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.

Table of Contents

[**1.** **Увод** 6](#_Toc114926396)

[**2.** **Софтвери за праћење регуларности полагања тестова** 7](#_Toc114926397)

[2.1. Аутоматско надгледање онлајн тестирања 8](#_Toc114926398)

[2.2. Примери софтвера за аутоматско надгледање онлајн тестирања 8](#_Toc114926399)

[**3.** **Технике рачунарског вида за праћење регуларности полагања тестова** 11](#_Toc114926400)

[3.1. Детекција објеката 11](#_Toc114926401)

[3.1.1. Viola-Jones детектор 12](#_Toc114926404)

[3.1.2. HOG детектор 12](#_Toc114926406)

[3.1.3. R-CNN – Region-based Convolutional Neural Networks 12](#_Toc114926408)

[3.1.4. YOLO детектор 13](#_Toc114926410)

[3.1.5. SSD детектор 14](#_Toc114926412)

[**4.** **Опис практичног дела** 15](#_Toc114926414)

[**5.** **Закључак** 16](#_Toc114926415)

[**Литература** 17](#_Toc114926416)

# **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, 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 интердисциплинарно 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 и препознавање лица, издвајање карактеристичних особина лица (енг. facial landmarks), праћење окретања главе и 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) како би се побољшали резултати препознавања лица и праћења покрета очију и усана.

## Детекција објеката

Детекција објеката је технологија 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 (енг. features) које одређују припадност објекта тој класи. Детекција објеката има широку примену у свакодневном животу, на пример за детекцију пешака и аутомобила, мерење растојања између људи у некој просторији, препознавање регистарских таблица, у спорту и медицини. Користи се и за праћење објеката (нпр. праћење особа на видео снимцима или лопте током фудбалског меча) и не казује нам само који објекат је на слици, већ и где се он налази. 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 особине 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 (енг. CNN, convolutional neural networks). Примери ових метода су: Region Proposals (R-CNN, Fast R-CNN, Faster R-CNN, cascade 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, графичких процесора (енг. GPU, graphical processing unit) и AI акцелератора попут TPU (енг. tensor processing unit). Ово омогућава детекцију објеката у скоро реалном времену. Дубоке конволуционе неуронске мреже у комбинацији са убрзањем које доноси GPU значајно су унапредиле процес детекције објеката. Ако поредимо технике које су се раније користиле и модерне технике засноване на дубоком учењу приметићемо да старије методе не захтевају податке за тренирање и анотиране слике објеката које би људи ручно означавали, али, са друге стране, могу бити ограничене услед постојања комплексне позадине, делимично прекривених објеката, лошег осветљења и шума. Технике дубоког учења су значајно отпорније на прекривеност објеката, комплексне сцене и промене у осветљењу, док им је недостатак што захтевају велику количину података за тренирање и процес означавања слика може бити временски захтеван (мада постоје и велики референтни dataset-ови као што су MS COCO, Caltech, KITTI, PASCAL VOC, V5 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 особинама из детектованих региона и bounding-box регресијом. Овакви детектори имају велику поузданост, али су спорији. Најпознатији детектори овог типа су: RCNN (2014), Fast RCNN and Faster RCNN (2015), Mask R-CNN (2017), granulated RCNN (G-RCNN) (2021). Детектори који раде у једном кораку предвиђају bounding boxes 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 (2015) који има више верзија, SSD (2016), i RetinaNet (2017). За поређење алгоритама за детекцију објеката најпознатији репер је Microsoft COCO dataset. Метрика која 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. [11]

### 3.1.1. Viola-Jones детектор

-2001.

### 3.1.2. HOG детектор

-2006.

