира вредности на основу фактора скалирања, врши замену плавог и црвеног канала. [15]

1. Image – улазна слика

2. Scalefactor- Фактор скалирања који се примњује након одузимања средишње вредности

3. Size – величина излазне слике

4. Mean - scalar with mean values which are subtracted from channels. Values are intended to be in (mean-R, mean-G, mean-B) order if image has BGR ordering and swapRB is true

5. swapRB - OpenCV assumes images are in BGR channel order; however, the `mean` value assumes we are using RGB order. To resolve this discrepancy we can swap the R and B channels in image by setting this value to `True`.

6. crop – flag which indicates wheather image will be cropped after resize or not

7. ddepth – depth of output blob, Choose CV\_32F or CV\_8U

Функција враћа блоб који представља улазну слику након одузимања средње вредности, нормализације и замене канала. [16]

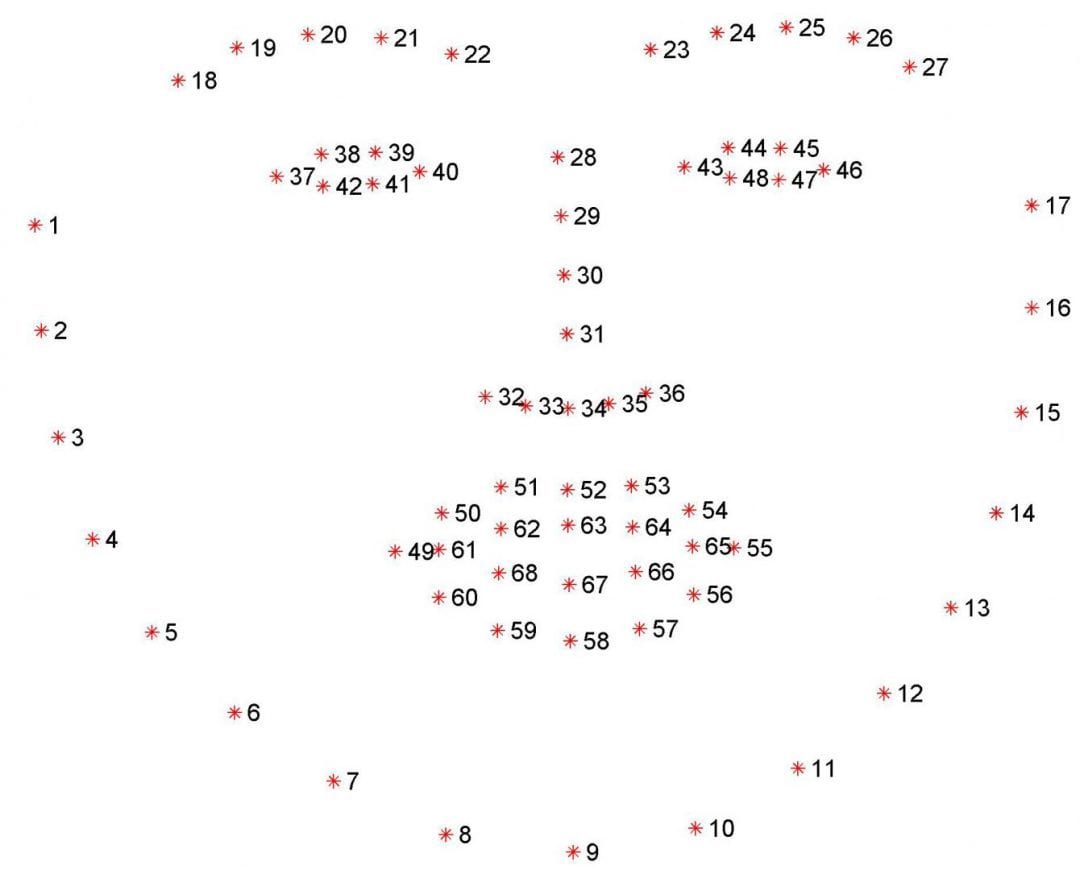
Резултат прослеђивања блоба мрежи је 4-D низ који садржи информацију о поузданости и координатама лица скалираних на опсег [0, 1], те ћемо множењем тих бројева са оригиналном ширином и висином слике добити предикцију за оригиналну слику. Трећа димензија итерира кроз детектована лица. Четврта димензија садржи информацију о оквирном превоугаонику и вероватноћи за свако лице. На пример, detections[0,0,0,2] даје информацију о поузданости, а detections[0,0,0,3:6] оквирни правоугаоник.