### 3.1.3. R-CNN – Region-based Convolutional Neural Networks

Конволуционе мреже за детекцију објеката базиране на регионима су прве које примењују дубоко учење за детекцију објеката. R-CNN модел из 2014. године прво селектује регионе са слике и онда означава њихове класе и обухватајуће правоугаонике. Користи се конволуциона неуронска мрежа да издвоји фичере из сваке од одабраних области. Слика се дели на око 2000 региона и конволуциона мрежа се примењује за сваки од њих. Након класификације, пост-процесирање се примењује да се пречисте оквирни правоугаоници и елиминишу дупликати. Ови кораци су спори и тешки за оптимизацију јер сваку индивидуалну компоненту треба одвојено тренирати. Из тог разлога је 2015. године развијен Fast R-CNN. Овај модел покреће неуронску мрежу једном за целу слику. CNN на основу улазне слике генерише мапу фичера из које се идентификују региони који се коришћењем ROI (Region of Interest) слоја преобликују на фиксну величину и прослеђују потпуно повезаном слоју за класификацију. Претходна два алгоритма користима су селективну претрагу за проналажење региона која је веома спора. Уместо овог алгоритма Faster R-CNN користи дубоку конволуциону мрежу која предлаже регионе, а други модул је Fast R-CNN детектор који користи преложене регионе. Комплетан систем је јединствена неуронска мрежа за детекцију објеката. Mask R-CNN из 2017. године је унапређење Fast 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. [14]

3.1.4. 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“). 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[[2]](#footnote-2) (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. 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. [12]

3.1.5. SSD детектор

Овај метод из 2016. године објекте на сликама коришћењем дубоке конволуционе неуронске мреже тако што дискретизује излазне оквирне правоугаонике у скуп подразумеваних правоугаоника са различитим аспектима и скалирањем по feature map location. Генерише вероватноће за присуство сваке од класа објеката у сваком од подразумеваних правоугаоника и прилагођава правоугоник да боње одговара облику објекта. Мрежа комбинује предикције из различитих feature maps са различитим резолуцијама како би се обухватили објекти различитих величина. Детектор је једноставан за тренирање и интеграцију са системима који захтевају компоненту за детекцију објеката. За улаз 300×300 достиже 72.1% mAP on VOC2007 и 58 FPS на Nvidia Titan X а за 500×500 улаз, достиже 75.1% mAP, чиме превазилази Faster R-CNN модел. The fundamental improvement in speed comes from eliminating bounding box proposals and the subsequent pixel or feature resampling stage Our improvements include using a small convolutional filter to predict object categories and offsets in bounding box locations, using separate predictors (filters) for different aspect ratio detections, and applying these filters to multiple feature maps from the later stages of a network in order to perform detection at multiple scales. With these modifications—especially using multiple layers for prediction at different scales—we can achieve high-accuracy using relatively low resolution input, further increasing detection speed We summarize our contributions as follows: – We introduce SSD, a single-shot detector for multiple categories that is faster than the previous state-of-the-art for single shot detectors (YOLO), and significantly more accurate, in fact as accurate as slower techniques that perform explicit region proposals and pooling (including Faster R-CNN). – 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. – Experiments include timing and accuracy analysis on models with varying input size evaluated on PASCAL VOC, COCO, and ILSVRC and are compared to a range of recent state-of-the-art approaches. Fig. 1: SSD framework. (a) SSD only needs an input image and ground truth boxes for each object during training. In a convolutional fashion, we evaluate a small set (e.g. 4) of default boxes of different aspect ratios at each location in several feature maps with different scales (e.g. 8 × 8 and 4 × 4 in (b) and (c)). For each default box, we predict both the shape offsets and the confidences for all object categories ((c1, c2, · · · , cp)). At training time, we first match these default boxes to the 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). [15]

# Детекција лица

Detekcija lica je računarska tehnika koja se koristi za identifikaciju ljudskog lica na digitalnim slikama. Može se posmatrati kao specifičan slučaj detekcije, odnosno, klasifikacije objekata. Kod klasifikacije objekata zadatak je pronaći i locirati sve objekte na slici koji pripadaju određenoj klasi. Npr. pronaći pešake, automobile, znakove pored puta... Detekcija lica se odnosi na ispitivanje da li se na slici ili video snimku nalazi ljudsko lice, gde je locirano i koje je veličine, nezavisno od njegovog položaja i osvetljenja. [1]

Detekcija lica je prvi korak bilo koje obrade lica, kao što su nalaženje karakterističnih tačaka lica, prepoznavanje lica, detekcija pokreta lica, ... Ima široku primenu u različitim oblastima poput biometrije, bezbednosti i zabave. Pronalaženje lica je izazov za računarski vid jer je neophodno uzeti u obzir orijentaciju lica, izraz lica, osvetljenje, prekrivenost lica (npr. naočare, kosa, brada, kapa), rezoluciju slike, kompleksnost pozadine (prisustvo velikog broja objekata), boju kože itd. [2]

Metode i algoritmi za detekciju lica

Metode

[3] [4] Yang, Kriegman, and Ahuja predstavili su u svom radu klasifikaciju metoda za detekciju lica u 4 kategorije. Algoritmi za detekciju lica mogu pripadati dvema ili većem broju kategorija.