Овај детектор има еквивалентне бафере као детектор особа, с тим што је имплементирана и функција за ресетовање која се позива када детектор не добије фрејм. Ова функција проверава да ли је до тог тренутка било најмање 15 узастопних невалидних фрејмова и ако јесте пребацује их у бафер намењен овом детектору из главне класе. Ако то није случај, проверава да ли је обрађено више од две трећине прекинутог прозора. Ако јесте, проверава да ли је више од половине тих фрејмова нерегуларно и ако то јесте случај пребацује прекинути прозор у главни бафер.

За детекцију карактеристичних тачака лица на фрејму коришћена је dlib библиотека. Модел је преузет са линка: <http://dlib.net/files/shape_predictor_68_face_landmarks.dat.bz2>. Овај детектор као повратну вредност има shape објекат који садржи (x, y) координате 68 карактеристичних тачака лица. Коришћењем shape\_to\_np функције овај објекат се конвертује у NumPy низ, што омогућава лакшу обраду у наставку. У оквиру класе овог детектора имплементиране су функције које враћају специфичне тачке лица које се користе у наставку обраде: тачке левог и десног ока, горње и доње усне и 6 карактеристичних тачака за естимацију позе главе.

Detektor oznaka lica iz biblioteke dlib

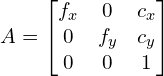
Метода користи тренинг скуп слика на којима су значајне тачке ручно обележене специфицирањем (x, y) координата. На основу тренинг података, низ (ensemble?) регресионих стабала је обучен да установи позиције значајних тачака директно на основу проређеног (sparse) поскупа интензитета пиксела на слици (без издвајања фичера). Тачке се могу означити у реалном времену са веома добрим резултатима. Детектују се координате 68 тачака које се мапирају на структуру лица и оне су приказане на следећој слици:



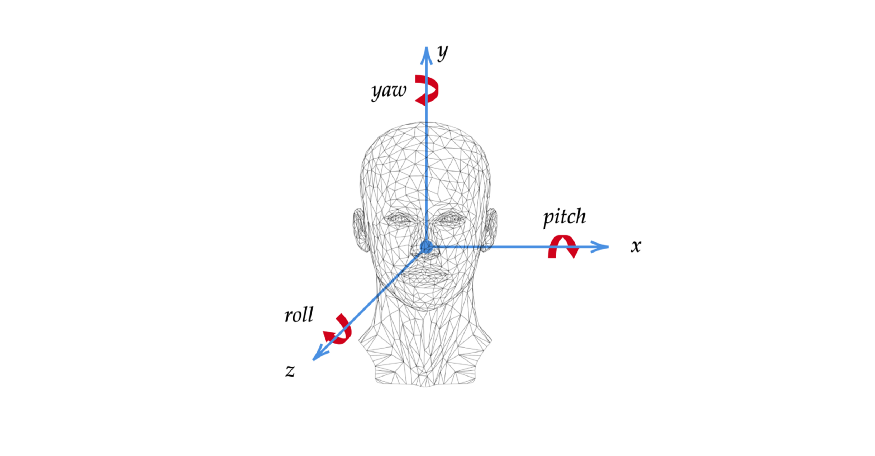
**Figure 2:** Visualizing the 68 facial landmark coordinates from the iBUG 300-W dataset.