1. Feature-based metode lociraju lice izdvajanjem strukturnih karakteristika lica poput nosa, usta i očiju. Najpre se trenira klasifikator koji se nakon toga koristi za razlikovanje regiona na kojima je lice i na kojima nije lice.

-Haar Feature Selection relies on similar properties of human faces to form matches from facial features: location and size of the eye, mouth, bridge of the nose, and the oriented gradients of pixel intensities. There are 38 layers of cascaded classifiers to obtain the total number of 6061 features from each frontal face. You can find some pre-trained classifiers here. Histogram of Oriented Gradients (HOG) is a feature extractor for object detection. The features extracted are the distribution (histograms) of directions of gradients (oriented gradients) of the image.

-Histogram of Oriented Gradients (HOG) is a feature extractor za detekciju objekata. Izdvojeni features su distribution (histograms) of directions of gradients (oriented gradients) of the image. Gradients are typically large round edges and corners and allow us to detect those regions. Instead of considering the pixel intensities, they count the occurrences of gradient vectors to represent the light direction to localize image segments.

1. Knowledge-based metoda zavisi od skupa pravila zasnovanih na ljudskom znanju. Na primer, lice obično sadrži dva oka, nos i usta koji se nalaze na određenom rastojanju i poziciji jedni u odnosu na druge. Problem kod ovog modela je kreirati adekvatan skup pravila. Ako su pravila suviše opšta, rezultovaće dobijanje velikog broja lažno pozitivnih lica. Slično, ako su pravila suviše detaljna, rezultovaće dobijanje velikog broja lažno negativnih lica. Nisu primenljivi za sve boje kože i zavise od osvetljenja koje može znatno uticati na nijansu kože osobe na slici.
2. Template matching metoda koristi predefinisane ili parametrizovane šablone lica za lociranje i detekciju lica, računajući korelaciju između šablona i ulazne slike. Npr. šablon može pokazati da se lice podeljeno na regione kao što su nos, usta, oči i kontura lica. Takođe, može biti sačinjen samo od ivica i onda se može koristiti metod za detekciju ivica-implementacija ovog pristupa je jednostavna, ali nedovoljno dobra za detekciju lica. Ovi metodi ne podržavaju varijacije u pozi, skaliranju i obliku.
3. Appearance-based metoda zavisi od skupa trening slika koje se koriste za nalaženje modela lica. Zasniva se na mašinskom učenju i statističkim metodama kako bi se odredile relevantne karakteristike lica. Ove metode su pokazale znatno bolje performanse u poređenju sa ostalim.

-Viola-Jones detektor lica imao je najviše uticaja na istraživanje u ovoj oblasti. Široko je korišćen u aplikacijama i omogućava detekciju lica u realnom vremenu sa visokom pouzdanošću. Zasniva se na treniranju modela da razazna šta je lice, a šta ne. Jednom istreniran, model se čuva u fajl tako da karakteristike sa novih slika mogu biti upoređene sa prethodno sačuvanim karakteristikama. Ako slika prođe kroz sve faze upoređivanja feature-a, lice je detektovano. Ovaj metod neće raditi ako je lice prekriveno nečim, ili nije adekvatno orijentisano.

This method unites several algorithms:

* 1. Eigenface-based algoritam koji efikasno predstavlja lica korišćenjem Principal Component Analysis (PCA). PCA se primenjuje nad skupom slika kako bi se smanjila dimanzija dataset-a, najbolje opisujući varijansu podataka. Lice može biti modelovano kao linearna kombinacija eigenfaces (set of eigenvectors). Prepoznavanje lica je zasnovano na upoređivanju koeficijenata linearne reprezentacije.
  2. Distribution-based algorithms like PCA and Fisher’s Discriminant define the subspace representing facial patterns. They usually have a trained classifier that identifies instances of the target pattern class from the background image patterns.
  3. Neural networks, such as GANs, are among the most recent and most powerful methods for detection problems, including face detection, emotion detection, and face recognition.
  4. Support Vector Machines are linear classifiers that maximise the margin between the decision hyperplane and the examples in the training set. Osuna et al. first applied this classifier to face detection
  5. Sparse Network of Winnows defines two linear units or target nodes: one for face patterns and the other for non-face patterns. It is less time consuming and efficient.
  6. Naive Bayes Classifiers compute the probability of a face to appear in the picture based on the frequency of occurrence of a series of the pattern over the training images. The classifier captured the joint statistics of local appearance and position of the faces
  7. Hidden Markov Model is a standard method for detection tasks. Its states would be the facial features, usually described as strips of pixels.
  8. Inductive learning uses such algorithms as Quinlan’s C4.5 or Mitchell’s FIND-S to detect faces starting with the most specific hypothesis and generalizing.
  9. Video Processing: Motion-based face detection. In video images, you can use movement as a guide. One specific face movement is blinking, so if the software can determine a regular blinking pattern, it determines the face. Various other motions indicate that the image may contain a face, such as flared nostrils, raised eyebrows, wrinkled foreheads, and opened mouths
  10. Information Theoretical Approach:- Markov Random Fields (MRF) can use for face pattern and correlated features. The Markov process maximises the discrimination between classes using Kullback-Leibler divergence. Therefore this method can be used in Face Detection

**Odabir detektora lica [5] [6]**

U ovom poglavlju biće dat pregled nekoliko najpoznatijih detektora koji se mogu koristiti za detekciju lica, kao što su Haar cascades i DNN detektor iz OpenCV biblioteke, HOG + Linear SVM i CNN iz dlib biblioteke i MTCNN (Multi-task Cascaded Convolutional Neural Network).

1. OpenCV and Haar cascades

Ovaj model je prvi put predstavljen 2001. godine od strane autora Paul Viola and Micheal Jones. Veoma je brz i izdvaja mnogo features sa slike, ali i zahteva veliki broj pozitivnih i negativnih slika za treniranje. Najbolje features su nakon toga selektovani pomoću Adaboost. Ovim se originalnih 160000+ features redukuje na 6000 features. Primena svih ovih features u klizećem prozoru bi zahtevala mnogo vremena, te je preložena kaskada klasifikatora, gde su features grupisani. Ako prozor otpadne u prvoj fazi, preostali features u kaskadi se ne procesiraju. Ako prođe, onda se testira sledeći feature i postupak se ponavlja. Ako prozor prođe sve features onda se klasifikuje kao region koji sadrži lice. [7]

Ovo je originalni detektor lica koji je isporučen uz OpenCV biblioteku, istovremeno i najpoznatiji detektor. Veoma je brz i pogodan za izvršavanje u realnom vremenu, nema velike hardverske zahteve i ima mali model (oko 400KB). Sa druge strane, podležan je lažno pozitivnim detekcijama, zahteva ručno podešavanje parametara i mnogo manje je pouzdan nego ostali navedeni detektori. Treba ga koristiti kada je brzina izvršavanja prioritet i spremni smo da žrtvujemo pouzdanost.

1. Dlib’s HOG + Linear SVM implementation

Dlib je biblioteka toolkit koja sadrži algoritme mašinskog učenja za rešavanje problema iz realnog sveta. Iako je napisana u jeziku C++, moguće je izvršavati i u jeziku Python. Ovaj detektor radi tako što features izdvojene preko Histogram of Oriented Gradients (HOG) prosleđuje SVM. In the HOG feature descriptor, the distribution of the directions of gradients is used as features. Prvi put predstavljen od strane Dalal and Triggs u njihovom radu iz 2005, Histograms of Oriented Gradients for Human Detection [8]. Slično kao Haar cascades, ovaj algoritam se zasniva na piramidama slika i klizećem prozoru (napomena). Pouzdaniji je od Haar cascades, ima stabilniju detekciju i dobro je dokumentovan. Nedostaci su što radi samo sa frontalnim licima jer HOG deskriptor ne toleriše promene u rotaciji ili uglu gledanja, zahteva instaliranje dlib biblioteke, nije toliko pouzdan kao detektori zasnovani na dubokom učenju i skup je u vidu zahteva za računanjem jer radi pyramid construction, sliding windows, and computing HOG features at every stop of the window.