Ова анотација је део iBUG 300-W скупа података на којем је dlib детектор трениран. Dlib нуди и детектор који проналази 5 значајних тачака лица и који је знатно бржи од варијанте која детектује 68 тачака. Препорука је да се он користи ако су нам само потребне локације носа и очију. [17]

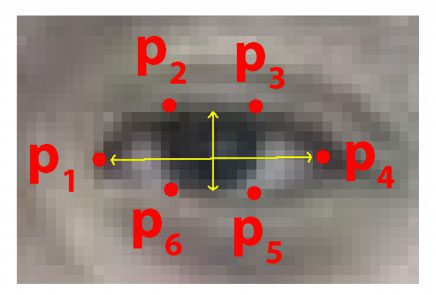
## Детекција окретања главе

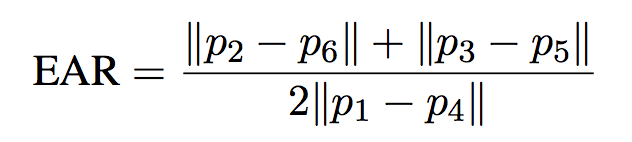
У OpenCV библитеци постоји више функција које су намењене овом проблему, а овде је коришћена solvePnP. Ова функција имплементира неколико алгоритама који могу бити селектовани приликом позива функције, а подразумевани је SOLVEPNP\_ITERATIVE који је DLT решење праћено Levenberg-Marquardt оптимизацијом. Функција као улазне параметре има: низ 3D тачака објекта (у светском координатама), низ одговарајућих тачака на слици (2D тачке лица), унутрашњу (intrinsic) матрицу камере, низ коефицијената дисторзије и метод за решавање PnP проблема. Матрица камере је облика:

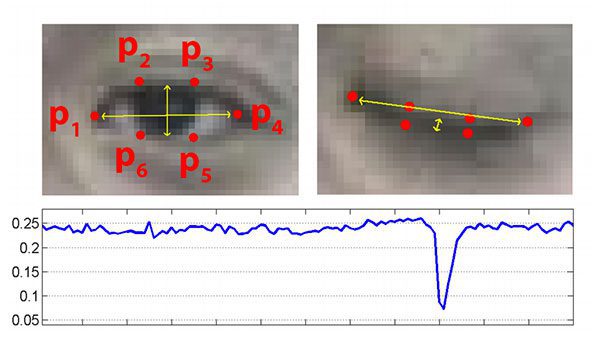
**Функција враћа ротациони и транслациони вектор који трансформишу** 3D тачке објекта у координате камере. [25]

На крају треба издвојити информације за roll, pitch, and yaw, односно Ојлерове углове коришћењем RQDecomp3x3 функције. Ова функција захтева ротациону матрицу, а не ротациони вектор, чију конверзију нам омогућава функција Rodrigues. [27]

## Провера да ли је лице реално

У овом пројекту коришћена је детекција трептања ученика за проверу да ли лице није лажно. Детектовани су и бројани трептаји ока. Како би се детектовао трептај коришћена је метрика eye aspect ratio (EAR). За разлику од традиционалних метода за детекцију трептаја које укључују локализацију ока, thresholding да би се издвојила бела површина у оку и одређивање да ли бела површина нестаје на одређени временски период, eye aspect ratio је елегантније решење које врши једноставну калкулацију базирану на односу растојања између одговарајућих карактеристичних тачака очију. Свако око представљено је помоћу 6 тачака, почевши од левог угла ока и крећући се у смеру казаљке на сату.

Постоји релација између ширине и висине ових координата:

где су p1, …, p6 значајне тачке које одређују око. Ова величина је приближно константна док је око отворено, а нагло опада до нуле када се деси трептај. Коришћењем ове метода избегавамо технике процесирања слике и ослањамо се на растојања између одговарајућих тачака.

На графику се може видети the eye aspect ratio креоз време на видео снимку. Вредност је константна, затим нагло опада до нуле, а онда поново расте, што индицира да се десио трептај.