Napomena: Piramida slika je multi-scale reprezentacija slike. Omogućava nam da pronađemo objekte na slici posmatrajući njene skalirane verzije. Na dnu piramide je slika u originalnoj veličini, a u narednim slojevima resized slike koje su opciono smoothed (obično korišćenjem Gaussian blurring). Slika se smanjuje dok ne dostigne određenu minimalnu veličinu. [9]

Klizeći prozor je pravougaoni region fiksne širine i visine koji klizi preko slike sa leva na desno i odozgo na dole. Za svaki od prozora obično se primenjuje klasifikator za određivanje da li je objekat od interesa sadržan u prozoru. [10]

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

Davis King, kreator dlib-a, trenirao je CNN face detector na osnovu svog rada [max-margin object detection](https://arxiv.org/abs/1502.00046) [11]. Metoda je veoma pouzdana, zahvaljujući samom dizajnu algoritma i kvalitetnom trening skupu. Model je mali (manji od 1MB), veoma dobro implementiran i dokumentovan. Sa druge strane, zahteva da dlib bude instalirana, zahteva konvertovanje prozora za oukviravanje lica ako se koristi OpenCV (napomena), nemoguće ga je koristiti u realnom vremenu bez GPU ubrzanja. Preporuka je da se koristi kada ne treba brinuti o performansama u realnom vremenu.

Napomena: U OpenCV-u, uokvirujući pravougaonik je predstavljen uređenom četvorkom ( početna x koordinata, početna y koordinata, širina, visina), dok je u dlib-u predstavljen objektom pravougaonika sa left, top, right, and bottom atributima.

1. MTCNN [13]

Predstavljen je 2016. godine u paper, “Joint Face Detection and Alignment Using Multi-task Cascaded Convolutional Networks.” Pored detekcije lica, detektuje i 5 karakterističnih tačaka lica. Koristi kaskadnu strukturu sa 3 nivoa CNN-a. Prvo, koristi potpunu konvolucionu mrežu da prikupi prozore-kandidate i njihove regresione vektore, koji su preklopljeni korišćenjem on-maximum suppression (NMS). Dalje, ovi kandidati se prisleđuju drugoj CNN koja eliminiše veliki broj lažno pozitivnih i kalibriše obuhvatajuće pravougaonike. U poslednjoj fazi obavlja se detekcija karakterističnih tačaka lica.

1. Detektor lica baziran na dubokom učenju iz OpenCV biblioteke

U današnje vreme duboko učenje je jedna od najpopularnijih i najbrže rastućih oblasti u računarskom vidu. Od verzije 3.1 OpenCV biblioteke postoji DNN modul, a verziji 3.3 je prebačen u glavni repozitorijum i znatno je poboljšan. Ovaj modul podržava veliki broj framework-a za duboko učenje, uključujući Caffe, TensorFlow i Torch/PyTorch. [14]

U pitanju je Caffe model zasnovan na Single Shot Detector (SSD) i ResNet mreži, što ga čini i brzim i pouzdanim. Može se izvršavati u realnom vremenu na modernim laptopovima i desktop računarima, model nije preveliki (oko 10MB) i deo je OpenCV biblioteke. Pouzdaniji je od Haar cascades i HOG + Linear SVM, ali manje pouzdan od dlib’s CNN MMOD detektora. Najbolji izbor je za većinu primena, ne zahteva dodatne biblioteke i uključen je u OpenCV. Jedan od najvećih nedostataka je što je manje pouzdan za detekciju lica sa tamnijom bojom kože, ali to se može rešiti treniranjem modela skupom slika koji sadrži slike ljudi različitih rasa. Postoji i kvantovana Tensorflow verzija. More specifically, the model used (opencv\_face\_detector\_uint8.pb) has been quantized (with the TensorFlow library) on 8-bit unsigned int to reduce the size of the training model (2.7 mo vs 10.7 mo for res10\_300x300\_ssd\_iter\_140000.caffemodel). U poređenju sa prethodno navedenim metodama, DNN je dao najbolje rezultate prilikom testiranja real time videa (članak). Ovaj model radi dobro kada je lice delimično prekriveno, kada postoje brzi pokreti glave, može detektovati bočna lica, i ima najbrži fps.