## Детекција погледа

Ученик би током полагања теста требало да гледа право у екран и да не скреће поглед лево и десно. У модулу за праћење погледа детектују се периоди када постоји дуже гледање у једну или другу страну, а краткотрајно гледање са стране се толерише јер то могу бити случајне детекције које не треба санкционисати. И овај детектор садржи бафер који представља прозор фрејмова и функционише на сличан начин као претходно описани детектори. Како се на крају прозора тестира да ли је број фрејмова у којима ученик гледа са стране већи од трећине укупног прозора, биће детектовано ако је за време трајања прозора ученик често гледао са стране на кратко и то ће бити пријављено као проблематична ситуација. Оба ока су праћена истовремено и мерено је растојање зенице од центалног положаја по хоризонтали, чиме се детектује гледање у лево или десно. Поглед на горе или доле није детектован јер су по вертикали знатно мања растојања и незгодно је прецизно мерити их. Детектору се прослеђује улазна слика и карактеристичне тачке једног и другог ока. Прати се центар зенице и рачуна се однос између ширине ока и растојања између левог угла ока и центра зенице. Овај однос је опсегу од 0 до 1, а док особа гледа право вредност је 0.5. Ако је однос мањи од 0,35 сматра се да особа гледа лево, а ако је већи од 0,65 десно.

## Детекција говора

Причање је реализовано праћењем покрета усана. Коришћене су карактеристичне тачке горње и доње усне. Детектор је иницијализован сликом ученика из базе података на којој су ученику уста затворена. Мерена су растојања одговарајућих тачака горње и доње усне. Ова растојања су упоређивана са истим растојањима на улазној слици (фрејму) и ако је вредност већа од унапред задате вредности (експериментално је одабрана вредност 0.2) уста се сматрају отвореним. Креирани су бафери као код претходних детектора. Један бафер представља прозор фрејмова и његова величина је лимитирана на 30 елемената. У оквиру прозора броје се фрејмови на којима су уста ученика отворена и уколико је при преласку на наредни прозор тај број већи од четвртине обрађених фрејмова пријављује се да је ученик причао и тај прозор се премешта у бафер намењен овом детектору. Како се може десити да се ученик прозева или из неког разлога држи уста отвореним дуже време, те ситуације се толеришу и не сматрају невалидним. Постоји бафер који чува фрејмове у којима су уста узастопно отворена. Уколико на пресласку између прозора овај бафер има елемената, отварање новог прозора се одлаже док су уста отворена и ти фрејмови се сматрају валидним. Експериментално је утврђено да приликом причања неће бити више од 4 узастопних фрејмова када су уста отворена, тако да се низови дужи од 4 сматрају валидним „зевањем“ и уколико се налазе у оквиру прозора (не на граници) неће се урачунавати у бројање. При ресетовању услед прекида прозора проузрокованог неким од детектора изнад, испитује се да ли је обрађено више од две трећине фрејмова и ако јесте, проверава се да ли је бројач фрејмова са отвореним устима већи од половине обрађених фрејмова, па ако јесте, прозор се сматра нерегуларним и пребацује у главни бафер.

## Препознавање лица

За препознавање лица као прозор коришћен је бафер величине 50 фрејмова и у оквиру сваког прозора по једном је вршено препознавање с обзиром на то да је ово захтевнија операција и проузроковала би знатно слабије перформансе када би била извршавана над сваким фрејмом. На почетку је постављена ученикова слика које је учитана из базе и израчунати су encodings. Лице са камере упоређивано је са овом сликом, тачније упоређивани су encodings и уколико је разлика већа од 0.5, лице није препознато, тај прозор се сматра невалидним и пребацује се у бафер намењен овом детектору. При ресетовању овог детектора, уколико се до момента ресета детектовао невалидни фрејм, прозор се пребацује у одговарајући бафер и пријављује се нерегуларност.