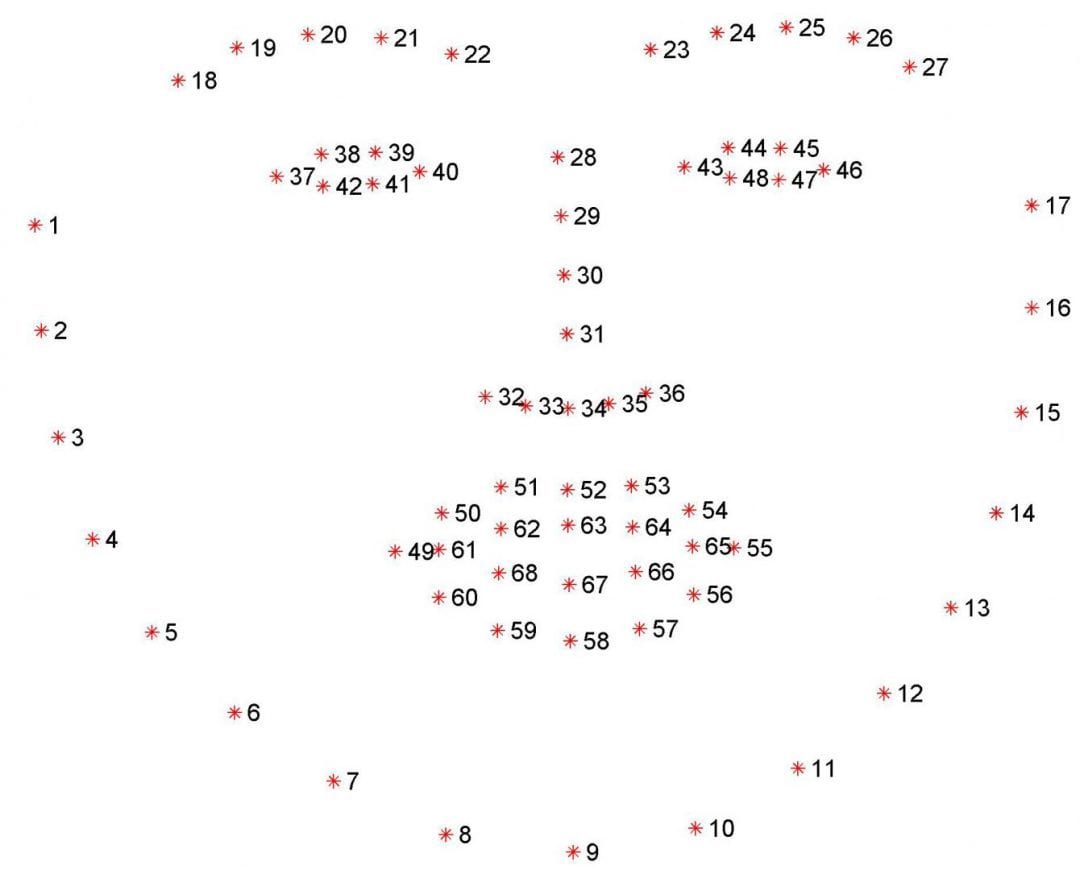
DETEKCIJA KARAKTERISTIČNIH TAČAKA LICA [26]

http://dlib.net/files/shape\_predictor\_68\_face\_landmarks.dat.bz2"

Detekcija karakterističnih oznaka lica je podskup problema predikcije oblika. Na osnovu ulazne slike (na kojoj je region od interesa), prediktor oblika locira ključne tačke za taj oblik. Oznake lica se koriste za lokalizaciju i istaknutih delova lica, poput očiju, obrva, nosa, usta i vilice. Uspešno se primenjuju za poravnanje lica, određivanje poze glave, detekciju treptaja itd. Mi ćemo za detekciju karakterističnih oznaka lica koristiti dlib, OpenCV, and Python. Detekcija se obavlja u 2 koraka: Pronalaženje lica na slici, a nakon toga pronalaženje karakteristika lica u regionu od interesa. Prvi korak se može obaviti na više načina. Kao što je opisano u prethodnom poglavlju, mi ćemo za detekciju lica koristiti OpenCV DNN model. Detektor lica nam vraća region od interesa predstavljen obuhvatajućim pravougaonikom.

Detektor oznaka lica iz biblioteke dlib

Detektor uključen u dlib je implementacija One Millisecond Face Alignment with an Ensemble of Regression Trees rada autora Kazemi and Sullivan (2014).

Metoda kreće sa korišćenjem trening skupa obeleženih oznaka lica na slici. Ove slike su ručno označene, specificiranjem (x, y) koordinata regiona koji okružuju svaku od struktura lica. Imajući trening podatke, an ensemble of regresionih stabala je treniran da ustanovi pozicije oznaka lica direkno na osnovu intenziteta piksela (bez izdvajanja features). Ovaj detektor može detektovati oznake lica u realnom vremu sa veoma dobrim rezultatom. Detektuju se koordinate 68 tačaka koje se mapiraju na strukturu lica. Indeksi svih 68 tačaka prikazani su na slici:

Ova anotacija je deo iBUG 300-W dataset na kojem je dlib detektor treniran. Dlib detektor oznaka lica je najpoznatiji zbog svoje brzine i pouzdanosti dlib biblioteke. Dlib nudi i detektor koji pronalazi 5 karakteritičnih tačaka lica i brži je od 68-tačaka varijante. Preporučuje se za korišćenje ako su nam samo potrebne likacije nosa i očiju.

Jedan od najpoznatijih novijih detektira je iz MediaPipe biblioteke koji je sposoban da detektuje 3D mrežu lica. Ukoliko želimo da koristimo samo OpenCV biblioteku, i ona nudi ugrađeni detektor karakterističnih tačaka lica.

PORAVNANJE LICA

There are many forms of face alignment.

Some methods try to impose a (pre-defined) 3D model and then apply a transform to the input image such that the landmarks on the input face match the landmarks on the 3D model.

Other, more simplistic methods (like the one discussed in this blog post), rely only on the facial landmarks themselves (in particular, the eye regions) to obtain a normalized rotation, translation, and scale representation of the face.

The reason we perform this normalization is due to the fact that many facial recognition algorithms, including Eigenfaces, LBPs for face recognition, Fisherfaces, and deep learning/metric methods can all benefit from applying facial alignment before trying to identify the face.

Thus, face alignment can be seen as a form of “data normalization”. Just as you may normalize a set of feature vectors via zero centering or scaling to unit norm prior to training a machine learning model, it’s very common to align the faces in your dataset before training a face recognizer.

By performing this process, you’ll enjoy higher accuracy from your face recognition models.

The purpose of this blog post is to demonstrate how to align a face using OpenCV, Python, and facial landmarks.

Given a set of facial landmarks (the input coordinates) our goal is to warp and transform the image to an output coordinate space.

In this output coordinate space, all faces across an entire dataset should:

Be centered in the image.

Be rotated that such the eyes lie on a horizontal line (i.e., the face is rotated such that the eyes lie along the same y-coordinates).

Be scaled such that the size of the faces are approximately identical.

To accomplish this, we’ll first implement a dedicated Python class to align faces using an affine transformation. I’ve already implemented this FaceAligner class in imutils.

Note: Affine transformations are used for rotating, scaling, translating, etc. We can pack all three of the above requirements into a single cv2.warpAffine call; the trick is creating the rotation matrix, M .

We’ll then create an example driver Python script to accept an input image, detect faces, and align them.

Finally, we’ll review the results from our face alignment with OpenCV process.

desiredLeftEye : An optional (x, y) tuple with the default shown, specifying the desired output left eye position. For this variable, it is common to see percentages within the range of 20-40%. These percentages control how much of the face is visible after alignment. The exact percentages used will vary on an application-to-application basis. With 20% you’ll basically be getting a “zoomed in” view of the face, whereas with larger values the face will appear more “zoomed out.”