## Генерисање извештаја

Као што је речено, сваки од појединачних детектора има у главном програму бафер који садржи прозоре фрејмова који су означени као нерегуларни. На крају теста, ови бафери се спајају у јединствени бафер који садржи све нерегуларне ситуације. С обзиром на то да може да се деси да су се нерегуларни прозори из различитих детектора преклапали, односно да се исти фрејмови налазе у више различитих прозора, из коначног бафера су елиминисани дупликати. Коначни бафер се сортира тако да фрејмови буду временски уређени и његов садржај се уписује у видео фајл. Генерише се још један видео фајл који садржи комплетан тест, односно све фрејмове, при чему су нерегуларни фрејмови обележени као и у претходном видео фајлу. За креирање видео фајла на основу појединачних фрејмова коришћена је функција cv2.VideoWriter. Направљена је посебна класа која представља један фрејм и која има три атрибута: слику, редни број фрејма и поруку која ће бити исписана на фрејму. Сваки фрејм има испис вредности тајмера у тренутку обраде тог фрејма како би се знало када се тачно десила нека нерегуларна ситуација. Атрибут који чува информацију о редном броју се користи приликом сортирања елемената коначног бафера, а атрибут порука садржи информацију о томе која нерегуларност је детектована. Пошто у једном фрејму може истовремено бити више нерегуларности, сваки од детектора ажурира овај атрибут дописивањем специфичног текста.

# **Закључак**

Софтвер за полагање тестова је све напреднији и све више коришћен како у образовним установама, тако и за потребе тестирања у различитим компанијама и организацијама. Вештачка интелигенција и машинско учење у великој мери доприносе реализацији тестирања на даљину. Софтвер за аутоматско надгледање полагања онлајн тестова у доброј мери омогућава детекцију неправилности које можда не би биле детектоване при стандардном тестирању у учионици. Појединци се могу надгледати детаљније и надгледање је константно за разлику од ситуације у учионици када наставник у неком тренутку скрене поглед и створи прилику ученику за преписивање, међутим, ученици/студенти ће вероватно увек моћи да пронађу неки начин да варају. Предности оваквог начина тестирања су и смањене потешкоће око организације јер нема потребе за проналажењем одговараће просторије и особа које би дежурале. Штеди се на времену људи и брже се генеришу и анализирају резултати теста. Са друге стране, овакав софтвер може бити скуп.

Онлајн преписивање може бити спречено на пример увођењем временског ограничења за одговор на питање како ученик не би имао времена да потражи помоћ. Може се и онемогућити враћање на претходно одговорено питање. Како ученици не би делили међусобно питања и решења након што су завршили тест, може се захтевати да сви полажу тест истовремено. Међутим, ове мере некада нису довољне и студенти ипак пронађу начин да преписују. Једна од најчешћих ситуација је да неко други ради тест уместо особе која треба да полаже, што се може превазићи методама за препознавање лица тако што ће се повремено бележити слика са камере и проверавати.

Главне предности софтвера за полагање онлајн тестова су доступност, мање ангажованог особља, константно надгледање током израде и могућност напредније анализе резултата.

Увек је било преписивања и могу се пронаћи начини да се превари дежурни наставник или софтвер који се користи за надгледање онлајн полагања. Међутим, ако узмемо у обзир напредак технологије и развој машинског учења, све више покушаја преваре може се детектовати и санкционисати. И даље се не можемо у потпуности ослонити на аутоматско надгледање и потребно је укључити људски фактор, али чињеница је да су системи све напреднији и да имају све већу примену како у образовању, тако и у осталим гранама друштва. У овом раду дат је преглед техника које се могу приметити у обради података са камере приликом тестирања и реализован је прототип система који примељује неке од њих.