Next, let’s will compute the center of each eye as well as the angle between the eye centroids.

This angle serves as the key component for aligning our image.

On Lines 34 and 35 we compute the centroid, also known as the center of mass, of each eye by averaging all (x, y) points of each eye, respectively.

Given the eye centers, we can compute differences in (x, y)-coordinates and take the arc-tangent to obtain angle of rotation between eyes.

This angle will allow us to correct for rotation.

To determine the angle, we start by computing the delta in the y-direction, dY . This is done by finding the difference between the rightEyeCenter and the leftEyeCenter on Line 38.

Similarly, we compute dX , the delta in the x-direction on Line 39.

Next, on Line 40, we compute the angle of the face rotation. We use NumPy’s arctan2 function with arguments dY and dX , followed by converting to degrees while subtracting 180 to obtain the angle.

In the following code block we compute the desired right eye coordinate (as a function of the left eye placement) as well as calculating the scale of the new resulting image.

On Line 44, we calculate the desired right eye based upon the desired left eye x-coordinate. We subtract self.desiredLeftEye[0] from 1.0 because the desiredRightEyeX value should be equidistant from the right edge of the image as the corresponding left eye x-coordinate is from its left edge.

We can then determine the scale of the face by taking the ratio of the distance between the eyes in the current image to the distance between eyes in the desired image

First, we compute the Euclidean distance ratio, dist , on Line 50.

Next, on Line 51, using the difference between the right and left eye x-values we compute the desired distance, desiredDist . We update the desiredDist by multiplying it by the desiredFaceWidth on Line 52. This essentially scales our eye distance based on the desired width.

Finally, our scale is computed by dividing desiredDist by our previously calculated dist .

Now that we have our rotation angle and scale , we will need to take a few steps before we compute the affine transformation. This includes finding the midpoint between the eyes as well as calculating the rotation matrix and updating its translation component On Lines 57 and 58, we compute eyesCenter , the midpoint between the left and right eyes. This will be used in our rotation matrix calculation. In essence, this midpoint is at the top of the nose and is the point at which we will rotate the face around:

To compute our rotation matrix, M , we utilize cv2.getRotationMatrix2D specifying eyesCenter , angle , and scale (Line 61). Each of these three values have been previously computed, so refer back to Line 40, Line 53, and Line 57 as needed.

A description of the parameters to cv2.getRotationMatrix2D follow:

eyesCenter : The midpoint between the eyes is the point at which we will rotate the face around.

angle : The angle we will rotate the face to to ensure the eyes lie along the same horizontal line.

scale : The percentage that we will scale up or down the image, ensuring that the image scales to the desired size.

Now we must update the translation component of the matrix so that the face is still in the image after the affine transform.

On Line 64, we take half of the desiredFaceWidth and store the value as tX , the translation in the x-direction.

To compute tY , the translation in the y-direction, we multiply the desiredFaceHeight by the desired left eye y-value, desiredLeftEye[1] .

Using tX and tY , we update the translation component of the matrix by subtracting each value from their corresponding eyes midpoint value, eyesCenter (Lines 66 and 67).

We can now apply our affine transformation to align the face:

For convenience we store the desiredFaceWidth and desiredFaceHeight into w and h respectively (Line 70).

Then we perform our last step on Lines 70 and 71 by making a call to cv2.warpAffine . This function call requires 3 parameters and 1 optional parameter:

image : The face image.

M : The translation, rotation, and scaling matrix.

(w, h) : The desired width and height of the output face.

flags : The interpolation algorithm to use for the warp, in this case INTER\_CUBIC . To read about the other possible flags and image transformations, please consult the OpenCV documentation.

Finally, we return the aligned face on Line 75.

# **Опис практичног дела**

# **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/deep-learning/object-detection/>
12. <https://arxiv.org/abs/1506.02640>
13. <https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>
14. <https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e>
15. https://arxiv.org/abs/1512.02325

1. LMS (Learning Management System) платформа је софтвер који се користи за креирање, администрацију и праћење тренинга и обука. [↑](#footnote-ref-1)
2. 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. [13] [↑](#footnote-ref-2)