# **Литература**

1. <https://www.testreach.com/blog-post/proctoring-online-exams.html?fbclid=IwAR0r8C4Cg4BLoVv-VJdbadDFxZk8qAiA_HHEkq7mLrEGkpa-U_7iah8jG10>
2. <https://www.techjockey.com/blog/proctoring-software-for-online-exam>
3. https://www.masterteachingonline.com/can-an-online-exam-detect-cheating/?fbclid=IwAR0CTcD3-2ixdQQaBXkrii\_eVHSyffVz2AZ4P08q0tc4cSRFCm0-VVLCOPs
4. <https://honorlock.com/>
5. <https://proctoredu.com/industries/corporate/pre-employment>
6. <https://pages.mettl.com/faq-online-examination#Cheating-Prevention>
7. <https://www.speedexam.net/>
8. <https://examus.com/>
9. https://en.wikipedia.org/wiki/Computer\_vision
10. <https://en.wikipedia.org/wiki/Object_detection>
11. https://viso.ai/edge-ai/ai-hardware-accelerators-overview/
12. <https://viso.ai/deep-learning/object-detection/>
13. https://www.sciencedirect.com/topics/engineering/haar-function
14. https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf
15. <http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>
16. <https://learnopencv.com/histogram-of-oriented-gradients/>
17. <https://arxiv.org/pdf/1311.2524.pdf>
18. <https://arxiv.org/abs/1504.08083?so>
19. <https://arxiv.org/abs/1506.01497>
20. <https://arxiv.org/abs/1703.06870>
21. <https://arxiv.org/abs/1506.02640>
22. <https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>
23. <https://arxiv.org/abs/1512.02325>
24. <https://www.researchgate.net/publication/3193340_Detecting_Faces_in_Images_A_Survey>
25. <https://towardsdatascience.com/face-detection-models-which-to-use-and-why-d263e82c302c>
26. <https://pyimagesearch.com/2021/05/01/what-is-face-recognition/>
27. <https://pyimagesearch.com/2021/05/10/opencv-eigenfaces-for-face-recognition/>
28. <https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/>
29. https://pyimagesearch.com/2021/05/03/face-recognition-with-local-binary-patterns-lbps-and-opencv/

1. LMS (Learning Management System) платформа је софтвер који се користи за креирање, администрацију и праћење тренинга и обука. [↑](#footnote-ref-1)
2. AI акцелератори су врста специјализованих хардверских акцелератора који се користе да убрзају аплиакције које користе вештачку интелигенцију и машинско учење. Неки од примера су VPU (енг. Vision Processing Unit), GPU (енг. Graphics Processing Unit) и TPU (енг. Tensor Processing Unit). [11] [↑](#footnote-ref-2)
3. Bounding-box регресија је популарна техника за is a popular technique to пречишћавање или предикцију граничних правоугаоника приликом детекције објеката. [↑](#footnote-ref-3)
4. The Haar functions are an orthogonal family of switched rectangular waveforms where the amplitudes can differ from one function to another. [13] [↑](#footnote-ref-4)
5. IoU (пресек над унијом) је метрика за евалуацију тачности детектора објеката. Представња количник површине преклапања предиктованог правоугаоника и ручно означеног правоугаоника из тест скупа података и површине уније та два правоугаоника. Добром предикцијом се сматра IoU већи од 0,5. [22] [↑](#footnote-ref-5)
6. Пирамида слика омогућава проналажење објеката на слици посматрајући њене скалиране верзије. На дну пирамиде је слика у оригиналној величини, а у наредним слојевима рисајзоване слике.Слика се смањује док не достигне одређену минималну величину. [↑](#footnote-ref-6)
7. Клизећи прозор је правоугаони регион фиксне ширине и висине који клизи преко слике са лева на десно и одозго на доле. Над сваким од прозора се обично примењује класификатор за одређивање да ли је објекат од интереса садржан у прозору. [↑](#footnote-ref-7)
8. У OpenCV-у, оквирни правоугаоник је представњен уређеном четворком (x, y, w, h), док је у dlib-у представљен објектом правоугаоника са left, top, right, and bottom атрибутима. [↑](#footnote-ref-8)
9. Ојлерове углове је први увео Леонард Ојлер (нем. Leonhard Euler), како би описао ротацију крутог тела око непокретне тачке. Положај тела при овој ротацији може се једнозначно одредити помоћу три угла. Свакој ротацији крутог тела одговара ротација тродимензионог Еуклидског простора, а свака оваква ротација може се представити као композиција три ротације око координатних оса (елементарне ротације). [25] [↑](#footnote-ref-9)
10. Firebase је Гуглова (енг. Google) платформа за развој апликација која нуди различите сервисе попут база података, аутентификације и аналитике. [↑](#footnote-ref-10)